OpenShift Container Platform 4.16

安装

安装并配置 OpenShift Container Platform 集群

Last Updated: 2024-07-03

OpenShift Container Platform 4.16 安装

安装并配置 OpenShift Container Platform 集群

法律通告

摘要

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

目录

第 1 章 OPENSHIFT CONTAINER PLATFORM 安装概述

1.1. 关于 OPENSHIFT CONTAINER PLATFORM 安装

OpenShift Container Platform 安装程序提供了四个部署集群的方法，相关信息包括在以下列表中：

- **交互式** ：您可以使用基于 Web 的 辅助安装程序（Assisted Installer）部署集群。对于可以连接到互联网的网络，这是一个理想的方法。Assisted Installer 是安装 OpenShift Container Platform 的最简单方法，它提供智能默认值，并在安装集群前执行预动态验证。它还提供了一个 RESTful API 用于自动化和高级配置场景。

- **本地基于代理的** ：对于断开连接的环境或有网络限制的环境，您可以使用基于代理的安装程序。它提供了 Assisted Installer 的许多优点，但您必须首先下载并配置基于代理的安装程序。使用命令行界面完成配置。这个方法适用于断开连接的环境。

- **自动** ：您可以在安装程序置备的基础架构中部署集群。安装程序使用每个集群主机的基板管理控制器 (BMC) 进行置备。您可以在有连接或断开连接的环境中部署集群。

- **完全控制**：您可以在自己准备和维护的基础架构上部署集群，这种方法提供了最大的定制性。您可以在有连接或断开连接的环境中部署集群。

每种方法部署的集群具有以下特征：

- 没有单点故障的高可用性基础架构，默认可用。

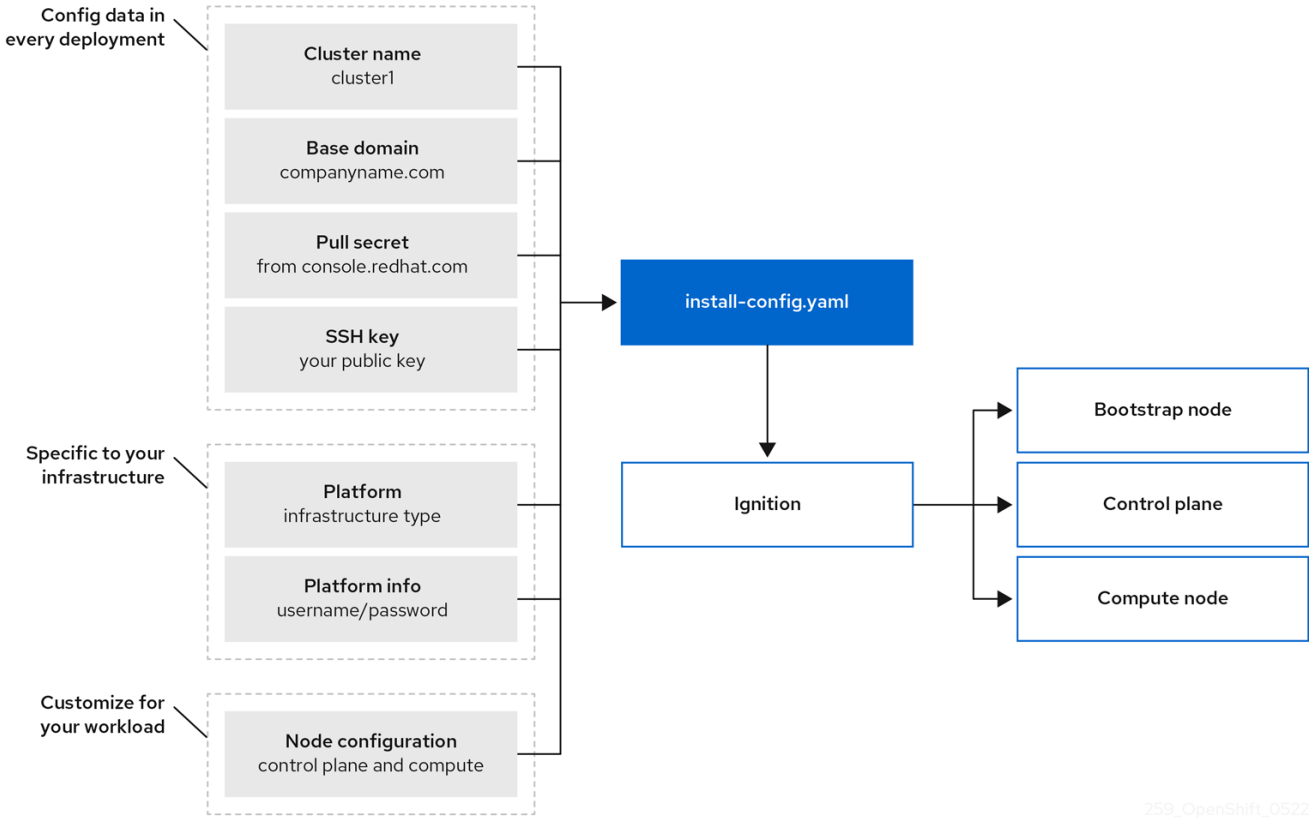
- 管理员可以控制要应用的更新，以及应用的时间。

1.1.1. 关于安装程序

您可以使用安装程序部署每种集群。安装程序会生成主要资产，如 bootstrap、control plane 和计算机器的 Ignition 配置文件。您可以使用这三个机器配置开始使用 OpenShift Container Platform 集群，它为您提供了正确配置的基础架构。

OpenShift Container Platform 安装程序使用一组目标和依赖项来管理集群安装。安装程序具有一组必须实现的目标，并且每个目标都有一组依赖项。因为每个目标仅关注其自己的依赖项，所以安装程序可以并行地实现多个目标，最终组成一个正常运行的集群。安装程序会识别并使用现有组件，而不是运行命令来再次创建它们，因为程序满足依赖项。

图 1.1. OpenShift Container Platform 安装目标和依赖项



1.1.2. 关于 Red Hat Enterprise Linux CoreOS (RHCOS)

在安装后，每一个集群机器都将使用 Red Hat Enterprise Linux CoreOS (RHCOS) 作为操作系统。RHCOS 是 Red Hat Enterprise Linux (RHEL) 的不可变容器主机版本，具有默认启用 SELinux 的 RHEL 内核。RHCOS 包括作为 Kubernetes 节点代理的 **kubelet**，以及为 Kubernetes 优化的 CRI-O 容器运行时。

OpenShift Container Platform 4.16 集群中的每一 control plane 机器都必须使用 RHCOS，其中包括一个关键的首次启动置备工具，称为 Ignition。这一工具让集群能够配置机器。操作系统更新作为可引导容器镜像（使用 **OSTree** 作为后端）提供，该镜像由 Machine Config Operator 在集群中部署。实际的操作系统更改通过使用 **rpm-ostree** 在每台机器上作为原子操作原位进行。通过结合使用这些技术，OpenShift Container Platform 可以像管理集群上的任何其他应用程序一样管理操作系统，通过原位升级使整个平台保持最新状态。这些原位更新可以减轻运维团队的负担。

如果将 RHCOS 用作所有集群机器的操作系统，则集群将管理其组件和机器的所有方面，包括操作系统在内。因此，只有安装程序和 Machine Config Operator 才能更改机器。安装程序使用 Ignition 配置文件设置每台机器的确切状态，安装后则由 Machine Config Operator 完成对机器的更多更改，例如应用新证书或密钥等。

1.1.3. OpenShift Container Platform 安装的常见术语表

术语表定义了与安装内容相关的常用术语。参阅以下术语列表以更好地了解安装过程。

支持的安装程序

在 console.redhat.com 中托管的安装程序，它提供基于 web 用户界面或 RESTful API 用于创建集群配置。Assisted Installer 会生成一个发现镜像。集群机器使用发现镜像进行引导，它会安装 RHCOS 和代理。Assisted Installer 和代理一起为集群提供了预安装验证和安装功能。

基于代理的安装程序

与 Assisted Installer 类似的安装程序，但您必须首先下载基于代理的安装程序。基于代理的安装程序是断开连接的环境的理想选择。

Bootstrap 节点

一个临时的机器，它运行最小需要的 Kubernetes 配置来部署 OpenShift Container Platform 控制平面（control plane）。

Control plane（控制平面）

一个容器编配层，用于公开 API 和接口来定义、部署和管理容器的生命周期。也称为 control plane 机器。

Compute 节点

负责执行集群用户工作负载的节点。也称为 worker 节点。

断开连接的安装

在有些情况下，数据中心的部分环境可能无法访问互联网，甚至无法通过代理服务器访问。您仍可在这些环境中安装 OpenShift Container Platform，但需要先下载所需的软件和镜像，并将其保存在离线环境中。

OpenShift Container Platform 安装程序

置备基础架构并部署集群的程序。

安装程序置备的基础架构

安装程序部署并配置运行集群的基础架构。

Ignition 配置文件

Ignition 工具用于在操作系统初始化过程中配置 Red Hat Enterprise Linux CoreOS (RHCOS)的文件。安装程序生成不同的 Ignition 配置文件来初始化 bootstrap、control plane 和 worker 节点。

Kubernetes 清单

JSON 或 YAML 格式的 Kubernetes API 对象的规格。配置文件可以包含部署、配置映射、secret 和 daemonset 等。

Kubelet

在集群的每个节点上运行的一个主节点代理，以确保容器在 pod 中运行。

负载均衡器

负载均衡器是客户端的单点联系。API 的负载均衡器在 control plane 节点之间分布传入的流量。

Machine Config Operator

一个 Operator，管理并应用基本操作系统和容器运行时的配置和更新，包括内核和 kubelet 之间的所有配置和更新。

Operator

在 OpenShift Container Platform 集群中打包、部署和管理 Kubernetes 应用程序的首选方法。Operator 将人类操作知识编码到一个软件程序中，易于打包并与客户共享。

用户置备的基础架构

您可以在自己提供的基础架构上安装 OpenShift Container Platform。您可以使用安装程序来生成置备集群基础架构所需的资产，再创建集群基础架构，然后将集群部署到您提供的基础架构中。

1.1.4. 安装过程

除了 Assisted Installer 外，当安装 OpenShift Container Platform 集群时，您必须从 OpenShift Cluster Manager Hybrid Cloud Console 上的适当的 **Cluster Type** 页面下载安装程序。此控制台管理：

- 帐户的 REST API。
- registry 令牌，这是用于获取所需组件的 pull secret。

- 集群注册，将集群身份与您的红帽帐户相关联，以便收集使用指标。

在 OpenShift Container Platform 4.16 中，安装程序是对一组资产执行一系列文件转换的 Go 二进制文件。与安装程序交互的方式因您的安装类型而异。考虑以下安装用例：

- 要使用 Assisted Installer 部署集群，您可以使用 Assisted Installer 配置集群设置。没有安装程序可以下载和配置。设置完集群配置后，您可以下载发现 ISO，然后使用该镜像引导集群机器。您可以使用 Assisted Installer 在完全集成的 Nutanix、vSphere 和裸机上安装集群，以及以前没有集成的环境中安装集群。如果在裸机上安装，您需要提供所有集群基础架构和资源，包括网络、负载均衡、存储和所有集群机器。

- 要使用基于代理的安装程序部署集群，您可以首先下载基于代理的安装程序。然后，您可以配置集群并生成发现镜像。您可以使用发现镜像引导集群机器，它会安装一个与安装程序进行通信的代理，并为您处理置备，您不需要与安装程序进行交互或自行设置置备程序机器。您需要提供所有集群基础架构和资源，包括网络、负载均衡、存储和单个集群机器。这个方法适用于断开连接的环境。

- 对于具有安装程序置备的基础架构集群，您可以将基础架构启动和置备委派给安装程序，而不是亲自执行。安装程序将创建支持集群所需的所有网络、机器和操作系统，除非您裁裸机上安装。如果在裸机上安装，您必须提供所有集群基础架构和资源，包括 bootstrap 机器、网络、负载均衡、存储和单个集群机器。

- 如果亲自为集群置备和管理基础架构，则必须提供所有集群基础架构和资源，包括 Bootstrap 机器、网络、负载均衡、存储和独立的集群机器。

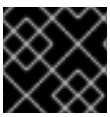
对于安装程序，在安装过程中会使用三组文件：名为 **install-config.yaml** 的安装配置文件、Kubernetes 清单，以及您的集群类型的 Ignition 配置文件。

重要

在安装过程中，您可以修改控制基础 RHCOS 操作系统的 Kubernetes 和 Ignition 配置文件。但是，没有可用的验证机制来确认您对这些对象所做修改是适当的。如果修改了这些对象，集群可能会无法运行。由于存在这种风险，修改 Kubernetes 和 Ignition 配置文件不受支持，除非您遵循记录的流程或在红帽支持指示下操作。

安装配置文件转换为 Kubernetes 清单，然后清单嵌套到 Ignition 配置文件中。安装程序使用这些 Ignition 配置文件来创建集群。

运行安装程序时，所有配置文件会被修剪，因此请务必备份需要再次使用的所有配置文件。



重要

安装之后，您无法修改在安装过程中设置的参数，但可以修改一些集群属性。

使用辅助安装程序的安装过程

使用辅助安装程序进行安装涉及使用基于 Web 的用户界面或使用 RESTful API 以互动方式创建集群配置。Assisted Installer 用户界面会提示您输入所需的值，并为其余参数提供合理的默认值，除非在用户界面或使用 API 中更改它们。Assisted Installer 生成发现镜像，您可以下载并用来引导集群机器。镜像安装 RHCOS 和代理，代理会为您处理置备。您可以使用 Assisted Installer 安装 OpenShift Container Platform，并在 Nutanix、vSphere 和裸机上完全集成。另外，您可以在其他没有集成的情况下使用 Assisted Installer 安装 OpenShift Container Platform。

OpenShift Container Platform 管理集群的所有方面，包括操作系统本身。每台机器在启动时使用的配置引用其加入的集群中托管的资源。此配置允许集群在应用更新时自行管理。

如果可能，请使用 Assisted Installer 功能来避免下载和配置基于代理的安装程序。

基于代理的基础架构的安装过程
基于代理的安装与使用 Assisted Installer 类似，唯一的不同是需要在最初下载并安装基于代理的安装程序。当您希望利用 Assisted Installer 所带来的变量，并需要在断开连接的环境中安装集群时，可以使用基于代理的安装。

如果可能，请使用基于代理的安装功能来避免创建带有 bootstrap 虚拟机的置备程序机器，然后置备和维护集群基础架构。

采用安装程序置备的基础架构的安装过程
默认安装类型为使用安装程序置备的基础架构。默认情况下，安装程序充当安装向导，提示您输入它无法自行确定的值，并为其余参数提供合理的默认值。您还可以自定义安装过程来支持高级基础架构场景。安装程序将为集群置备底层基础架构。

您可以安装标准集群或自定义集群。对于标准集群，您要提供安装集群所需的最低限度详细信息。对于自定义集群，您可以指定有关平台的更多详细信息，如 control plane 使用的机器数量、集群部署的虚拟机的类型，或 Kubernetes 服务网络的 CIDR 范围。

若有可能，可以使用此功能来避免置备和维护集群基础架构。在所有其他环境中，可以使用安装程序来生成置备集群基础架构所需的资产。

对于安装程序置备的基础架构的集群，OpenShift Container Platform 可以管理集群的所有方面，包括操作系统本身。每台机器在启动时使用的配置引用其加入的集群中托管的资源。此配置允许集群在应用更新时自行管理。

采用用户置备的基础架构的安装过程
您还可以在自己提供的基础架构上安装 OpenShift Container Platform。您可以使用安装程序来生成置备集群基础架构所需的资产，再创建集群基础架构，然后将集群部署到您提供的基础架构中。

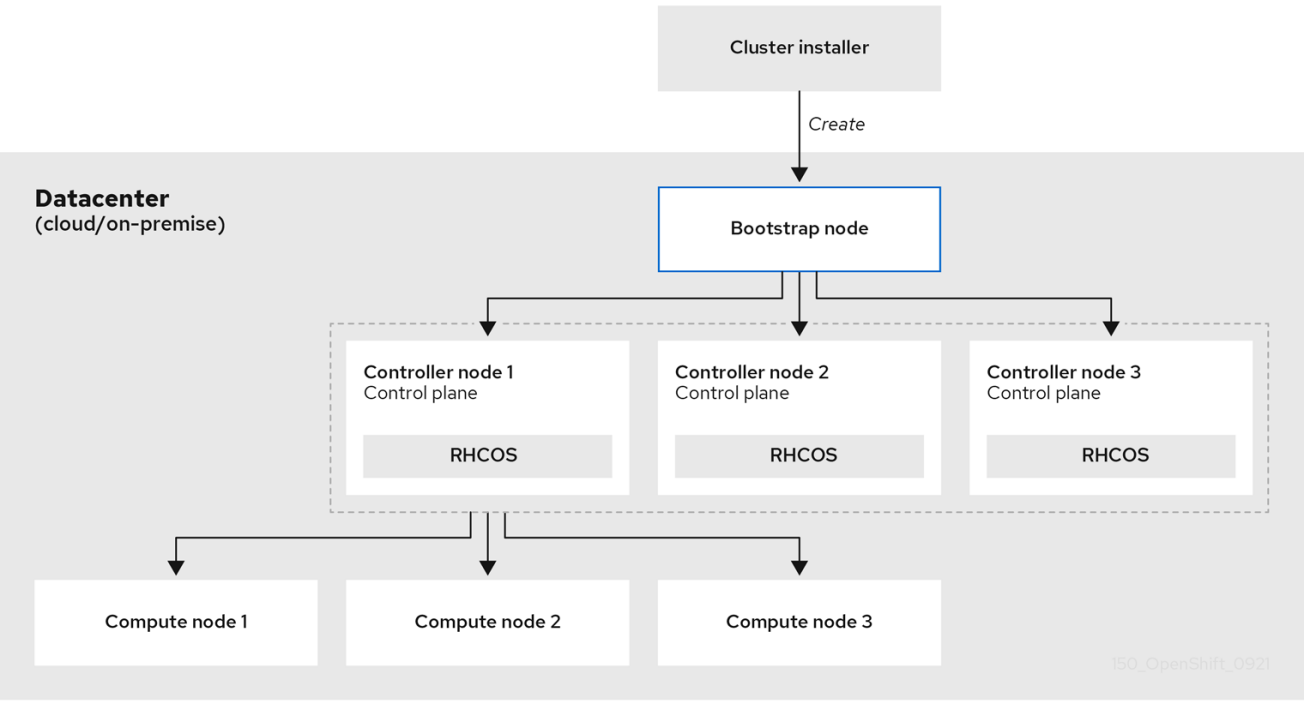
如果不使用安装程序置备的基础架构，您必须自己管理和维护集群资源。以下列表详细介绍了其中一些自我管理的资源：

- 组成集群的 control plane 和计算机器的底层基础架构
- 负载均衡器
- 集群网络，包括 DNS 记录和所需的子网
- 集群基础架构和应用程序的存储

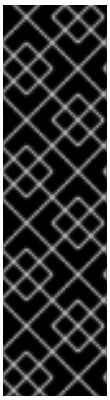
如果您的集群使用用户置备的基础架构，您可以选择将 RHEL 计算机器添加到集群中。

安装过程详细信息
置备集群时，集群中的每台机器都需要有关集群的信息。OpenShift Container Platform 在初始配置过程中使用临时 bootstrap 机器为永久 control plane 提供所需的信息。临时 bootstrap 机器使用一个带有描述如何创建集群的 Ignition 配置文件进行引导。bootstrap 机器创建组成控制平面（control plane）的 control plane 机器。然后，control plane 机器创建计算（compute）机器。下图说明了这一过程：

图 1.2. 创建 bootstrap、control plane 和计算机器



集群机器初始化后，Bootstrap 机器将被销毁。所有集群都使用 Bootstrap 过程来初始化集群，但若您自己置备集群的基础架构，则必须手动完成许多步骤。



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

bootstrapp 集群涉及以下步骤：

1. bootstrap 机器启动并开始托管 control plane 机器引导所需的远程资源。如果您置备基础架构，此步骤需要人工干预。
2. bootstrap 机器启动单节点 etcd 集群和一个临时 Kubernetes control plane。
3. control plane 机器从 bootstrap 机器获取远程资源并完成启动。如果您置备基础架构，此步骤需要人工干预。
4. 临时 control plane 将生产环境的 control plane 调度到生产环境 control plane 机器。
5. Cluster Version Operator（CVO）在线并安装 etcd Operator。etcd Operator 在所有 control plane 节点上扩展 etcd。
6. 临时 control plane 关机，并将控制权交给生产环境 control plane。

7. bootstrap 机器将 OpenShift Container Platform 组件注入生产环境 control plane。
8. 安装程序关闭 bootstrap 机器。如果您置备基础架构，此步骤需要人工干预。
9. control plane 设置计算节点。
10. control plane 以一组 Operator 的形式安装其他服务。

完成此 bootstrap 过程后，将生成一个全面运作的 OpenShift Container Platform 集群。然后，集群会下载并配置日常运作所需的其余组件，包括在受支持的环境中创建计算（compute）机器。

其他资源

- Red Hat OpenShift Network Calculator

1.1.5. 安装后验证节点状态

当以下安装健康检查成功时，OpenShift Container Platform 安装会完成：

- 置备程序可以访问 OpenShift Container Platform Web 控制台。
- 所有 control plane 节点都已就绪。
- 所有集群 Operator 都可用。

注意

安装完成后，负责 worker 节点的特定集群 Operator 持续尝试置备所有 worker 节点。在所有 worker 节点都报告为 **READY** 之前需要一段时间。对于在裸机上的安装，请等待至少 60 分钟，然后对 worker 节点进行故障排除。对于所有其他平台上安装，请等待至少 40 分钟后再对 worker 节点进行故障排除。负责 worker 节点的集群 Operator 的 **DEGRADED** 状态取决于 Operator 自己的资源，而不是节点的状态。

安装完成后，您可以继续监控集群中的节点条件。

先决条件

- 安装程序在终端中成功解决。

流程

1. 显示所有 worker 节点的状态：

```
$ oc get nodes
```

输出示例

```
NAME                         STATUS   ROLES    AGE   VERSION
example-compute1.example.com   Ready    worker   13m   v1.21.6+bb8d50a
example-compute2.example.com   Ready    worker   13m   v1.21.6+bb8d50a
example-compute4.example.com   Ready    worker   14m   v1.21.6+bb8d50a
example-control1.example.com   Ready    master   52m   v1.21.6+bb8d50a
example-control2.example.com   Ready    master   55m   v1.21.6+bb8d50a
example-control3.example.com   Ready    master   55m   v1.21.6+bb8d50a
```

2. 显示所有 worker 机器节点的阶段：

```
$ oc get machines -A
```

输出示例

```
NAMESPACE               NAME                              PHASE     TYPE   REGION   ZONE   AGE
openshift-machine-api   example-zbbt6-master-0            Running                          95m
openshift-machine-api   example-zbbt6-master-1            Running                          95m
openshift-machine-api   example-zbbt6-master-2            Running                          95m
openshift-machine-api   example-zbbt6-worker-0-25bhp      Running                          49m
openshift-machine-api   example-zbbt6-worker-0-8b4c2      Running                          49m
openshift-machine-api   example-zbbt6-worker-0-jkbqt      Running                          49m
openshift-machine-api   example-zbbt6-worker-0-qrl5b      Running                          49m
```

其他资源

- 获取 BareMetalHost 资源
- 安装后
- 验证安装
- 基于代理的安装程序
- OpenShift Container Platform 支持的安装程序

安装范围

OpenShift Container Platform 安装程序的作用范围特意设计得比较狭窄。它旨在简化操作并确保成功。安装完成后，您可以完成更多的配置任务。

其他资源

- 如需有关 OpenShift Container Platform 配置资源的详细信息，请参阅可用的集群自定义。

1.1.6. OpenShift Local 概述

OpenShift Local 支持快速应用程序开发，以开始构建 OpenShift Container Platform 集群。OpenShift Local 设计为在本地计算机上运行，以简化设置和测试，并使用开发基于容器的应用所需的所有工具在本地模拟云环境。

无论您使用什么编程语言，OpenShift Local 都可以托管您的应用程序，并将最小预配置的 Red Hat OpenShift Container Platform 集群引入本地 PC，而无需基于服务器的基础架构。

在托管环境中，OpenShift Local 可以创建微服务，将它们转换为镜像，并在运行 Linux、macOS 或 Windows 10 或更高版本的笔记本电脑或桌面上直接运行它们。

如需有关 OpenShift Local 的更多信息，请参阅 Red Hat OpenShift Local Overview 。

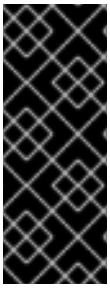
1.2. OPENSHIFT CONTAINER PLATFORM 集群支持的平台

在 OpenShift Container Platform 4.16 中，您可以在以下平台上安装使用安装程序置备的基础架构集群：

- Alibaba Cloud

- Amazon Web Services (AWS)
- 裸机
- Google Cloud Platform (GCP)
- IBM Cloud®
- Microsoft Azure
- Microsoft Azure Stack Hub
- Nutanix
- Red Hat OpenStack Platform (RHOSP)
  - 最新的 OpenShift Container Platform 版本支持最新的 RHOSP 长生命版本和中间版本。如需完整的 RHOSP 发行版本兼容性信息，请参阅 RHOSP 上的 OpenShift Container Platform 支持列表。
- VMware vSphere

对于所有这些集群，包括用来运行安装过程的计算机在内的所有机器都必须可直接访问互联网，以便为平台容器拉取镜像并向红帽提供 telemetry 数据。



重要

安装后，不支持以下更改：

- 混合云供应商平台.
- 混合云供应商组件。例如，在安装集群的平台上使用另一个平台的持久性存储框架。

在 OpenShift Container Platform 4.16 中，您可以在以下平台上安装使用用户置备的基础架构集群：

- AWS
- Azure
- Azure Stack Hub
- 裸机
- GCP
- IBM Power®
- IBM Z® 或 IBM® LinuxONE
- RHOSP
  - 最新的 OpenShift Container Platform 版本支持最新的 RHOSP 长生命版本和中间版本。如需完整的 RHOSP 发行版本兼容性信息，请参阅 RHOSP 上的 OpenShift Container Platform 支持列表。
- AWS 上的 VMware Cloud

- VMware vSphere

根据平台支持的情况，您可以在用户置备的基础架构上执行安装，以便您可以运行具有完整互联网访问的机器，将集群放在一个代理的后面，或者执行断开连接的安装。

在断开连接的网络安装中，您可以下载安装集群所需的镜像（image），将它们放在镜像 registry（mirror registry）中，然后使用那些数据安装集群。虽然您需要访问互联网来为平台容器拉取镜像，但在 vSphere 或裸机基础架构上进行断开连接的网络安装，您的集群机器不需要直接访问互联网。

OpenShift Container Platform 4.x Tested Integrations 页面中提供了有关针对不同平台进行集成测试的详细信息。

其他资源

- 如需了解每个支持的平台可用的安装类型的更多信息，请参阅不同平台支持的安装方法。
- 有关选择安装方法以及准备所需资源的信息，请参阅选择集群安装方法并为用户准备。
- Red Hat OpenShift Network Calculator 可帮助您在部署和扩展阶段设计集群网络。它解决了与集群网络相关的常见问题，并以方便的 JSON 格式提供输出。

第 2 章 选择集群安装方法并为用户准备它

在安装 OpenShift Container Platform 前，请确定您拥有为用户准备集群所需的所有所需资源。

2.1. 选择集群安装类型

在安装 OpenShift Container Platform 集群前，需要选择最佳安装说明。请考虑您对以下问题的回答，以选择最佳选择。

2.1.1. 您要自己安装和管理 OpenShift Container Platform 集群吗？

如果要自己安装和管理 OpenShift Container Platform，您可以在以下平台上安装它：

- 64 位 x86 实例上的 Amazon Web Services (AWS)
- 64 位 ARM 实例上的 Amazon Web Services (AWS)
- 64 位 x86 实例上的 Microsoft Azure
- 64 位 ARM 实例上的 Microsoft Azure
- Microsoft Azure Stack Hub
- 64 位 x86 实例上的 Google Cloud Platform (GCP)
- 64 位 ARM 实例上的 Google Cloud Platform (GCP)
- Red Hat OpenStack Platform (RHOSP)
- IBM Cloud®
- IBM Z® 或 IBM® LinuxONE
- IBM Z® or IBM® LinuxONE for Red Hat Enterprise Linux (RHEL) KVM
- IBM Power®
- IBM Power® Virtual Server
- Nutanix
- VMware vSphere
- 裸机或其他平台基础架构

您可以将 OpenShift Container Platform 4 集群部署到内部硬件环境，或部署到云托管服务中，但集群中的所有机器都必须位于相同的数据中心或云托管服务中。

如果要使用 OpenShift Container Platform，但不想自行管理集群，则有几个受管服务选项。如果要完全由红帽管理的集群，可以使用 OpenShift Dedicated 或 OpenShift Online。您还可以在 Azure、AWS、IBM Cloud® 或 Google Cloud 上使用 OpenShift 作为受管服务。有关受管服务的更多信息，请参阅 OpenShift 产品页。如果您安装了使用云虚拟机作为虚拟裸机的 OpenShift Container Platform 集群，则其对应的基于云的存储不被支持。

2.1.2. 您是否已使用了 OpenShift Container Platform 3 且要使用 OpenShift Container Platform 4?

如果您已使用了 OpenShift Container Platform 3 并希望尝试 OpenShift Container Platform 4，则需要了解 OpenShift Container Platform 4 的不同。OpenShift Container Platform 4 将无缝地集成了软件包、部署和管理 Kubernetes 应用程序以及平台在 Red Hat Enterprise Linux CoreOS（RHCOS）上运行的 Operator。与其他需要部署机器并配置其操作系统以便在其中安装 OpenShift Container Platform 的系统不同，RHCOS 操作系统是 OpenShift Container Platform 集群的一个内部组成部分。为集群机器部署操作系统是 OpenShift Container Platform 的安装过程的一部分。请参阅 OpenShift Container Platform 3 和 4 之间的差别。

由于需要置备机器作为 OpenShift Container Platform 集群安装过程的一部分，所以无法将 OpenShift Container Platform 3 集群升级到 OpenShift Container Platform 4。相反，您必须创建新的 OpenShift Container Platform 4 集群，并将 OpenShift Container Platform 3 工作负载迁移到它们。有关迁移的更多信息，请参阅从 OpenShift Container Platform 3 迁移到 4 概述 。由于必须迁移到 OpenShift Container Platform 4，因此可以使用任何类型的生产环境集群安装过程来创建新集群。

2.1.3. 您是否希望在您的集群中使用已存在的组件？

由于操作系统是 OpenShift Container Platform 集成的一部分，因此让安装程序可以更轻松地支持所有基础架构。它们被称为*安装程序置备的基础架构*安装。在这种安装中，您可以为集群提供一些现有的基础架构，但安装程序会部署集群初始需要的所有机器。

您可以在不对集群或其底层机器自定义 AWS, Azure, Azure Stack Hub, GCP, 或 Nutanix 的情况下部署安装程序置备的基础架构集群。

如果需要为安装程序置备的基础架构集群执行基本配置，如集群机器的实例类型，您可以自定义 AWS、Azure 、GCP、Nutanix 的安装。

对于安装程序置备的基础架构安装，您可以使用现有的 VPC in AWS, vNet in Azure, 或 VPC in GCP。您还可以重复使用网络基础架构的一部分，以便 AWS、Azure、GCP 中的集群可以与环境中的现有 IP 地址分配共存，并与现有的 MTU 和 VXLAN 配置集成。如果在这些云上已有帐户和凭证，您可以重复使用这些帐户，但可能需要修改帐户，以便具有在它们上安装 OpenShift Container Platform 集群所需的权限。

您可以使用安装程序置备的基础架构方法在硬件上为 vSphere 和裸机创建适当的机器实例。另外，对于 vSphere，您还可以在安装过程中自定义额外网络参数。

如果要重复使用广泛的云基础架构，可以完成*用户置备的基础架构*安装。使用这些安装，您可以在安装过程中手动部署集群所需的机器。如果在 AWS、Azure、Azure Stack Hub 上执行用户置备的基础架构安装，您可以使用提供的模板来帮助备份所有需要的组件。您还可以重复使用一个共享的 VPC on GCP。或者，您可以使用 供应商安装方法 将集群部署到其他云中。

您还可以在现有硬件上完成用户置备的基础架构安装。如果您使用 RHOSP, IBM Z® or IBM® LinuxONE, IBM Z® and IBM® LinuxONE with RHEL KVM, IBM Power, 或 vSphere，请使用特定的安装说明来部署集群。如果您使用其他支持的硬件，请按照裸机安装过程进行操作。对于其中一些平台，如 vSphere 和裸机，您也可以在安装过程中自定义额外网络参数。

2.1.4. 您的集群是否需要额外的安全性？

如果使用用户置备的安装方法，您可以为集群配置代理。这些说明包含在每个安装过程中。

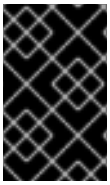
如果要防止公有云中的集群从外部公开端点，您可以在 AWS、Azure 或 GCP 上使用安装程序置备的基础架构部署私有集群。

如果您需要安装对互联网有限访问的集群，如断开连接的或受限的网络集群，您可以镜像安装软件包并从中安装集群。按照用户置备的基础架构安装到 AWS, GCP, IBM Z® or IBM® LinuxONE, IBM Z® or IBM®

LinuxONE with RHEL KVM, IBM Power®, vSphere, 或裸机受限网络中的详细说明。您还可以按照 AWS,GCP,IBM Cloud®, Nutanix,RHOSP, 和 vSphere 的详细信息，使用安装程序置备的基础架构将集群安装到受限网络中。

如果需要将集群部署到 AWS GovCloud 区域、AWS 中国区域或 Azure 政府区域，您可以在安装程序置备的基础架构安装过程中配置这些自定义区域。

您还可以将集群机器配置为使用 RHEL 加密库在安装过程中为 FIPS 140-2/140-3 Validation 提交给 NIST。



重要

当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS)时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。

2.2. 安装后为用户准备集群

在安装集群时不需要进行一些配置，但建议在用户访问集群前进行操作。您可以通过自定义组成集群的 Operator，并将集群与其他所需系统（如身份提供程序）集成，从而自定义集群本身。

对于生产环境集群，您必须配置以下集成：

- 持久性存储
- 身份供应商
- 监控 OpenShift Container Platform 核心组件

2.3. 为工作负载准备集群

根据工作负载需要，您可能需要在开始部署应用程序前执行额外的步骤。例如，在为应用程序构建策略准备了基础架构后，您可能需要为低延迟工作负载置备或保护敏感工作负载。您还可以为应用程序工作负载配置 监控。

2.4. 支持的用于不同平台的安装方法

您可以在不同的平台上执行不同类型的安装。

注意

不是所有安装选项都支持所有平台，如下表所示。勾选标记代表支持的选项，并链接到相关部分。

表 2.1. 安装程序置备的基础架构选项

	AWS (64位x86)	AWS (64位ARM)	Azure (64位x86)	Azure (64位ARM)	Azure Stack Hub	GCP (64位x86)	GCP (64位ARM)	Nutanix	RHOSP	裸机 (64位x86)	裸机 (64位ARM)	vSphere	IBM Cloud®	IBM Z®	IBM Power®	IBM Power® Virtual Server
Default（默认）	✓	✓	✓	✓	✓	✓	✓	✓		✓	✓	✓	✓			
Custom	✓	✓	✓	✓	✓	✓	✓	✓	✓			✓	✓			✓
网络自定义	✓	✓	✓	✓	✓	✓	✓					✓	✓			
Restrictednetwork	✓	✓	✓	✓		✓	✓	✓	✓	✓	✓	✓	✓			✓
私有集群	✓	✓	✓	✓		✓	✓						✓			✓

	AWS (64 位 x86)	AWS (64 位 ARM)	Azure (64 位 x86)	Azure (64 位 ARM)	Azure Stack Hub	GCP (64 位 x86)	GCP (64 位 ARM)	Nutanix	RHOSP	裸机 (64 位 x86)	裸机 (64 位 ARM)	vSphere	IBM Cloud®	IBM Z®	IBM Power®	IBM Power® Virtual Server
现有的虚拟私有网络	✓	✓	✓	✓		✓	✓						✓			✓
政府区域	✓		✓													
Secret 区域	✓															
中国区域	✓															

表 2.2. 用户置备的基础架构

	AWS (64位x86)	AWS (64位ARM)	Azure (64位x86)	Azure (64位ARM)	Azure Stack Hub	GCP (64位x86)	GCP (64位ARM)	Nutanix	RHOSP	裸机 (64位x86)	裸机 (64位ARM)	vSphere	IBM Cloud®	IBM Z®	使用RHEL KVM的IBM Z®	IBM Power®	平台无关
Custom	✓	✓	✓	✓	✓	✓	✓		✓	✓	✓	✓		✓	✓	✓	✓
网络自定义										✓	✓	✓					
Restricted network	✓	✓				✓	✓			✓	✓	✓		✓	✓	✓	
在集群项目外托管共享VPC						✓	✓										

第 3 章 集群功能

集群管理员可在安装前使用集群功能启用或禁用可选组件。集群管理员可以在安装后随时启用集群功能。

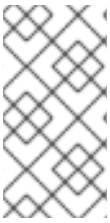
注意

集群管理员无法在启用集群后禁用集群功能。

3.1. 选择集群功能

您可以根据包含自定义集群的安装方法之一来选择集群功能，如"使用自定义配置"在 AWS 上安装集群，或使用自定义在 GCP 上安装集群"。

在自定义安装过程中，您可以创建一个 **install-config.yaml** 文件，其中包含集群的配置参数。



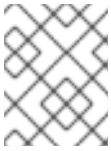
注意

如果通过启用或禁用特定集群功能自定义集群，则需要手动维护 **install-config.yaml** 文件。新的 OpenShift Container Platform 更新可能会为现有组件声明新的功能处理，或者完全引入新的组件。自定义 **install-config.yaml** 文件的用户应该会在 OpenShift Container Platform 更新时定期更新其 **install-config.yaml** 文件。

您可以使用以下配置参数来选择集群功能：

```
capabilities:
  baselineCapabilitySet: v4.11 (1)
  additionalEnabledCapabilities: (2)
  - CSISnapshot
  - Console
  - Storage
```

(1) 定义要安装的一组基准功能。有效值为 **None**、**vCurrent** 和 **v4.x**。如果选择 **None**，则会禁用所有可选功能。默认值为 **vCurrent**，它启用了所有可选功能。



注意

v4.x 代表最高为当前集群版本（包括当前版本）的任何值。例如，OpenShift Container Platform 4.12 集群的有效值为 **v4.11** 和 **v4.12**。

(2) 定义要显式启用的功能列表。除了 **baselineCapabilitySet** 中指定的能力外，它们也会启用。

注意

在本例中，默认能力被设置为 **v4.11**。**additionalEnabledCapabilities** 字段启用了默认的 **v4.11** 功能集的额外功能。

下表描述了 **baselineCapabilitySet** 值。

表 3.1. 集群功能 baselineCapabilitySet 值描述

值	描述
vCurrent	当您要自动添加新版本中引入的新功能时，指定这个选项。
v4.11	当要为 OpenShift Container Platform 4.11 启用默认功能时指定这个选项。通过指定 **v4.11**，不会启用较新版本的 OpenShift Container Platform 中引入的功能。OpenShift Container Platform 4.11 中的默认功能是 **baremetal**、**MachineAPI**、**marketplace** 和 **openshift-samples**。
v4.12	当您要为 OpenShift Container Platform 4.12 启用默认功能时指定这个选项。通过指定 **v4.12**，不会启用较新版本的 OpenShift Container Platform 中引入的功能。OpenShift Container Platform 4.12 中的默认功能是 **baremetal**,**MachineAPI**,**marketplace**,**openshift-samples**,**Console**,**Insights**,**Storage**, 和 **CSISnapshot**。
v4.13	当要为 OpenShift Container Platform 4.13 启用默认功能时，指定这个选项。通过指定 **v4.13**，不会启用较新版本的 OpenShift Container Platform 中引入的功能。OpenShift Container Platform 4.13 中的默认功能是 **baremetal**,**MachineAPI**,**marketplace**,**openshift-samples**,**Console**,**Insights**,**Storage**,**CSISnapshot**, 和 **NodeTuning**。
v4.14	当要为 OpenShift Container Platform 4.14 启用默认功能时指定这个选项。通过指定 **v4.14**，不会启用较新版本的 OpenShift Container Platform 中引入的功能。OpenShift Container Platform 4.14 中的默认功能是 **baremetal**,**MachineAPI**,**marketplace**,**openshift-samples**,**Console**,**Insights**,**Storage**,**CSISnapshot**,**NodeTuning**,**ImageRegistry**,**Build**, 和 **DeploymentConfig**。
v4.15	当要为 OpenShift Container Platform 4.15 启用默认功能时指定这个选项。通过指定 **v4.15**，不会启用较新版本的 OpenShift Container Platform 中引入的功能。OpenShift Container Platform 4.15 中的默认功能是 **baremetal**,**MachineAPI**,**marketplace**,**OperatorLifecycleManager**,**openshift-samples**,**Console**,**Insights**,**Storage**,**CSISnapshot**,**NodeTuning**,**ImageRegistry**,**Build**,**CloudCredential**, 和 **DeploymentConfig**。
v4.16	当要为 OpenShift Container Platform 4.16 启用默认功能时指定这个选项。通过指定 **v4.16**，不会启用较新版本的 OpenShift Container Platform 中引入的功能。OpenShift Container Platform 4.16 中的默认功能是 **baremetal**, **MachineAPI**, **marketplace**, **OperatorLifecycleManager**, **openshift-samples**, **Console**, **Insights**, **Storage**, **CSISnapshot**, **NodeTuning**, **ImageRegistry**, **Build**, **CloudCredential**, **DeploymentConfig**, 和 **CloudControllerManager**。

值	描述
None	指定其他集合太大，您不需要任何功能，或者想要 **通过额外的 EnabledCapabilities** 进行微调。

其他资源

- 使用自定义在 AWS 上安装集群
- 使用自定义在 GCP 上安装集群

3.2. OPENSHIFT CONTAINER PLATFORM 4.16 中的可选集群功能

目前，集群 Operator 为这些可选功能提供功能。以下总结了每个功能提供的功能，并在禁用时丢失的功能。

其他资源

- 集群 Operator 参考

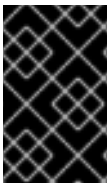
3.2.1. 裸机功能

用途
Cluster Baremetal Operator 为 **baremetal** 功能提供功能。

Cluster Baremetal Operator (CBO) 会部署使裸机服务器成为一个可完全正常工作的节点以运行 OpenShift Container Platform 计算节点所需的所有组件。CBO 确保 metal3 部署（由 Bare Metal Operator (BMO) 和 Ironic 容器组成）在 OpenShift Container Platform 集群内的一个 control plane 节点上运行。CBO 还会侦听 OpenShift Container Platform 对资源的更新，它会监视并采取适当的操作。

使用安装程序置备的基础架构部署需要裸机功能。禁用裸机功能可能会导致这些部署出现意外问题。

建议集群管理员仅在带有集群中没有任何 **BareMetalHost** 资源的用户置备的基础架构禁用裸机功能。



重要

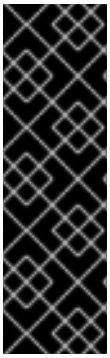
如果禁用裸机功能，集群将无法置备或管理裸机节点。只有在部署中没有 **BareMetalHost** 资源时才禁用该功能。**baremetal** 能力取决于 **MachineAPI** 功能。如果启用 **baremetal** 功能，还必须启用 **MachineAPI**。

其他资源

- 在裸机上部署安装程序置备的集群
- 准备裸机集群安装
- 裸机配置

3.2.2. 构建功能

用途
Build 功能启用 **Build** API。**Build** API 管理 **Build** 和 **BuildConfig** 对象的生命周期。



重要

如果您禁用 **Build** 功能，集群中没有提供以下资源：

- **Build** 和 **BuildConfig** 资源
- **builder** 服务帐户

仅在不需要 **Build** 或 **BuildConfig** 资源，或集群中有 **builder** 服务帐户时，才禁用 **Build** 功能。

3.2.3. 云控制器管理器功能

用途
Cloud Controller Manager Operator 为 **CloudControllerManager** 提供功能。

注意

目前，在所有平台上都不支持禁用 **CloudControllerManager** 功能。

您可以通过检查集群的安装配置 (**install-config.yaml**) 文件中的值来确定集群是否支持禁用 **CloudControllerManager** 功能。

在 **install-config.yaml** 文件中，找到 **platform** 参数。

- 如果 **platform** 参数的值是 **Baremetal** 或 **None**，您可以在集群中禁用 **CloudControllerManager** 功能。
- 如果 **platform** 参数的值是 **External**，找到 **platform.external.cloudControllerManager** 参数。如果 **platform.external.cloudControllerManager** 参数的值为 **None**，您可以在集群中禁用 **CloudControllerManager** 功能。

重要

如果这些参数包含除列出的值以外的其他值，则无法在集群中禁用 **CloudControllerManager** 功能。

注意

对于 Amazon Web Services (AWS), Google Cloud Platform (GCP), IBM Cloud®, global Microsoft Azure, Microsoft Azure Stack Hub, Nutanix, Red Hat OpenStack Platform (RHOSP), 和 VMware vSphere，这个 Operator 的状态是正式发布（GA）。

对于 Alibaba Cloud 和 IBM Power® Virtual Server，这个 Operator 是技术预览。

Cloud Controller Manager Operator 管理并更新在 OpenShift Container Platform 上部署的云控制器管理器。Operator 基于 Kubebuilder 框架和 **controller-runtime** 库。它通过 Cluster Version Operator (CVO) 安装。

它包含以下组件：

- Operator
- 云配置观察

默认情况下，Operator 通过 **metrics** 服务公开 Prometheus 指标数据。

3.2.4. 云凭证功能

用途
Cloud Credential Operator 为 **CloudCredential** 提供功能。

注意

目前，禁用 **CloudCredential** 只在裸机集群中被支持。

Cloud Credential Operator(CCO)将云供应商凭证作为 Kubernetes 自定义资源定义(CRD)进行管理。**CredentialsRequest** 自定义资源（CR）的 CCO 同步，允许 OpenShift Container Platform 组件使用集群运行所需的特定权限请求云供应商凭证。

通过在 **install-config.yaml** 文件中为 **credentialsMode** 参数设置不同的值，可将 CCO 配置为以几种不同模式操作。如果没有指定模式，或将 **credentialsMode** 参数被设置为空字符串（""）。

其他资源

- 关于 Cloud Credential Operator

3.2.5. 集群存储功能

用途
Cluster Storage Operator 为**存储**功能提供功能。

Cluster Storage Operator 设置 OpenShift Container Platform 集群范围内的存储默认设置。它确保了 OpenShift Container Platform 集群存在默认**存储类**。它还安装 Container Storage Interface (CSI) 驱动程序，使集群能够使用各种存储后端。

重要

如果禁用了集群存储功能，集群将没有默认的 **storageclass** 或任何 CSI 驱动程序。具有管理员特权的用户可以创建默认**存储类**，并在禁用集群存储功能时手动安装 CSI 驱动程序。

备注

- Operator 创建的存储类可以通过编辑其注解来实现非默认设置，但只要 Operator 运行，这个存储类就无法被删除。

3.2.6. 控制台功能

用途
Console Operator 为 **Console** 功能提供功能。

Console Operator 在集群中安装和维护 OpenShift Container Platform web 控制台。Console Operator 会被默认安装，并自动维护控制台。

其他资源

- Web 控制台概述

3.2.7. CSI 快照控制器功能

用途
Cluster CSI Snapshot Controller Operator 为 **CSISnapshot** 功能提供功能。

Cluster CSI Snapshot Controller Operator 安装和维护 CSI Snapshot Controller。CSI Snapshot Controller 负责监视 **VolumeSnapshot** CRD 对象，并管理卷快照的创建和删除生命周期。

其他资源

- CSI 卷快照

3.2.8. DeploymentConfig 功能

用途
DeploymentConfig 功能启用和管理 **DeploymentConfig** API。

重要

如果禁用 **DeploymentConfig** 功能，集群中将无法使用以下资源：

- **DeploymentConfig** 资源
- **deployer** 服务帐户

仅在不需要 **DeploymentConfig** 资源以及集群中有 **deployer** 服务帐户时才禁用 DeploymentConfig 功能。

3.2.9. Ingress 能力

用途
Ingress Operator 为 **Ingress** 提供功能。

Ingress Operator 配置并管理 OpenShift Container Platform 路由。

项目
openshift-ingress-operator

CRD

- **clusteringresses.ingress.openshift.io**
  - Scope: Namespaced
  - CR: **clusteringresses**
  - Validation: No

Configuration objects

- Cluster config
  - 类型名：**clusteringresses.ingress.openshift.io**
  - 实例名称： **default**
  - 查看命令：

```
$ oc get clusteringresses.ingress.openshift.io -n openshift-ingress-operator default -o
yaml
```

备注
Ingress Operator 在 **openshift-ingress** 项目中设置路由，并为路由创建部署：

```
$ oc get deployment -n openshift-ingress
```

Ingress Operator 使用来自 **network/cluster** 状态的 **clusterNetwork[].cidr** 来决定受管入口控制器（路由器）应该在其中操作的模式(IPv4、IPv6 或双堆栈）。例如，如果 **clusterNetwork** 只包含 v6 **cidr**，则 Ingress Controller 在只纯 IPv6 模式下运行。

在以下示例中，Ingress Operator 管理的 ingress 控制器将以 IPv4 模式运行，因为只有一个集群网络存在，网络是 IPv4 **cidr**：

```
$ oc get network/cluster -o jsonpath='{.status.clusterNetwork[*]}'
```

输出示例

```
map[cidr:10.128.0.0/14 hostPrefix:23]
```

3.2.10. Insights 功能

用途
Insights Operator 为 **Insights** 功能提供功能。

Insights Operator 收集 OpenShift Container Platform 配置数据并将其发送到红帽。数据用于生成有关集群可能暴露的潜在问题的主动分析建议。这些建议通过 console.redhat.com 上的 Insights Advisor 与集群管理员通信。

备注
Insights Operator 补充 OpenShift Container Platform Telemetry。

其他资源

- 使用 Insights Operator

3.2.11. 机器 API 功能

用途
machine-api-operator、**cluster-autoscaler-operator** 和 **cluster-control-plane-machine-set-operator** Operator 提供了与 **MachineAPI** 功能相关的功能。只有在使用用户置备的基础架构安装集群时，才能禁用此功能。

Machine API 功能负责集群中的所有机器配置和管理。如果在安装过程中禁用 Machine API 功能，则需要手动管理所有与机器相关的任务。

其他资源

- 机器管理概述
- Machine API Operator

- Cluster Autoscaler Operator
- Control Plane Machine Set Operator

3.2.12. Marketplace 功能

用途
Marketplace Operator 提供了 **marketplace** 功能。

Marketplace Operator 通过使用集群中的一组默认 Operator Lifecycle Manager (OLM) 目录简化了将非集群 Operator 引入集群的过程。安装 Marketplace Operator 时，它会创建 **openshift-marketplace** 命名空间。OLM 确保在 **openshift-marketplace** 命名空间中安装的目录源可用于集群中的所有命名空间。

如果禁用 **marketplace** 功能，Marketplace Operator 不会创建 **openshift-marketplace** 命名空间。目录源仍可在集群中配置和管理，但 OLM 依赖于 **openshift-marketplace** 命名空间，以便目录可供集群中的所有命名空间使用。有权创建带 **openshift-** 前缀的命名空间（如系统或集群管理员）的用户可以手动创建 **openshift-marketplace** 命名空间。

如果启用 **marketplace** 功能，您可以通过配置 Marketplace Operator 来启用和禁用单个目录。

其他资源

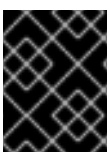
- 红帽提供的 Operator 目录

3.2.13. Operator Lifecycle Manager 功能

用途
Operator Lifecycle Manager （OLM）可帮助用户安装、更新和管理所有 Kubernetes 原生应用程序（Operator）以及在 OpenShift Container Platform 集群中运行的关联服务的生命周期。它是 Operator Framework 的一部分，后者是一个开源工具包，用于以有效、自动化且可扩展的方式管理 Operator。

如果 Operator 需要以下 API，则必须启用 **OperatorLifecycleManager** 功能：

- **ClusterServiceVersion**
- **CatalogSource**
- **订阅**
- **InstallPlan**
- **OperatorGroup**



重要

marketplace 功能取决于 **OperatorLifecycleManager** 功能。您无法禁用 **OperatorLifecycleManager** 功能并启用 **marketplace** 功能。

其他资源

- Operator Lifecycle Manager 概念和资源

3.2.14. 节点调优功能

用途

Node Tuning Operator 为 **NodeTuning** 功能提供功能。

Node Tuning Operator 可以帮助您通过编排 TuneD 守护进程来管理节点级别的性能优化，并使用 Performance Profile 控制器获得低延迟性能。大多数高性能应用程序都需要一定程度的内核级性能优化。Node Tuning Operator 为用户提供了一个统一的、节点一级的 sysctl 管理接口，并可以根据具体用户的需要灵活地添加自定义性能优化设置。

如果您禁用了 NodeTuning 功能，一些默认的性能优化设置不会应用到 control-plane 节点。这可能会限制具有 900 个节点或 900 路由的大型集群的可扩展性和性能。

其他资源

- 使用 Node Tuning Operator

3.2.15. OpenShift 示例功能

用途
Cluster Samples Operator 为 **openshift-samples** 功能提供功能。

Cluster Samples Operator 管理存储在 **openshift** 命名空间中的示例镜像流和模板。

在初始启动时，Operator 会创建默认样本配置资源来启动镜像流和模板的创建。配置对象是一个集群范围内的对象，它带有一个键 **cluster** 和类型 **configs.samples**。

镜像流是基于 Red Hat Enterprise Linux CoreOS (RHCOS) 的 OpenShift Container Platform 镜像流，指向 **registry.redhat.io** 上的镜像。同样，模板也被归类为 OpenShift Container Platform 模板。

如果您禁用示例功能，用户无法访问它提供的镜像流、示例和模板。根据您的部署，如果不需要，您可能需要禁用此组件。

其他资源

- 配置 Cluster Samples Operator

3.2.16. 集群 Image Registry 功能

用途
Cluster Image Registry Operator 为 **ImageRegistry** 功能提供功能。

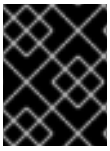
Cluster Image Registry Operator 管理 OpenShift 镜像 registry 的单个实例。它管理 registry 的所有配置，包括创建存储。

在初始启动时，Operator 会基于集群中检测到的配置创建默认的 **image-registry** 资源实例。这代表了根据云供应商要使用的云存储类型。

如果没有足够的信息来定义完整的 **image-registry** 资源，则会定义一个不完整的资源，Operator 将更新资源状态以提供缺失的内容。

Cluster Image Registry Operator在**openshift-image-registry**命名空间中运行，并管理该位置中的 registry 实例。registry的所有配置和工作负载资源都位于该命名空间中。

要将镜像 registry 集成到集群的用户身份验证和授权系统中，会为集群中的每个服务帐户生成一个镜像 pull secret。



重要

如果您禁用 **ImageRegistry** 功能，或者在 Cluster Image Registry Operator 配置中禁用集成的 OpenShift 镜像 registry，则不会为每个服务帐户生成镜像 pull secret。

如果禁用 **ImageRegistry** 功能，您可以在 Telco 环境中减少 OpenShift Container Platform 的整体资源占用空间。根据您的部署，如果需要，可以禁用此组件。

项目
cluster-image-registry-operator

其他资源

- OpenShift Container Platform中的Image Registry Operator
- 自动生成的 secret

3.3. 其他资源

- 安装后启用集群功能

第 4 章 断开连接的安装镜像

4.1. 关于断开连接的安装镜像

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

4.1.1. 创建镜像 registry

如果您已经有一个容器镜像 registry，如 Red Hat Quay，您可以使用它作为您的镜像 registry。如果您还没有 registry，可以使用 *mirror registry for Red Hat OpenShift* 创建镜像 registry。

4.1.2. 为断开连接的安装 mirror 镜像

您可以使用以下流程之一将 OpenShift Container Platform 镜像存储库镜像到您的镜像 registry：

- 为断开连接的安装 mirror 镜像
- 使用 oc-mirror 插件为断开连接的安装镜像镜像

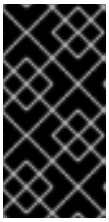
4.2. 为 RED HAT OPENSHIFT 创建带有 MIRROR REGISTRY 的 MIRROR REGISTRY

mirror registry for Red Hat OpenShift 是一个小型灵活的容器 registry，作为目标，用于为断开连接的安装镜像(mirror)的 OpenShift Container Platform 所需的容器镜像。

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

4.2.1. 先决条件

- OpenShift Container Platform 订阅。
- 安装了 Podman 3.4.2 或更高版本以及 OpenSSL 的 Red Hat Enterprise Linux (RHEL) 8 和 9。
- Red Hat Quay 服务的完全限定域名，它必须通过 DNS 服务器解析。
- 目标主机上的基于密钥的 SSH 连接。为本地安装自动生成 SSH 密钥。对于远程主机，您必须生成自己的 SSH 密钥。
- 2 个或更多 vCPU。
- 8 GB RAM。
- OpenShift Container Platform 4.16 发行镜像大约需要 12 GB；OpenShift Container Platform 4.16 发行镜像和 OpenShift Container Platform 4.16 Red Hat Operator 镜像大约需要 358 GB。每个流推荐具有 1 TB 或更多空间。



重要

这些要求基于本地测试结果，且只测试了发行镜像和 Operator 镜像。存储要求可能会因您的组织的需求而有所不同。例如，当镜像了多个 z-streams 时，则可能需要更多空间。您可以使用标准 Red Hat Quay 功能 或适当的 API 调用来删除不必要的镜像并释放空间。

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

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

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

mirror registry for Red Hat OpenShift 提供了一个预先确定的网络配置，并在成功时报告部署的组件凭证并访问 URL。另外还提供了一组有限的可选配置输入，如完全限定域名(FQDN)服务、超级用户名称和密码，以及自定义 TLS 证书。这为用户提供了一个容器 registry，以便在受限网络环境中运行 OpenShift Container Platform 时，轻松创建所有 OpenShift Container Platform 发行版本内容的离线镜像。

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

4.2.2.1. Mirror registry for Red Hat OpenShift 限制

以下限制适用于 *mirror registry for Red Hat OpenShift* ：

- *mirror registry for Red Hat OpenShift* 并不是一个高度可用的 registry，且只支持本地文件系统存储。它并不适用于 OpenShift Container Platform 替换 Red Hat Quay 或内部镜像 registry。
- *mirror registry for Red Hat OpenShift* 只支持托管安装断开连接的 OpenShift Container Platform 集群（如发行镜像或 Red Hat Operator 镜像）所需的镜像。它使用 Red Hat Enterprise Linux(RHEL)机器上的本地存储，而 RHEL 支持的存储也被 *mirror registry for Red Hat OpenShift* 支持。

注意

因为 *mirror registry for Red Hat OpenShift* 使用本地存储，所以您应该了解镜像镜像时消耗的存储使用量，并使用 Red Hat Quay 的垃圾回收功能来缓解潜在的问题。有关此功能的更多信息，请参阅"Red Hat Quay 垃圾回收"。

- 对推送到 *mirror registry for* bootstrap 的红帽产品镜像的支持会被每个相应产品的有效订阅涵盖。进一步启用 bootstrap 体验的例外列表可在 自助管理的 Red Hat OpenShift 大小和订阅指南 中找到。
- 由客户创建的内容不应由 *mirror registry for Red Hat OpenShift* 托管。
- 不建议将 *mirror registry for Red Hat OpenShift* 与多个集群一起使用，因为多个集群可以在更新集群时造成单点故障。建议利用 *mirror registry for Red Hat OpenShift* 安装一个集群，通过这个集群托管一个生产环境级别的、具有高可用性的 registry（如 Red Hat Quay）的集群，用于为其他集群提供 OpenShift Container Platform 内容。

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

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

注意

使用 **mirror-registry** CLI 工具安装 *mirror registry for Red Hat OpenShift* 对您的系统会有一些变化。安装后，会创建一个 **$HOME/quay-install** 目录，其中包含安装文件、本地存储和配置捆绑包。如果部署目标是本地主机，则生成可信 SSH 密钥，并且设置主机计算机上的 systemd 文件，以确保容器运行时持久。另外，会创建一个名为 **init** 的初始用户，并自动生成的密码。所有访问凭证都会在安装例程的末尾打印。

流程

1. 从 OpenShift console **Downloads** 页下载最新版本的 *mirror registry for Red Hat OpenShift* 的 **mirror-registry.tar.gz** 软件包。
2. 使用 **mirror-registry** 工具，在本地主机上安装 *mirror registry for Red Hat OpenShift*。有关可用标志的完整列表，请参阅 "mirror registry for Red Hat OpenShift flags"。

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

3. 运行以下命令，使用安装期间生成的用户名和密码登录 registry：

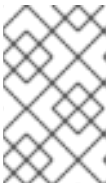
```
$ podman login -u init \
  -p <password> \
  <host_example_com>:8443> \
  --tls-verify=false 1
```

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

注意

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

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



注意

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

4.2.4. 从一个本地主机为 Red Hat OpenShift 更新 mirror registry

此流程解释了如何使用 **upgrade** 命令从本地主机*更新 Red Hat OpenShift 的镜像 registry*。更新至最新版本可确保新的功能、错误修复和安全漏洞修复。

重要

更新时，镜像 registry 会发生间歇性停机时间，因为它在更新过程中重启。

先决条件

- 您已在本地主机上安装了 *Red Hat OpenShift 的镜像 registry*。

流程

- 如果要将 *mirror registry for Red Hat OpenShift* 从 1.2.z 升级到 1.3.0，您的安装目录默认为 **/etc/quay-install**，您可以输入以下命令：

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

注意

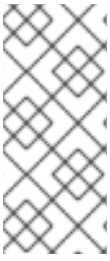
  - *mirror registry for Red Hat OpenShift* 将 Quay 的 Podman 卷、Postgres 数据和 **/etc/quay-install** 数据迁移到新的 **$HOME/quay-install** 位置。这可让让您在以后的升级过程中，在没有 **--quayRoot** 标志的情况下，使用 *mirror registry for Red Hat OpenShift*。
  - 在使用 **./mirror-registry upgrade -v** 标记升级 *mirror registry for Red Hat OpenShift* 时需要包括在创建 mirror registry 时使用的相同的凭证。例如，如果使用 **--quayHostname <host_example_com>** 和 **--quayRoot <example_directory_name>** 安装 *Red Hat OpenShift 镜像 registry*，则必须包括该字符串来正确地升级镜像 registry。

- 如果您要将 *mirror registry for Red Hat OpenShift* 从 1.2.z 升级到 1.3.0，且您在 1.2.z 部署中使用了指定目录，则必须传递新的 **--pgStorage** 和 **--quayStorage** 标志。例如：

```
$ sudo ./mirror-registry upgrade --quayHostname <host_example_com> --quayRoot
<example_directory_name> --pgStorage <example_directory_name>/pg-data --quayStorage
<example_directory_name>/quay-storage -v
```

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

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



注意

使用 **mirror-registry** CLI 工具安装 *mirror registry for Red Hat OpenShift* 对您的系统会有一些变化。安装后，会创建一个 **$HOME/quay-install** 目录，其中包含安装文件、本地存储和配置捆绑包。如果部署目标是本地主机，则生成可信 SSH 密钥，并且设置主机计算机上的 systemd 文件，以确保容器运行时持久。另外，会创建一个名为 **init** 的初始用户，并自动生成的密码。所有访问凭证都会在安装例程的末尾打印。

流程

1. 从 OpenShift console **Downloads** 页下载最新版本的 *mirror registry for Red Hat OpenShift* 的 **mirror-registry.tar.gz** 软件包。

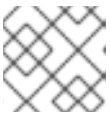
2. 使用 **mirror-registry** 工具，在本地主机上安装 *mirror registry for Red Hat OpenShift*。有关可用标志的完整列表，请参阅 "mirror registry for Red Hat OpenShift flags"。

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

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

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

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



注意

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

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

注意

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

4.2.6. 从一个远程主机为 Red Hat OpenShift 更新 mirror registry

此流程解释了如何使用 **upgrade** 命令从远程主机*更新 Red Hat OpenShift 的镜像 registry*。更新至最新版本可确保程序错误修复和安全漏洞。

重要

更新时，镜像 registry 会发生间歇性停机时间，因为它在更新过程中重启。

先决条件

- 您已在远程主机上安装了 *Red Hat OpenShift 的镜像 registry*。

流程

- 要从远程主机升级 *Red Hat OpenShift 的镜像 registry*，请输入以下命令：

```
$ ./mirror-registry upgrade -v --targetHostname <remote_host_url> --targetUsername
<user_name> -k ~/.ssh/my_ssh_key
```

注意

在使用 **./mirror-registry upgrade -v** 标记升级 *mirror registry for Red Hat OpenShift* 时需要包括在创建 mirror registry 时使用的相同的凭证。例如，如果使用 **--quayHostname <host_example_com>** 和 **--quayRoot <example_directory_name>** 安装 *Red Hat OpenShift 镜像 registry*，则必须包括该字符串来正确地升级镜像 registry。

4.2.7. 替换 mirror registry for Red Hat OpenShift SSL/TLS 证书

在某些情况下，您可能想要为 *mirror registry for Red Hat OpenShift* 更新 SSL/TLS 证书。这在以下情况中很有用：

- 如果您需要替换当前的 *mirror registry for Red Hat OpenShift* 证书。
- 如果您使用与之前 *mirror registry for Red Hat OpenShift* 安装相同的证书。
- 如果您希望定期更新 *mirror registry for Red Hat OpenShift* 证书。

使用以下步骤替换 *mirror registry for Red Hat OpenShift* SSL/TLS 证书。

先决条件

- 您已从 OpenShift 控制台 **Downloads** 页面下载 **./mirror-registry** 二进制文件。

流程

1. 输入以下命令安装 *mirror registry for Red Hat OpenShift*：

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

   这会将 *mirror registry for Red Hat OpenShift* 安装到 **$HOME/quay-install** 目录中。

2. 准备一个新的证书颁发机构(CA)捆绑包，并生成新的 **ssl.key** 和 **ssl.crt** 密钥文件。如需更多信息，请参阅使用 SSL/TLS 保护到 Red Hat Quay 的连接 。

3. 输入以下命令为 **/$HOME/quay-install** 分配一个环境变量，如 **QUAY** ：

```
$ export QUAY=/$HOME/quay-install
```

4. 输入以下命令将新的 **ssl.crt** 文件复制到 **/$HOME/quay-install** 目录中：

```
$ cp ~/ssl.crt $QUAY/quay-config
```

5. 输入以下命令将新的 **ssl.key** 文件复制到 **/$HOME/quay-install** 目录中：

```
$ cp ~/ssl.key $QUAY/quay-config
```

6. 输入以下命令重启 **quay-app** 应用程序 pod：

```
$ systemctl restart quay-app
```

4.2.8. 为 Red Hat OpenShift 卸载镜像 registry

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

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

注意

- 删除 *mirror registry for Red Hat OpenShift* 会在删除前提示用户。您可以使用 **--autoApprove** 来跳过此提示。
- 如果使用 **--quayRoot** 标志安装了 *mirror registry for Red Hat OpenShift*，则卸载时也需要使用 **--quayRoot** 标志。例如，如果使用 **--quayRoot example_directory_name** 安装了 *mirror registry for Red Hat OpenShift*，则在卸载镜像 registry 时也需要包括这个字符串才能正确卸载。

4.2.9. Mirror registry for Red Hat OpenShift 标记

以下标记可用于 *mirror registry for Red Hat OpenShift* ：

标记	描述
--autoApprove	禁用交互式提示的布尔值。如果设置为 **true**，则在卸载镜像 registry 时自动删除 **quayRoot** 目录。如果未指定，则默认为 **false**。
--initPassword	在 Quay 安装过程中创建的 init 用户的密码。必须至少包含八个字符，且不包含空格。
--initUser string	显示初始用户的用户名。若未指定，则默认为 **init**。
--no-color, **-c**	允许用户禁用颜色序列，在运行安装、卸载和升级命令时将其传播到 Ansible。
--pgStorage	保存 Postgres 持久性存储数据的文件夹。默认为 **pg-storage** Podman 卷。卸载需要 root 权限。
--quayHostname	客户端用来联系 registry 的镜像 registry 的完全限定域名。等同于 Quay **config.yaml** 中的 **SERVER_HOSTNAME**。必须可以被 DNS 解析。如果未指定，则默认为 **<targetHostname>:8443**。[1]
--quayStorage	保存 Quay 持久性存储数据的文件夹。默认为 **quay-storage** Podman 卷。卸载需要 root 权限。
--quayRoot, **-r**	保存容器镜像层和配置数据的目录，包括 **rootCA.key**、rootCA**.pem** 和 **rootCA.srl** 证书。如果未指定，则默认为 **$HOME/quay-install**。

标记	描述
--ssh-key,-k	SSH 身份密钥的路径。如果未指定，则默认为 **~/.ssh/quay_installer**。
--sslCert	SSL/TLS 公钥/证书的路径。默认为 **{quayRoot}/quay-config**，并在未指定时自动生成。
--sslCheckSkip	跳过对 **config.yaml** 文件中的 **SERVER_HOSTNAME** 的检查证书主机名。[2]
--sslKey	用于 HTTPS 通信的 SSL/TLS 私钥路径。默认为 **{quayRoot}/quay-config**，并在未指定时自动生成。
--targetHostname, -H	要安装 Quay 的目标的主机名。默认为 **$HOST**，如本地主机（如果未指定）。
--targetUsername, -u	目标主机上的用户，将用于 SSH。默认为 **$USER**，例如，如果未指定，则默认为当前用户。
--verbose,-v	显示调试日志和 Ansible playbook 输出。
--version	显示 *mirror registry for Red Hat OpenShift*的版本。

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

4.2.10. Mirror registry for Red Hat OpenShift 发现注记

mirror registry for Red Hat OpenShift 是一个小型灵活的容器 registry，作为目标，用于为断开连接的安装镜像(mirror)的 OpenShift Container Platform 所需的容器镜像。

本发行注记介绍了 OpenShift Container Platform 的 *mirror registry for Red Hat OpenShift*。

如需了解 *mirror registry for Red Hat OpenShift*，请参阅 Creating a mirror registry with mirror registry for Red Hat OpenShift。

4.2.10.1. Mirror registry for Red Hat OpenShift 1.3.11

发布日期： 2024 年 4 月 23 日

Mirror registry for Red Hat OpenShift 现在包括在 Red Hat Quay 3.8.15 中。

以下公告适用于 *mirror registry for Red Hat OpenShift* ：

- RHBA-2024:1758 - mirror registry for Red Hat OpenShift 1.3.11

4.2.10.2. Mirror registry for Red Hat OpenShift 1.3.10

发布日期：2023 年 12 月 7 日

Mirror registry for Red Hat OpenShift 现在包括在 Red Hat Quay 3.8.14 中。

以下公告适用于 *mirror registry for Red Hat OpenShift* ：

- RHBA-2023:7628 - mirror registry for Red Hat OpenShift 1.3.10

4.2.10.3. Mirror registry for Red Hat OpenShift 1.3.9

发布日期：2023 年 9 月 19 日

Mirror registry for Red Hat OpenShift 现在包括在 Red Hat Quay 3.8.12 中。

以下公告适用于 *mirror registry for Red Hat OpenShift* ：

- RHBA-2023:5241 - mirror registry for Red Hat OpenShift 1.3.9

4.2.10.4. Mirror registry for Red Hat OpenShift 1.3.8

发布日期：2023 年 8 月 16 日

Mirror registry for Red Hat OpenShift 现在包括在 Red Hat Quay 3.8.11 中。

以下公告适用于 *mirror registry for Red Hat OpenShift* ：

- RHBA-2023:4622 - mirror registry for Red Hat OpenShift 1.3.8

4.2.10.5. Mirror registry for Red Hat OpenShift 1.3.7

发布日期：2023 年 7 月 19 日

Mirror registry for Red Hat OpenShift 现在包括在 Red Hat Quay 3.8.10 中。

以下公告适用于 *mirror registry for Red Hat OpenShift* ：

- RHBA-2023:4087 - mirror registry for Red Hat OpenShift 1.3.7

4.2.10.6. Mirror registry for Red Hat OpenShift 1.3.6

发布日期：2023 年 5 月 30 日

Mirror registry for Red Hat OpenShift 现在包括在 Red Hat Quay 3.8.8 中。

以下公告适用于 *mirror registry for Red Hat OpenShift* ：

- RHBA-2023:3302 - mirror registry for Red Hat OpenShift 1.3.6

4.2.10.7. Mirror registry for Red Hat OpenShift 1.3.5

发布日期：2023 年 5 月 18 日

Mirror registry for Red Hat OpenShift 现在包括在 Red Hat Quay 3.8.7 中。

以下公告适用于 *mirror registry for Red Hat OpenShift* ：

- RHBA-2023:3225 - mirror registry for Red Hat OpenShift 1.3.5

4.2.10.8. Mirror registry for Red Hat OpenShift 1.3.4

发布日期：2023 年 4 月 25 日

Mirror registry for Red Hat OpenShift 现在包括在 Red Hat Quay 3.8.6 中。

以下公告适用于 *mirror registry for Red Hat OpenShift* ：

- RHBA-2023:1914 - mirror registry for Red Hat OpenShift 1.3.4

4.2.10.9. Mirror registry for Red Hat OpenShift 1.3.3

发布日期：2023 年 4 月 5 日

Mirror registry for Red Hat OpenShift 现在包括在 Red Hat Quay 3.8.5 中。

以下公告适用于 *mirror registry for Red Hat OpenShift* ：

- RHBA-2023:1528 - mirror registry for Red Hat OpenShift 1.3.3

4.2.10.10. Mirror registry for Red Hat OpenShift 1.3.2

发布日期：2023 年 3 月 21 日

Mirror registry for Red Hat OpenShift 现在包括在 Red Hat Quay 3.8.4 中。

以下公告适用于 *mirror registry for Red Hat OpenShift* ：

- RHBA-2023:1376 - mirror registry for Red Hat OpenShift 1.3.2

4.2.10.11. Mirror registry for Red Hat OpenShift 1.3.1

发布日期：2023 年 3 月 7 日

Mirror registry for Red Hat OpenShift 现在包括在 Red Hat Quay 3.8.3 中。

以下公告适用于 *mirror registry for Red Hat OpenShift* ：

- RHBA-2023:1086 - mirror registry for Red Hat OpenShift 1.3.1

4.2.10.12. Mirror registry for Red Hat OpenShift 1.3.0

发布日期：2023 年 2 月 20 日

Mirror registry for Red Hat OpenShift 现在包括在 Red Hat Quay 3.8.1 中。

以下公告适用于 *mirror registry for Red Hat OpenShift* ：

- RHBA-2023:0558 - mirror registry for Red Hat OpenShift 1.3.0

4.2.10.12.1. 新功能

- *Mirror registry for Red Hat OpenShift* 现在支持在 Red Hat Enterprise Linux (RHEL) 9 安装中。
- 现在，*mirror registry for Red Hat OpenShift* 本地主机安装提供了 IPv6 支持。
  目前， *mirror registry for Red Hat OpenShift* 远程主机安装不支持 IPv6。

- 添加了新功能标志 **--quayStorage**。通过指定此标志，您可以手动设置 Quay 持久性存储的位置。
- 添加了新功能标志 **--pgStorage**。通过指定此标志，您可以手动设置 Postgres 持久性存储的位置。
- 在以前的版本中，用户需要具有 root 权限 (**sudo**) 才能安装 *mirror registry for Red Hat OpenShift*。在这个版本中，安装 *mirror registry for Red Hat OpenShift* 不再需要 **sudo**。当使用 **sudo** 安装 *mirror registry for Red Hat OpenShift* 时，会创建一个包含安装文件、本地存储和配置捆绑包的 **/etc/quay-install** 目录。移除对 **sudo** 的要求后，安装文件和配置捆绑包现在安装到 **$HOME/quay-install** 中。本地存储（如 Postgres 和 Quay）现在由 Podman 自动创建的命名卷中。

  要覆盖存储这些文件的默认目录，您可以为 *mirror registry for Red Hat OpenShift* 使用命令行参数。如需有关 *mirror registry for Red Hat OpenShift* 命令行参数的更多信息，请参阅"*Mirror registry for Red Hat OpenShift* 标志"。

4.2.10.12.2. 程序错误修复

- 在以前的版本中，当试图卸载 *mirror registry for Red Hat OpenShift* 时会返回以下错误：**["Error: no container with name or ID \"quay-postgres\" found: no such container"], "stdout": "", "stdout_lines": []***。在这个版本中，*mirror registry for Red Hat OpenShift* 服务的停止和卸载的顺序有所变化，因此在卸载 *mirror registry for Red Hat OpenShift* 时不再发生错误。如需更多信息，请参阅 PROJQUAY-4629。

4.2.10.13. Mirror registry for Red Hat OpenShift 1.2.9

Mirror registry for Red Hat OpenShift 现在包括在 Red Hat Quay 3.7.10 中。

以下公告适用于 *mirror registry for Red Hat OpenShift* ：

- RHBA-2022:7369 - mirror registry for Red Hat OpenShift 1.2.9

4.2.10.14. Mirror registry for Red Hat OpenShift 1.2.8

Mirror registry for Red Hat OpenShift 现在包括在 Red Hat Quay 3.7.9 中。

以下公告适用于 *mirror registry for Red Hat OpenShift* ：

- RHBA-2022:7065 - mirror registry for Red Hat OpenShift 1.2.8

4.2.10.15. Mirror registry for Red Hat OpenShift 1.2.7

限制，Red Hat Quay 3.7.8 提供了 *Mirror registry for Red Hat OpenShift*。

以下公告适用于 *mirror registry for Red Hat OpenShift* ：

- RHBA-2022:6500 - mirror registry for Red Hat OpenShift 1.2.7

4.2.10.15.1. 程序错误修复

- 在以前的版本中，**getFQDN()** 依赖于完全限定域名 (FQDN) 库来确定其 FQDN，FQDN 库会尝试直接读取 **/etc/hosts** 文件夹。因此，在一些带有不常见的 DNS 配置的 Red Hat Enterprise Linux CoreOS (RHCOS) 安装中，FQDN 库将无法安装并导致安装过程被终止。在这个版本

中，*mirror registry for Red Hat OpenShift* 使用 **hostname** 来决定 FQDN。因此，FQDN 库不会导致安装被中止。(PROJQUAY-4139)

4.2.10.16. Mirror registry for Red Hat OpenShift 1.2.6

Mirror registry for Red Hat OpenShift 现在包括在 Red Hat Quay 3.7.7 中。

以下公告适用于 *mirror registry for Red Hat OpenShift* ：

- RHBA-2022:6278 - mirror registry for Red Hat OpenShift 1.2.6

4.2.10.16.1. 新功能

添加了新功能标志 **--no-color** (**-c**)。通过此功能标志，用户可以禁用颜色序列，在运行安装、卸载和升级命令时将其传播到 Ansible。

4.2.10.17. Mirror registry for Red Hat OpenShift 1.2.5

Mirror registry for Red Hat OpenShift 现在包括在 Red Hat Quay 3.7.6 中。

以下公告适用于 *mirror registry for Red Hat OpenShift* ：

- RHBA-2022:6071 - mirror registry for Red Hat OpenShift 1.2.5

4.2.10.18. Mirror registry for Red Hat OpenShift 1.2.4

Mirror registry for Red Hat OpenShift 现在包括在 Red Hat Quay 3.7.5 中。

以下公告适用于 *mirror registry for Red Hat OpenShift* ：

- RHBA-2022:5884 - mirror registry for Red Hat OpenShift 1.2.4

4.2.10.19. Mirror registry for Red Hat OpenShift 1.2.3

Mirror registry for Red Hat OpenShift 现在包括在 Red Hat Quay 3.7.4 中。

以下公告适用于 *mirror registry for Red Hat OpenShift* ：

- RHBA-2022:5649 - mirror registry for Red Hat OpenShift 1.2.3

4.2.10.20. Mirror registry for Red Hat OpenShift 1.2.2

Mirror registry for Red Hat OpenShift 现在包括在 Red Hat Quay 3.7.3 中。

以下公告适用于 *mirror registry for Red Hat OpenShift* ：

- RHBA-2022:5501 - mirror registry for Red Hat OpenShift 1.2.2

4.2.10.21. Mirror registry for Red Hat OpenShift 1.2.1

Mirror registry for Red Hat OpenShift 现在包括在 Red Hat Quay 3.7.2 中。

以下公告适用于 *mirror registry for Red Hat OpenShift* ：

- RHBA-2022:4986 - mirror registry for Red Hat OpenShift 1.2.1

4.2.10.22. Mirror registry for Red Hat OpenShift 1.2.0

Mirror registry for Red Hat OpenShift 现在包括在 Red Hat Quay 3.7.1 中。

以下公告适用于 *mirror registry for Red Hat OpenShift* ：

- RHBA-2022:4986 - mirror registry for Red Hat OpenShift 1.2.0

4.2.10.22.1. 程序错误修复

- 在以前的版本中，在 Quay pod Operator 内运行的所有组件和 worker 都被设置为 **DEBUG**。因此，在创建大量流量日志时使用了不必要的空间。在这个版本中，日志级别被设置为 **WARN**，这可减少流量信息，同时突出了问题的情况。(PROJQUAY-3504)

4.2.10.23. Mirror registry for Red Hat OpenShift 1.1.0

以下公告适用于 *mirror registry for Red Hat OpenShift* ：

- RHBA-2022:0956 - mirror registry for Red Hat OpenShift 1.1.0

4.2.10.23.1. 新功能

- 添加了一个新的命令 **mirror-registry upgrade**。此命令升级所有容器镜像，而不会干扰配置或数据。

注意

如果 **quayRoot** 之前设为默认值以外的内容，则必须将其传递到 upgrade 命令。

4.2.10.23.2. 程序错误修复

- 在以前的版本中，没有 **quayHostname** 或 **targetHostname** 默认为本地主机名。在这个版本中，如果缺少 **quayHostname** 和 **targetHostname**，则将其默认为本地主机名。(PROJQUAY-3079)
- 在以前的版本中，命令 **./mirror-registry --version** 返回一个 **unknown flag** 错误。现在，运行 **./mirror-registry --version** 会返回 *mirror registry for Red Hat OpenShift* 的当前版本。(PROJQUAY-3086)
- 在以前的版本中，用户无法在安装过程中设置密码，例如在运行 **./mirror-registry install --initUser <user_name> --initPassword <password> --verbose** 时。在这个版本中，用户可以在安装过程中设置密码。(PROJQUAY-3149)
- 在以前的版本中，如果已销毁，则 *mirror registry for Red Hat OpenShift* 不会重新创建 pod。现在，如果 pod 被销毁，则会重新创建 pod。(PROJQUAY-3261)

4.2.11. Mirror registry for Red Hat OpenShift 故障排除

为了更好地对 *mirror registry for Red Hat OpenShift* 进行故障排除，您可以收集由镜像 registry 安装的 systemd 服务的日志。安装以下服务：

- quay-app.service
- quay-postgres.service

- quay-redis.service
- quay-pod.service

先决条件

- 您已安装了 *mirror registry for Red Hat OpenShift*。

流程

- 如果使用 root 权限安装 *mirror registry for Red Hat OpenShift*，您可以输入以下命令获取其 systemd 服务的状态信息：

```
$ sudo systemctl status <service>
```

- 如果作为标准用户安装 *mirror registry for Red Hat OpenShift*，您可以输入以下命令获取其 systemd 服务的状态信息：

```
$ systemctl --user status <service>
```

4.2.12. 其他资源

- Red Hat Quay 垃圾回收
- 使用 SSL 保护到 Red Hat Quay 的连接
- 将系统配置为信任证书颁发机构
- 镜像 OpenShift Container Platform 镜像存储库
- 镜像用于断开连接的集群的 Operator 目录

4.3. 为断开连接的安装 MIRROR 镜像

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

重要

您必须可以访问互联网来获取所需的容器镜像。在这一流程中，您要将镜像 registry 放在可访问您的网络以及互联网的镜像（mirror）主机上。如果您没有镜像主机的访问权限，请使用镜像 Operator 目录与断开连接的集群流程一起使用，将镜像复制到可跨网络界限的设备。

4.3.1. 先决条件

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

- Sonatype Nexus 仓库
- Harbor

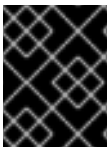
如果您有 Red Hat Quay 权利，请参阅有关部署 Red Hat Quay 以了解概念验证的文档，或使用 Red Hat Quay Operator。如果您需要额外的帮助来选择并安装 registry，请联络您的销售代表或红帽支持。

- 如果您还没有容器镜像 registry，OpenShift Container Platform 可以为订阅者提供一个 mirror registry for Red Hat OpenShift。*Red Hat OpenShift 的镜像 registry* 包含在您的订阅中，它是一个小型容器 registry，可用于在断开连接的安装中镜像 OpenShift Container Platform 所需的容器镜像。

4.3.2. 关于镜像 registry

您可以镜像 OpenShift Container Platform 安装所需的镜像，以及容器镜像 registry 的后续产品更新，如 Red Hat Quay、JFrog Artifactory、Sonatype Nexus Repository 或 Harbor。如果您无法访问大型容器 registry，可以使用 *mirror registry for Red Hat OpenShift*，它是包括在 OpenShift Container Platform 订阅中的一个小型容器 registry。

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



重要

OpenShift 镜像 registry 不能用作目标 registry，因为它不支持没有标签的推送，在镜像过程中需要这个推送。

如果选择的容器 registry 不是 *mirror registry for Red Hat OpenShift*，则需要集群中置备的每台机器都可以访问它。如果 registry 无法访问，安装、更新或常规操作（如工作负载重新定位）可能会失败。因此，您必须以高度可用的方式运行镜像 registry，镜像 registry 至少必须与 OpenShift Container Platform 集群的生产环境可用性相匹配。

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

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

注意

红帽没有针对 OpenShift Container Platform 测试第三方 registry。

附加信息

有关查看 CRI-O 日志以查看镜像源的详情，请参阅查看镜像拉取源。

4.3.3. 准备您的镜像主机

执行镜像步骤前，必须准备主机以检索内容并将其推送到远程位置。

4.3.3.1. 安装 OpenShift CLI

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

重要

如果安装了旧版本的 **oc**，则可能无法使用 OpenShift Container Platform 4.16 中的所有命令。下载并安装新版本的 **oc**。

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

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **产品变体** 下拉列表中选择架构。
3. 从 **版本** 下拉列表中选择适当的版本。
4. 点 **OpenShift v4.16 Linux Client** 条目旁的 **Download Now** 来保存文件。
5. 解包存档：

```
$ tar xvf <file>
```

6. 将 **oc** 二进制文件放到 **PATH 中的目录中**。
   要查看您的 **PATH**，请执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

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

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 Windows Client** 条目旁的 **Download Now** 来保存文件。
4. 使用 ZIP 程序解压存档。
5. 将 **oc** 二进制文件移到 **PATH 中的目录中**。
   要查看您的 **PATH**，请打开命令提示并执行以下命令：

```
C:\> path
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

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

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

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 macOS Client** 条目旁的 **Download Now** 来保存文件。

注意

对于 macOS arm64，请选择 **OpenShift v4.16 macOS arm64 Client** 条目。

4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
要查看您的 **PATH**，请打开终端并执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

4.3.4. 配置允许对容器镜像进行镜像的凭证

创建容器镜像 registry 凭证文件，可让您将镜像从红帽 mirror 到您的镜像。

警告

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

警告

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

先决条件

- 您已将镜像 registry 配置为在断开连接的环境中使用。
- 您在镜像 registry 中标识了镜像仓库的位置，以将容器镜像镜像(mirror)到这个位置。
- 您置备了一个镜像 registry 帐户，允许将镜像上传到该镜像仓库。

流程

在安装主机上完成以下步骤：

1. 从 Red Hat OpenShift Cluster Manager 下载 **registry.redhat.io** pull secret。
2. 以 JSON 格式创建您的 pull secret 副本：

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

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

该文件类似于以下示例：

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

1. 为您的镜像 registry 生成 base64 编码的用户名和密码或令牌：

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

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

2. 编辑 JSON 文件并添加描述 registry 的部分：

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

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

(2) 指定镜像 registry 的 base64 编码用户名和密码。

该文件类似于以下示例：

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

4.3.5. 镜像 OpenShift Container Platform 镜像存储库

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

先决条件

- 您的镜像主机可访问互联网。
- 您已将镜像 registry 配置为在受限网络中使用，并可访问您配置的证书和凭证。
- 您已从 Red Hat OpenShift Cluster Manager 下载了 pull secret，并已修改为包含镜像存储库的身份验证。
- 如果您使用自签名证书，已在证书中指定 Subject Alternative Name。

流程

在镜像主机上完成以下步骤：

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

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

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

   b. 导出本地 registry 名称和主机端口：

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

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

   c. 导出本地存储库名称：

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

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

   d. 导出要进行镜像的存储库名称：

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

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

   e. 导出 registry pull secret 的路径：

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

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

   f. 导出发行版本镜像：

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

对于生产环境版本，您必须指定 **ocp-release**。

g. 为您的集群导出构架类型：

```
$ ARCHITECTURE=<cluster_architecture> 1
```

1 指定集群的构架，如 **x86_64**, **aarch64**, **s390x**, 获 **ppc64le**。

h. 导出托管镜像的目录的路径：

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

1 指定完整路径，包括开始的前斜杠(/)字符。

3. 将版本镜像(mirror)到镜像 registry：

- 如果您的镜像主机无法访问互联网，请执行以下操作：

i. 将可移动介质连接到连接到互联网的系统。

ii. 查看要镜像的镜像和配置清单：

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

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

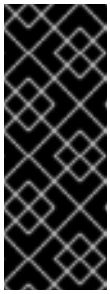
iv. 将镜像镜像到可移动介质的目录中：

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

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

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

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



重要

运行 **oc image mirror** 可能会导致以下错误： **error: unable to retrieve source image**。当镜像索引包括对镜像 registry 中不再存在的镜像的引用时，会发生此错误。镜像索引可能会保留旧的引用，以便为运行这些镜像的用户在升级图表中显示新的升级路径。作为临时解决方案，您可以使用 **--skip-missing** 选项绕过错误并继续下载镜像索引。如需更多信息，请参阅 Service Mesh Operator 镜像失败。

- 如果本地容器 registry 连接到镜像主机，请执行以下操作：

  i. 使用以下命令直接将发行版镜像推送到本地 registry:

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

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

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

注意

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

4. 要创建基于您镜像内容的安装程序，请提取内容并将其固定到发行版中：

- 如果您的镜像主机无法访问互联网，请运行以下命令：

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

- 如果本地容器 registry 连接到镜像主机，请运行以下命令：

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

重要

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

您必须在有活跃互联网连接的机器上执行这个步骤。

5. 对于使用安装程序置备的基础架构的集群，运行以下命令：

```
$ openshift-install
```

4.3.6. 在断开连接的环境中的 Cluster Samples Operator

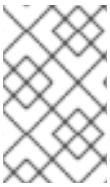
在断开连接的环境中，在安装集群后执行额外的步骤来配置 Cluster Samples Operator。在准备过程中查阅以下信息：

4.3.6.1. 协助镜像的 Cluster Samples Operator

在安装过程中，OpenShift Container Platform 在 **openshift-cluster-samples-operator** 命名空间中创建一个名为 **imagestreamtag-to-image** 的配置映射。**imagestreamtag-to-image** 配置映射包含每个镜像流标签的条目（填充镜像）。

配置映射中 data 字段中每个条目的键格式为 **<image_stream_name>_<image_stream_tag_name>**。

在断开连接的 OpenShift Container Platform 安装过程中，Cluster Samples Operator 的状态被设置为 **Removed**。如果您将其改为 **Managed**，它会安装示例。



注意

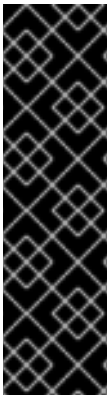
在网络限制或断开连接的环境中使用示例可能需要通过网络访问服务。某些示例服务包括：Github、Maven Central、npm、RubyGems、PyPi 等。这可能需要执行额外的步骤，让集群 samples operator 对象能够访问它们所需的服务。

您可以使用此配置映射作为导入镜像流所需的镜像的引用。

- 在 Cluster Samples Operator 被设置为 **Removed** 时，您可以创建镜像的 registry，或决定您要使用哪些现有镜像 registry。
- 使用新的配置映射作为指南来镜像您要镜像的 registry 的示例。
- 将没有镜像的任何镜像流添加到 Cluster Samples Operator 配置对象的 **skippedImagestreams** 列表中。
- 将 Cluster Samples Operator 配置对象的 **samplesRegistry** 设置为已镜像的 registry。
- 然后，将 Cluster Samples Operator 设置为 **Managed** 来安装您已镜像的镜像流。

4.3.7. 镜像用于断开连接的集群的 Operator 目录

您可以使用 **oc adm catalog mirror** 命令将红帽提供的目录或自定义目录的 Operator 内容镜像到容器镜像 registry 中。目标 registry 必须支持 Docker v2-2。对于受限网络中的集群，此 registry 可以是集群有网络访问权限的 registry，如在受限网络集群安装过程中创建的镜像 registry。



重要

- OpenShift 镜像 registry 不能用作目标 registry，因为它不支持没有标签的推送，在镜像过程中需要这个推送。
- 运行 **oc adm catalog mirror** 可能会导致以下错误： **error: unable to retrieve source image**。当镜像索引包括对镜像 registry 中不再存在的镜像的引用时，会发生此错误。镜像索引可能会保留旧的引用，以便为运行这些镜像的用户在升级图表中显示新的升级路径。作为临时解决方案，您可以使用 **--skip-missing** 选项绕过错误并继续下载镜像索引。如需更多信息，请参阅 Service Mesh Operator 镜像失败。

oc adm catalog mirror 命令还会自动将在镜像过程中指定的索引镜像（无论是红帽提供的索引镜像还是您自己的自定义构建索引镜像）镜像到目标 registry。然后，您可以使用镜像的索引镜像创建一个目录源，允许 Operator Lifecycle Manager（OLM）将镜像目录加载到 OpenShift Container Platform 集群。

其他资源

- 在受限网络中使用 Operator Lifecycle Manager

4.3.7.1. 先决条件

与断开连接的集群一起使用的 Operator 目录具有以下先决条件：

- 没有网络访问限制的工作站
- **podman** 1.9.3 或更高版本。
- 如果要过滤或 *prune* 一个现存的目录，且仅选择性地镜像部分 Operator，请参阅以下部分：
  - 安装 opm CLI
  - 更新或过滤基于文件的目录镜像
- 如果要镜像红帽提供的目录，请在具有无网络访问限制的工作站中运行以下命令，以便与 **registry.redhat.io** 进行身份验证：

```
$ podman login registry.redhat.io
```

- 访问支持 Docker v22 的镜像 registry。
- 在镜像 registry 上，决定使用哪个存储库或命名空间来存储已镜像的 Operator 内容。例如，您可以创建一个 **olm-mirror** 存储库。
- 如果您的镜像 registry 无法访问互联网，请将可移动介质连接到您的没有网络访问限制的工作站。
- 如果您正在使用私有 registry，包括 **registry.redhat.io**，请将 **REG_CREDS** 环境变量设置为 registry 凭证的文件路径，以便在后续步骤中使用。例如，对于 **podman** CLI:

```
$ REG_CREDS=${XDG_RUNTIME_DIR}/containers/auth.json
```

4.3.7.2. 提取和镜像目录内容

oc adm catalog mirror 命令提取索引镜像的内容，以生成镜像所需的清单。命令的默认行为会生成清单，然后会自动将索引镜像以及索引镜像本身中的所有镜像内容镜像（mirror）到您的镜像 registry。

另外，如果您的镜像 registry 位于完全断开连接的主机上，或者断开连接的或 *airgapped* 主机上，您可以首先将内容镜像到可移动介质，将介质移到断开连接的环境中，然后将内容从介质镜像到 registry。

4.3.7.2.1. 将目录内容镜像到同一网络上的 registry

如果您的镜像 registry 与您的没有网络访问限制的工作站位于同一个网络中，请在您的工作站上执行以下操作：

流程

1. 如果您的镜像 registry 需要身份验证，请运行以下命令登录到 registry:

```
$ podman login <mirror_registry>
```

2. 运行以下命令，将内容提取并镜像到镜像 registry:

```
$ oc adm catalog mirror \
    <index_image> \ 1
    <mirror_registry>:<port>[/<repository>] \ 2
    [-a ${REG_CREDS}] \ 3
    [--insecure] \ 4
    [--index-filter-by-os='<platform>/<arch>'] \ 5
    [--manifests-only] 6
```

1. 指定您要镜像的目录的索引镜像。
2. 指定要将 Operator 内容镜像到的目标 registry 的完全限定域名(FQDN)。镜像 registry **<repository>** 可以是 registry 上的任何现有存储库或命名空间，如先决条件中所述，如 **olm-mirror**。如果在镜像过程中找到现有的存储库，存储库名称将添加到生成的镜像名称中。如果您不希望镜像名称包含存储库名称，请省略此行中的 **<repository>** 值，例如 **<mirror_registry>:<port>**。
3. 可选：如果需要，指定 registry 凭证文件的位置。**registry.redhat.io** 需要 **{REG_CREDS}**。
4. 可选：如果您不想为目标 registry 配置信任，请添加 **--insecure** 标志。
5. 可选：在有多个变体可用时，指定索引镜像的平台和架构。镜像被传递为 **'<platform>/<arch>[/<variant>]'**。这不适用于索引引用的镜像。有效值为 **linux/amd64**, **linux/ppc64le**, **linux/s390x**, **linux/arm64**.
6. 可选：只生成镜像所需的清单，而不实际将镜像内容镜像到 registry。这个选项对检查哪些将被镜像（mirror）非常有用，如果您只需要一小部分软件包，可以对映射列表进行修改。然后，您可以使用带有 **oc image mirror** 命令的 **mapping.txt** 文件来在以后的步骤中镜像修改的镜像列表。此标志用于从目录中对内容进行高级选择性镜像。

输出示例

```
src image has index label for database path: /database/index.db
using database path mapping: /database/index.db:/tmp/153048078
wrote database to /tmp/153048078 1
```

```
...
wrote mirroring manifests to manifests-redhat-operator-index-1614211642 (2)
```

(1) 命令生成的临时 **index.db** 数据库的目录。

(2) 记录生成的 manifests 目录名称。该目录在后续过程中被引用。

注意

Red Hat Quay 不支持嵌套存储库。因此，运行 **oc adm catalog mirror** 命令会失败，并显示 **401** 未授权错误。作为临时解决方案，您可以在运行 **oc adm catalog mirror** 命令时使用 **--max-components=2** 选项来禁用嵌套存储库的创建。有关此临时解决方案的更多信息，请参阅 Unauthorized error thrown while using catalog mirror command with Quay registry。

其他资源

- Operator 的架构和操作系统支持

4.3.7.2.2. 将目录内容镜像到 airgapped registry

如果您的镜像 registry 位于完全断开连接的主机上，或 airgapped 主机上，请执行以下操作。

流程

1. 在您的工作站中运行以下命令，且没有网络访问权限将内容镜像到本地文件中：

```
$ oc adm catalog mirror \
    <index_image> \ (1)
    file:///local/index \ (2)
    -a ${REG_CREDS} \ (3)
    --insecure \ (4)
    --index-filter-by-os='<platform>/<arch>' (5)
```

(1) 指定您要镜像的目录的索引镜像。

(2) 指定要镜像到当前目录中的本地文件的内容。

(3) 可选：如果需要，指定 registry 凭证文件的位置。

(4) 可选：如果您不想为目标 registry 配置信任，请添加 **--insecure** 标志。

(5) 可选：在有多个变体可用时，指定索引镜像的平台和架构。镜像被指定为 **'<platform>/<arch>[/<variant>]'**。这不适用于索引引用的镜像。有效值为 **linux/amd64**, **linux/ppc64le**, **linux/s390x**, **linux/arm64**, 和 **.***

输出示例

```
...
info: Mirroring completed in 5.93s (5.915MB/s)
wrote mirroring manifests to manifests-my-index-1614985528 (1)
```

```
To upload local images to a registry, run:

oc adm catalog mirror file://local/index/myrepo/my-index:v1 REGISTRY/REPOSITORY (2)
```

(1) 记录生成的 manifests 目录名称。该目录在后续过程中被引用。

(2) 记录根据您提供的索引镜像扩展的 **file://** 路径。这个路径在后续步骤中被引用。

此命令会在当前目录中创建 **v2/** 目录中。

2. 将 **v2/** 目录复制到可移动介质。
3. 物理删除该介质并将其附加到断开连接的环境中可访问镜像 registry 的主机。
4. 如果您的镜像 registry 需要身份验证，请在断开连接的环境中的主机上运行以下命令以登录到 registry：

```
$ podman login <mirror_registry>
```

5. 从包含 **v2/** 目录的父目录运行以下命令，将镜像从本地文件上传到镜像 registry：

```
$ oc adm catalog mirror \
    file://local/index/<repository>/<index_image>:<tag> \ (1)
    <mirror_registry>:<port>[/<repository>] \ (2)
    -a ${REG_CREDS} \ (3)
    --insecure \ (4)
    --index-filter-by-os='<platform>/<arch>' (5)
```

(1) 指定上一命令输出中的 **file://** 路径。

(2) 指定要将 Operator 内容镜像到的目标 registry 的完全限定域名(FQDN)。镜像 registry **<repository>** 可以是 registry 上的任何现有存储库或命名空间，如先决条件中所述，如 **olm-mirror**。如果在镜像过程中找到现有的存储库，存储库名称将添加到生成的镜像名称中。如果您不希望镜像名称包含存储库名称，请省略此行中的 **<repository>** 值，例如 **<mirror_registry>:<port>**。

(3) 可选：如果需要，指定 registry 凭证文件的位置。

(4) 可选：如果您不想为目标 registry 配置信任，请添加 **--insecure** 标志。

(5) 可选：在有多个变体可用时，指定索引镜像的平台和架构。镜像被指定为 **'<platform>/<arch>[/<variant>]'**。这不适用于索引引用的镜像。有效值为 **linux/amd64**, **linux/ppc64le**, **linux/s390x**, **linux/arm64**, 和 **.***

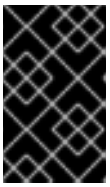
注意

Red Hat Quay 不支持嵌套存储库。因此，运行 **oc adm catalog mirror** 命令会失败，并显示 **401** 未授权错误。作为临时解决方案，您可以在运行 **oc adm catalog mirror** 命令时使用 **--max-components=2** 选项来禁用嵌套存储库的创建。有关此临时解决方案的更多信息，请参阅 Unauthorized error thrown while using catalog mirror command with Quay registry。

6. 再次运行 **oc adm catalog mirror** 命令。使用新镜像的索引镜像作为源，以及上一步中使用的同一镜像 registry 目标：

```
$ oc adm catalog mirror \
    <mirror_registry>:<port>/<index_image> \
    <mirror_registry>:<port>[/<repository>] \
    --manifests-only \ 1
    [-a ${REG_CREDS}] \
    [--insecure]
```

1 此步骤需要 **--manifests-only** 标志，以便该命令不会再次复制所有镜像的内容。



重要

这一步是必需的，因为上一步中生成的 **imageContentSourcePolicy.yaml** 文件中的镜像映射必须从本地路径更新为有效的镜像位置。如果不这样做，会在稍后的步骤中创建 **ImageContentSourcePolicy** 对象时会导致错误。

在镜像目录后，您可以继续执行集群的其余部分。在集群安装成功完成后，您必须指定此流程中的 manifests 目录来创建 **ImageContentSourcePolicy** 和 **CatalogSource** 对象。需要这些对象才能从 OperatorHub 安装 Operator。

其他资源

- Operator 的架构和操作系统支持

4.3.7.3. 生成的清单

将 Operator 目录内容镜像到镜像 registry 后，会在当前目录中生成清单目录。

如果您将内容镜像到同一网络上的 registry，则目录名称采用以下模式：

```
manifests-<index_image_name>-<random_number>
```

如果您在上一节中将内容镜像到断开连接的主机上的 registry，则目录名称采用以下模式：

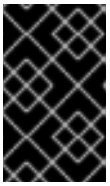
```
manifests-index/<repository>/<index_image_name>-<random_number>
```

注意

清单目录名称在后续过程中被引用。

manifests 目录包含以下文件，其中的一些文件可能需要进一步修改：

- **catalogSource.yaml** 文件是 **CatalogSource** 对象的基本定义，它预先填充索引镜像标签及其他相关元数据。此文件可原样使用，或进行相应修改来在集群中添加目录源。



重要

如果将内容镜像到本地文件，您必须修改 catalogSource **.yaml** 文件，从 **metadata.name** 字段中删除任何反斜杠(/)字符。否则，当您试图创建对象时，会失败并显示 "invalid resource name" 错误。

- 用来定义 **ImageContentSourcePolicy** 对象的 **imageContentSourcePolicy.yaml**，它可以将节点配置为在 Operator 清单中存储的镜像（image）引用和镜像 (mirror) 的 registry 间进行转换。

注意

如果您的集群使用 **ImageContentSourcePolicy** 对象来配置存储库镜像，则只能将全局 pull secret 用于镜像 registry。您不能在项目中添加 pull secret。

- **mapping.txt** 文件，在其中包含所有源镜像，并将它们映射到目标 registry。此文件与 **oc image mirror** 命令兼容，可用于进一步自定义镜像（mirror）配置。

重要

如果您在镜像过程中使用 **--manifests-only** 标志，并希望进一步调整要镜像的软件包子集，请参阅 OpenShift Container Platform 4.7 文档中的镜像软件包清单格式目录镜像流程中有关修改 **mapping.txt** 文件并使用 **oc image mirror** 命令的步骤。

4.3.7.4. 安装后的要求

在镜像目录后，您可以继续执行集群的其余部分。在集群安装成功完成后，您必须指定此流程中的 manifests 目录来创建 **ImageContentSourcePolicy** 和 **CatalogSource** 对象。这些对象需要填充和启用从 OperatorHub 安装 Operator。

其他资源

- 从镜像的 Operator 目录填充 OperatorHub
- 更新或过滤基于文件的目录镜像

4.3.8. 后续步骤

- 在您在受限网络中置备的基础架构上安装集群，如 VMware vSphere、裸机或 Amazon Web Services。

4.3.9. 其他资源

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

4.4. 使用 OC-MIRROR 插件为断开连接的安装镜像镜像

可以在没有直接的互联网连接的受限网络中运行集群，方法是使用在一个私有 registry 中的 mirror OpenShift Container Platform 容器镜像安装集群。集群运行时必须始终运行此 registry。如需更多信息，请参阅先决条件部分。

您可以使用 oc-mirror OpenShift CLI (**oc**)插件在完全或部分断开连接的环境中将镜像镜像到镜像 registry。您必须从具有互联网连接的系统运行 oc-mirror，以便从官方红帽 registry 中下载所需的镜像。

4.4.1. 关于 oc-mirror 插件

您可以使用 oc-mirror OpenShift CLI(**oc**)插件，使用单个工具将所有所需的 OpenShift Container Platform 内容和其他镜像(mirror)镜像到您的镜像 registry。它提供以下功能：

- 提供镜像 OpenShift Container Platform 发行版本、Operator、helm chart 和其他镜像的集中方法。
- 维护 OpenShift Container Platform 和 Operator 的更新路径。
- 使用声明的镜像设置配置文件来仅包含集群所需的 OpenShift Container Platform 发行版本、Operator 和镜像。
- 执行增量镜像，从而减少将来镜像集的大小。
- 从上一执行以来，从镜像集配置中排除的目标镜像 registry 中修剪镜像的镜像。
- （可选）为 OpenShift Update Service (OSUS) 使用生成支持工件。

使用 oc-mirror 插件时，您可以在镜像设置配置文件中指定要镜像的内容。在这个 YAML 文件中，您可以将配置微调为仅包含集群需要的 OpenShift Container Platform 发行版本和 Operator。这可减少您下载和传输所需的数据量。oc-mirror 插件也可以镜像任意 helm chart 和附加容器镜像，以帮助用户将其工作负载无缝同步到镜像 registry 中。

第一次运行 oc-mirror 插件时，它会使用所需内容填充您的镜像 registry，以执行断开连接的集群安装或更新。要让断开连接的集群继续接受更新，您必须更新镜像 registry。要更新您的镜像 registry，请使用与第一次运行相同的配置运行 oc-mirror 插件。oc-mirror 插件引用存储后端的元数据，并只下载上次运行该工具后所发布的元数据。这为 OpenShift Container Platform 和 Operator 提供了更新路径，并根据需要执行依赖项解析。

4.4.1.1. 高级别工作流

下列步骤概述了如何使用 oc-mirror 插件将镜像镜像到镜像 registry 的高级别工作流：

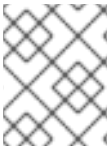
1. 创建镜像设置配置文件。
2. 使用以下方法之一将镜像设置为目标镜像 registry：
   - 将镜像直接设置为目标镜像 registry。
   - 镜像集合镜像到磁盘，将镜像设置为目标环境，然后将镜像上传到目标镜像 registry。
3. 配置集群以使用 oc-mirror 插件生成的资源。
4. 根据需要重复这些步骤以更新目标镜像 registry。

重要

当使用 oc-mirror CLI 插件填充镜像 registry 时，必须使用 oc-mirror 插件对目标镜像 registry 进行进一步的更新。

4.4.2. oc-mirror 插件兼容性和支持

oc-mirror 插件支持为 OpenShift Container Platform 版本 4.12 及之后的版本的镜像 OpenShift Container Platform 有效负载镜像和 Operator 目录。



注意

在 **aarch64**, **ppc64le**, 和 **s390x** 架构中，oc-mirror 插件只支持 OpenShift Container Platform 版本 4.14 及更新的版本。

使用 oc-mirror 插件的最新版本，无论您需要镜像的 OpenShift Container Platform 版本是什么。

其他资源

- 有关更新 oc-mirror 的详情，请参阅 查看镜像拉取源。

4.4.3. 关于镜像 registry

您可以将 OpenShift Container Platform 安装和后续的产品更新镜像(mirror)到支持 Docker v2-2 （如 Red Hat Quay）的容器镜像（如 Red Hat Quay）的容器镜像。如果您无法访问大型容器 registry，可以使用 *Red Hat OpenShift 的镜像 registry*，这是 OpenShift 中包含的小型容器 registry。

无论您所选 registry 是什么，都会将互联网上红帽托管站点的内容镜像到隔离的镜像 registry 相同。镜像内容后，您要将每个集群配置为从镜像 registry 中检索此内容。

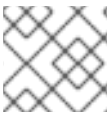
重要

OpenShift 镜像 registry 不能用作目标 registry，因为它不支持没有标签的推送，在镜像过程中需要这个推送。

如果选择的容器 registry 不是 *mirror registry for Red Hat OpenShift*，则需要集群中置备的每台机器都可以访问它。如果 registry 无法访问，安装、更新或常规操作（如工作负载重新定位）可能会失败。因此，您必须以高度可用的方式运行镜像 registry，镜像 registry 至少必须与 OpenShift Container Platform 集群的生产环境可用性相匹配。

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

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



注意

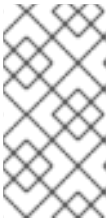
红帽没有针对 OpenShift Container Platform 测试第三方 registry。

其他资源

- 有关查看 CRI-O 日志以查看镜像源的详情，请参阅查看镜像拉取源。

4.4.4. 先决条件

- 您必须在托管 OpenShift Container Platform 集群的位置（如 Red Hat Quay）中有一个支持 Docker v2-2 的容器镜像 registry。



注意

如果使用 Red Hat Quay，则必须在 oc-mirror 插件中使用 3.6 或更高版本的版本。如果您有 Red Hat Quay 权利，请参阅有关部署 Red Hat Quay 以了解概念验证的文档，或使用 Red Hat Quay Operator。如果您需要额外的帮助来选择并安装 registry，请联络您的销售代表或红帽支持。

如果您还没有容器镜像 registry，OpenShift Container Platform 可以为订阅者提供一个 mirror registry for Red Hat OpenShift。*Red Hat OpenShift 的镜像 registry* 包含在您的订阅中，它是一个小型容器 registry，可用于在断开连接的安装中镜像 OpenShift Container Platform 所需的容器镜像。

4.4.5. 准备您的镜像主机

在使用 oc-mirror 插件镜像(mirror)前，您必须安装插件并创建容器镜像 registry 凭据文件，以允许从红帽镜像到您的镜像。

4.4.5.1. 安装 oc-mirror OpenShift CLI 插件

安装 oc-mirror OpenShift CLI 插件以在断开连接的环境中管理镜像集。

先决条件

- 已安装 OpenShift CLI（**oc**）。如果您在完全断开连接的环境中镜像镜像集，请确保以下内容：
  - 您已在可访问互联网的主机上安装了 oc-mirror 插件。
  - 在断开连接的环境中的主机可以访问目标镜像 registry。
- 您已在使用 oc-mirror 的操作系统中，将 **umask** 参数设置为 **0022**。
- 您已为您要使用的 RHEL 版本安装了正确的二进制文件。

流程

1. 下载 oc-mirror CLI 插件。
   a. 导航到 OpenShift Cluster Manager 的 Downloads 页面。
   b. 在 **OpenShift disconnected 安装工具**部分下，点 **Download** for **OpenShift Client(oc)mirror 插件** 并保存该文件。
2. 解压归档：

```
$ tar xvzf oc-mirror.tar.gz
```

3. 如有必要，将插件文件更新为可执行。

```
$ chmod +x oc-mirror
```

注意

不要重命名 **oc-mirror** 文件。

4. 通过将文件放在 **PATH** 中，例如 **/usr/local/bin**，安装 oc-mirror CLI 插件：

```
$ sudo mv oc-mirror /usr/local/bin/.
```

验证

- 运行以下命令，验证 oc-mirror v1 的插件是否已成功安装：

```
$ oc mirror help
```

其他资源

- 安装和使用 CLI 插件

4.4.5.2. 配置允许对容器镜像进行镜像的凭证

创建容器镜像 registry 凭证文件，可让您将镜像从红帽 mirror 到您的镜像。

警告

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

警告

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

先决条件

- 您已将镜像 registry 配置为在断开连接的环境中使用。
- 您在镜像 registry 中标识了镜像仓库的位置，以将容器镜像镜像(mirror)到这个位置。
- 您置备了一个镜像 registry 帐户，允许将镜像上传到该镜像仓库。

流程

在安装主机上完成以下步骤：

1. 从 Red Hat OpenShift Cluster Manager 下载 **registry.redhat.io** pull secret。
2. 以 JSON 格式创建您的 pull secret 副本：

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

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

该文件类似于以下示例：

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

3. 将文件保存为 **~/.docker/config.json** 或 **$XDG_RUNTIME_DIR/containers/auth.json**。

1. 为您的镜像 registry 生成 base64 编码的用户名和密码或令牌：

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

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

2. 编辑 JSON 文件并添加描述 registry 的部分：

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

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

2 指定镜像 registry 的 base64 编码用户名和密码。

该文件类似于以下示例：

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

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

4.4.6. 创建镜像设置配置

在使用 oc-mirror 插件镜像集之前，必须先创建镜像设置配置文件。此镜像设置配置文件定义哪些 OpenShift Container Platform 发行版本、Operator 和其他镜像要镜像，以及 oc-mirror 插件的其他配置设置。

您必须在镜像设置配置文件中指定存储后端。此存储后端可以是本地目录或支持 Docker v2-2 的 registry。oc-mirror 插件在创建镜像的过程中将元数据存储在这个存储后端中。

重要

不要删除或修改 oc-mirror 插件生成的元数据。每次针对同一镜像 registry 运行 oc-mirror 插件时，都必须使用相同的存储后端。

先决条件

- 您已创建了容器镜像 registry 凭证文件。具体步骤请参阅"配置允许镜像镜像的凭证"。

流程

1. 使用 **oc mirror init** 命令为镜像设置配置创建模板，并将其保存到名为 **imageset-config.yaml** 的文件中：

```
$ oc mirror init <--registry <storage_backend> > imageset-config.yaml (1)
```

(1) 指定存储后端的位置，如 **example.com/mirror/oc-mirror-metadata**。

2. 编辑该文件并根据需要调整设置：

```
kind: ImageSetConfiguration
```

```
apiVersion: mirror.openshift.io/v1alpha2
archiveSize: 4                                                    1
storageConfig:                                                    2
  registry:
    imageURL: example.com/mirror/oc-mirror-metadata               3
    skipTLS: false
mirror:
  platform:
    channels:
    - name: stable-4.16                                           4
      type: ocp
    graph: true                                                   5
  operators:
  - catalog: registry.redhat.io/redhat/redhat-operator-index:v4.16 6
    packages:
    - name: serverless-operator                                   7
      channels:
      - name: stable                                              8
  additionalImages:
  - name: registry.redhat.io/ubi9/ubi:latest                      9
  helm: {}
```

1. 添加 **archiveSize** 以设置镜像集合中的每个文件的最大大小（以 GiB 为单位）。
2. 设置后端位置，以将镜像设置元数据保存到。此位置可以是 registry 或本地目录。必须指定 **storageConfig** 值。
3. 设置存储后端的 registry URL。
4. 将频道设置为从中检索 OpenShift Container Platform 镜像。
5. 添加 **graph: true** 以构建并推送 graph-data 镜像推送到镜像 registry。创建 OpenShift Update Service (OSUS) 需要 graph-data 镜像。**graph: true** 字段还会生成 **UpdateService** 自定义资源清单。**oc** 命令行界面 (CLI) 可以使用 **UpdateService** 自定义资源清单来创建 OSUS。如需更多信息，请参阅*关于 OpenShift Update Service*。
6. 将 Operator 目录设置为从中检索 OpenShift Container Platform 镜像。
7. 仅指定要包含在镜像集中的某些 Operator 软件包。删除此字段以检索目录中的所有软件包。
8. 仅指定要包含在镜像集中的 Operator 软件包的某些频道。即使您没有使用该频道中的捆绑包，还必须始终包含 Operator 软件包的默认频道。您可以运行以下命令来找到默认频道：**oc mirror list operators --catalog=<catalog_name> --package=<package_name>**。
9. 指定要在镜像集中包含的任何其他镜像。

注意

graph: true 字段还会镜像 **ubi-micro** 镜像，以及其他镜像的镜像。

有关各种镜像用例，请参阅"镜像设置配置参数"和"镜像设置配置示例"。

3. 保存更新的文件。
   在镜像内容时，**oc mirror** 命令需要此镜像设置配置文件。

其他资源

- 镜像设置配置参数
- 镜像设置配置示例
- 在断开连接的环境中使用 OpenShift Update Service

4.4.7. 将镜像集镜像(mirror)到镜像 registry

您可以使用 oc-mirror CLI 插件在 部分断开连接的环境中或完全断开连接的环境中将镜像镜像到镜像 registry。

这些步骤假定您已设置了镜像 registry。

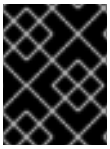
4.4.7.1. 在部分断开连接的环境中镜像设置的镜像

在部分断开连接的环境中，您可以直接镜像到目标镜像 registry 的镜像。

4.4.7.1.1. 镜像(mirror)到镜像(mirror)的镜像

您可以使用 oc-mirror 插件将镜像直接设置为在镜像设置过程中可访问的目标镜像 registry。

您必须在镜像设置配置文件中指定存储后端。这个存储后端可以是本地目录或 Docker v2 registry。oc-mirror 插件在创建镜像的过程中将元数据存储在这个存储后端中。



重要

不要删除或修改 oc-mirror 插件生成的元数据。每次针对同一镜像 registry 运行 oc-mirror 插件时，都必须使用相同的存储后端。

先决条件

- 您可以访问互联网来获取所需的容器镜像。
- 已安装 OpenShift CLI(**oc**)。
- 已安装 **oc-mirror** CLI 插件。
- 您已创建了镜像设置配置文件。

流程

- 运行 **oc mirror** 命令将指定镜像集配置中的镜像镜像到指定的 registry:

```
$ oc mirror --config=./<imageset-config.yaml> \ 1
  docker://registry.example:5000 2
```

1. 指定您创建的镜像设置配置文件。例如，**imageset-config.yaml**。
2. 指定要镜像设置文件的 registry。registry 必须以 **docker://** 开头。如果为镜像 registry 指定顶层命名空间，则必须在后续执行时使用此命名空间。

验证

1. 进入生成的 **oc-mirror-workspace/** 目录。
2. 导航到结果目录，例如，**results-1639608409/**。
3. 验证 **ImageContentSourcePolicy** 和 **CatalogSource** 资源是否存在 YAML 文件。

注意

ImageContentSourcePolicy YAML 文件的 **repositoryDigestMirrors** 部分在安装过程中用于 **install-config.yaml** 文件。

后续步骤

- 配置集群以使用 oc-mirror 生成的资源。

故障排除

- 无法检索源镜像。

4.4.7.2. 镜像在完全断开连接的环境中设置的镜像

要镜像在完全断开连接的环境中设置的镜像，您必须首先将镜像集镜像到磁盘， 然后将磁盘上的镜像集文件镜像到一个镜像。

4.4.7.2.1. 从镜像镜像到磁盘

您可以使用 oc-mirror 插件生成镜像集，并将内容保存到磁盘。然后，生成的镜像集可以转移到断开连接的环境中，并镜像到目标 registry。

重要

根据镜像设置配置文件中指定的配置，使用 oc-mirror 的镜像可能会将几百 GB 数据下载到磁盘。

您填充镜像 registry 时初始镜像集下载通常是最大镜像。因为您只下载自上次运行命令以来更改的镜像，所以再次运行 oc-mirror 插件时，所生成的镜像集通常比较小。

您必须在镜像设置配置文件中指定存储后端。这个存储后端可以是本地目录或 docker v2 registry。oc-mirror 插件在创建镜像的过程中将元数据存储在这个存储后端中。

重要

不要删除或修改 oc-mirror 插件生成的元数据。每次针对同一镜像 registry 运行 oc-mirror 插件时，都必须使用相同的存储后端。

先决条件

- 您可以访问互联网来获取所需的容器镜像。

- 已安装 OpenShift CLI(**oc**)。
- 已安装 **oc-mirror** CLI 插件。
- 您已创建了镜像设置配置文件。

流程

- 运行 **oc mirror** 命令将指定镜像集配置镜像到磁盘：

```
$ oc mirror --config=./imageset-config.yaml \ 1
  file://<path_to_output_directory>           2
```

1. 传递创建的镜像设置配置文件。此流程假设它名为 **imageset-config.yaml**。
2. 指定要输出镜像集文件的目标目录。目标目录路径必须以 **file://** 开头。

验证

1. 进入您的输出目录：

```
$ cd <path_to_output_directory>
```

2. 验证是否创建了镜像设置 **.tar** 文件：

```
$ ls
```

输出示例

```
mirror_seq1_000000.tar
```

后续步骤

- 将镜像集 .tar 文件移动到断开连接的环境中。

故障排除

- 无法检索源镜像。

4.4.7.2.2. 从磁盘镜像到镜像

您可以使用 oc-mirror 插件将生成的镜像集的内容镜像到目标镜像 registry。

先决条件

- 您已在断开连接的环境中安装了 OpenShift CLI(**oc**)。
- 您已在断开连接的环境中安装了 **oc-mirror** CLI 插件。
- 已使用 **oc mirror** 命令生成镜像集文件。
- 您已将镜像集文件传送到断开连接的环境中。

流程

- 运行 **oc mirror** 命令，以处理磁盘上镜像集文件，并将内容镜像到目标镜像 registry：

```
$ oc mirror --from=./mirror_seq1_000000.tar \ 1
  docker://registry.example:5000 2
```

1. 传递镜像集 .tar 文件以进行镜像，在本例中名为 **mirror_seq1_000000.tar**。如果在镜像设置配置文件中指定了 **archiveSize** 值，则镜像集可能会划分为多个 .tar 文件。在这种情况下，您可以传递一个包含镜像设置 .tar 文件的目录。
2. 指定要镜像设置文件的 registry。registry 必须以 **docker://** 开头。如果为镜像 registry 指定顶层命名空间，则必须在后续执行时使用此命名空间。

此命令使用镜像集更新镜像 registry，并生成 **ImageContentSourcePolicy** 和 **CatalogSource** 资源。

验证

1. 进入生成的 **oc-mirror-workspace/** 目录。
2. 导航到结果目录，例如，**results-1639608409/**。
3. 验证 **ImageContentSourcePolicy** 和 **CatalogSource** 资源是否存在 YAML 文件。

后续步骤

- 配置集群以使用 oc-mirror 生成的资源。

故障排除

- 无法检索源镜像。

4.4.8. 配置集群以使用 oc-mirror 生成的资源

将镜像设置为镜像 registry 后，您必须将生成的 **ImageContentSourcePolicy**、**CatalogSource** 和发行版本镜像签名资源应用到集群。

ImageContentSourcePolicy 资源将镜像 registry 与源 registry 关联，并将在线 registry 中的镜像拉取请求重定向到镜像 registry。Operator Lifecycle Manager(OLM)使用 **CatalogSource** 资源检索有关镜像 registry 中可用 Operator 的信息。发行镜像签名用于验证镜像的发行镜像。

先决条件

- 您已将镜像设置为断开连接的环境中的 registry 镜像。
- 您可以使用具有 **cluster-admin** 角色的用户访问集群。

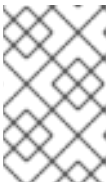
流程

1. 以具有 **cluster-admin** 角色的用户身份登录 OpenShift CLI。
2. 运行以下命令，将结果目录中的 YAML 文件应用到集群：

```
$ oc apply -f ./oc-mirror-workspace/results-1639608409/
```

3. 如果镜像(mirror)镜像，请运行以下命令将发行版本镜像签名应用到集群：

```
$ oc apply -f ./oc-mirror-workspace/results-1639608409/release-signatures/
```



注意

如果要镜像 Operator 而不是集群，则不需要运行 **$ oc apply -f ./oc-mirror-workspace/results-1639608409/release-signatures/**。运行该命令将返回错误，因为没有要应用的发行版本镜像签名。

验证

1. 运行以下命令验证 **ImageContentSourcePolicy** 资源是否已成功安装：

```
$ oc get imagecontentsourcepolicy
```

2. 运行以下命令验证 **CatalogSource** 资源是否已成功安装：

```
$ oc get catalogsource -n openshift-marketplace
```

4.4.9. 更新您的镜像 registry 内容

您可以通过更新镜像设置配置文件并将镜像集镜像到镜像 registry 来更新镜像 registry 内容。下次运行 oc-mirror 插件时，会生成一个镜像集，该镜像集仅包含之前执行以来的新和更新镜像。

在更新镜像 registry 时，您必须考虑以下注意事项：

- 如果镜像不再包含在生成和镜像的最新镜像集中，则会从目标镜像 registry 中修剪镜像。因此，请确保为以下关键组件相同的组合更新镜像，以便只创建并镜像不同的镜像集：
  - 镜像设置配置
  - 目标 registry
  - 存储配置
- 当要 mirror 或 mirror 到 mirror 工作流时，可以修剪镜像。
- 生成的镜像集必须按顺序推送到目标镜像 registry。您可以从生成的镜像设置归档文件的文件名中获取序列号。
- 不要删除或修改 oc-mirror 插件生成的元数据镜像。
- 如果您在初始镜像集创建过程中为镜像 registry 指定顶层命名空间，则每次针对同一镜像 registry 运行 oc-mirror 插件时都必须使用此命名空间。

有关更新镜像 registry 内容的工作流的更多信息，请参阅"高级别工作流"部分。

4.4.9.1. 镜像 registry 更新示例

本节论述了将镜像 registry 从磁盘更新到镜像的用例。

以前用于镜像的 ImageSetConfiguration 文件示例：

```
apiVersion: mirror.openshift.io/v1alpha2
kind: ImageSetConfiguration
storageConfig:
  local:
    path: /home/user/metadata
mirror:
  platform:
    channels:
      - name: stable-4.12
        minVersion: 4.12.1
        maxVersion: 4.12.1
  operators:
    - catalog: registry.redhat.io/redhat/redhat-operator-index:v4.14
      packages:
        - name: rhacs-operator
          channels:
          - name: stable
```

通过修剪现有镜像来镜像特定的 OpenShift Container Platform 版本

更新了 ImageSetConfiguration 文件：

```
apiVersion: mirror.openshift.io/v1alpha2
kind: ImageSetConfiguration
storageConfig:
  local:
    path: /home/user/metadata
mirror:
  platform:
    channels:
      - name: stable-4.13 (1)
  operators:
    - catalog: registry.redhat.io/redhat/redhat-operator-index:v4.14
      packages:
        - name: rhacs-operator
          channels:
          - name: stable
```

(1) 使用 **stable-4.13** 修剪 **stable-4.12** 的所有镜像。

通过修剪现有镜像升级到 Operator 的最新版本

更新了 ImageSetConfiguration 文件：

```
apiVersion: mirror.openshift.io/v1alpha2
kind: ImageSetConfiguration
storageConfig:
  local:
    path: /home/user/metadata
mirror:
  platform:
    channels:
```

```
    - name: stable-4.12
      minVersion: 4.12.1
      maxVersion: 4.12.1
  operators:
   - catalog: registry.redhat.io/redhat/redhat-operator-index:v4.14
     packages:
      - name: rhacs-operator
        channels:
        - name: stable 1
```

1 使用相同的频道而没有指定版本会修剪现有镜像，并使用最新版本的镜像进行更新。

通过修剪现有的 Operator 来镜像新 Operator

更新了 ImageSetConfiguration 文件：

```
apiVersion: mirror.openshift.io/v1alpha2
kind: ImageSetConfiguration
storageConfig:
  local:
    path: /home/user/metadata
mirror:
  platform:
    channels:
      - name: stable-4.12
        minVersion: 4.12.1
        maxVersion: 4.12.1
  operators:
    - catalog: registry.redhat.io/redhat/redhat-operator-index:v4.14
      packages:
        - name: <new_operator_name> 1
          channels:
          - name: stable
```

1 使用 **new_operator_name** 替换 **rhacs-operator** 修剪 Red Hat Advanced Cluster Security for Kubernetes Operator。

修剪所有 OpenShift Container Platform 镜像

更新了 ImageSetConfiguration 文件：

```
apiVersion: mirror.openshift.io/v1alpha2
kind: ImageSetConfiguration
storageConfig:
  local:
    path: /home/user/metadata
mirror:
  platform:
    channels:
  operators:
    - catalog: registry.redhat.io/redhat/redhat-operator-index:v4.14
      packages:
```

其他资源

- 镜像设置配置示例
- 在部分断开连接的环境中镜像设置的镜像
- 镜像在完全断开连接的环境中设置的镜像
- 配置集群以使用 oc-mirror 生成的资源

4.4.10. 执行空运行

您可以使用 oc-mirror 来执行空运行，而无需实际镜像(mirror)。这可让您查看要镜像的镜像列表，以及从镜像 registry 修剪的所有镜像。使用空运行（dry run）还允许您在早期版本中捕获与镜像集配置相关的任何错误，或使用生成的镜像列表以及其他工具来执行镜像操作。

先决条件

- 您可以访问互联网来获取所需的容器镜像。
- 已安装 OpenShift CLI(**oc**)。
- 已安装 **oc-mirror** CLI 插件。
- 您已创建了镜像设置配置文件。

流程

1. 使用 **--dry-run** 标志运行 **oc mirror** 命令来执行空运行：

```
$ oc mirror --config=./imageset-config.yaml \ 1
  docker://registry.example:5000           \ 2
  --dry-run                                  3
```

1. 传递创建的镜像设置配置文件。此流程假设它名为 **imageset-config.yaml**。
2. 指定镜像 registry。在使用 **--dry-run** 标志时，不会镜像这个 registry。
3. 使用 **--dry-run** 标志来生成空运行工件，而不是实际的镜像设置文件。

输出示例

```
Checking push permissions for registry.example:5000
Creating directory: oc-mirror-workspace/src/publish
Creating directory: oc-mirror-workspace/src/v2
Creating directory: oc-mirror-workspace/src/charts
Creating directory: oc-mirror-workspace/src/release-signatures
No metadata detected, creating new workspace
wrote mirroring manifests to oc-mirror-workspace/operators.1658342351/manifests-redhat-operator-index

...
```

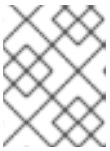
```
info: Planning completed in 31.48s
info: Dry run complete
Writing image mapping to oc-mirror-workspace/mapping.txt
```

2. 进入生成的工作区目录：

```
$ cd oc-mirror-workspace/
```

3. 查看生成的 **mapping.txt** 文件。
此文件包含将要镜像的所有镜像的列表。

4. 查看生成的 **prune-plan.json** 文件。
此文件包含在发布镜像集时从镜像 registry 中修剪的所有镜像的列表。



注意

只有在 oc-mirror 命令指向您的镜像 registry 且需要修剪的镜像时，才会生成 **prune-plan.json** 文件。

4.4.11. 包括本地 OCI Operator 目录

虽然将 OpenShift Container Platform 发行版本、Operator 目录和其他额外的镜像从 registry mirror 到一个部分断开连接的集群中，但您还可以在一个本地磁盘中的基于文件的目录中包含 Operator 目录镜像。本地目录必须采用开放容器项目 (OCI) 格式。

本地目录及其内容会根据镜像设置配置文件中的过滤信息，mirror 到您的目标 mirror registry。

重要

在镜像本地 OCI 目录时，所有您要 mirror 的 OpenShift Container Platform 发行版本或其他需要和本地 OCI 格式目录一起 mirror 的镜像都必须从 registry 中拉取。

您无法在磁盘中一起 mirror OCI 目录和一个 oc-mirror 镜像集文件镜像。

使用 OCI 功能的一个用例是，您有一个 CI/CD 系统将 OCI 目录构建到磁盘上的位置，并需要将 OCI 目录以及 OpenShift Container Platform 发行版本一起 mirror 到您的 mirror 镜像 registry。

注意

如果您为 OpenShift Container Platform 4.12 的 oc-mirror 插件使用了预览预览的 OCI 本地目录功能，则无法再使用 oc-mirror 插件的 OCI 本地目录功能在本地复制目录，并将其转换为 OCI 格式作为 mirror 到一个完全断开连接的集群中的第一步。

先决条件

- 您可以访问互联网来获取所需的容器镜像。
- 已安装 OpenShift CLI(**oc**)。
- 已安装 **oc-mirror** CLI 插件。

流程

1. 创建镜像设置配置文件，并根据需要调整设置。
以下示例镜像设置配置在磁盘上 mirror 一个 OCI 目录，以及来自 **registry.redhat.io** 的 OpenShift Container Platform 发行版本和 UBI 镜像。

```
kind: ImageSetConfiguration
apiVersion: mirror.openshift.io/v1alpha2
storageConfig:
  local:
    path: /home/user/metadata                                    1
mirror:
  platform:
    channels:
    - name: stable-4.16                                          2
      type: ocp
    graph: false
  operators:
  - catalog: oci:///home/user/oc-mirror/my-oci-catalog           3
    targetCatalog: my-namespace/redhat-operator-index            4
    packages:
    - name: aws-load-balancer-operator
  - catalog: registry.redhat.io/redhat/redhat-operator-index:v4.16   5
    packages:
    - name: rhacs-operator
  additionalImages:
  - name: registry.redhat.io/ubi9/ubi:latest                     6
```

1. 设置后端位置，以将镜像设置元数据保存到。此位置可以是 registry 或本地目录。必须指定 **storageConfig** 值。
2. （可选）包括一个 OpenShift Container Platform 发行版本，以便从 **registry.redhat.io** mirror。
3. 指定磁盘上 OCI 目录位置的绝对路径。使用 OCI 功能时，路径必须以 **oci:**// 开头。
4. 另外，还可指定替代命名空间和名称来镜像目录。
5. （可选）指定要从 registry 中拉取的额外 Operator 目录。
6. （可选）指定要从 registry 中拉取的额外镜像。

2. 运行 **oc mirror** 命令将 OCI 目录 mirror 到目标 mirror registry：

```
$ oc mirror --config=./imageset-config.yaml \                   1
  docker://registry.example:5000                                2
```

1. 传递镜像设置配置文件。此流程假设它名为 **imageset-config.yaml**。
2. 指定要将内容 mirror 到的 registry。registry 必须以 **docker:**// 开头。如果为镜像 registry 指定顶层命名空间，则必须在后续执行时使用此命名空间。

另外，您可以指定其他标记来调整 OCI 功能的行为：

--oci-insecure-signature-policy

不要将签名推送到目标 mirror registry。

--oci-registries-config

指定 TOML 格式的 **registry.conf** 文件的路径。您可以使用它来从不同的 registry 中镜像，如用于测试的预生产位置，而无需更改镜像设置配置文件。这个标志只会影响本地 OCI 目录，而不会影响任何其他被镜像的内容。

registry.conf 文件示例

```
[[registry]]
 location = "registry.redhat.io:5000"
 insecure = false
 blocked = false
 mirror-by-digest-only = true
 prefix = ""
 [[registry.mirror]]
    location = "preprod-registry.example.com"
    insecure = false
```

后续步骤

- 配置集群以使用 oc-mirror 生成的资源。

其他资源

- 配置集群以使用 oc-mirror 生成的资源

4.4.12. 镜像设置配置参数

oc-mirror 插件需要一个镜像设置配置文件，该文件定义哪些镜像要镜像(mirror)。下表列出了 **ImageSetConfiguration** 资源的可用参数。

表 4.1. ImageSetConfiguration 参数

参数	描述	值
apiVersion	**ImageSetConfiguration** 内容的 API 版本。	字符串.例如：**mirror.openshift.io/v1alpha2**。
archiveSize	镜像集中的每个存档文件的最大大小（以 GiB 为单位）。	整数.例如： **4**
mirror	镜像集的配置。	对象

参数	描述	值

mirror.additionalImages	镜像集的额外镜像配置。	对象数组。例如：
		`additionalImages:`
		`  - name: registry.redhat.io/ubi8/ubi:latest`

mirror.additionalImages.name	要 mirror 的镜像的标签或摘要。	字符串.例如：**registry.redhat.io/ubi8/ubi:latest**
mirror.blockedImages	阻止 mirror 的镜像的完整标签、摘要或模式。	字符串数组。例如：**docker.io/library/alpine**
mirror.helm	镜像集的 helm 配置。请注意，oc-mirror 插件只支持 helm chart，在呈现时不需要用户输入。	对象

mirror.helm.local	要镜像的本地 helm chart。	对象数组。例如：
		`local:`
		`  - name: podinfo`
		`  path: /test/podinfo-5.0.0.tar.gz`

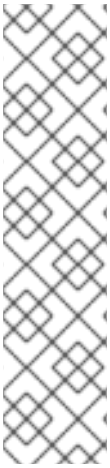
mirror.helm.local.name	要镜像的本地 helm chart 的名称。	字符串.例如：**podinfo**。
mirror.helm.local.path	到镜像的本地 helm chart 的路径。	字符串.例如：**/test/podinfo-5.0.0.tar.gz**。

参数	描述	值
mirror.helm.repositories	从其中镜像的的远程 helm 软件仓库。	对象数组。例如： repositories: - name: podinfo url: https://example.github.io/podinfo charts: - name: podinfo version: 5.0.0
mirror.helm.repositories.name	从其中镜像(mirror)的 helm 存储库的名称。	字符串.例如：**podinfo**。
mirror.helm.repositories.url	从其中镜像(mirror)的 helm 存储库的 URL。	字符串.例如：**https://example.github.io/podinfo**。
mirror.helm.repositories.charts	要镜像的远程 helm chart。	对象数组。
mirror.helm.repositories.charts.name	要镜像的 helm chart 的名称。	字符串.例如：**podinfo**。
mirror.helm.repositories.charts.version	要镜像命名 helm chart 的版本。	字符串.例如：**5.0.0**。
mirror.operators	镜像集的 Operator 配置。	对象数组。例如： operators: - catalog: registry.redhat.io/redhat/redhat-operator-index:v4.16 packages: - name: elasticsearch-operator minVersion: '2.4.0'

参数	描述	值
mirror.operators.catalog	包括在镜像集中的 Operator 目录。	字符串.例如：**registry.redhat.io/redhat/redhat-operator-index:v4.16**。
mirror.operators.full	为 **true** 时，下载完整的目录、Operator 软件包或 Operator 频道。	布尔值.默认值为 **false**。
mirror.operators.packages	Operator 软件包配置。	对象数组。例如： operators: - catalog: registry.redhat.io/redhat/redhat-operator-index:v4.16 packages: - name: elasticsearch-operator minVersion: '5.2.3-31'
mirror.operators.packages.name	镜像集中要包含的 Operator 软件包名称	字符串.例如：**elasticsearch-operator**。
mirror.operators.packages.channels	Operator 软件包频道配置。	对象
mirror.operators.packages.channels.name	Operator 频道名称（软件包中唯一）要包括在镜像集中。	字符串.例如：**fast** 或 **stable-v4.16**。
mirror.operators.packages.channels.maxVersion	Operator 镜像的最高版本，在其中存在所有频道。详情请查看以下备注。	字符串.例如：**5.2.3-31**
mirror.operators.packages.channels.minBundle	要包含的最小捆绑包的名称，以及频道头更新图中的所有捆绑包。仅在命名捆绑包没有语义版本元数据时设置此字段。	字符串.例如：**bundleName**
mirror.operators.packages.channels.minVersion	Operator 的最低版本，用于镜像存在的所有频道。详情请查看以下备注。	字符串.例如：**5.2.3-31**

参数	描述	值
mirror.operators.packages.maxVersion	Operator 最高版本，可跨所有存在的频道进行镜像。详情请查看以下备注。	字符串.例如：**5.2.3-31**。
mirror.operators.packages.minVersion	Operator 的最低版本，用于镜像存在的所有频道。详情请查看以下备注。	字符串.例如：**5.2.3-31**。
mirror.operators.skipDependencies	如果为 **true**，则不会包含捆绑包的依赖项。	布尔值.默认值为 **false**。
mirror.operators.targetCatalog	要镜像引用的目录的替代名称和可选命名空间层次结构。	字符串.例如：**my-namespace/my-operator-catalog**
mirror.operators.targetName	将引用的目录镜像为。 **targetName** 参数已弃用。改为使用 **targetCatalog** 参数。	字符串.例如： **my-operator-catalog**
mirror.operators.targetTag	附加到 **targetName** 或 **targetCatalog** 的替代标签。	字符串.例如： **v1**
mirror.platform	镜像集的平台配置。	对象
mirror.platform.architectures	要镜像的平台发行版本有效负载的架构。	字符串数组。例如： architectures: - amd64 - arm64 - multi - ppc64le - s390x 默认值为 **amd64**。值 **multi** 确保镜像支持所有可用架构，无需指定单个架构。
mirror.platform.channels	镜像集的平台频道配置。	对象数组。例如： channels: - name: stable-4.10 - name: stable-4.16

参数	描述	值
mirror.platform.channels.full	为 **true** 时，将 **minVersion** 设置为频道中的第一个发行版本，将 **maxVersion** 设置为该频道的最后一个发行版本。	布尔值.默认值为 **false**。
mirror.platform.channels.name	发行频道的名称。	字符串.例如：**stable-4.16**
mirror.platform.channels.minVersion	要镜像引用的平台的最低版本。	字符串.例如：**4.12.6**
mirror.platform.channels.maxVersion	要镜像引用的平台的最高版本。	字符串.例如：**4.16.1**
mirror.platform.channels.shortestPath	切换最短的路径镜像或完整范围镜像。	布尔值.默认值为 **false**。
mirror.platform.channels.type	要镜像的平台的类型。	字符串.例如： **ocp** 或 **okd**。默认为 **ocp**。
mirror.platform.graph	指明是否将 OSUS 图表添加到镜像集中，然后发布到镜像。	布尔值.默认值为 **false**。
storageConfig	镜像集的后端配置。	对象
storageConfig.local	镜像集的本地后端配置。	对象
storageConfig.local.path	包含镜像设置元数据的目录路径。	字符串.例如：**./path/to/dir/**。
storageConfig.registry	镜像集的 registry 后端配置。	对象
storageConfig.registry.imageURL	后端 registry URI。可以选择在 URI 中包含命名空间引用。	字符串.例如：**quay.io/myuser/imageset:metadata**。
storageConfig.registry.skipTLS	（可选）跳过引用的后端 registry 的 TLS 验证。	布尔值.默认值为 **false**。



注意

使用 **minVersion** 和 **maxVersion** 属性过滤特定 Operator 版本范围可能会导致多个频道头错误。错误信息将显示有**多个频道**头。这是因为在应用过滤器时，Operator 的更新图会被截断。

Operator Lifecycle Manager 要求每个 operator 频道都包含一个端点组成更新图表的版本，即 Operator 的最新版本。在应用图形的过滤器范围时，可以进入两个或多个独立图形或具有多个端点的图形。

要避免这个错误，请不要过滤 Operator 的最新版本。如果您仍然遇到错误，具体取决于 Operator，则必须增加 **maxVersion** 属性，或者 **minVersion** 属性必须减少。因为每个 Operator 图都可以不同，所以您可能需要调整这些值，直到错误解决为止。

4.4.13. 镜像设置配置示例

以下 **ImageSetConfiguration** 文件示例演示了各种镜像用例的配置。

使用案例：包含最短的 OpenShift Container Platform 更新路径
以下 **ImageSetConfiguration** 文件使用本地存储后端，并包括所有 OpenShift Container Platform 版本，以及从最低 **4.11.37** 版本到最大 **4.12.15** 版本的更新路径。

ImageSetConfiguration 文件示例

```
apiVersion: mirror.openshift.io/v1alpha2
kind: ImageSetConfiguration
storageConfig:
  local:
    path: /home/user/metadata
mirror:
  platform:
    channels:
      - name: stable-4.12
        minVersion: 4.11.37
        maxVersion: 4.12.15
        shortestPath: true
```

使用案例：包含对于多架构的版本的从最低到最新版本的所有 OpenShift Container Platform 版本
以下 **ImageSetConfiguration** 文件使用一个 registry 存储后端，并包括从最小 **4.13.4** 迁移到频道中最新版本的所有 OpenShift Container Platform 版本。对于每个使用此镜像集合配置的 oc-mirror，评估 **stable-4.13** 频道的最新发行版本，因此定期运行 oc-mirror 可确保您自动收到最新版本的 OpenShift Container Platform 镜像。

通过将 **platform.architectures** 的值设置为 **multi**，您可以确保支持多架构版本的镜像。

ImageSetConfiguration 文件示例

```
apiVersion: mirror.openshift.io/v1alpha2
kind: ImageSetConfiguration
storageConfig:
  registry:
    imageURL: example.com/mirror/oc-mirror-metadata
    skipTLS: false
mirror:
  platform:
```

```
    architectures:
      - "multi"
    channels:
      - name: stable-4.13
        minVersion: 4.13.4
        maxVersion: 4.13.6
```

使用案例：包含从最低到最新的 Operator 版本
以下 **ImageSetConfiguration** 文件使用本地存储后端，仅包含 **stable** 频道中从 4.0.1 及之后的版本开始的 Red Hat Advanced Cluster Security for Kubernetes Operator。

注意

当您指定了一个最小或最大版本范围时，可能不会接收该范围内的所有 Operator 版本。

默认情况下，oc-mirror 排除了 Operator Lifecycle Manager (OLM)规格中跳过或被较新的版本替换的任何版本。跳过的 Operator 版本可能会受到 CVE 或包含错误的影响。改为使用较新版本。有关跳过和替换版本的更多信息，请参阅使用 OLM 创建更新图表。

要接收指定范围内的所有 Operator 版本，您可以将 **mirror.operators.full** 字段设置为 **true**。

ImageSetConfiguration 文件示例

```
apiVersion: mirror.openshift.io/v1alpha2
kind: ImageSetConfiguration
storageConfig:
  local:
    path: /home/user/metadata
mirror:
  operators:
    - catalog: registry.redhat.io/redhat/redhat-operator-index:v4.16
      packages:
        - name: rhacs-operator
          channels:
          - name: stable
            minVersion: 4.0.1
```

注意

要指定最大版本而不是最新的版本，请设置 **mirror.operators.packages.channels.maxVersion** 字段。

使用案例：包含 Nutanix CSI Operator
以下 **ImageSetConfiguration** 文件使用本地存储后端，并包括 Nutanix CSI Operator、OpenShift Update Service (OSUS)图形镜像以及额外的 Red Hat Universal Base Image (UBI)。

ImageSetConfiguration 文件示例

```
kind: ImageSetConfiguration
apiVersion: mirror.openshift.io/v1alpha2
storageConfig:
  registry:
```

```
    imageURL: mylocalregistry/ocp-mirror/openshift4
    skipTLS: false
mirror:
  platform:
    channels:
    - name: stable-4.11
      type: ocp
    graph: true
  operators:
  - catalog: registry.redhat.io/redhat/certified-operator-index:v4.16
    packages:
    - name: nutanixcsioperator
      channels:
      - name: stable
  additionalImages:
  - name: registry.redhat.io/ubi9/ubi:latest
```

使用案例：包含默认 Operator 频道
以下 **ImageSetConfiguration** 文件包括 OpenShift Elasticsearch Operator 的 **stable-5.7** 和 **stable** 频道。即使只需要 **stable-5.7** 频道中的软件包，**stable** 频道也必须包含在 **ImageSetConfiguration** 文件中，因为它是 Operator 的默认频道。即使您没有使用该频道中的捆绑包，还必须始终包含 Operator 软件包的默认频道。

提示

您可以运行以下命令来找到默认频道：**oc mirror list operators --catalog=<catalog_name> --package=<package_name>**。

ImageSetConfiguration 文件示例

```
apiVersion: mirror.openshift.io/v1alpha2
kind: ImageSetConfiguration
storageConfig:
  registry:
    imageURL: example.com/mirror/oc-mirror-metadata
    skipTLS: false
mirror:
  operators:
  - catalog: registry.redhat.io/redhat/redhat-operator-index:v4.16
    packages:
    - name: elasticsearch-operator
      channels:
      - name: stable-5.7
      - name: stable
```

使用案例：包含整个目录（所有版本）
以下 **ImageSetConfiguration** 文件将 **mirror.operators.full** 字段设置为 **true**，使其包含整个 Operator 目录的所有版本。

ImageSetConfiguration 文件示例

```
apiVersion: mirror.openshift.io/v1alpha2
kind: ImageSetConfiguration
storageConfig:
```

```
  registry:
    imageURL: example.com/mirror/oc-mirror-metadata
    skipTLS: false
mirror:
  operators:
    - catalog: registry.redhat.io/redhat/redhat-operator-index:v4.16
      full: true
```

使用案例：包含整个目录（仅限频道头）
以下 **ImageSetConfiguration** 文件包含整个 Operator 目录的频道头。

默认情况下，对于目录中的每个 Operator，oc-mirror 都包含来自默认频道的最新 Operator 版本（频道头）。如果要镜像所有 Operator 版本，而不仅仅是频道头，您必须将 **mirror.operators.full** 字段设置为 **true**。

本例还使用 **targetCatalog** 字段指定替代命名空间和名称来镜像目录。

ImageSetConfiguration 文件示例

```
apiVersion: mirror.openshift.io/v1alpha2
kind: ImageSetConfiguration
storageConfig:
  registry:
    imageURL: example.com/mirror/oc-mirror-metadata
    skipTLS: false
mirror:
  operators:
  - catalog: registry.redhat.io/redhat/redhat-operator-index:v4.16
    targetCatalog: my-namespace/my-operator-catalog
```

用例：包含任意镜像和 helm chart
以下 **ImageSetConfiguration** 文件使用 registry 存储后端，并包含 helm chart 和额外的 Red Hat Universal Base Image(UBI)。

ImageSetConfiguration 文件示例

```
apiVersion: mirror.openshift.io/v1alpha2
kind: ImageSetConfiguration
archiveSize: 4
storageConfig:
  registry:
    imageURL: example.com/mirror/oc-mirror-metadata
    skipTLS: false
mirror:
 platform:
   architectures:
     - "s390x"
   channels:
     - name: stable-4.16
 operators:
   - catalog: registry.redhat.io/redhat/redhat-operator-index:v4.16
 helm:
   repositories:
     - name: redhat-helm-charts
       url: https://raw.githubusercontent.com/redhat-developer/redhat-helm-charts/master
```

```
    charts:
      - name: ibm-mongodb-enterprise-helm
        version: 0.2.0
additionalImages:
  - name: registry.redhat.io/ubi9/ubi:latest
```

4.4.14. oc-mirror 的命令参考

下表描述了 **oc mirror** 子命令和标志：

表 4.2. oc mirror 子命令

子命令	描述
completion	为指定的 shell 生成自动完成脚本。
describe	输出镜像集合的内容。
帮助	显示有关任何子命令的帮助。
init	输出初始镜像设置配置模板。
list	列出可用平台和 Operator 内容及其版本。
version	输出 oc-mirror 版本。

表 4.3. oc mirror 标记

标记	描述
-c, --config <string>	指定镜像设置配置文件的路径。
--continue-on-error	如果发生任何非镜像拉取相关的错误，请继续并尝试进行镜像(mirror)。
--dest-skip-tls	禁用目标 registry 的 TLS 验证。
--dest-use-http	使用 HTTP 用于目标 registry。
--dry-run	仅输出操作情况，不实际 mirror 镜像。生成 **mapping.txt** 和 **pruning-plan.json** 文件。
--from <string>	指定由执行 oc-mirror 生成的镜像设置归档的路径，以加载到目标 registry 中。
-h,--help	显示帮助。
--ignore-history	下载镜像和打包层时，忽略过去的镜像。禁用增量镜像，并可能会下载更多数据。

标记	描述
--manifests-only	为 **ImageContentSourcePolicy** 对象生成清单，将集群配置为使用镜像 registry，但不实际镜像任何镜像。要使用此标志，您必须使用 **--from** 标志传递镜像集存档。
--max-nested-paths <int>	指定限制嵌套路径的目标 registry 的最大嵌套路径数。默认值为 **0**。
--max-per-registry <int>	指定每个 registry 允许的并发请求数。默认值为 **6**。
--oci-insecure-signature-policy	在镜像本地 OCI 目录时不要推送签名（使用 **--include-local-oci-catalogs**）。
--oci-registries-config	提供 registry 配置文件，以指定在镜像本地 OCI 目录（使用 **--include-local-oci-catalogs**）时复制的替代 registry 位置。
--skip-cleanup	跳过删除工件目录。
--skip-image-pin	不要将镜像标签替换为 Operator 目录中的摘要。
--skip-metadata-check	发布镜像集时跳过元数据。只有在使用 **--ignore-history** 创建镜像集时才建议使用。
--skip-missing	如果没有找到镜像，则跳过它而不是报告错误并中止执行。不适用于在镜像设置配置中明确指定的自定义镜像。
--skip-pruning	从目标镜像 registry 禁用自动修剪镜像。
--skip-verification	跳过摘要验证。
--source-skip-tls	为源 registry 禁用 TLS 验证。
--source-use-http	将普通 HTTP 用于源 registry。
-v, --verbose <int>	指定日志级别详细程度的数量。有效值为 **0** - **9**。默认值为 **0**。

4.4.15. 其他资源

- 关于在断开连接的环境中的集群更新

4.5. 使用 OC-MIRROR 插件 V2 为断开连接的安装 MIRROR 镜像

如果从私有 registry 中的镜像 OpenShift Container Platform 容器镜像安装集群，则可以在没有直接互联网连接的受限网络中运行集群。每当集群运行时，此 registry 必须正在运行。

正如您可以使用 **oc-mirror** OpenShift CLI (**oc**)插件一样，您也可以使用 oc-mirror 插件 v2 在完全或部分断开连接的环境中将镜像镜像到镜像 registry。要从官方红帽 registry 下载所需的镜像，您必须从具有互联网连接的系统运行 oc-mirror 插件 v2。

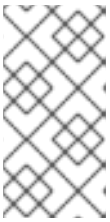
重要

oc-mirror 插件 v2 只是一个技术预览功能。技术预览功能不受红帽产品服务等级协议（SLA）支持，且功能可能并不完整。红帽不推荐在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。

有关红帽技术预览功能支持范围的更多信息，请参阅技术预览功能支持范围。

4.5.1. 先决条件

- 您必须在托管 OpenShift Container Platform 集群的位置（如 Red Hat Quay）中有一个支持 Docker V2-2 的容器镜像 registry。



注意

如果使用 Red Hat Quay，请在 oc-mirror 插件中使用 3.6 或更高版本。请参阅有关在 OpenShift Container Platform 上部署 Red Hat Quay Operator 的文档 (Red Hat Quay 文档)。如果您需要额外的帮助来选择并安装 registry，请联络您的销售代表或红帽支持。

- 如果您没有容器镜像 registry 的现有解决方案，OpenShift Container Platform 订阅者会收到一个 mirror registry for Red Hat OpenShift。此镜像 registry 包含在您的订阅中，并充当小型容器 registry。您可以使用此 registry 为断开连接的安装镜像 OpenShift Container Platform 所需的容器镜像。

- 置备的集群中的每个机器都必须有权访问镜像 registry。如果 registry 无法访问，安装、更新或常规操作等任务（如工作负载重新定位）可能会失败。镜像 registry 必须以高可用性的方式运行，确保其可用性与 OpenShift Container Platform 集群的生产环境可用性一致。

高级别工作流

以下步骤概述了如何使用 oc-mirror 插件 v2 将镜像镜像到镜像 registry 的高级别工作流：

1. 创建镜像设置配置文件。
2. 使用以下工作流之一将镜像设置为目标镜像 registry：
    - 将镜像直接设置为目标镜像 registry (mirror 到 mirror)。
        - 镜像设置为磁盘(Mirror-to-Disk)，将 **tar** 文件传送到目标环境，然后将镜像设置为目标镜像 registry (Disk-to-Mirror)。
3. 配置集群以使用 oc-mirror 插件 v2 生成的资源。
4. 根据需要重复这些步骤以更新目标镜像 registry。

4.5.2. 关于 oc-mirror 插件 v2

oc-mirror OpenShift CLI (**oc**) 插件是一个单一工具，可将所有所需的 OpenShift Container Platform 内容和其他镜像 (mirror)镜像到您的镜像 registry。

要使用 oc-mirror 的新技术预览版本，请在 oc-mirror 插件 **v2** 命令行中添加--v2 标志。

oc-mirror 插件 v2 具有以下功能：

- 验证镜像设置配置中指定的完整镜像集是否已镜像到已镜像的 registry，无论镜像之前是否被镜像(mirror)。
- 使用缓存系统而不是元数据。
- 通过将新镜像合并到存档中来维护最小归档大小。
- 使用通过镜像日期选择的内容生成镜像存档。
- 可以为完整镜像集生成 **ImageDigestMirrorSet** (IDMS)、**ImageTagMirrorSet** (ITMS) 而不是 **ImageContentSourcePolicy** (ICSP)，而不是只针对增量更改。
- 根据捆绑包名称保存过滤器 Operator 版本。
- 不执行自动修剪。V2 现在有一个 **Delete** 功能，它授予用户对删除镜像的更多控制。
- 引入了对 **registry.conf** 的支持。此更改有助于在使用相同的缓存时镜像到多个 enclaves。

4.5.2.1. oc-mirror 插件 v2 兼容性和支持

OpenShift Container Platform 支持 oc-mirror 插件 v2。

注意

在 **aarch64**,**ppc64le**, 和 **s390x** 架构上，oc-mirror 插件 v2 只支持 OpenShift Container Platform 版本 4.14 及更新的版本。

使用 oc-mirror 插件 v2 的最新可用版本，无论您需要镜像的 OpenShift Container Platform 版本是什么。

4.5.3. 准备您的镜像主机

要将 oc-mirror 插件 v2 用于镜像镜像，您需要安装插件并使用容器镜像凭证创建文件，使您能够从红帽镜像到您的镜像。

4.5.3.1. 安装 oc-mirror OpenShift CLI 插件

安装 oc-mirror OpenShift CLI 插件以在断开连接的环境中管理镜像集。

先决条件

- 已安装 OpenShift CLI(**oc**)。如果您在完全断开连接的环境中镜像镜像集，请确保以下内容：
  - 您已在可访问互联网的主机上安装了 oc-mirror 插件。
  - 在断开连接的环境中的主机可以访问目标镜像 registry。
- 您已在使用 oc-mirror 的操作系统中，将 **umask** 参数设置为 **0022**。
- 您已为您要使用的 RHEL 版本安装了正确的二进制文件。

流程

1. 下载 oc-mirror CLI 插件。

a. 导航到 OpenShift Cluster Manager 的 Downloads 页面。

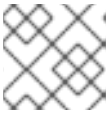
b. 在 **OpenShift disconnected 安装工具**部分下，点 **Download** for **OpenShift Client(oc)mirror 插件** 并保存该文件。

2. 解压归档：

```
$ tar xvzf oc-mirror.tar.gz
```

3. 如有必要，将插件文件更新为可执行。

```
$ chmod +x oc-mirror
```



注意

不要重命名 **oc-mirror** 文件。

4. 通过将文件放在 **PATH** 中，例如 **/usr/local/bin**，安装 oc-mirror CLI 插件：

```
$ sudo mv oc-mirror /usr/local/bin/.
```

验证

- 运行以下命令，验证 oc-mirror v2 的插件是否已成功安装：

```
$ oc mirror --v2 --help
```

4.5.3.2. 配置允许对容器镜像进行镜像的凭证

创建容器镜像 registry 凭证文件，可让您将镜像从红帽 mirror 到您的镜像。

警告

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

警告

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

先决条件

- 您已将镜像 registry 配置为在断开连接的环境中使用。
- 您在镜像 registry 中标识了镜像仓库的位置，以将容器镜像镜像(mirror)到这个位置。
- 您置备了一个镜像 registry 帐户，允许将镜像上传到该镜像仓库。

流程

在安装主机上完成以下步骤：

1. 从 Red Hat OpenShift Cluster Manager 下载 **registry.redhat.io** pull secret。
2. 以 JSON 格式创建您的 pull secret 副本：

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

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

该文件类似于以下示例：

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

3. 将文件保存为 **$XDG_RUNTIME_DIR/containers/auth.json**。
4. 为您的镜像 registry 生成 base64 编码的用户名和密码或令牌：

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

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

5. 编辑 JSON 文件并添加描述 registry 的部分：

```
  "auths": {
    "<mirror_registry>": { 1
```

```
    "auth": "<credentials>", (2)
    "email": "you@example.com"
  }
},
```

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

(2) 指定镜像 registry 的 base64 编码用户名和密码。

该文件类似于以下示例：

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

4.5.4. 将镜像集镜像(mirror)到镜像 registry

将镜像集镜像到镜像 registry 可确保所需的镜像在安全且受控的环境中可用，从而促进平稳部署、更新和维护任务。

4.5.4.1. 构建镜像设置配置

oc-mirror 插件 v2 使用镜像设置配置作为输入文件，以确定镜像镜像所需的镜像。

ImageSetConfiguration 输入文件示例

```
kind: ImageSetConfiguration
apiVersion: mirror.openshift.io/v2alpha1
mirror:
  platform:
    channels:
    - name: stable-4.13
```

```
    minVersion: 4.13.10
    maxVersion: 4.13.10
  graph: true
 operators:
  - catalog: registry.redhat.io/redhat/redhat-operator-index:v4.15
    packages:
     - name: aws-load-balancer-operator
     - name: 3scale-operator
     - name: node-observability-operator
 additionalImages:
  - name: registry.redhat.io/ubi8/ubi:latest
  - name:
registry.redhat.io/ubi9/ubi@sha256:20f695d2a91352d4eaa25107535126727b5945bff38ed36a3e5959
0f495046f0
```

4.5.4.2. 在部分断开连接的环境中镜像设置的镜像

您可以在带有受限互联网访问的环境中使用 oc-mirror 插件 v2 将镜像集镜像到 registry。

先决条件

- 在运行 oc-mirror 插件 v2 的环境中，您可以访问互联网和镜像 registry。

流程

- 运行以下命令，将指定镜像设置配置中的镜像镜像到指定的 registry：

```
$ oc mirror -c isc.yaml --workspace file://<file_path> docker://<mirror_registry_url> --v2 1
```

1 指定存储镜像并从中删除镜像的镜像 registry 的 URL 或地址。

验证

1. 进入到 **working-dir** 中的 **cluster-resources** 目录，它在 **<file_path>** 目录中生成。
2. 验证 **ImageDigestMirrorSet**、**ImageTagMirrorSet** 和 **CatalogSource** 资源是否存在 YAML 文件。

后续步骤

- 配置集群以使用 oc-mirror 插件 v2 生成的资源。

4.5.4.3. 镜像在完全断开连接的环境中设置的镜像

您可以在 OpenShift Container Platform 集群无法访问互联网的完全断开连接的环境中镜像镜像集。

1. **镜像到磁盘** ：准备包含为镜像设置的镜像的存档。需要访问互联网。
2. **手动步骤** ：将存档传输到断开连接的镜像 registry 的网络。
3. **磁盘到镜像**: 要将存档中的镜像集镜像到目标断开连接的 registry，请从可访问镜像 registry 的环境运行 oc-mirror 插件 v2。

4.5.4.3.1. 从镜像镜像到磁盘

您可以使用 oc-mirror 插件 v2 生成镜像集，并将内容保存到磁盘。然后，您可以将生成的镜像集传送到断开连接的环境中，并镜像到目标 registry。

oc-mirror 插件 v2 从镜像设置配置中指定的源中检索容器镜像，并将它们打包到本地目录中的 tar 存档中。

流程

- 运行以下命令，将指定镜像集配置中的镜像镜像到磁盘：

```
$ oc mirror -c isc.yaml file://<file_path> --v2 1
```

1 添加所需的文件路径。

验证

1. 进入生成的 **<file_path>** 目录。
2. 验证存档文件是否已生成。

后续步骤

- 配置集群以使用 oc-mirror 插件 v2 生成的资源。

4.5.4.3.2. 从磁盘镜像到镜像

您可以使用 **oc-mirror** 插件 v2 将磁盘中的镜像集镜像到目标镜像 registry。

oc-mirror 插件 v2 从本地磁盘检索容器镜像并将其传送到指定的镜像 registry。

流程

- 运行以下命令，处理磁盘上的镜像集文件，并将内容镜像到目标镜像 registry：

```
$ oc mirror -c isc.yaml --from file://<file_path> docker://<mirror_registry_url> --v2 1
```

1 指定存储镜像并从中删除镜像的镜像 registry 的 URL 或地址。

验证

1. 进入到 **working-dir** 中的 **cluster-resources** 目录，它在 **<file_path>** 目录中生成。
2. 验证 **ImageDigestMirrorSet**、**ImageTagMirrorSet** 和 **CatalogSource** 资源是否存在 YAML 文件。

后续步骤

- 配置集群以使用 oc-mirror 插件 v2 生成的资源。

4.5.5. 其他资源

- 使用 OpenShift Update Service 在断开连接的环境中更新集群。

4.5.6. 关于 v2 生成的自定义资源

使用 oc-mirror 插件 v2 时，如果至少找到一个标签，则默认生成 **ImageDigestMirrorSet** (IDMS) 和 **ImageTagMirrorSet** (ITMS)。这些集合包含由版本、Operator 目录和其他镜像中的摘要或标签引用的镜像的镜像。

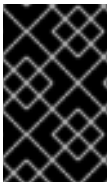
ImageDigestMirrorSet (IDMS) 将镜像 registry 链接到源 registry，并使用摘要规格转发镜像拉取请求。但是，**ImagetagMirrorSet** (ITMS) 资源使用镜像标签重定向镜像拉取请求。

Operator Lifecycle Manager (OLM) 使用 **CatalogSource** 资源来检索有关镜像 registry 中可用 Operator 的信息。

OSUS 服务使用 **UpdateService** 资源为断开连接的环境提供 Cincinnati 图。

4.5.6.1. 配置集群以使用 oc-mirror 插件 v2 生成的资源

将镜像设置为镜像 registry 后，您必须将生成的 **ImageDigestMirrorSet** (IDMS)、**ImageTagMirrorSet** (ITMS)、**CatalogSource** 和 **UpdateService** 应用到集群。



重要

在 oc-mirror 插件 v2 中，IDMS 和 ITMS 文件涵盖了整个镜像集，这与 oc-mirror 插件 v1 中的 ICSP 文件不同。因此，即使您仅在增量镜像过程中添加新镜像，IDMS 和 ITMS 文件还包含集合的所有镜像。

先决条件

- 您可以使用具有 **cluster-admin** 角色的用户访问集群。

流程

- 运行以下命令，将结果目录中的 YAML 文件应用到集群：

```
$ oc apply -f <path_to_oc-mirror_workspace>/working-dir/cluster-resources
```

验证

1. 运行以下命令验证 **ImageDigestMirrorSet** 资源是否已成功安装：

```
$ oc get imagedigestmirrorset
```

2. 运行以下命令验证 **ImageTagMirrorSet** 资源是否已成功安装：

```
$ oc get imagetagmirrorset
```

3. 运行以下命令验证 **CatalogSource** 资源是否已成功安装：

```
$ oc get catalogsource -n openshift-marketplace
```

4.5.7. 在断开连接的环境中删除镜像

在使用 oc-mirror 插件 v2 之前，您必须删除之前部署的镜像。oc-mirror 插件 v2 不再执行自动修剪。

在使用 oc-mirror 插件 v2 时，您必须创建 **DeleteImageSetConfiguration** 文件来删除镜像配置。这可防止在使用 **ImageSetConfig.yaml** 更改时意外删除必要的或部署的镜像。

在以下示例中，**DeleteImageSetConfiguration** 会删除以下内容：

- OpenShift Container Platform release 4.13.3 的所有镜像。
- 目录镜像 **redhat-operator-index v4.12**。
- **aws-load-balancer-operator** v0.0.1 捆绑包及其所有相关镜像。
- 相应的摘要引用的额外镜像 **ubi** 和 **ubi-minimal**。

示例： DeleteImageSetConfig

```
apiVersion: mirror.openshift.io/v2alpha1
kind: DeleteImageSetConfiguration
delete:
  platform:
    channels:
      - name: stable-4.13
        minVersion: 4.13.3
        maxVersion: 4.13.3
  operators:
    - catalog: registry.redhat.io/redhat/redhat-operator-index:v4.12
      packages:
      - name: aws-load-balancer-operator
          minVersion: 0.0.1
          maxVersion: 0.0.1
  additionalImages:
    - name:
registry.redhat.io/ubi8/ubi@sha256:bce7e9f69fb7d4533447232478fd825811c760288f87a35699f9c8f03
0f2c1a6
    - name: registry.redhat.io/ubi8/ubi-
minimal@sha256:8bedbe742f140108897fb3532068e8316900d9814f399d676ac78b46e740e34e
```

重要

考虑使用 mirror-to-disk 和 disk-to-mirror 工作流来减少镜像问题。

在镜像删除工作流中，oc-mirror 插件 v2 只删除镜像的清单，这不会减少 registry 中占用的存储。

要从不必要的镜像（如带有删除的清单）中释放存储空间，您必须在容器 registry 上启用垃圾收集器。启用垃圾收集器后，registry 将删除不再引用任何清单的镜像 Blob，从而减少之前由已删除 Blob 占用的存储。启用垃圾收集器会因容器 registry 而异。

4.5.7.1. 从断开连接的环境中删除镜像

要使用 oc-mirror 插件 v2 从断开连接的环境中删除镜像，请按照以下步骤操作。

流程

1. 创建删除之前镜像的 YAML 文件：

```
$ oc mirror delete --config delete-image-set-config.yaml --workspace
file://<previously_mirrored_work_folder> --v2 --generate docker://<remote_registry>
```

其中：

- **<previously_mirrored_work_folder>** ：使用镜像之前在镜像过程中镜像或存储的目录。
- **<remote_registry>** : 插入从中删除镜像的远程容器 registry 的 URL 或地址。

2. 进入创建的 **<previously_mirrored_work_folder>/delete directory**。
3. 验证 **delete-images.yaml** 文件是否已生成。
4. 手工确保文件中列出的每个镜像不再被集群需要，可以安全地从 registry 中删除。
5. 在生成 **delete** YAML 文件后，从远程 registry 中删除镜像：

```
$ oc mirror delete --v2 --delete-yaml-file <previously_mirrored_work_folder>/delete/delete-
images.yaml docker:/ <remote_registry>
```

其中：

- **<previously_mirrored_work_folder>**：指定之前镜像的工作文件夹。

重要

使用 mirror-to-mirror 时，镜像不会在本地缓存，因此您无法从本地缓存中删除镜像。

4.5.8. 验证您选择的镜像以进行镜像

您可以使用 oc-mirror 插件 v2 执行不实际镜像任何镜像的测试运行(dry run)。这可让您查看要镜像的镜像列表。您也可以在早期使用空运行来捕获与镜像设置配置的任何错误。在镜像到磁盘工作流上运行空运行时，oc-mirror 插件 v2 会检查镜像集中的所有镜像是否在其缓存中可用。**missing.txt** 文件中列出了任何缺少的镜像。在镜像前执行空运行时，**missing.txt** 和 **mapping.txt** 文件都包含相同的镜像列表。

4.5.8.1. 为 oc-mirror 插件 v2 执行空运行

通过在不镜像的情况下执行空运行来验证您的镜像设置。这样可确保您的设置正确，并防止意外更改。

流程

- 要执行测试运行，请运行 **oc mirror** 命令，并在命令中使用 **--dry-run** 参数：

```
$ oc mirror -c <image_set_config_yaml> --from file://<oc_mirror_workspace_path>
docker://<mirror_registry_url> --dry-run --v2
```

其中：

  - **<image_set_config_yaml>** ：使用您刚才创建的镜像设置配置文件。
  - **<oc_mirror_workspace_path>**：插入 workspace 路径的地址。
  - **<mirror_registry_url>** : 插入从中删除镜像的远程容器 registry 的 URL 或地址。

输出示例

```
$ oc mirror --config /tmp/isc_dryrun.yaml file://<oc_mirror_workspace_path> --dry-run --v2

[INFO]   : :warning:  --v2 flag identified, flow redirected to the oc-mirror v2 version. This is Tech Preview, it is still under development and it is not production ready.
[INFO]   : :wave: Hello, welcome to oc-mirror
[INFO]   : :gear:  setting up the environment for you...
[INFO]   : :twisted_rightwards_arrows: workflow mode: mirrorToDisk
[INFO]   : :sleuth_or_spy:  going to discover the necessary images...
[INFO]   : :mag: collecting release images...
[INFO]   : :mag: collecting operator images...
[INFO]   : :mag: collecting additional images...
[WARN]   : :warning:  54/54 images necessary for mirroring are not available in the cache.
[WARN]   : List of missing images in : CLID-19/working-dir/dry-run/missing.txt.
please re-run the mirror to disk process
[INFO]   : :page_facing_up: list of all images for mirroring in : CLID-19/working-dir/dry-run/mapping.txt
[INFO]   : mirror time     : 9.641091076s
[INFO]   : :wave: Goodbye, thank you for using oc-mirror
```

验证

1. 进入生成的工作区目录：

```
$ cd <oc_mirror_workspace_path>
```

2. 检查生成的 **mapping.txt** 和 **missing.txt** 文件。这些文件包含将要镜像的所有镜像的列表。

4.5.9. enclave 支持的好处

enclave 支持限制内部访问网络的特定部分。与一个 demilitarized zone (DMZ)网络不同，它允许通过防火墙界限和出站流量访问，enclaves 不跨防火墙界限。

重要

enclave Support 只是一个技术预览功能。技术预览功能不受红帽产品服务等级协议（SLA）支持，且功能可能并不完整。红帽不推荐在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。

有关红帽技术预览功能支持范围的更多信息，请参阅技术预览功能支持范围。

新的 enclave 支持功能适用于在至少一个中间断开连接的网络中保护的多个 enclave 时需要镜像 (mirror)。

enclave 支持有以下优点：

- 您可以镜像多个 enclaves 的内容，并将其集中到一个内部 registry 中。因为有些客户希望在镜像的内容上运行安全检查，所以此设置他们可以一次性运行这些检查。然后，在镜像到下游 enclaves 之前，内容会被检查。
- 您可以直接从集中式内部 registry 中镜像内容，而无需为每个 enclave 重启镜像过程。

- 您可以最小化网络阶段的数据传输，以确保 Blob 或镜像仅从一个阶段传输到另一个阶段。

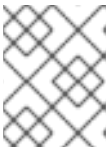
4.5.9.1. 镜像到 enclave

当镜像到 enclave 时，您必须首先将必要的镜像从一个或多个 enclaves 传送到企业中央 registry 中。

中央 registry 位于安全网络中，特别是断开连接的环境，且不会直接链接到公共互联网。但是，用户必须在一个可访问公共互联网的环境中执行 **oc mirror**。

流程

1. 在断开连接的环境中运行 oc-mirror 插件 v2 之前，请创建一个 **registry.conf** 文件。该文件的 TOML 格式如下所述：



注意

建议将文件存储在 **$HOME/.config/containers/registries.conf** 或 **/etc/containers/registries.conf** 下。

registry.conf 示例

```
[[registry]]
location="registry.redhat.io"
[[registry.mirror]]
location="<enterprise-registry.in>"

[[registry]]
location="quay.io"
[[registry.mirror]]
location="<enterprise-registry.in>"
```

2. 生成镜像存档。

   a. 要将所有 OpenShift Container Platform 内容收集到磁盘的一个归档中（**<file_path>/enterprise-content**），请运行以下命令：

   ```
   $ oc mirror --v2 -c isc.yaml file://<file_path>/enterprise-content
   ```

   isc.yaml 示例：

   ```
   apiVersion: mirror.openshift.io/v2alpha1
   kind: ImageSetConfiguration
   mirror:
     platform:
       architectures:
         - "amd64"
       channels:
         - name: stable-4.15
           minVersion: 4.15.0
           maxVersion: 4.15.3
   ```

   生成存档后，它将传送到断开连接的环境中。传输机制不是 oc-mirror 插件 v2 的一部分。企业网络管理员需要负责制定传输策略。

在某些情况下，传输是手动完成的，例如在一个没有网络连接的环境中，将磁盘从一个物理机器中拔出，并插入到另外一个系统。在其他情况下，使用安全文件传输协议 (SFTP) 或其他协议。

3. 完成存档传输后，您可以再次执行 oc-mirror 插件 v2，以便将相关存档内容镜像到 registry (在示例中是 **entrerpise_registry.in**)，如下例所示：

```
$ oc mirror --v2 -c isc.yaml --from file://<disconnected_environment_file_path>/enterprise-content docker://<enterprise_registry.in>/
```

其中：

- **--from** 指向包含存档的文件夹。它以 **file://** 开头。
- **docker://** 是镜像 (mirror) 的目的地，这是最后的参数。因为它是一个 docker registry。
- **-c** (**--config**)是一个强制参数。它可让 oc-mirror 插件 v2 最终仅将存档的子部分镜像到 registry。一个存档可能包含几个 OpenShift Container Platform 版本，但断开连接的环境或 enclave 可能只镜像几个。

4. 准备 **imageSetConfig** YAML 文件，该文件描述了要镜像到 enclave 的内容：

示例 isc-enclave.yaml

```
apiVersion: mirror.openshift.io/v2alpha1
kind: ImageSetConfiguration
mirror:
  platform:
    architectures:
      - "amd64"
    channels:
      - name: stable-4.15
        minVersion: 4.15.2
        maxVersion: 4.15.2
```

您必须在可访问断开连接的 registry 的机器上运行 oc-mirror 插件 v2。在上例中，可以访问断开连接的环境 **enterprise-registry.in**。

5. 更新图形 URL
如果您使用 **graph:true**，oc-mirror 插件 v2 会尝试访问 **cincinnati** API 端点。由于此环境断开连接，因此请务必导出环境变量 **UPDATE_URL_OVERRIDE**，以引用 OpenShift Update Service (OSUS)的 URL：

```
$ export UPDATE_URL_OVERRIDE=https://<osus.enterprise.in>/graph
```

有关在 OpenShift 集群上设置 OSUS 的更多信息，请参阅"使用 OpenShift Update Service 在断开连接的环境中更新集群"。

6. 为 enclave 从企业 registry 中生成镜像存档。
要为 **enclave1** 准备存档，用户使用针对那个 enclave 专用的 **imageSetConfiguration** 在企业断开连接的环境中执行 oc-mirror 插件 v2。这样可确保仅镜像 enclave 需要的镜像：

```
$ oc mirror --v2 -c isc-enclave.yaml
file:///disk-enc1/
```

此操作将所有 OpenShift Container Platform 内容收集到存档中，并在磁盘上生成存档。

7. 生成存档后，它将传送到 **enclave1** 网络。传输机制不是 oc-mirror 插件 v2 的责任。

8. 将内容镜像到 enclave registry
完成存档传输后，用户可以再次执行 oc-mirror 插件 v2，以便将相关存档内容镜像到 registry。

```
$ oc mirror --v2 -c isc-enclave.yaml --from file://local-disk docker://registry.enc1.in
```

enclave1 中的 OpenShift Container Platform 集群的管理员现在可以安装或升级该集群。

4.5.10. 过滤在 Operator 目录中如何工作

oc-mirror 插件 v2 通过处理 **imageSetConfig** 中的信息来选择用于镜像的捆绑包列表。

当 oc-mirror 插件 v2 选择用于镜像的捆绑包时，它不会推断 Group Version Kind (GVK) 或捆绑包依赖项，从镜像集中省略它们。相反，它严格遵循用户指令。您必须明确指定任何所需的依赖软件包及其版本。

捆绑包版本通常使用语义版本标准 (SemVer)，您可以根据版本在频道中对捆绑包进行排序。您可以选择位于 **ImageSetConfig** 中的特定范围的 buncles。

此选择算法确保与 oc-mirror 插件 v1 相比的一致性结果。但是，它不包括升级图形详情，如 **replaces**, **skip**, 和 **skipRange**。这种方法与 OLM 算法不同。它可能会比升级集群所需的更多捆绑包进行镜像 (mirror)，因为 **minVersion** 和 **maxVersion** 之间的升级路径可能比较短。

表 4.4. 使用下表查看在不同场景中包括哪些捆绑包版本：

ImageSetConfig operator 过滤	预期的捆绑包版本

场景 1
 	
`mirror:`
`operators:`
`- catalog:`
`registry.redhat.io/redhat/redhat-operator-index:v4.10`
	对于目录中每个包(1 个捆绑包)，对应于该软件包的默认频道的头版本。

场景 2
 	
`mirror:`
`operators:`
`- catalog:`
`registry.redhat.io/redhat/redhat-operator-index:v4.10`
`full: true`
	指定目录的所有频道的所有捆绑包

ImageSetConfig operator 过滤	预期的捆绑包版本
场景 3 mirror: operators: - catalog: registry.redhat.io/redhat/redhat-operator-index:v4.10 packages: - name: compliance-operator	一个捆绑包，对应于该软件包的默认频道的头版本
场景 4 mirror: operators: - catalog: registry.redhat.io/redhat/redhat-operator-index:v4.10 full: true - packages: - name: elasticsearch-operator	指定软件包的所有频道捆绑包
场景 5 mirror: operators: - catalog: registry.redhat.io/redhat/redhat-operator-index:v4.15 packages: - name: compliance-operator minVersion: 5.6.0	默认频道中的所有捆绑包（从 **minVersion** ）到频道头，该软件包不依赖于升级图表的最短路径。
场景 6 mirror: operators: - catalog: registry.redhat.io/redhat/redhat-operator-index:v4.15 packages: - name: compliance-operator maxVersion: 6.0.0	默认频道中的所有捆绑包都低于该软件包的 **maxVersion**。

ImageSetConfig operator 过滤	预期的捆绑包版本
场景 7 mirror: operators: - catalog: registry.redhat.io/redhat/redhat-operator-index:v4.15 packages: - name: compliance-operator minVersion: 5.6.0 maxVersion: 6.0.0	默认频道中的所有捆绑包，在那个软件包的 **minVersion** 和 **maxVersion** 之间。频道头不包含，即使过滤中包含多个频道。
场景 8 mirror: operators: - catalog: registry.redhat.io/redhat/redhat-operator-index:v4.15 packages: - name: compliance-operator channels - name: stable	该软件包所选频道的头捆绑包。
场景 9 mirror: operators: - catalog: registry.redhat.io/redhat/redhat-operator-index:v4.10 full: true - packages: - name: elasticsearch-operator channels: - name: 'stable-v0'	指定软件包和频道的所有捆绑包。

ImageSetConfig operator 过滤	预期的捆绑包版本
场景 10 mirror: operators: - catalog: registry.redhat.io/redhat/redhat-operator-index:v4.15 packages: - name: compliance-operator channels - name: stable - name: stable-5.5	该软件包每个所选频道的头捆绑包。
场景 11 mirror: operators: - catalog: registry.redhat.io/redhat/redhat-operator-index:v4.15 packages: - name: compliance-operator channels - name: stable minVersion: 5.6.0	在该软件包所选频道中，所有版本都以 **minVersion** 开头，直到频道头。这个场景并不能依赖升级图表中的最短路径。
场景 12 mirror: operators: - catalog: registry.redhat.io/redhat/redhat-operator-index:v4.15 packages: - name: compliance-operator channels - name: stable maxVersion: 6.0.0	在该软件包所选频道中，所有版本都最多为 **maxVersion**（不依赖于升级图中的最佳路径）。频道头不包含，即使过滤中包含多个频道。

ImageSetConfig operator 过滤	预期的捆绑包版本
场景 13 mirror: operators: - catalog: registry.redhat.io/redhat/redhat-operator-index:v4.15 packages: - name: compliance-operator channels - name: stable minVersion: 5.6.0 maxVersion: 6.0.0	在该软件包的所选频道中，**minVersion** 和 **maxVersion** 之间的所有版本都不依赖于升级图表中最短的路径。频道头不包含，即使过滤中包含多个频道。
场景 14 mirror: operators: - catalog: registry.redhat.io/redhat/redhat-operator-index:v4.14 packages: - name: aws-load-balancer-operator bundles: - name: aws-load-balancer-operator.v1.1.0 - name: 3scale-operator bundles: - name: 3scale-operator.v0.10.0-mas	只有为每个软件包指定的捆绑包才会包含在过滤中。
场景 15 mirror: operators: - catalog: registry.redhat.io/redhat/redhat-operator-index:v4.15 packages: - name: compliance-operator channels - name: stable minVersion: 5.6.0 maxVersion: 6.0.0	不要使用这个场景。通过频道过滤，以及不允许带有 **minVersion** 或 **maxVersion** 的软件包。

ImageSetConfig operator 过滤	预期的捆绑包版本
场景 16 mirror: operators: - catalog: registry.redhat.io/redhat/redhat-operator-index:v4.15 packages: - name: compliance-operator channels - name: stable minVersion: 5.6.0 maxVersion: 6.0.0	请勿使用此场景。您不能使用 **full:true** 和 **minVersion** 或 **maxVersion** 进行过滤。
场景 17 mirror: operators: - catalog: registry.redhat.io/redhat/redhat-operator-index:v4.15 full: true packages: - name: compliance-operator channels - name: stable minVersion: 5.6.0 maxVersion: 6.0.0	请勿使用此场景。您不能使用 **full:true** 和 **minVersion** 或 **maxVersion** 进行过滤。

4.5.11. oc-mirror 插件 v2 的 ImageSet 配置参数

oc-mirror 插件 v2 需要一个镜像设置配置文件，该文件定义哪些镜像要镜像(mirror)。下表列出了 **ImageSetConfiguration** 资源的可用参数。

注意

使用 **minVersion** 和 **maxVersion** 属性过滤特定 Operator 版本范围可能会导致多个频道头错误。错误信息将显示有**多个频道**头。这是因为在应用过滤器时，Operator 的更新图会被截断。

OLM 要求每个 Operator 频道都包含组成一个更新图表的版本，它只有一个端点，即 Operator 的最新版本。在应用图形的过滤器范围时，可以进入两个或多个独立图形或具有多个端点的图形。

要避免这个错误，请不要过滤 Operator 的最新版本。如果您仍然遇到错误，具体取决于 Operator，则必须增加 **maxVersion** 属性，或者 **minVersion** 属性必须减少。因为每个 Operator 图都可以不同，所以您可能需要调整这些值，直到错误解决为止。

表 4.5. **ImageSetConfiguration 参数**

参数	描述	值
apiVersion	**ImageSetConfiguration** 内容的 API 版本。	字符串示例：**mirror.openshift.io/v2alpha1**
archiveSize	镜像集中的每个存档文件的最大大小（以 GiB 为单位）。	整数示例：**4**
mirror	镜像集的配置。	对象
mirror.additionalImages	镜像集的额外镜像配置。	对象数组 Example: additionalImages: - name: registry.redhat.io/ubi8/ubi:latest
mirror.additionalImages.name	要 mirror 的镜像的标签或摘要。	字符串示例：**registry.redhat.io/ubi8/ubi:latest**
mirror.blockedImages	阻止 mirror 的镜像的完整标签、摘要或模式。	字符串数组示例：**docker.io/library/alpine**
mirror.operators	镜像集的 Operator 配置。	对象数组 Example: operators: - catalog: registry.redhat.io/redhat/redhat-operator-index:4.16 packages: - name: elasticsearch-operator minVersion: '2.4.0'

参数	描述	值
mirror.operators.catalog	包括在镜像集中的 Operator 目录。	字符串示例：**registry.redhat.io/redhat/redhat-operator-index:v4.15**
mirror.operators.full	为 **true** 时，下载完整的目录、Operator 软件包或 Operator 频道。	布尔值，默认值为 **false**。
mirror.operators.packages	Operator 软件包配置。	对象数组 Example: operators: - catalog: registry.redhat.io/redhat/redhat-operator-index:4.16 packages: - name: elasticsearch-operator minVersion: '5.2.3-31'
mirror.operators.packages.name	镜像集中要包含的 Operator 软件包名称。	字符串 例：**elasticsearch-operator**
mirror.operators.packages.channels	Operator 软件包频道配置	对象
mirror.operators.packages.channels.name	Operator 频道名称（软件包中唯一）要包括在镜像集中。	字符串示例：**fast** 或 **stable-v4.15**
mirror.operators.packages.channels.maxVersion	Operator 镜像的最高版本，在其中存在所有频道。	字符串 例：**5.2.3-31**
mirror.operators.packages.channels.minVersion	Operator 的最低版本，用于镜像存在的所有频道	字符串 例：**5.2.3-31**
mirror.operators.packages.maxVersion	Operator 最高版本，可跨所有存在的频道进行镜像。	字符串 例：**5.2.3-31**

参数	描述	值
mirror.operators.packages.minVersion	Operator 的最低版本，用于镜像存在的所有频道。	字符串 例：**5.2.3-31**
mirror.operators.packages.bundles	选择的捆绑包配置	对象数组 Example: operators: - catalog: registry.redhat.io/redhat/redhat-operator-index:4.16 packages: - name: 3scale-operator bundles: - name: 3scale-operator.v0.10.0-mas
mirror.operators.packages.bundles.name	为镜像选择的捆绑包的名称（因为它出现在目录中）。	字符串 例：**3scale-operator.v0.10.0-mas**
mirror.operators.targetCatalog	将引用的目录镜像为的替代名称和可选命名空间层次结构	字符串示例：**my-namespace/my-operator-catalog**

参数	描述	值

mirror.operators.targetCatalogSourceTemplate	用于完成 oc-mirror 插件 v2 生成的 catalogSource 自定义资源的模板的路径。	字符串
		示例：**/tmp/catalog-source_template.yaml** 模板文件
		示例：
		apiVersion: operators.coreos.com/v2alpha1
		kind: CatalogSource
		metadata:
		  name: discarded
		  namespace: openshift-marketplace
		spec:
		  image: discarded
		  sourceType: grpc
		
		updateStrategy:
		registryPoll:
		interval: 30m0s

mirror.operators.targetTag	附加到 **targetName** 或 **targetCatalog** 的替代标签。	字符串示例：**v1**
mirror.platform	镜像集的平台配置。	对象

参数	描述	值

mirror.platform.architectures	要镜像的平台发行版本有效负载的架构。	字符串数组示例：
		`architectures:`
		`- amd64`
		`- arm64`
		`- multi`
		`- ppc64le`
		`- s390x`
		默认值为 **amd64**。值 **multi** 确保镜像支持所有可用架构，无需指定单个架构。

mirror.platform.channels	镜像集的平台频道配置。	对象数组示例：
		`channels:`
		`- name: stable-4.12`
		`- name: stable-4.16`

mirror.platform.channels.full	为 **true** 时，将 **minVersion** 设置为频道中的第一个发行版本，将 **maxVersion** 设置为该频道的最后一个发行版本。	布尔值，默认为 **false**
mirror.platform.channels.name	版本频道的名称	字符串示例：**stable-4.15**
mirror.platform.channels.minVersion	要镜像引用的平台的最低版本。	字符串示例：**4.12.6**
mirror.platform.channels.maxVersion	要镜像引用的平台的最高版本。	字符串示例：**4.15.1**
mirror.platform.channels.shortestPath	切换最短的路径镜像或完整范围镜像。	布尔值，默认为 **false**
mirror.platform.channels.type	要镜像的平台的类型	字符串示例：**ocp** 或 **okd**。默认为 **ocp**。
mirror.platform.graph	指明是否将 OSUS 图表添加到镜像集中，然后发布到镜像。	布尔值，默认为 **false**

4.5.11.1. 删除 ImageSet 配置参数

要使用 oc-mirror 插件 v2，您必须有删除镜像设置配置文件，该文件定义要从镜像 registry 中删除哪些镜像。下表列出了 **DeleteImageSetConfiguration** 资源的可用参数。

表 4.6. DeleteImageSetConfiguration 参数

参数	描述	值
apiVersion	**DeleteImageSetConfiguration** 内容的 API 版本。	字符串示例：**mirror.openshift.io/v2alpha1**
delete	镜像集要删除的配置。	对象

delete.additionalImages	删除镜像集的额外镜像配置。	对象数组示例：
		`additionalImages:`
		`  - name: registry.redhat.io/ubi8/ubi:latest`

delete.additionalImages.name	要删除的镜像的标签或摘要。	字符串示例：**registry.redhat.io/ubi8/ubi:latest**

delete.operators	删除镜像集的 Operator 配置。	对象数组示例：
		`operators:`
		`  - catalog: registry.redhat.io/redhat/redhat-operator-index:{product-version}`
		`    packages:`
		`     - name: elasticsearch-operator`
		`    minVersion: '2.4.0'`

delete.operators.catalog	要在 delete 镜像集中包含的 Operator 目录。	字符串示例：**registry.redhat.io/redhat/redhat-operator-index:v4.15**

参数	描述	值
delete.operators.full	如果为 true，则删除完整目录、Operator 软件包或 Operator 频道。	布尔值，默认为 **false**
delete.operators.packages	Operator 软件包配置	对象数组示例： operators: - catalog: registry.redhat.io/redhat/redhat-operator-index: {product-version} packages: - name: elasticsearch-operator minVersion: '5.2.3-31'
delete.operators.packages.name	要在 delete 镜像集中包含的 Operator 软件包名称。	字符串 例：**elasticsearch-operator**
delete.operators.packages.channels	Operator 软件包频道配置	对象
delete.operators.packages.channels.name	Operator 频道名称（在软件包中是唯一的）包括在 delete 镜像集中。	字符串示例：**fast** 或 **stable-v4.15**
delete.operators.packages.channels.maxVersion	在所选频道中删除的 Operator 的最高版本。	字符串 例：**5.2.3-31**
delete.operators.packages.channels.minVersion	在存在的选择中删除 Operator 的最低版本。	字符串 例：**5.2.3-31**
delete.operators.packages.maxVersion	在存在的所有频道中删除 Operator 的最高版本。	字符串 例：**5.2.3-31**
delete.operators.packages.minVersion	在存在的所有频道中删除 Operator 的最低版本。	字符串 例：**5.2.3-31**

参数	描述	值
delete.operators.packages.bundles	所选捆绑包配置	对象数组 您不能为同一 Operator 选择频道和捆绑包。 Example: operators: - catalog: registry.redhat.io/redhat/redhat-operator-index:{product-version} packages: - name: 3scale-operator bundles: - name: 3scale-operator.v0.10.0-mas
delete.operators.packages.bundles.name	选择要删除的捆绑包的名称（因为它显示在目录中）	字符串 例：**3scale-operator.v0.10.0-mas**
delete.platform	镜像集的平台配置	对象
delete.platform.architectures	要删除的平台发行版本有效负载的架构。	字符串数组示例： architectures: - amd64 - arm64 - multi - ppc64le - s390x 默认值为 **amd64**

参数	描述	值
delete.platform.channels	镜像集的平台频道配置。	对象数组 Example: channels: - name: stable-4.12 - name: stable-4.16
delete.platform.channels.full	为 **true** 时，将 **minVersion** 设置为频道中的第一个发行版本，将 **maxVersion** 设置为该频道的最后一个发行版本。	布尔值，默认为 **false**
delete.platform.channels.name	版本频道的名称	字符串 例：**stable-4.15**
delete.platform.channels.minVersion	要删除的引用平台的最低版本。	字符串 例：**4.12.6**
delete.platform.channels.maxVersion	要删除的引用平台的最高版本。	字符串 例：**4.15.1**
delete.platform.channels.shortestPath	在删除最短路径并删除完整范围之间切换。	布尔值，默认为 **false**
delete.platform.channels.type	要删除的平台的类型	字符串示例：**ocp** 或 **okd**。默认为 **ocp**
delete.platform.graph	确定是否在镜像 registry 上也删除 OSUS 图形。	布尔值，默认为 **false**

4.5.12. oc-mirror 插件 v2 的命令参考

下表描述了 **oc mirror** 子命令和 oc-mirror 插件 v2 的标志：

表 4.7. oc-mirror 插件 v2 的子命令和标志：

子命令	描述
帮助	显示有关任何子命令的帮助
version	输出 oc-mirror 版本
delete	删除远程 registry 和本地缓存中的镜像。

表 4.8. oc mirror 标记

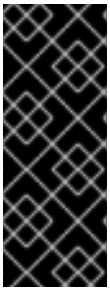
标记	描述
--authfile	显示身份验证文件的字符串路径。默认为 **${XDG_RUNTIME_DIR}/containers/auth.json**。
-c, --config <string>	指定镜像设置配置文件的路径。
--dest-tls-verify	在访问容器 registry 或守护进程时，需要 HTTPS 并验证证书。
--dry-run	打印没有 mirror 镜像的操作
--from <string>	指定通过执行 oc-mirror 插件 v2 来加载目标 registry 生成的镜像设置存档的路径。
-h,--help	显示帮助
--loglevel	显示字符串日志级别。支持的值包括 info、debug、trace、error。默认为 **info**。
-p, --port	确定 oc-mirror 插件 v2 本地存储实例使用的 HTTP 端口。默认值为 **55000**。
--max-nested-paths <int>	指定限制嵌套路径的目标 registry 的最大嵌套路径数。默认值为 **0**。
--secure-policy	默认值为 **false**。如果您设置了非默认值，命令会启用签名验证，这是签名验证的安全策略。
--since	包含自指定日期以来的所有新内容（格式：**yyyy-mm-dd**）。如果没有提供，自以前的镜像(mirror)以来的新内容会被镜像。
--src-tls-verify	在访问容器 registry 或守护进程时，需要 HTTPS 并验证证书。
--strict-archive	默认值为 **false**。如果您设置了值，命令会生成比 **imageSetConfig** 自定义资源 (CR) 中设置的 **archiveSize** 的存档。如果正在归档的文件超过 **archiveSize** (GB)，则镜像已存在。
-v, --version	显示 oc-mirror 插件 v2 的版本。
--workspace	决定字符串 oc-mirror 插件 v2 工作区，其中生成资源和内部工件。

- 配置集群以使用 oc-mirror 生成的资源

第 5 章 在 ALIBABA CLOUD 上安装

5.1. 使用 ASSISTED INSTALLER 在 ALIBABA CLOUD 上安装集群

Alibaba Cloud 为在线企业和全球企业提供广泛的云计算和数据存储服务。您可以使用 Assisted Installer 在 Alibaba Cloud 上安装 OpenShift Container Platform 集群。



重要

使用 Assisted Installer 安装 Alibaba Cloud 只是一个技术预览功能。技术预览功能不受红帽产品服务等级协议（SLA）支持，且功能可能并不完整。红帽不推荐在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。

有关红帽技术预览功能支持范围的更多信息，请参阅技术预览功能支持范围。

5.1.1. 使用 Assisted Installer 创建集群的流程概述

安装过程的主要步骤如下：

1. 使用 Assisted Installer 创建集群并下载生成的镜像。
2. 将镜像转换为 **QCOW2** 格式。如需更多信息，请参阅以下部分。
3. 将镜像上传到 Alibaba Cloud 中的对象存储服务存储桶。
4. 在 Alibaba Cloud 中将镜像导入到 Elastic Compute Service。
5. 置备 Alibaba 云资源：
    a. 在 Virtual Private Cloud (VPC) 控制台中设置网络配置。
    b. 在 Alibaba Cloud DNS 控制台中，定义域名系统。
    c. 在 Elastic Compute Service (ECS) 控制台中，调配计算实例。
6. 在 Assisted Installer 中完成主机发现。
7. 在 Alibaba Cloud 中完成网络配置。
8. 在 Assisted Installer 中完成集群配置并安装。

其他资源

- 使用辅助安装程序安装 OpenShift Container Platform

5.1.2. 将发现镜像转换为 QCOW2 格式

在将生成的 ISO 导入到 Alibaba Cloud 之前，将生成的 ISO 转换为 **QCOW2** 格式。

先决条件

- 您已创建了集群，并在 Assisted Installer 中下载发现镜像。
- 您可以访问集群外的 Linux 机器，如您的桌面机器。

流程

1. 打开 Linux 计算机上的命令行界面。
2. 运行以下命令验证系统是否启用了虚拟化标记：

```
$ grep -e lm -e svm -e vmx /proc/cpuinfo
```

3. 运行以下命令，在 RHEL 或 Fedora 机器上安装 **qemu-img** 软件包：

```
$ sudo dnf install -y qemu-img
```

注意

如果您的系统使用 **APT** 软件包管理器，使用名称 **qemu-utils** 安装软件包。

4. 运行以下命令将镜像转换为 **QCOW2** ：

```
$ qemu-img convert -O qcow2 ${CLUSTER_NAME}.iso ${CLUSTER_NAME}.qcow2
```

第 6 章 在 AWS 上安装

6.1. 安装方法

您可以使用安装程序置备的基础架构在 Amazon Web Services (AWS) 上安装 OpenShift Container Platform。默认安装类型使用安装程序置备的基础架构，安装程序会在其中为集群置备底层基础架构。您还可以在您置备的基础架构上安装 OpenShift Container Platform。如果不使用安装程序置备的基础架构，您必须自己管理和维护集群资源。您还可以在单一节点上安装 OpenShift Container Platform，这是适用于边缘计算环境的专用安装方法。

6.1.1. 在安装程序置备的基础架构上安装集群

您可以使用以下方法之一在 OpenShift Container Platform 安装程序置备的 AWS 基础架构上安装集群：

- **在 AWS 上快速安装集群**：您可以在由 OpenShift Container Platform 安装程序置备的 AWS 基础架构上安装 OpenShift Container Platform。您可以使用默认配置选项快速安装集群。
- **在 WS 上安装自定义集群**：您可以在安装程序置备的 AWS 基础架构上安装自定义集群。安装程序允许在安装阶段应用一些自定义。其它自定义选项可在安装后使用。
- **使用自定义网络在 AWS 上安装集群**：您可以在安装过程中自定义 OpenShift Container Platform 网络配置，以便集群可以与现有的 IP 地址分配共存,并遵循您的网络要求。
- **在受限网络中的 AWS 上安装集群**：您可以使用安装发行内容的内部镜像在 AWS 上安装 OpenShift Container Platform。您可以使用此方法安装不需要活跃互联网连接的集群来获取软件组件。
- **在现有 Virtual Private Cloud 上安装集群**：您可以在现有 AWS Virtual Private Cloud（VPC）上安装 OpenShift Container Platform。如果您按照公司的说明设置了限制，可以使用这个安装方法，例如在创建新帐户或基础架构时的限制。
- **在现有 VPC 上安装私有集群**：您可以在现有 AWS VPC 上安装私有集群。您可以使用此方法将 OpenShift Container Platform 部署到互联网中不可见的内部网络中。
- **在 WS 上将集群安装到一个政府或机密区域**：OpenShift Container Platform 可以部署到 AWS 区域，这些区域是为需要运行云中敏感工作负载的美国政府机构、州和本地级别的政府机构、企业和其他需要运行敏感工作负载的美国客户准备的。

6.1.2. 在用户置备的基础架构上安装集群

您可以使用以下方法之一在您置备的 AWS 基础架构上安装集群：

- **在您提供的 AWS 基础架构上安装集群**：您可以在您提供的 AWS 基础架构上安装 OpenShift Container Platform。您可以使用提供的 CloudFormation 模板来创建 AWS 资源堆栈，这些资源代表 OpenShift Container Platform 安装所需的每个组件。
- 在**带有用户置备的受限网络中的 AWS 上安装集群**：您可以使用安装发行内容的内部镜像在 AWS 基础架构上安装 OpenShift Container Platform。您可以使用此方法安装不需要活跃互联网连接的集群来获取软件组件。您还可以使用此安装方法确保集群只使用满足您机构对外部内容控制的容器镜像。虽然您可以使用镜像内容安装 OpenShift Container Platform，但您的集群仍需要访问互联网才能使用 AWS API。

6.1.3. 在单一节点上安装集群

在单一节点上安装 OpenShift Container Platform 可降低高可用性和大型集群的一些要求。但是，您必须

满足在单一节点上安装的要求，以及在云供应商上安装单节点 OpenShift 的额外要求。在满足了在单一节点安装要求后，请按照在 AWS 中安装自定义的集群来安装集群。在单一节点上安装 OpenShift Container Platform 集群时，手动安装单节点 OpenShift 部分包含一个 **install-config.yaml** 示例文件。

6.1.4. 其他资源

- 安装过程

6.2. 配置 AWS 帐户

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

6.2.1. 配置路由 53（Route 53）

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

流程

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

注意

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

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

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

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

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

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

6.2.1.1. AWS Route 53 的 Ingress Operator 端点配置

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

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

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

重要

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

Route 53 配置示例

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

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

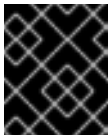
2 只有 US-West 区域有标记端点。如果集群位于另一个区域，则省略此参数。

6.2.2. AWS 帐户限值

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

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

组件	默认可用的集群数	默认 AWS 限值	描述

组件	默认可用的集群数	默认 AWS 限值	描述
实例限值	可变	可变	默认情况下，每个集群创建以下实例： • 一台 Bootstrap 机器，在安装后删除 • 三个 control plane 节点 • 三个 worker 节点 这些实例类型数量在新帐户的默认限值之内。若要部署更多 worker 节点、启用自动扩展、部署大型工作负载或使用不同的实例类型，请检查您的帐户限制，以确保集群可以部署您需要的机器。 在大多数区域中，worker 机器使用 **m6i.large** 实例，bootstrap 和 control plane 机器使用 **m6i.xlarge** 实例。在一些区域，包括所有不支持这些实例类型的区域，则使用 **m5.large** 和 **m5.xlarge** 实例。
弹性 IP (EIP)	0 到 1	每个帐户 5 个 EIP	要在高可用性配置中置备集群，安装程序将为区域中的每个可用区创建一个公共和专用子网。每个专用子网都需要 NAT 网关，每个 NAT 网关需要单独的弹性 IP。查看 AWS 区域图来确定每个区域有多少个可用区。要利用默认高可用性，请在至少含有三个可用区的区域安装集群。要在有超过五个可用区的区域安装集群，您必须提高 EIP 限值。  **重要** 要使用 **us-east-1** 区域，必须提高您帐户的 EIP 限值。
虚拟私有云 (VPC)	5	每个区域 5 个 VPC	每个集群创建自己的 VPC。
弹性负载均衡 (ELB/NLB)	3	每个区域 20 个	在默认情况下，每个集群为 master API 服务器创建一个内部和外部网络负载均衡器，并为路由器创建一个典型的弹性负载均衡器。使用类型 **LoadBalancer** 部署更多 Kubernetes **Service** 对象将创建额外的负载均衡器。
NAT 网关	5	每个可用区 5 个	集群在每个可用区中部署一个 NAT 网关。
弹性网络接口 (ENI)	至少 12 个	每个区域 350 个	默认安装创建 21 个 ENI，并为区域中的每个可用区创建一个 ENI。例如，**us-east-1** 区域包含六个可用区，因此在该区部署的集群将使用 27 个 ENI。查看 AWS 区域图来确定每个区域有多少个可用区。 为额外的机器和 ELB 负载均衡器（它们由集群的使用以及部署的工作负载创建）创建额外的 ENI。

组件	默认可用的集群数	默认 AWS 限值	描述
VPC 网关	20	每个帐户 20 个	每个集群创建一个 VPC 网关来访问 S3。
S3 存储桶	99	每个帐户有 100 个存储桶	因为安装过程会创建一个临时存储桶，并且每个集群中的 registry 组件会创建一个存储桶，所以您只能为每个 AWS 帐户创建 99 个 OpenShift Container Platform 集群。
安全组	250	每个帐户 2,500 个	每个集群创建 10 个不同的安全组。

6.2.3. IAM 用户所需的 AWS 权限

注意

您的 IAM 用户必须在区域 **us-east-1** 中有权限 **tag:GetResources** 来删除基本集群资源。作为 AWS API 的要求的一部分，OpenShift Container Platform 安装程序在此区域中执行各种操作。

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

例 6.1. 安装所需的 EC2 权限

- **ec2:AttachNetworkInterface**
- **ec2:AuthorizeSecurityGroupEgress**
- **ec2:AuthorizeSecurityGroupIngress**
- **ec2:CopyImage**
- **ec2:CreateNetworkInterface**
- **ec2:CreateSecurityGroup**
- **ec2:CreateTags**
- **ec2:CreateVolume**
- **ec2:DeleteSecurityGroup**
- **ec2:DeleteSnapshot**
- **ec2:DeleteTags**
- **ec2:DeregisterImage**
- **ec2:DescribeAccountAttributes**
- **ec2:DescribeAddresses**

- **ec2:DescribeAvailabilityZones**
- **ec2:DescribeDhcpOptions**
- **ec2:DescribeImages**
- **ec2:DescribeInstanceAttribute**
- **ec2:DescribeInstanceCreditSpecifications**
- **ec2:DescribeInstances**
- **ec2:DescribeInstanceTypes**
- **ec2:DescribeInternetGateways**
- **ec2:DescribeKeyPairs**
- **ec2:DescribeNatGateways**
- **ec2:DescribeNetworkAcls**
- **ec2:DescribeNetworkInterfaces**
- **ec2:DescribePrefixLists**
- **ec2:DescribePublicIpv4Pools** （仅在 **install-config.yaml**中指定 **publicIpv4Pool** 时才需要）
- **ec2:DescribeRegions**
- **ec2:DescribeRouteTables**
- **ec2:DescribeSecurityGroupRules**
- **ec2:DescribeSecurityGroups**
- **ec2:DescribeSubnets**
- **ec2:DescribeTags**
- **ec2:DescribeVolumes**
- **ec2:DescribeVpcAttribute**
- **ec2:DescribeVpcClassicLink**
- **ec2:DescribeVpcClassicLinkDnsSupport**
- **ec2:DescribeVpcEndpoints**
- **ec2:DescribeVpcs**
- **ec2:DisassociateAddress** （仅在 **install-config.yaml**中指定 **publicIpv4Pool** 时才需要）
- **ec2:GetEbsDefaultKmsKeyId**
- **ec2:ModifyInstanceAttribute**

- **ec2:ModifyNetworkInterfaceAttribute**
- **ec2:RevokeSecurityGroupEgress**
- **ec2:RevokeSecurityGroupIngress**
- **ec2:RunInstances**
- **ec2:TerminateInstances**

例 6.2. 安装过程中创建网络资源所需的权限

- **ec2:AllocateAddress**
- **ec2:AssociateAddress**
- **ec2:AssociateDhcpOptions**
- **ec2:AssociateRouteTable**
- **ec2:AttachInternetGateway**
- **ec2:CreateDhcpOptions**
- **ec2:CreateInternetGateway**
- **ec2:CreateNatGateway**
- **ec2:CreateRoute**
- **ec2:CreateRouteTable**
- **ec2:CreateSubnet**
- **ec2:CreateVpc**
- **ec2:CreateVpcEndpoint**
- **ec2:ModifySubnetAttribute**
- **ec2:ModifyVpcAttribute**

注意

如果您使用现有的 Virtual Private Cloud (VPC)，您的帐户不需要这些权限来创建网络资源。

例 6.3. 安装所需的 Elastic Load Balancing 权限(ELB)

- **elasticloadbalancing:AddTags**
- **elasticloadbalancing:ApplySecurityGroupsToLoadBalancer**
- **elasticloadbalancing:AttachLoadBalancerToSubnets**

- **elasticloadbalancing:ConfigureHealthCheck**
- **elasticloadbalancing:CreateListener**
- **elasticloadbalancing:CreateLoadBalancer**
- **elasticloadbalancing:CreateLoadBalancerListeners**
- **elasticloadbalancing:CreateTargetGroup**
- **elasticloadbalancing:DeleteLoadBalancer**
- **elasticloadbalancing:DeregisterInstancesFromLoadBalancer**
- **elasticloadbalancing:DeregisterTargets**
- **elasticloadbalancing:DescribeInstanceHealth**
- **elasticloadbalancing:DescribeListeners**
- **elasticloadbalancing:DescribeLoadBalancerAttributes**
- **elasticloadbalancing:DescribeLoadBalancers**
- **elasticloadbalancing:DescribeTags**
- **elasticloadbalancing:DescribeTargetGroupAttributes**
- **elasticloadbalancing:DescribeTargetHealth**
- **elasticloadbalancing:ModifyLoadBalancerAttributes**
- **elasticloadbalancing:ModifyTargetGroup**
- **elasticloadbalancing:ModifyTargetGroupAttributes**
- **elasticloadbalancing:RegisterInstancesWithLoadBalancer**
- **elasticloadbalancing:RegisterTargets**
- **elasticloadbalancing:SetLoadBalancerPoliciesOfListener**
- **elasticloadbalancing:SetSecurityGroups**

重要

OpenShift Container Platform 使用 ELB 和 ELBv2 API 服务来置备负载均衡器。权限列表显示这两个服务所需的权限。AWS web 控制台中存在一个已知问题，其中这两个服务都使用相同的 **elasticloadbalancing** 操作前缀，但无法识别相同的操作。您可以忽略有关服务没有识别某些 **elasticloadbalancing** 操作的警告。

例 6.4. 安装所需的 IAM 权限

- **iam:AddRoleToInstanceProfile**

- **iam:CreateInstanceProfile**
- **iam:CreateRole**
- **iam:DeleteInstanceProfile**
- **iam:DeleteRole**
- **iam:DeleteRolePolicy**
- **iam:GetInstanceProfile**
- **iam:GetRole**
- **iam:GetRolePolicy**
- **iam:GetUser**
- **iam:ListInstanceProfilesForRole**
- **iam:ListRoles**
- **iam:ListUsers**
- **iam:PassRole**
- **iam:PutRolePolicy**
- **iam:RemoveRoleFromInstanceProfile**
- **iam:SimulatePrincipalPolicy**
- **iam:TagInstanceProfile**
- **iam:TagRole**

注意

如果您还没有在 AWS 帐户中创建负载均衡器，IAM 用户还需要 **iam:CreateServiceLinkedRole** 权限。

例 6.5. 安装所需的 Route 53 权限

- **route53:ChangeResourceRecordSets**
- **route53:ChangeTagsForResource**
- **route53:CreateHostedZone**
- **route53:DeleteHostedZone**
- **route53:GetChange**
- **route53:GetHostedZone**
- **route53:ListHostedZones**

- **route53:ListHostedZonesByName**
- **route53:ListResourceRecordSets**
- **route53:ListTagsForResource**
- **route53:UpdateHostedZoneComment**

例 6.6. 安装所需的 Amazon Simple Storage Service (S3) 权限

- **s3:CreateBucket**
- **s3:DeleteBucket**
- **s3:GetAccelerateConfiguration**
- **s3:GetBucketAcl**
- **s3:GetBucketCors**
- **s3:GetBucketLocation**
- **s3:GetBucketLogging**
- **s3:GetBucketObjectLockConfiguration**
- **s3:GetBucketPolicy**
- **s3:GetBucketRequestPayment**
- **s3:GetBucketTagging**
- **s3:GetBucketVersioning**
- **s3:GetBucketWebsite**
- **s3:GetEncryptionConfiguration**
- **s3:GetLifecycleConfiguration**
- **s3:GetReplicationConfiguration**
- **s3:ListBucket**
- **s3:PutBucketAcl**
- **s3:PutBucketPolicy**
- **s3:PutBucketTagging**
- **s3:PutEncryptionConfiguration**

例 6.7. 集群 Operators 所需的 S3 权限

- **s3:DeleteObject**

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

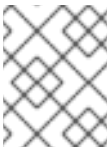
例 6.8. 删除基本集群资源所需的权限

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

例 6.9. 删除网络资源所需的权限

- **ec2:DeleteDhcpOptions**
- **ec2:DeleteInternetGateway**
- **ec2:DeleteNatGateway**

- **ec2:DeleteRoute**
- **ec2:DeleteRouteTable**
- **ec2:DeleteSubnet**
- **ec2:DeleteVpc**
- **ec2:DeleteVpcEndpoints**
- **ec2:DetachInternetGateway**
- **ec2:DisassociateRouteTable**
- **ec2:ReleaseAddress**
- **ec2:ReplaceRouteTableAssociation**



注意

如果您使用现有的 VPC，您的帐户不需要这些权限来删除网络资源。您的帐户只需要有 **tag:UntagResources** 权限就能删除网络资源。

例 6.10. 使用自定义密钥管理服务 (KMS) 密钥安装集群的可选权限

- **kms:CreateGrant**
- **kms:Decrypt**
- **kms:DescribeKey**
- **kms:Encrypt**
- **kms:GenerateDataKey**
- **kms:GenerateDataKeyWithoutPlainText**
- **kms:ListGrants**
- **kms:RevokeGrant**

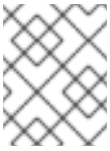
例 6.11. 使用共享实例角色删除集群所需的权限

- **iam:UntagRole**

例 6.12. 创建清单所需的额外 IAM 和 S3 权限

- **iam:GetUserPolicy**
- **iam:ListAccessKeys**
- **iam:PutUserPolicy**

- **iam:TagUser**
- **s3:AbortMultipartUpload**
- **s3:GetBucketPublicAccessBlock**
- **s3:ListBucket**
- **s3:ListBucketMultipartUploads**
- **s3:PutBucketPublicAccessBlock**
- **s3:PutLifecycleConfiguration**



注意

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

例 6.13. 实例的可选权限和安装配额检查

- **ec2:DescribeInstanceTypeOfferings**
- **servicequotas:ListAWSDefaultServiceQuotas**

例 6.14. 在共享 VPC 上安装集群时集群所有者帐户的可选权限

- **sts:AssumeRole**

例 6.15. 为安装启用您自己的公共 IPv4 地址 (BYOIP) 功能所需的权限

- **ec2:DescribePublicIpv4Pools**
- **ec2:DisassociateAddress**

6.2.4. 创建 IAM 用户

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

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

流程

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

注意

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

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

重要

在部署集群时，您无法在使用多因素验证设备来验证 AWS 的同时使用您生成的临时会话令牌。集群将继续使用您当前的 AWS 凭证为集群的整个生命周期创建 AWS 资源，因此您必须使用基于密钥的长期凭证。

6.2.5. IAM 策略和 AWS 身份验证

默认情况下，安装程序会为集群操作所需的权限为 bootstrap、control plane 和计算实例创建实例配置集。

但是，您可以创建自己的 IAM 角色，并将其指定为安装过程的一部分。您可能需要指定自己的角色来部署集群或在安装后管理集群。例如：

- 机构的安全策略要求您使用更严格的权限集来安装集群。
- 安装后，集群使用需要访问其他服务的 Operator 配置。

如果选择指定自己的 IAM 角色，您可以执行以下步骤：

- 从默认策略开始，并根据需要进行调整。如需更多信息，请参阅"IAM 实例配置集的默认权限"。
- 使用 AWS Identity and Access Management Access Analyzer (IAM Access Analyzer) 创建一个基于集群的活动的策略模板。如需更多信息，请参阅"使用 AWS IAM 分析器来创建策略模板"。

6.2.5.1. IAM 实例配置集的默认权限

默认情况下，安装程序会为集群操作所需的权限为 bootstrap、control plane 和 worker 实例创建 IAM 实例配置集。

以下列表指定 control plane 和计算机器的默认权限：

例 6.16. control plane 实例配置集的默认 IAM 角色权限

- **ec2:AttachVolume**
- **ec2:AuthorizeSecurityGroupIngress**
- **ec2:CreateSecurityGroup**
- **ec2:CreateTags**
- **ec2:CreateVolume**

- **ec2:DeleteSecurityGroup**
- **ec2:DeleteVolume**
- **ec2:Describe***
- **ec2:DetachVolume**
- **ec2:ModifyInstanceAttribute**
- **ec2:ModifyVolume**
- **ec2:RevokeSecurityGroupIngress**
- **elasticloadbalancing:AddTags**
- **elasticloadbalancing:AttachLoadBalancerToSubnets**
- **elasticloadbalancing:ApplySecurityGroupsToLoadBalancer**
- **elasticloadbalancing:CreateListener**
- **elasticloadbalancing:CreateLoadBalancer**
- **elasticloadbalancing:CreateLoadBalancerPolicy**
- **elasticloadbalancing:CreateLoadBalancerListeners**
- **elasticloadbalancing:CreateTargetGroup**
- **elasticloadbalancing:ConfigureHealthCheck**
- **elasticloadbalancing:DeleteListener**
- **elasticloadbalancing:DeleteLoadBalancer**
- **elasticloadbalancing:DeleteLoadBalancerListeners**
- **elasticloadbalancing:DeleteTargetGroup**
- **elasticloadbalancing:DeregisterInstancesFromLoadBalancer**
- **elasticloadbalancing:DeregisterTargets**
- **elasticloadbalancing:Describe***
- **elasticloadbalancing:DetachLoadBalancerFromSubnets**
- **elasticloadbalancing:ModifyListener**
- **elasticloadbalancing:ModifyLoadBalancerAttributes**
- **elasticloadbalancing:ModifyTargetGroup**
- **elasticloadbalancing:ModifyTargetGroupAttributes**
- **elasticloadbalancing:RegisterInstancesWithLoadBalancer**

- **elasticloadbalancing:RegisterTargets**
- **elasticloadbalancing:SetLoadBalancerPoliciesForBackendServer**
- **elasticloadbalancing:SetLoadBalancerPoliciesOfListener**
- **kms:DescribeKey**

例 6.17. 计算实例配置集的默认 IAM 角色权限

- **ec2:DescribeInstances**
- **ec2:DescribeRegions**

6.2.5.2. 指定现有的 IAM 角色

您可以使用 **install-config.yaml** 文件为 control plane 和计算实例指定现有 IAM 角色，而不是让安装程序创建具有默认权限的 IAM 实例配置集。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。

流程

1. 使用计算机器的现有角色更新 **compute.platform.aws.iamRole**。

   带有计算实例的 IAM 角色的 install-config.yaml 文件示例

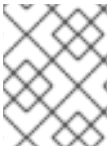
   ```
   compute:
   - hyperthreading: Enabled
     name: worker
     platform:
       aws:
         iamRole: ExampleRole
   ```

2. 使用 control plane 机器的现有角色更新 **controlPlane.platform.aws.iamRole**。

   带有 control plane 实例的 IAM 角色的 install-config.yaml 文件示例

   ```
   controlPlane:
     hyperthreading: Enabled
     name: master
     platform:
       aws:
         iamRole: ExampleRole
   ```

3. 保存文件并在安装 OpenShift Container Platform 集群时引用。



注意

要在安装集群后更改或更新 IAM 帐户，请参阅 RHOCP 4 AWS cloud-credentials access key is expired (Red Hat Knowledgebase)。

其他资源

- 部署集群

6.2.5.3. 使用 AWS IAM 分析器创建策略模板

control plane 和计算实例配置集需要的最小权限集取决于如何为每日操作配置集群。

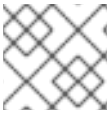
要确定集群实例需要哪些权限的一种方法是使用 AWS Identity and Access Management Access Analyzer (IAM Access Analyzer) 创建策略模板：

- 策略模板包含集群在指定时间段内使用的权限。
- 然后，您可以使用模板来创建具有细粒度权限的策略。

流程

整个过程可以是：

1. 确保启用了 CloudTrail。CloudTrail 记录 AWS 帐户中的所有操作和事件，包括创建策略模板所需的 API 调用。如需更多信息，请参阅 AWS 文档的使用 CloudTrail 。
2. 为 control plane 实例创建实例配置集，并为计算实例创建一个实例配置集。确保为每个角色分配一个 permissive 策略，如 PowerUserAccess。如需更多信息，请参阅 AWS 文档的创建实例配置集角色。
3. 在开发环境中安装集群，并根据需要进行配置。务必在生产环境中部署集群托管的所有应用程序。
4. 全面测试集群。测试集群可确保记录所有必需的 API 调用。
5. 使用 IAM Access Analyzer 为每个实例配置集创建策略模板。如需更多信息，请参阅 AWS 文档基于 CloudTrail 日志生成策略。
6. 创建并为每个实例配置文件添加一个精细的策略。
7. 从每个实例配置集中删除 permissive 策略。
8. 使用现有实例配置集和新策略部署生产集群。



注意

您可以在策略中添加 IAM 条件，使其更严格且符合您的机构安全要求。

6.2.6. 支持的 AWS Marketplace 区域

使用 AWS Marketplace 镜像安装 OpenShift Container Platform 集群可供购买北美提供的客户提供。

虽然优惠必须在北美购买，但您可以将集群部署到以下任意一种支持：

- 公开

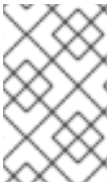
- GovCloud

注意

AWS secret 区域或中国区域不支持使用 AWS Marketplace 镜像部署 OpenShift Container Platform 集群。

6.2.7. 支持的 AWS 区域

您可以将 OpenShift Container Platform 集群部署到以下区域。



注意

您的 IAM 用户必须在区域 **us-east-1** 中有权限 **tag:GetResources** 来删除基本集群资源。作为 AWS API 的要求的一部分，OpenShift Container Platform 安装程序在此区域中执行各种操作。

6.2.7.1. AWS 公共区域

支持以下 AWS 公共区域：

- **af-south-1** (Cape Town)
- **ap-east-1** (Hong Kong)
- **ap-northeast-1** (Tokyo)
- **ap-northeast-2** (Seoul)
- **ap-northeast-3** (Osaka)
- **ap-south-1** (Mumbai)
- **ap-south-2** (Hyderabad)
- **ap-southeast-1** (Singapore)
- **ap-southeast-2** (Sydney)
- **ap-southeast-3** (Jakarta)
- **ap-southeast-4** (Melbourne)
- **ca-central-1** (Central)
- **ca-west-1** (Calgary)
- **eu-central-1** (Frankfurt)
- **eu-central-2** (Zurich)
- **eu-north-1** (Stockholm)
- **eu-south-1** (Milan)
- **eu-south-2** (Spain)

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

6.2.7.2. AWS GovCloud 区域

支持以下 AWS GovCloud 区域：

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

6.2.7.3. AWS SC2S 和 C2S secret 区域

支持以下 AWS secret 区域：

- **us-isob-east-1** Secret Commercial Cloud Services (SC2S)
- **us-iso-east-1** Commercial Cloud Services (C2S)

6.2.7.4. AWS 中国区域

支持以下 AWS 中国区域：

- **cn-north-1** (Beijing)
- **cn-northwest-1** (Ningxia)

6.2.8. 后续步骤

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

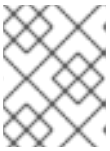
- 使用 CloudFormation 模板在 AWS 中用户置备的基础架构上安装集群

6.3. 安装程序置备的基础架构

6.3.1. 准备在 AWS 上安装集群

您可以通过完成以下步骤准备在 AWS 上安装 OpenShift Container Platform 集群：

- 为集群验证互联网连接。
- 配置 AWS 帐户.
- 下载安装程序。



注意

如果您要在断开连接的环境中安装，您可以从镜像内容中提取安装程序。如需更多信息，请参阅为断开连接的安装镜像镜像。

- 安装 OpenShift CLI (**oc**)。

注意

如果要在断开连接的环境中安装，请将 **oc** 安装到镜像主机上。

- 生成 SSH 密钥对。您可以在部署后使用此密钥对在 OpenShift Container Platform 集群的节点上进行身份验证。
- 如果环境中无法访问云身份和访问管理(IAM) API，或者不想将管理员级别的凭证 secret 存储在 **kube-system** 命名空间中，请参阅为 AWS 手动创建长期凭证，或将 AWS 集群配置为使用 Amazon Web Services Security Token Service (AWS STS)的短期凭证。

6.3.1.1. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来获得用来安装集群的镜像。

您必须具有以下互联网访问权限：

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

6.3.1.2. 获取安装程序

在安装 OpenShift Container Platform 之前，将安装文件下载到镜像主机上。

先决条件

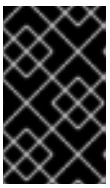
- 您有一台运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB。

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐户，请使用您的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入到安装类型的页面，下载与您的主机操作系统和架构对应的安装程序，并将该文件放在您要存储安装配置文件的目录中。

重要

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



重要

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

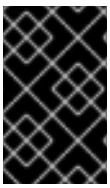
4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

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

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。此 pull secret 允许您与所含授权机构提供的服务进行身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

6.3.1.3. 安装 OpenShift CLI

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



重要

如果安装了旧版本的 **oc**，则可能无法使用 OpenShift Container Platform 4.16 中的所有命令。下载并安装新版本的 **oc**。如果要在断开连接的环境中更新集群，请安装您要升级到的 **oc** 版本。

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

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **产品变体** 下拉列表中选择架构。
3. 从 **版本** 下拉列表中选择适当的版本。
4. 点 **OpenShift v4.16 Linux Client** 条目旁的 **Download Now** 来保存文件。
5. 解包存档：

```
$ tar xvf <file>
```

6. 将 **oc** 二进制文件放到 **PATH 中的目录中**。
要查看您的 **PATH**，请执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

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

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 Windows Client** 条目旁的 **Download Now** 来保存文件。
4. 使用 ZIP 程序解压存档。
5. 将 **oc** 二进制文件移到 **PATH 中的目录中**。
要查看您的 **PATH**，请打开命令提示并执行以下命令：

```
C:\> path
```

验证

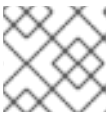
- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

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

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

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 macOS Client** 条目旁的 **Download Now** 来保存文件。



注意

对于 macOS arm64，请选择 **OpenShift v4.16 macOS arm64 Client** 条目。

4. 解包和解压存档。

5. 将 **oc** 二进制文件移到 PATH 的目录中。
要查看您的 **PATH**，请打开终端并执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

6.3.1.4. 为集群节点 SSH 访问生成密钥对

在 OpenShift Container Platform 安装过程中，您可以为安装程序提供 SSH 公钥。密钥通过它们的 Ignition 配置文件传递给 Red Hat Enterprise Linux CoreOS(RHCOS)节点，用于验证对节点的 SSH 访问。密钥添加到每个节点上 **core** 用户的 **~/.ssh/authorized_keys** 列表中，这将启用免密码身份验证。

将密钥传递给节点后，您可以使用密钥对作为用户 **核心** 通过 SSH 连接到 RHCOS 节点。若要通过 SSH 访问节点，必须由 SSH 为您的本地用户管理私钥身份。

如果要通过 SSH 连接到集群节点来执行安装调试或灾难恢复，则必须在安装过程中提供 SSH 公钥。**./openshift-install gather** 命令还需要在集群节点上设置 SSH 公钥。

重要

不要在生产环境中跳过这个过程，在生产环境中需要灾难恢复和调试。

注意

您必须使用本地密钥，而不是使用特定平台方法配置 的密钥，如 AWS 密钥对。

流程

1. 如果您在本地计算机上没有可用于在集群节点上进行身份验证的现有 SSH 密钥对，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

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

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_ed25519**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

注意

如果您计划在 **x86_64**、**ppc64le** 和 **s390x** 架构上安装使用 RHEL 加密库（这些加密库已提交给 NIST 用于 FIPS 140-2/140-3 验证）的 OpenShift Container Platform 集群，则不要创建使用 **ed25519** 算法的密钥。相反，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

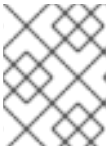
2. 查看公共 SSH 密钥：

```
$ cat <path>/<file_name>.pub
```

例如，运行以下命令来查看 ~/**.ssh/id_ed25519.pub** 公钥：

```
$ cat ~/.ssh/id_ed25519.pub
```

3. 将 SSH 私钥身份添加到本地用户的 SSH 代理（如果尚未添加）。在集群节点上，或者要使用 **./openshift-install gather** 命令，需要对该密钥进行 SSH 代理管理，才能在集群节点上进行免密码 SSH 身份验证。



注意

在某些发行版中，自动管理默认 SSH 私钥身份，如 ~/**.ssh/id_rsa** 和 ~/**.ssh/id_dsa**。

a. 如果 **ssh-agent** 进程尚未为您的本地用户运行，请将其作为后台任务启动：

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

输出示例

```
Agent pid 31874
```

注意

如果集群处于 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

4. 将 SSH 私钥添加到 **ssh-agent** ：

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

1 指定 SSH 私钥的路径和文件名，如 ~/**.ssh/id_ed25519.pub**

输出示例

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

后续步骤

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

6.3.1.5. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

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

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于 远程健康监控

6.3.2. 在 AWS 上安装集群

在 OpenShift Container Platform 版本 4.16 中，您可以使用默认配置选项在 Amazon Web Services (AWS) 上安装集群。

6.3.2.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读选择集群安装方法并为用户准备它的文档。
- 已将 AWS 帐户配置为托管集群。

重要

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

- 如果使用防火墙，则会 将其配置为允许集群需要访问的站点。

6.3.2.2. 部署集群

您可以在兼容云平台上安装 OpenShift Container Platform。

重要

在初始安装过程中，您只能运行安装程序的 **create cluster** 命令一次。

先决条件

- 您已使用托管集群的云平台配置了帐户。
- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。
- 已确认主机上的云供应商帐户具有部署集群的正确权限。权限不正确的帐户会导致安装过程失败，并显示包括缺失权限的错误消息。

流程

1. 进入包含安装程序的目录并初始化集群部署：

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

1 对于 **<installation_directory>**，请指定要存储安装程序创建的文件的目录名称。

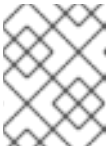
2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不是 **info**。

在指定目录时：

- 验证该目录是否具有**执行**权限。在安装目录中运行 Terraform 二进制文件需要这个权限。
- 使用空目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

2. 在提示符处提供值：

a. 可选：选择用于访问集群机器的 SSH 密钥。



注意

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

b. 选择 **aws** 作为目标平台。

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

注意

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

d. 选择要将集群部署到的 AWS 区域。

e. 选择您为集群配置的 Route 53 服务的基域。

f. 为集群输入描述性名称。

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

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

注意

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

验证

当集群部署成功完成时：

- 终端会显示用于访问集群的说明，包括指向 Web 控制台和 **kubeadmin** 用户的凭证的链接。
- 凭证信息还会输出到 **<installation_directory>/.openshift_install.log**.

重要

不要删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

输出示例

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

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

其他资源

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

6.3.2.3. 使用 CLI 登录集群

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

先决条件

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

流程

1. 导出 **kubeadmin** 凭证：

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

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

2. 验证您可以使用导出的配置成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

6.3.2.4. 使用 Web 控制台登录到集群

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

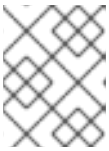
先决条件

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

流程

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

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



注意

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

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

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

注意

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

输出示例

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

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

其他资源

- 如需有关 访问和了解 OpenShift Container Platform Web 控制台的更多详情，请参阅 访问 Web 控制台。

6.3.2.5. 后续步骤

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

6.3.3. 使用自定义在 AWS 上安装集群

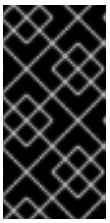
在 OpenShift Container Platform 版本 4.16 中，您可以在安装程序在 Amazon Web Services(AWS)中置备的基础架构上安装自定义集群。要自定义安装，请在安装集群前修改 **install-config.yaml** 文件中的参数。

注意

OpenShift Container Platform 安装配置的作用范围被特意设计为较小。它旨在简化操作并确保成功。在安装完成后，您可以进行更多的 OpenShift Container Platform 配置任务。

6.3.3.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读选择集群安装方法并为用户准备它的文档。
- 已将 AWS 帐户配置为托管集群。



重要

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

- 如果使用防火墙，则会 将其配置为允许集群需要访问的站点。

6.3.3.2. 获取 AWS Marketplace 镜像

如果要使用 AWS Marketplace 镜像部署 OpenShift Container Platform 集群，您必须首先通过 AWS 订阅。订阅提供的提供的 AMI ID 可让您使用安装程序用来部署计算节点的 AMI ID。

先决条件

- 有 AWS 账户购买的产品。此帐户不必与用于安装集群的帐户相同。

流程

1. 从 AWS Marketplace 完成 OpenShift Container Platform 订阅。
2. 记录特定 AWS 区域的 AMI ID。作为安装过程的一部分，您必须在部署集群前使用这个值更新 **install-config.yaml** 文件。

   使用 AWS Marketplace 计算节点的 install-config.yaml 文件示例

```
apiVersion: v1
baseDomain: example.com
compute:
- hyperthreading: Enabled
  name: worker
  platform:
    aws:
      amiID: ami-06c4d345f7c207239 (1)
      type: m5.4xlarge
  replicas: 3
metadata:
  name: test-cluster
platform:
  aws:
    region: us-east-2 (2)
sshKey: ssh-ed25519 AAAA...
pullSecret: '{"auths": ...}'
```

1. 来自 AWS Marketplace 订阅的 AMI ID。
2. 您的 AMI ID 与特定的 AWS 区域相关联。在创建安装配置文件时，请确保选择配置订阅时指定的相同 AWS 区域。

6.3.3.3. 创建安装配置文件

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

先决条件

- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。对于受限网络安装，这些文件位于您的镜像主机上。
- 您有创建镜像 registry 期间生成的 **imageContentSources** 值。
- 您已获取了镜像 registry 的证书内容。

流程

1. 创建 **install-config.yaml** 文件。
   a. 进入包含安装程序的目录并运行以下命令：

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

   1. 对于 **<installation_directory>**，请指定要存储安装程序创建的文件的目录名称。

   在指定目录时：

   - 验证该目录是否具有**执行**权限。在安装目录中运行 Terraform 二进制文件需要这个权限。
   - 使用空目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目

录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

b. 在提示符处，提供云的配置详情：

i. 可选： 选择用于访问集群机器的 SSH 密钥。

注意

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

ii. 选择 **AWS** 作为目标平台。

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

iv. 选择要将集群部署到的 AWS 区域。

v. 选择您为集群配置的 Route 53 服务的基域。

vi. 为集群输入描述性名称。

注意

如果要安装三节点集群，请确保将 **compute.replicas** 参数设置为 **0**。这样可确保集群的 control plane 可以调度。如需更多信息，请参阅"在 AWS 上安装三节点集群"。

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

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

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

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

b. 添加 **additionalTrustBundle** 参数和值。

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

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

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

c. 在以下位置定义 VPC 安装集群的子网：

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

```
    - subnet-3
```

d. 添加镜像内容资源，类似于以下 YAML 摘录：

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

对于这些值，请使用您在创建镜像 registry 时记录的 **imageContentSources**。

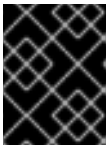
e. 可选：将发布策略设置为 **Internal**：

```
publish: Internal
```

通过设置这个选项，您可以创建一个内部 Ingress Controller 和一个私有负载均衡器。

3. 对您需要的 **install-config.yaml** 文件进行任何其他修改。
有关参数的更多信息，请参阅"安装配置参数"。

4. 备份 **install-config.yaml** 文件，以便您可以使用它安装多个集群。



重要

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

其他资源

- AWS 的安装配置参数

6.3.3.3.1. 集群安装的最低资源要求

每台集群机器都必须满足以下最低要求：

表 6.1. 最低资源要求

机器	操作系统	vCPU [1]	虚拟内存	Storage	每秒输入/输出 (IOPS) [2]
bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane（控制平面）	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS、RHEL 8.6 及更新版本 [3]	2	8 GB	100 GB	300

1. 当未启用并发多线程(SMT)或超线程时，一个 vCPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比例：（每个内核数的线程）× sockets = vCPU。

2. OpenShift Container Platform 和 Kubernetes 对磁盘性能非常敏感，建议使用更快的存储速度，特别是 control plane 节点上需要 10 ms p99 fsync 持续时间的 etcd。请注意，在许多云平台上，存储大小和 IOPS 可一起扩展，因此您可能需要过度分配存储卷来获取足够的性能。

3. 与所有用户置备的安装一样，如果您选择在集群中使用 RHEL 计算机器，则负责所有操作系统生命周期管理和维护，包括执行系统更新、应用补丁和完成所有其他必要的任务。RHEL 7 计算机器的使用已弃用，并已在 OpenShift Container Platform 4.10 及更新的版本中删除。

注意

从 OpenShift Container Platform 版本 4.13 开始，RHCOS 基于 RHEL 版本 9.2，它更新了微架构要求。以下列表包含每个架构需要的最小指令集架构 (ISA)：

- x86-64 体系结构需要 x86-64-v2 ISA
- ARM64 架构需要 ARMv8.0-A ISA
- IBM Power 架构需要 Power 9 ISA
- s390x 架构需要 z14 ISA

如需更多信息，请参阅 RHEL 架构。

如果平台的实例类型满足集群机器的最低要求，则 OpenShift Container Platform 支持使用它。

其他资源

- 优化存储

6.3.3.3.2. 为 AWS 测试的实例类型

以下 Amazon Web Services(AWS) 实例类型已经过 OpenShift Container Platform 测试。

注意

将以下图中包含的机器类型用于 AWS 实例。如果您使用没有在图表中列出的实例类型，请确保使用的实例大小与名为"最小资源要求"的部分中列出的最少资源要求匹配。

> **例 6.18. 基于 64 位 x86 架构的机器类型**
>
> - **c4.***
> - **c5.***
> - **c5a.***
> - **i3.***
> - **m4.***
> - **m5.***

- **m5a.***
- **m6a.***
- **m6i.***
- **r4.***
- **r5.***
- **r5a.***
- **r6i.***
- **t3.***
- **t3a.***

6.3.3.3.3. 在 64 位 ARM 基础架构上为 AWS 测试过的实例类型

OpenShift Container Platform 中已经测试了以下 Amazon Web Services (AWS) 64 位 ARM 实例类型。

注意

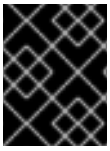
使用 AWS ARM 实例的以下图中包含的机器类型。如果您使用没有在图中列出的实例类型，请确保使用的实例大小与集群安装"最小资源要求"中列出的最少资源要求匹配。

例 6.19. 基于 64 位 ARM 架构的机器类型

- **c6g.***
- **m6g.***

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

您可以自定义安装配置文件 (**install-config.yaml**)，以指定有关 OpenShift Container Platform 集群平台的更多详细信息，或修改所需参数的值。



重要

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

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

```
      zones:
      - us-west-2a
      - us-west-2b
      rootVolume:
        iops: 4000
        size: 500
        type: io1 6
      metadataService:
        authentication: Optional 7
      type: m6i.xlarge
  replicas: 3
compute: 8
- hyperthreading: Enabled 9
  name: worker
  platform:
    aws:
      rootVolume:
        iops: 2000
        size: 500
        type: io1 10
      metadataService:
        authentication: Optional 11
      type: c5.4xlarge
      zones:
      - us-west-2c
  replicas: 3
metadata:
  name: test-cluster 12
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 13
  serviceNetwork:
  - 172.30.0.0/16
platform:
  aws:
    region: us-west-2 14
    propagateUserTags: true 15
    userTags:
      adminContact: jdoe
      costCenter: 7536
    subnets: 16
    - subnet-1
    - subnet-2
    - subnet-3
    amiID: ami-0c5d3e03c0ab9b19a 17
    serviceEndpoints: 18
      - name: ec2
        url: https://vpce-id.ec2.us-west-2.vpce.amazonaws.com
    hostedZone: Z3URY6TWQ91KVV 19
fips: false 20
```

```
sshKey: ssh-ed25519 AAAA... 21
pullSecret: '{"auths":{"<local_registry>": {"auth": "<credentials>","email": "you@example.com"}}}' 22
additionalTrustBundle: | 23
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
imageContentSources: 24
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-v4.0-art-dev
```

1 12 14 必需。安装程序会提示您输入这个值。

2 可选：添加此参数来强制 Cloud Credential Operator (CCO) 使用指定的模式。默认情况下，CCO 使用 **kube-system** 命名空间中的 root 凭证来动态尝试决定凭证的功能。有关 CCO 模式的详情，请参阅*身份验证和授权*指南中的"About the Cloud Credential Operator"部分。

3 8 15 如果没有提供这些参数和值，安装程序会提供默认值。

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

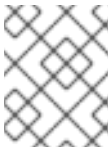
5 9 是否要启用或禁用并发多线程或 **超线程**。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将 参数值设置为 **Disabled** 来禁用它。如果在某些集群机器中禁用并发多线程，则必须在所有集群机器中禁用它。

重要

如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。如果您对机器禁用并发多线程，请使用较大的实例类型，如 **m4.2xlarge** 或 **m5.2xlarge**。

6 10 要为 etcd 配置更快的存储，特别是对于较大的集群，请将存储类型设置为 **io1**，并将 **iops** 设为 **2000**。

7 11 是否需要 Amazon EC2 实例元数据服务 v2 (IMDSv2)。为了要求 IMDSv2，请将参数值设置为 **Required**。要允许使用 IMDSv1 和 IMDSv2，请将参数值设置为 **Optional**。如果没有指定值，则允许 IMDSv1 和 IMDSv2。



注意

在集群安装过程中设置的 control plane 机器的 IMDS 配置只能使用 AWS CLI 更改。可以使用计算机器集来更改计算机器的 IMDS 配置。

13 要安装的集群网络插件。默认值 **OVNKubernetes** 是唯一支持的值。

16 如果您提供自己的 VPC，为集群使用的每个可用区指定子网。

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

18 AWS 服务端点。在安装到未知 AWS 区域时，需要自定义端点。端点 URL 必须使用 **https** 协议，主

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

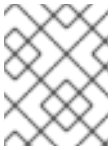
20 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 Red Hat Enterprise Linux (RHEL) 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅在 FIPS 模式中安装该系统。

当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS)时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。

21 您可以选择提供您用来访问集群中机器的 **sshKey** 值。



注意

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

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

23 提供用于镜像 registry 的证书文件内容。

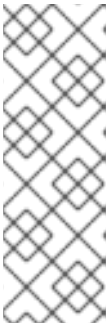
24 提供命令输出中的 **imageContentSources 部分来** 镜像存储库。

6.3.3.3.5. 在安装过程中配置集群范围的代理

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

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 **Proxy** 对象的 **spec.noProxy** 字段中添加站点来绕过代理。



注意

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

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

流程

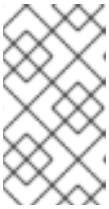
1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: ec2.<aws_region>.amazonaws.com,elasticloadbalancing.
<aws_region>.amazonaws.com,s3.<aws_region>.amazonaws.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 **.** 以仅匹配子域。例如，**.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代理。如果您已将 Amazon **EC2**、**Elastic Load Balancing** 和 **S3** VPC 端点添加到 VPC 中，您必须将这些端点添加到 **noProxy** 字段。
4. 如果提供，安装程序会在 **openshift-config** 命名空间中生成名为 **user-ca-bundle** 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 **trusted-ca-bundle** 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， **Proxy** 对象的 **trustedCA** 字段中也会引用此配置映射。**additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。
5. 可选：决定 **Proxy** 对象的配置以引用 **trustedCA** 字段中 **user-ca-bundle** 配置映射的策略。允许的值是 **Proxyonly** 和 **Always**。仅在配置了 **http/https** 代理时，使用 **Proxyonly** 引用 **user-ca-bundle** 配置映射。使用 **Always** 始终引用 **user-ca-bundle** 配置映射。默认值为 **Proxyonly**。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。



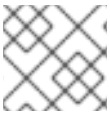
注意

如果安装程序超时，重启并使用安装程序的 **wait-for** 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

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

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



注意

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

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

默认情况下，管理员 secret 存储在 **kube-system** 项目中。如果您在 **install-config.yaml** 文件中将 **credentialsMode** 参数配置为 **Manual**，则必须使用以下替代方案之一：

- 要手动管理长期云凭证，请按照手动创建长期凭证中的步骤操作。
- 要实现在集群外为各个组件管理的短期凭证，请按照配置 A WS 集群以使用短期凭证 中的步骤操作。

6.3.3.4.1. 手动创建长期凭证

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

流程

1. 如果您没有将 **install-config.yaml** 配置文件中的 **credentialsMode** 参数设置为 **Manual**，请修改值，如下所示：

   配置文件片段示例

   ```
   apiVersion: v1
   baseDomain: example.com
   credentialsMode: Manual
   # ...
   ```

2. 如果您之前还没有创建安装清单文件，请运行以下命令：

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

   其中 **<installation_directory>** 是安装程序在其中创建文件的目录。

3. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

   ```
   $ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
   ```

4. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 自定义资源 (CR) 列表：

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
  --included \ 1
  --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
  --to=<path_to_directory_for_credentials_requests> 3
```

1. **--included** 参数仅包含特定集群配置所需的清单。
2. 指定 **install-config.yaml** 文件的位置。
3. 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

此命令为每个 **CredentialsRequest** 对象创建一个 YAML 文件。

CredentialsRequest 对象示例

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

5. 在之前生成的 **openshift-install** 清单目录中为 secret 创建 YAML 文件。secret 必须使用在 **spec.secretRef** 中为每个 **CredentialsRequest** 定义的命名空间和 secret 名称存储。

带有 secret 的 CredentialsRequest 对象示例

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component_credentials_request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
```

```
    kind: AWSProviderSpec
    statementEntries:
    - effect: Allow
      action:
      - s3:CreateBucket
      - s3:DeleteBucket
      resource: "*"
      ...
  secretRef:
    name: <component_secret>
    namespace: <component_namespace>
  ...
```

Secret 对象示例

```
apiVersion: v1
kind: Secret
metadata:
  name: <component_secret>
  namespace: <component_namespace>
data:
  aws_access_key_id: <base64_encoded_aws_access_key_id>
  aws_secret_access_key: <base64_encoded_aws_secret_access_key>
```

重要

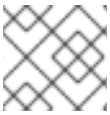
在升级使用手动维护凭证的集群前，您必须确保 CCO 处于可升级状态。

6.3.3.4.2. 将 AWS 集群配置为使用短期凭证

要安装配置为使用 AWS 安全令牌服务 (STS) 的集群，您必须配置 CCO 实用程序并为集群创建所需的 AWS 资源。

6.3.3.4.2.1. 配置 Cloud Credential Operator 工具

当 Cloud Credential Operator(CCO)以手动模式运行时，要从集群外部创建和管理云凭证，提取并准备 CCO 实用程序(**ccoctl**)二进制文件。



注意

ccoctl 工具是在 Linux 环境中运行的 Linux 二进制文件。

先决条件

- 您可以访问具有集群管理员权限的 OpenShift Container Platform 帐户。
- 已安装 OpenShift CLI(**oc**)。
- 您已为 **ccoctl** 工具创建了用于以下权限的 AWS 帐户：

  例 6.20. 所需的 AWS 权限

  所需的 iam 权限

  - **iam:CreateOpenIDConnectProvider**

- iam:CreateRole
- iam:DeleteOpenIDConnectProvider
- iam:DeleteRole
- iam:DeleteRolePolicy
- iam:GetOpenIDConnectProvider
- iam:GetRole
- iam:GetUser
- iam:ListOpenIDConnectProviders
- iam:ListRolePolicies
- iam:ListRoles
- iam:PutRolePolicy
- iam:TagOpenIDConnectProvider
- iam:TagRole

所需的 s3 权限

- s3:CreateBucket
- s3:DeleteBucket
- s3:DeleteObject
- s3:GetBucketAcl
- s3:GetBucketTagging
- s3:GetObject
- s3:GetObjectAcl
- s3:GetObjectTagging
- s3:ListBucket
- s3:PutBucketAcl
- s3:PutBucketPolicy
- s3:PutBucketPublicAccessBlock
- s3:PutBucketTagging
- s3:PutObject
- s3:PutObjectAcl

- **s3:PutObjectTagging**

所需的 cloudfront 权限

- **cloudfront:ListCloudFrontOriginAccessIdentities**
- **cloudfront:ListDistributions**
- **cloudfront:ListTagsForResource**

如果您计划通过公共 CloudFront 发行版 URL 将 OIDC 配置存储在 IAM 身份提供程序访问的私有 S3 存储桶中，则运行 **ccoctl** 工具的 AWS 帐户需要以下额外权限：

例 6.21. 使用 CloudFront 私有 S3 存储桶的额外权限

- **cloudfront:CreateCloudFrontOriginAccessIdentity**
- **cloudfront:CreateDistribution**
- **cloudfront:DeleteCloudFrontOriginAccessIdentity**
- **cloudfront:DeleteDistribution**
- **cloudfront:GetCloudFrontOriginAccessIdentity**
- **cloudfront:GetCloudFrontOriginAccessIdentityConfig**
- **cloudfront:GetDistribution**
- **cloudfront:TagResource**
- **cloudfront:UpdateDistribution**

注意

在使用 **ccoctl aws create-all** 命令处理凭证请求时，这些额外权限支持使用 **--create-private-s3-bucket** 选项。

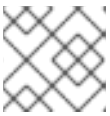
流程

1. 运行以下命令，为 OpenShift Container Platform 发行镜像设置变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. 运行以下命令，从 OpenShift Container Platform 发行镜像获取 CCO 容器镜像：

```
$ CCO_IMAGE=$(oc adm release info --image-for='cloud-credential-operator'
$RELEASE_IMAGE -a ~/.pull-secret)
```



注意

确保 **$RELEASE_IMAGE** 的架构与将使用 **ccoctl** 工具的环境架构相匹配。

3. 运行以下命令，将 CCO 容器镜像中的 **ccoctl** 二进制文件提取到 OpenShift Container Platform 发行镜像中：

```
$ oc image extract $CCO_IMAGE \
  --file="/usr/bin/ccoctl.<rhel_version>" \ 1
  -a ~/.pull-secret
```

1 对于 **<rhel_version>**，请指定与主机使用的 Red Hat Enterprise Linux (RHEL) 版本对应的值。如果没有指定值，则默认使用 **ccoctl.rhel8**。以下值有效：

- **rhel8**: 为使用 RHEL 8 的主机指定这个值。
- **rhel9** ：为使用 RHEL 9 的主机指定这个值。

4. 运行以下命令更改权限以使 **ccoctl** 可执行：

```
$ chmod 775 ccoctl.<rhel_version>
```

验证

- 要验证 **ccoctl** 是否可使用，请运行以下命令来显示帮助文件：

```
$ ccoctl --help
```

ccoctl --help 的输出

```
OpenShift credentials provisioning tool

Usage:
  ccoctl [command]

Available Commands:
  aws         Manage credentials objects for AWS cloud
  azure       Manage credentials objects for Azure
  gcp         Manage credentials objects for Google cloud
  help        Help about any command
  ibmcloud    Manage credentials objects for IBM Cloud
  nutanix     Manage credentials objects for Nutanix

Flags:
  -h, --help   help for ccoctl

Use "ccoctl [command] --help" for more information about a command.
```

6.3.3.4.2.2. 使用 Cloud Credential Operator 实用程序创建 AWS 资源

创建 AWS 资源时有以下选项：

- 您可以使用 **ccoctl aws create-all** 命令自动创建 AWS 资源。这是创建资源的最快速方法。请参阅 使用单个命令创建 AWS 资源。

- 如果您需要在修改 AWS 资源前查看 **ccoctl** 工具创建的 JSON 文件，或者 **ccoctl** 工具用于创建 AWS 资源的过程无法自动满足组织的要求，您可以单独创建 AWS 资源。请参阅 单独创建 AWS 资源。

6.3.3.4.2.2.1. 使用单个命令创建 AWS 资源

如果 **ccoctl** 工具用于创建 AWS 资源的过程自动满足机构的要求，您可以使用 **ccoctl aws create-all** 命令自动创建 AWS 资源。

否则，您可以单独创建 AWS 资源。如需更多信息，请参阅"单独创建 AWS 资源"。

注意

默认情况下，**ccoctl** 在运行命令的目录中创建对象。要在其他目录中创建对象，请使用 **--output-dir** 标志。此流程使用 **<path_to_ccoctl_output_dir>** 来引用这个目录。

先决条件

您必须：

- 提取并准备好 **ccoctl** 二进制文件。

流程

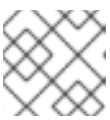
1. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 对象列表：

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
  --included \ 1
  --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
  --to=<path_to_directory_for_credentials_requests> 3
```

1. **--included** 参数仅包含特定集群配置所需的清单。
2. 指定 **install-config.yaml** 文件的位置。
3. 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。



注意

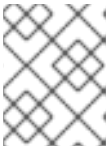
此命令可能需要一些时间才能运行。

3. 运行以下命令，使用 **ccoctl** 工具处理所有 **CredentialsRequest** 对象：

```
$ ccoctl aws create-all \
  --name=<name> \ 1
```

```
--region=<aws_region> \ 2
--credentials-requests-dir=<path_to_credentials_requests_directory> \ 3
--output-dir=<path_to_ccoctl_output_dir> \ 4
--create-private-s3-bucket 5
```

1. 指定用于标记创建用于跟踪的任何云资源的名称。
2. 指定在其中创建云资源的 AWS 区域。
3. 指定包含组件 **CredentialsRequest** 对象文件的目录。
4. 可选：指定您希望 **ccoctl** 实用程序在其中创建对象的目录。默认情况下，实用程序在运行命令的目录中创建对象。
5. 可选：默认情况下，**ccoctl** 实用程序将 OpenID Connect (OIDC) 配置文件存储在公共 S3 存储桶中，并使用 S3 URL 作为公共 OIDC 端点。要将 OIDC 配置存储在 IAM 身份提供程序通过公共 CloudFront 发行版 URL 访问的专用 S3 存储桶中，请使用 **--create-private-s3-bucket** 参数。



注意

如果您的集群使用 **TechPreviewNoUpgrade** 功能集启用的技术预览功能，则必须包含 **--enable-tech-preview** 参数。

验证

- 要验证 OpenShift Container Platform secret 是否已创建，列出 **<path_to_ccoctl_output_dir>/manifests** 目录中的文件：

```
$ ls <path_to_ccoctl_output_dir>/manifests
```

输出示例

```
cluster-authentication-02-config.yaml
openshift-cloud-credential-operator-cloud-credential-operator-iam-ro-creds-credentials.yaml
openshift-cloud-network-config-controller-cloud-credentials-credentials.yaml
openshift-cluster-api-capa-manager-bootstrap-credentials-credentials.yaml
openshift-cluster-csi-drivers-ebs-cloud-credentials-credentials.yaml
openshift-image-registry-installer-cloud-credentials-credentials.yaml
openshift-ingress-operator-cloud-credentials-credentials.yaml
openshift-machine-api-aws-cloud-credentials-credentials.yaml
```

您可以通过查询 AWS 来验证是否已创建 IAM 角色。如需更多信息，请参阅有关列出 IAM 角色的 AWS 文档。

6.3.3.4.2.2.2. 单独创建 AWS 资源

您可以使用 **ccoctl** 工具单独创建 AWS 资源。这个选项对于在不同用户或部门之间创建这些资源的组织可能很有用。

否则，您可以使用 **ccoctl aws create-all** 命令自动创建 AWS 资源。如需更多信息，请参阅"使用单个命令创建 AWS 资源"。

注意

默认情况下，**ccoctl** 在运行命令的目录中创建对象。要在其他目录中创建对象，请使用 **--output-dir** 标志。此流程使用 **<path_to_ccoctl_output_dir>** 来引用这个目录。

有些 **ccoctl** 命令会发出 AWS API 调用来创建或修改 AWS 资源。您可以使用 **--dry-run** 标志来避免 API 调用。使用此标志可在本地文件系统中创建 JSON 文件。您可以使用 **--cli-input-json** 参数查看和修改 JSON 文件，然后使用 AWS CLI 工具应用它们。

先决条件

- 提取并准备 **ccoctl** 二进制文件。

流程

1. 运行以下命令，生成用于为集群设置 OpenID Connect 供应商的公共和私有 RSA 密钥文件：

```
$ ccoctl aws create-key-pair
```

输出示例

```
2021/04/13 11:01:02 Generating RSA keypair
2021/04/13 11:01:03 Writing private key to /<path_to_ccoctl_output_dir>/serviceaccount-signer.private
2021/04/13 11:01:03 Writing public key to /<path_to_ccoctl_output_dir>/serviceaccount-signer.public
2021/04/13 11:01:03 Copying signing key for use by installer
```

其中 **serviceaccount-signer.private** 和 **serviceaccount-signer.public** 是生成的密钥文件。

此命令还会在 **/<path_to_ccoctl_output_dir>/tls/bound-service-account-signing-key.key** 中创建集群在安装过程中所需的私钥。

2. 运行以下命令，在 AWS 上创建 OpenID Connect 身份提供程序和 S3 存储桶：

```
$ ccoctl aws create-identity-provider \
  --name=<name> \ 1
  --region=<aws_region> \ 2
  --public-key-file=<path_to_ccoctl_output_dir>/serviceaccount-signer.public 3
```

1. **<name>** 是用于标记为跟踪而创建的云资源的名称。
2. **<aws-region>** 是将要在其中创建云资源的 AWS 区域。
3. **<path_to_ccoctl_output_dir>** 是 **ccoctl aws create-key-pair** 命令生成的公钥文件的路径。

输出示例

```
2021/04/13 11:16:09 Bucket <name>-oidc created
2021/04/13 11:16:10 OpenID Connect discovery document in the S3 bucket <name>-oidc at .well-known/openid-configuration updated
2021/04/13 11:16:10 Reading public key
```

```
2021/04/13 11:16:10 JSON web key set (JWKS) in the S3 bucket <name>-oidc at keys.json updated
2021/04/13 11:16:18 Identity Provider created with ARN: arn:aws:iam::<aws_account_id>:oidc-provider/<name>-oidc.s3.<aws_region>.amazonaws.com
```

其中 **openid-configuration** 是发现文档和 **key.json** 是一个 JSON Web 密钥集文件。

此命令还会在 **/<path_to_ccoctl_output_dir>/manifests/cluster-authentication-02-config.yaml** 中创建 YAML 配置文件。此文件为集群生成的服务帐户令牌设置签发者 URL 字段，以便 AWS IAM 身份提供程序信任令牌。

3. 为集群中的每个组件创建 IAM 角色：

   a. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

   ```
   $ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
   ```

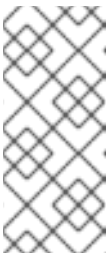
   b. 从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 对象列表：

   ```
   $ oc adm release extract \
     --from=$RELEASE_IMAGE \
     --credentials-requests \
     --included \ 1
     --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \
   2
     --to=<path_to_directory_for_credentials_requests> 3
   ```

   1. **--included** 参数仅包含特定集群配置所需的清单。
   2. 指定 **install-config.yaml** 文件的位置。
   3. 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

   c. 运行以下命令，使用 **ccoctl** 工具处理所有 **CredentialsRequest** 对象：

   ```
   $ ccoctl aws create-iam-roles \
     --name=<name> \
     --region=<aws_region> \
     --credentials-requests-dir=<path_to_credentials_requests_directory> \
     --identity-provider-arn=arn:aws:iam::<aws_account_id>:oidc-provider/<name>-oidc.s3.<aws_region>.amazonaws.com
   ```



注意

对于使用其他 IAM API 端点的 AWS 环境（如 GovCloud），还必须使用 **--region** 参数指定您的区域。

如果您的集群使用 **TechPreviewNoUpgrade** 功能集启用的技术预览功能，则必须包含 **--enable-tech-preview** 参数。

对于每个 **CredentialsRequest** 对象，**ccoctl** 创建一个带有信任策略的 IAM 角色，该角色与

对于每个 **CredentialsRequest** 对象，**ccoctl** 创建一个带有信任策略的 IAM 角色，该角色与指定的 OIDC 身份提供程序相关联，以及来自 OpenShift Container Platform 发行镜像的每个 **CredentialsRequest** 对象中定义的权限策略。

验证

- 要验证 OpenShift Container Platform secret 是否已创建，列出 **<path_to_ccoctl_output_dir>/manifests** 目录中的文件：

```
$ ls <path_to_ccoctl_output_dir>/manifests
```

输出示例

```
cluster-authentication-02-config.yaml
openshift-cloud-credential-operator-cloud-credential-operator-iam-ro-creds-credentials.yaml
openshift-cloud-network-config-controller-cloud-credentials-credentials.yaml
openshift-cluster-api-capa-manager-bootstrap-credentials-credentials.yaml
openshift-cluster-csi-drivers-ebs-cloud-credentials-credentials.yaml
openshift-image-registry-installer-cloud-credentials-credentials.yaml
openshift-ingress-operator-cloud-credentials-credentials.yaml
openshift-machine-api-aws-cloud-credentials-credentials.yaml
```

您可以通过查询 AWS 来验证是否已创建 IAM 角色。如需更多信息，请参阅有关列出 IAM 角色的 AWS 文档。

6.3.3.4.2.3. 整合 Cloud Credential Operator 实用程序清单

要为单个组件在集群外实现短期安全凭证，您必须将创建 Cloud Credential Operator 实用程序 (**ccoctl**) 的清单文件移到安装程序的正确目录中。

先决条件

- 您已使用托管集群的云平台配置了帐户。
- 您已配置了 Cloud Credential Operator 实用程序 (**ccoctl**)。
- 已使用 **ccoctl** 工具创建了集群所需的云供应商资源。

流程

1. 如果您没有将 **install-config.yaml** 配置文件中的 **credentialsMode** 参数设置为 **Manual**，请修改值，如下所示：

   配置文件片段示例

```
apiVersion: v1
baseDomain: example.com
credentialsMode: Manual
# ...
```

2. 如果您之前还没有创建安装清单文件，请运行以下命令：

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

其中 **<installation_directory>** 是安装程序在其中创建文件的目录。

3. 运行以下命令，将 **ccoctl** 工具生成的清单复制到安装程序创建的 **manifests** 目录中：

```
$ cp /<path_to_ccoctl_output_dir>/manifests/* ./manifests/
```

4. 运行以下命令，将 **ccoctl** 工具在 **tls** 目录中生成的私钥复制到安装目录中：

```
$ cp -a /<path_to_ccoctl_output_dir>/tls .
```

6.3.3.5. 部署集群

您可以在兼容云平台上安装 OpenShift Container Platform。

重要

在初始安装过程中，您只能运行安装程序的 **create cluster** 命令一次。

先决条件

- 您已使用托管集群的云平台配置了帐户。
- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。
- 已确认主机上的云供应商帐户具有部署集群的正确权限。权限不正确的帐户会导致安装过程失败，并显示包括缺失权限的错误消息。

流程

1. 进入包含安装程序的目录并初始化集群部署：

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

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

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

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

注意

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

验证

当集群部署成功完成时：

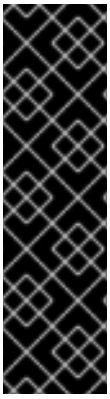
- 终端会显示用于访问集群的说明，包括指向 Web 控制台和 **kubeadmin** 用户的凭证的链接。
- 凭证信息还会输出到 **<installation_directory>/.openshift_install.log**.

重要

不要删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

输出示例

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



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

6.3.3.6. 使用 CLI 登录集群

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

先决条件

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

流程

1. 导出 **kubeadmin** 凭证：

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

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

2. 验证您可以使用导出的配置成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

6.3.3.7. 使用 Web 控制台登录到集群

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

先决条件

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

流程

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

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

注意

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

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

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

注意

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

输出示例

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

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

其他资源

- 如需有关 访问和了解 OpenShift Container Platform Web 控制台的更多详情，请参阅 访问 Web 控制台。

6.3.3.8. 后续步骤

- 验证安装.
- 自定义集群。

- 如果需要，您可以选择 不使用远程健康报告。
- 如果需要，您可以删除云供应商凭证。

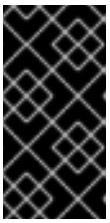
6.3.4. 使用自定义网络在 AWS 上安装集群

在 OpenShift Container Platform 版本 4.16 中，您可以使用自定义网络配置选项在 Amazon Web Services (AWS)上安装集群。通过自定义网络配置，您的集群可以与环境中现有的 IP 地址分配共存，并与现有的 MTU 和 VXLAN 配置集成。

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

6.3.4.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读选择集群安装方法并为用户准备它的文档。
- 已将 AWS 帐户配置为托管集群。



重要

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

- 如果使用防火墙，则会 将其配置为允许集群需要访问的站点。

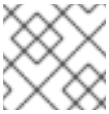
6.3.4.2. 网络配置阶段

OpenShift Container Platform 安装前有两个阶段，您可以在其中自定义网络配置。

第 1 阶段

在创建清单文件前，您可以自定义 **install-config.yaml** 文件中的以下与网络相关的字段：

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



注意

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

重要

CIDR 范围 **172.17.0.0/16** 由 libVirt 保留。对于集群中的任何网络，您无法使用此范围或与这个范围重叠的范围。

第 2 阶段

运行 **openshift-install create 清单创建** 清单文件后，您可以只使用您要修改的字段定义自定义 Cluster Network Operator 清单。您可以使用 清单指定高级网络配置。

您不能覆盖在 stage 2 阶段 1 中在 **install-config.yaml** 文件中指定的值。但是，您可以在第 2 阶段进一步自定义网络插件。

6.3.4.3. 创建安装配置文件

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

先决条件

- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。对于受限网络安装，这些文件位于您的镜像主机上。
- 您有创建镜像 registry 期间生成的 **imageContentSources** 值。
- 您已获取了镜像 registry 的证书内容。

流程

1. 创建 **install-config.yaml** 文件。
   a. 进入包含安装程序的目录并运行以下命令：

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

   1 对于 **<installation_directory>**，请指定要存储安装程序创建的文件的目录名称。

   在指定目录时：

   - 验证该目录是否具有**执行**权限。在安装目录中运行 Terraform 二进制文件需要这个权限。
   - 使用空目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

   b. 在提示符处，提供云的配置详情：
      i. 可选： 选择用于访问集群机器的 SSH 密钥。

注意

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

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

iv. 选择要将集群部署到的 AWS 区域。

v. 选择您为集群配置的 Route 53 服务的基域。

vi. 为集群输入描述性名称。

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

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

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

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

b. 添加 **additionalTrustBundle** 参数和值。

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

  ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
  -----END CERTIFICATE-----
```

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

c. 在以下位置定义 VPC 安装集群的子网：

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

d. 添加镜像内容资源，类似于以下 YAML 摘录：

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

对于这些值，请使用您在创建镜像 registry 时记录的 **imageContentSources**。

e. 可选：将发布策略设置为 **Internal**：

```
publish: Internal
```

通过设置这个选项，您可以创建一个内部 Ingress Controller 和一个私有负载均衡器。

3. 对您需要的 **install-config.yaml** 文件进行任何其他修改。
有关参数的更多信息，请参阅"安装配置参数"。

4. 备份 **install-config.yaml** 文件，以便您可以使用它安装多个集群。

重要

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

其他资源

- AWS 的安装配置参数

6.3.4.3.1. 集群安装的最低资源要求

每台集群机器都必须满足以下最低要求：

表 6.2. 最低资源要求

机器	操作系统	vCPU [1]	虚拟内存	Storage	每秒输入/输出 (IOPS) [2]
bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane（控制平面）	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS、RHEL 8.6 及更新版本 [3]	2	8 GB	100 GB	300

1. 当未启用并发多线程(SMT)或超线程时，一个 vCPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比例：（每个内核数的线程）× sockets = vCPU。

2. OpenShift Container Platform 和 Kubernetes 对磁盘性能非常敏感，建议使用更快的存储速度，特别是 control plane 节点上需要 10 ms p99 fsync 持续时间的 etcd。请注意，在许多云平台上，存储大小和 IOPS 可一起扩展，因此您可能需要过度分配存储卷来获取足够的性能。

3. 与所有用户置备的安装一样，如果您选择在集群中使用 RHEL 计算机器，则负责所有操作系统生命周期管理和维护，包括执行系统更新、应用补丁和完成所有其他必要的任务。RHEL 7 计算机器的使用已弃用，并已在 OpenShift Container Platform 4.10 及更新的版本中删除。

注意

从 OpenShift Container Platform 版本 4.13 开始，RHCOS 基于 RHEL 版本 9.2，它更新了微架构要求。以下列表包含每个架构需要的最小指令集架构 (ISA)：

- x86-64 体系结构需要 x86-64-v2 ISA
- ARM64 架构需要 ARMv8.0-A ISA
- IBM Power 架构需要 Power 9 ISA
- s390x 架构需要 z14 ISA

如需更多信息，请参阅 RHEL 架构。

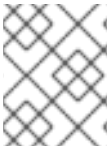
如果平台的实例类型满足集群机器的最低要求，则 OpenShift Container Platform 支持使用它。

其他资源

- 优化存储

6.3.4.3.2. 为 AWS 测试的实例类型

以下 Amazon Web Services(AWS) 实例类型已经过 OpenShift Container Platform 测试。



注意

将以下图中包含的机器类型用于 AWS 实例。如果您使用没有在图表中列出的实例类型，请确保使用的实例大小与名为"最小资源要求"的部分中列出的最少资源要求匹配。

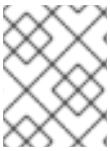
例 6.22. 基于 64 位 x86 架构的机器类型

- **c4.***
- **c5.***
- **c5a.***
- **i3.***
- **m4.***
- **m5.***
- **m5a.***
- **m6a.***
- **m6i.***
- **r4.***
- **r5.***
- **r5a.***

- **r6i.***
- **t3.***
- **t3a.***

6.3.4.3.3. 在 64 位 ARM 基础架构上为 AWS 测试过的实例类型

OpenShift Container Platform 中已经测试了以下 Amazon Web Services (AWS) 64 位 ARM 实例类型。



注意

使用 AWS ARM 实例的以下图中包含的机器类型。如果您使用没有在图中列出的实例类型，请确保使用的实例大小与集群安装"最小资源要求"中列出的最少资源要求匹配。

例 6.23. 基于 64 位 ARM 架构的机器类型

- **c6g.***
- **m6g.***

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

您可以自定义安装配置文件 (**install-config.yaml**)，以指定有关 OpenShift Container Platform 集群平台的更多详细信息，或修改所需参数的值。

重要

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

```
apiVersion: v1
baseDomain: example.com (1)
credentialsMode: Mint (2)
controlPlane: (3) (4)
  hyperthreading: Enabled (5)
  name: master
  platform:
    aws:
      zones:
      - us-west-2a
      - us-west-2b
      rootVolume:
        iops: 4000
        size: 500
        type: io1 (6)
      metadataService:
        authentication: Optional (7)
      type: m6i.xlarge
  replicas: 3
```

```
compute: 8
- hyperthreading: Enabled 9
  name: worker
  platform:
    aws:
      rootVolume:
        iops: 2000
        size: 500
        type: io1 10
      metadataService:
        authentication: Optional 11
      type: c5.4xlarge
      zones:
      - us-west-2c
  replicas: 3
metadata:
  name: test-cluster 12
networking: 13
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 14
  serviceNetwork:
  - 172.30.0.0/16
platform:
  aws:
    region: us-west-2 15
    propagateUserTags: true 16
    userTags:
      adminContact: jdoe
      costCenter: 7536
    subnets: 17
    - subnet-1
    - subnet-2
    - subnet-3
    amiID: ami-0c5d3e03c0ab9b19a 18
    serviceEndpoints: 19
      - name: ec2
        url: https://vpce-id.ec2.us-west-2.vpce.amazonaws.com
    hostedZone: Z3URY6TWQ91KVV 20
fips: false 21
sshKey: ssh-ed25519 AAAA... 22
pullSecret: '{"auths":{"<local_registry>": {"auth": "<credentials>","email": "you@example.com"}}}' 23
additionalTrustBundle: | 24
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
imageContentSources: 25
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-release
```

```
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-v4.0-art-dev
```

1 12 15 必需。安装程序会提示您输入这个值。

2 可选：添加此参数来强制 Cloud Credential Operator (CCO) 使用指定的模式。默认情况下，CCO 使用 **kube-system** 命名空间中的 root 凭证来动态尝试决定凭证的功能。有关 CCO 模式的详情，请参阅*身份验证和授权*指南中的"About the Cloud Credential Operator"部分。

3 8 13 16 如果没有提供这些参数和值，安装程序会提供默认值。

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

5 9 是否要启用或禁用并发多线程或 **超线程**。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将 参数值设置为 **Disabled** 来禁用它。如果在某些集群机器中禁用并发多线程，则必须在所有集群机器中禁用它。

重要

如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。如果您对机器禁用并发多线程，请使用较大的实例类型，如 **m4.2xlarge** 或 **m5.2xlarge**。

6 10 要为 etcd 配置更快的存储，特别是对于较大的集群，请将存储类型设置为 **io1**，并将 **iops** 设为 **2000**。

7 11 是否需要 Amazon EC2 实例元数据服务 v2 (IMDSv2)。为了要求 IMDSv2，请将参数值设置为 **Required**。要允许使用 IMDSv1 和 IMDSv2，请将参数值设置为 **Optional**。如果没有指定值，则允许 IMDSv1 和 IMDSv2。

注意

在集群安装过程中设置的 control plane 机器的 IMDS 配置只能使用 AWS CLI 更改。可以使用计算机器集来更改计算机器的 IMDS 配置。

14 要安装的集群网络插件。默认值 **OVNKubernetes** 是唯一支持的值。

17 如果您提供自己的 VPC，为集群使用的每个可用区指定子网。

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

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

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

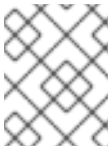
21 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 Red Hat Enterprise Linux (RHEL) 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅在 FIPS 模式中安装该系统。

当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS)时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。

22 您可以选择提供您用来访问集群中机器的 **sshKey** 值。



注意

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

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

24 提供用于镜像 registry 的证书文件内容。

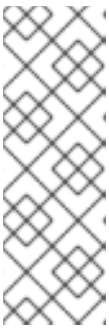
25 提供命令输出中的 **imageContentSources 部分来** 镜像存储库。

6.3.4.3.5. 在安装过程中配置集群范围的代理

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

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 **Proxy** 对象的 **spec.noProxy** 字段中添加站点来绕过代理。



注意

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

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

流程

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

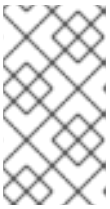
```
apiVersion: v1
```

```
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: ec2.<aws_region>.amazonaws.com,elasticloadbalancing.
<aws_region>.amazonaws.com,s3.<aws_region>.amazonaws.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 **.** 以仅匹配子域。例如，**.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代理。如果您已将 Amazon **EC2**、**Elastic Load Balancing** 和 **S3** VPC 端点添加到 VPC 中，您必须将这些端点添加到 **noProxy** 字段。
4. 如果提供，安装程序会在 **openshift-config** 命名空间中生成名为 **user-ca-bundle** 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 **trusted-ca-bundle** 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， **Proxy** 对象的 **trustedCA** 字段中也会引用此配置映射。**additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。
5. 可选：决定 **Proxy** 对象的配置以引用 **trustedCA** 字段中 **user-ca-bundle** 配置映射的策略。允许的值是 **Proxyonly** 和 **Always**。仅在配置了 **http/https** 代理时，使用 **Proxyonly** 引用 **user-ca-bundle** 配置映射。使用 **Always** 始终引用 **user-ca-bundle** 配置映射。默认值为 **Proxyonly**。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。



注意

如果安装程序超时，重启并使用安装程序的 **wait-for** 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

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

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

注意

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

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

默认情况下，管理员 secret 存储在 **kube-system** 项目中。如果您在 **install-config.yaml** 文件中将 **credentialsMode** 参数配置为 **Manual**，则必须使用以下替代方案之一：

- 要手动管理长期云凭证，请按照手动创建长期凭证中的步骤操作。
- 要实现在集群外为各个组件管理的短期凭证，请按照配置 A WS 集群以使用短期凭证 中的步骤操作。

6.3.4.4.1. 手动创建长期凭证

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

流程

1. 如果您没有将 **install-config.yaml** 配置文件中的 **credentialsMode** 参数设置为 **Manual**，请修改值，如下所示：

   配置文件片段示例

   ```
   apiVersion: v1
   baseDomain: example.com
   credentialsMode: Manual
   # ...
   ```

2. 如果您之前还没有创建安装清单文件，请运行以下命令：

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

   其中 **<installation_directory>** 是安装程序在其中创建文件的目录。

3. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

   ```
   $ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
   ```

4. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 自定义资源 (CR) 列表：

   ```
   $ oc adm release extract \
     --from=$RELEASE_IMAGE \
     --credentials-requests \
     --included \ 1
     --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
     --to=<path_to_directory_for_credentials_requests> 3
   ```

   1. **--included** 参数仅包含特定集群配置所需的清单。
   2. 指定 **install-config.yaml** 文件的位置。
   3. 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

此命令为每个 **CredentialsRequest** 对象创建一个 YAML 文件。

CredentialsRequest 对象示例

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

5. 在之前生成的 **openshift-install** 清单目录中为 secret 创建 YAML 文件。secret 必须使用在 **spec.secretRef** 中为每个 **CredentialsRequest** 定义的命名空间和 secret 名称存储。

带有 secret 的 CredentialsRequest 对象示例

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component_credentials_request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: AWSProviderSpec
    statementEntries:
    - effect: Allow
      action:
      - s3:CreateBucket
      - s3:DeleteBucket
      resource: "*"
      ...
  secretRef:
    name: <component_secret>
    namespace: <component_namespace>
  ...
```

Secret 对象示例

```
apiVersion: v1
kind: Secret
```

```
metadata:
  name: <component_secret>
  namespace: <component_namespace>
data:
  aws_access_key_id: <base64_encoded_aws_access_key_id>
  aws_secret_access_key: <base64_encoded_aws_secret_access_key>
```

重要

在升级使用手动维护凭证的集群前，您必须确保 CCO 处于可升级状态。

6.3.4.4.2. 将 AWS 集群配置为使用短期凭证

要安装配置为使用 AWS 安全令牌服务 (STS) 的集群，您必须配置 CCO 实用程序并为集群创建所需的 AWS 资源。

6.3.4.4.2.1. 配置 Cloud Credential Operator 工具

当 Cloud Credential Operator(CCO)以手动模式运行时，要从集群外部创建和管理云凭证，提取并准备 CCO 实用程序(**ccoctl**)二进制文件。

注意

ccoctl 工具是在 Linux 环境中运行的 Linux 二进制文件。

先决条件

- 您可以访问具有集群管理员权限的 OpenShift Container Platform 帐户。
- 已安装 OpenShift CLI(**oc**)。
- 您已为 **ccoctl** 工具创建了用于以下权限的 AWS 帐户：

  例 6.24. 所需的 AWS 权限

  所需的 iam 权限

  - **iam:CreateOpenIDConnectProvider**
  - **iam:CreateRole**
  - **iam:DeleteOpenIDConnectProvider**
  - **iam:DeleteRole**
  - **iam:DeleteRolePolicy**
  - **iam:GetOpenIDConnectProvider**
  - **iam:GetRole**
  - **iam:GetUser**
  - **iam:ListOpenIDConnectProviders**
  - **iam:ListRolePolicies**

- **iam:ListRoles**
- **iam:PutRolePolicy**
- **iam:TagOpenIDConnectProvider**
- **iam:TagRole**

所需的 s3 权限

- **s3:CreateBucket**
- **s3:DeleteBucket**
- **s3:DeleteObject**
- **s3:GetBucketAcl**
- **s3:GetBucketTagging**
- **s3:GetObject**
- **s3:GetObjectAcl**
- **s3:GetObjectTagging**
- **s3:ListBucket**
- **s3:PutBucketAcl**
- **s3:PutBucketPolicy**
- **s3:PutBucketPublicAccessBlock**
- **s3:PutBucketTagging**
- **s3:PutObject**
- **s3:PutObjectAcl**
- **s3:PutObjectTagging**

所需的 cloudfront 权限

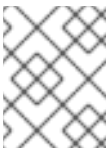
- **cloudfront:ListCloudFrontOriginAccessIdentities**
- **cloudfront:ListDistributions**
- **cloudfront:ListTagsForResource**

如果您计划通过公共 CloudFront 发行版 URL 将 OIDC 配置存储在 IAM 身份提供程序访问的私有 S3 存储桶中，则运行 **ccoctl** 工具的 AWS 帐户需要以下额外权限：

例 6.25. 使用 CloudFront 私有 S3 存储桶的额外权限

- **cloudfront:CreateCloudFrontOriginAccessIdentity**

- **cloudfront:CreateDistribution**
- **cloudfront:DeleteCloudFrontOriginAccessIdentity**
- **cloudfront:DeleteDistribution**
- **cloudfront:GetCloudFrontOriginAccessIdentity**
- **cloudfront:GetCloudFrontOriginAccessIdentityConfig**
- **cloudfront:GetDistribution**
- **cloudfront:TagResource**
- **cloudfront:UpdateDistribution**



注意

在使用 **ccoctl aws create-all** 命令处理凭证请求时，这些额外权限支持使用 **--create-private-s3-bucket** 选项。

流程

1. 运行以下命令，为 OpenShift Container Platform 发行镜像设置变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. 运行以下命令，从 OpenShift Container Platform 发行镜像获取 CCO 容器镜像：

```
$ CCO_IMAGE=$(oc adm release info --image-for='cloud-credential-operator'
$RELEASE_IMAGE -a ~/.pull-secret)
```

注意

确保 **$RELEASE_IMAGE** 的架构与将使用 **ccoctl** 工具的环境架构相匹配。

3. 运行以下命令，将 CCO 容器镜像中的 **ccoctl** 二进制文件提取到 OpenShift Container Platform 发行镜像中：

```
$ oc image extract $CCO_IMAGE \
  --file="/usr/bin/ccoctl.<rhel_version>" \ 1
  -a ~/.pull-secret
```

1 对于 **<rhel_version>**，请指定与主机使用的 Red Hat Enterprise Linux (RHEL) 版本对应的值。如果没有指定值，则默认使用 **ccoctl.rhel8**。以下值有效：

- **rhel8**: 为使用 RHEL 8 的主机指定这个值。
- **rhel9** ：为使用 RHEL 9 的主机指定这个值。

4. 运行以下命令更改权限以使 **ccoctl** 可执行：

-

```
$ chmod 775 ccoctl.<rhel_version>
```

验证

- 要验证 **ccoctl** 是否可使用，请运行以下命令来显示帮助文件：

```
$ ccoctl --help
```

ccoctl --help 的输出

```
OpenShift credentials provisioning tool

Usage:
  ccoctl [command]

Available Commands:
  aws         Manage credentials objects for AWS cloud
  azure       Manage credentials objects for Azure
  gcp         Manage credentials objects for Google cloud
  help        Help about any command
  ibmcloud    Manage credentials objects for IBM Cloud
  nutanix     Manage credentials objects for Nutanix

Flags:
  -h, --help   help for ccoctl

Use "ccoctl [command] --help" for more information about a command.
```

6.3.4.4.2.2. 使用 Cloud Credential Operator 实用程序创建 AWS 资源

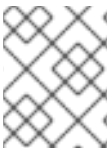
创建 AWS 资源时有以下选项：

- 您可以使用 **ccoctl aws create-all** 命令自动创建 AWS 资源。这是创建资源的最快速方法。请参阅使用单个命令创建 AWS 资源。
- 如果您需要在修改 AWS 资源前查看 **ccoctl** 工具创建的 JSON 文件，或者 **ccoctl** 工具用于创建 AWS 资源的过程无法自动满足组织的要求，您可以单独创建 AWS 资源。请参阅单独创建 AWS 资源。

6.3.4.4.2.2.1. 使用单个命令创建 AWS 资源

如果 **ccoctl** 工具用于创建 AWS 资源的过程自动满足机构的要求，您可以使用 **ccoctl aws create-all** 命令自动创建 AWS 资源。

否则，您可以单独创建 AWS 资源。如需更多信息，请参阅"单独创建 AWS 资源"。



注意

默认情况下，**ccoctl** 在运行命令的目录中创建对象。要在其他目录中创建对象，请使用 **--output-dir** 标志。此流程使用 **<path_to_ccoctl_output_dir>** 来引用这个目录。

先决条件

您必须：

- 提取并准备好 **ccoctl** 二进制文件。

流程

1. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 对象列表：

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
  --included \ 1
  --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
  --to=<path_to_directory_for_credentials_requests> 3
```

1. **--included** 参数仅包含特定集群配置所需的清单。
2. 指定 **install-config.yaml** 文件的位置。
3. 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

注意

此命令可能需要一些时间才能运行。

3. 运行以下命令，使用 **ccoctl** 工具处理所有 **CredentialsRequest** 对象：

```
$ ccoctl aws create-all \
  --name=<name> \ 1
  --region=<aws_region> \ 2
  --credentials-requests-dir=<path_to_credentials_requests_directory> \ 3
  --output-dir=<path_to_ccoctl_output_dir> \ 4
  --create-private-s3-bucket 5
```

1. 指定用于标记创建用于跟踪的任何云资源的名称。
2. 指定在其中创建云资源的 AWS 区域。
3. 指定包含组件 **CredentialsRequest** 对象文件的目录。
4. 可选：指定您希望 **ccoctl** 实用程序在其中创建对象的目录。默认情况下，实用程序在运行命令的目录中创建对象。
5. 可选：默认情况下，**ccoctl** 实用程序将 OpenID Connect (OIDC) 配置文件存储在公共 S3 存储桶中，并使用 S3 URL 作为公共 OIDC 端点。要将 OIDC 配置存储在 IAM 身份提供程序通过公共 CloudFront 发行版 URL 访问的专用 S3 存储桶中，请使用 **--create-private-s3-bucket** 参数。

注意

如果您的集群使用 **TechPreviewNoUpgrade** 功能集启用的技术预览功能，则必须包含 **--enable-tech-preview** 参数。

验证

- 要验证 OpenShift Container Platform secret 是否已创建，列出 **<path_to_ccoctl_output_dir>/manifests** 目录中的文件：

```
$ ls <path_to_ccoctl_output_dir>/manifests
```

输出示例

```
cluster-authentication-02-config.yaml
openshift-cloud-credential-operator-cloud-credential-operator-iam-ro-creds-credentials.yaml
openshift-cloud-network-config-controller-cloud-credentials-credentials.yaml
openshift-cluster-api-capa-manager-bootstrap-credentials-credentials.yaml
openshift-cluster-csi-drivers-ebs-cloud-credentials-credentials.yaml
openshift-image-registry-installer-cloud-credentials-credentials.yaml
openshift-ingress-operator-cloud-credentials-credentials.yaml
openshift-machine-api-aws-cloud-credentials-credentials.yaml
```

您可以通过查询 AWS 来验证是否已创建 IAM 角色。如需更多信息，请参阅有关列出 IAM 角色的 AWS 文档。

6.3.4.4.2.2.2. 单独创建 AWS 资源

您可以使用 **ccoctl** 工具单独创建 AWS 资源。这个选项对于在不同用户或部门之间创建这些资源的组织可能很有用。

否则，您可以使用 **ccoctl aws create-all** 命令自动创建 AWS 资源。如需更多信息，请参阅"使用单个命令创建 AWS 资源"。

注意

默认情况下，**ccoctl** 在运行命令的目录中创建对象。要在其他目录中创建对象，请使用 **--output-dir** 标志。此流程使用 **<path_to_ccoctl_output_dir>** 来引用这个目录。

有些 **ccoctl** 命令会发出 AWS API 调用来创建或修改 AWS 资源。您可以使用 **--dry-run** 标志来避免 API 调用。使用此标志可在本地文件系统中创建 JSON 文件。您可以使用 **--cli-input-json** 参数查看和修改 JSON 文件，然后使用 AWS CLI 工具应用它们。

先决条件

- 提取并准备 **ccoctl** 二进制文件。

流程

1. 运行以下命令，生成用于为集群设置 OpenID Connect 供应商的公共和私有 RSA 密钥文件：

```
$ ccoctl aws create-key-pair
```

输出示例

```
2021/04/13 11:01:02 Generating RSA keypair
2021/04/13 11:01:03 Writing private key to /<path_to_ccoctl_output_dir>/serviceaccount-signer.private
2021/04/13 11:01:03 Writing public key to /<path_to_ccoctl_output_dir>/serviceaccount-signer.public
2021/04/13 11:01:03 Copying signing key for use by installer
```

其中 **serviceaccount-signer.private** 和 **serviceaccount-signer.public** 是生成的密钥文件。

此命令还会在 **/<path_to_ccoctl_output_dir>/tls/bound-service-account-signing-key.key** 中创建集群在安装过程中所需的私钥。

2. 运行以下命令，在 AWS 上创建 OpenID Connect 身份提供程序和 S3 存储桶：

```
$ ccoctl aws create-identity-provider \
  --name=<name> \1
  --region=<aws_region> \2
  --public-key-file=<path_to_ccoctl_output_dir>/serviceaccount-signer.public 3
```

1. **<name>** 是用于标记为跟踪而创建的云资源的名称。
2. **<aws-region>** 是将要在其中创建云资源的 AWS 区域。
3. **<path_to_ccoctl_output_dir>** 是 **ccoctl aws create-key-pair** 命令生成的公钥文件的路径。

输出示例

```
2021/04/13 11:16:09 Bucket <name>-oidc created
2021/04/13 11:16:10 OpenID Connect discovery document in the S3 bucket <name>-oidc at .well-known/openid-configuration updated
2021/04/13 11:16:10 Reading public key
2021/04/13 11:16:10 JSON web key set (JWKS) in the S3 bucket <name>-oidc at keys.json updated
2021/04/13 11:16:18 Identity Provider created with ARN: arn:aws:iam::<aws_account_id>:oidc-provider/<name>-oidc.s3.<aws_region>.amazonaws.com
```

其中 **openid-configuration** 是发现文档和 **key.json** 是一个 JSON Web 密钥集文件。

此命令还会在 **/<path_to_ccoctl_output_dir>/manifests/cluster-authentication-02-config.yaml** 中创建 YAML 配置文件。此文件为集群生成的服务帐户令牌设置签发者 URL 字段，以便 AWS IAM 身份提供程序信任令牌。

3. 为集群中的每个组件创建 IAM 角色：

   a. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

   ```
   $ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
   ```

   b. 从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 对象列表：

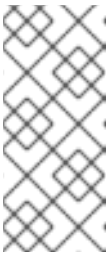
   ```
   $ oc adm release extract \
     --from=$RELEASE_IMAGE \
   ```

```
  --credentials-requests \
  --included \ 1
  --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \
2
  --to=<path_to_directory_for_credentials_requests> 3
```

1. **--included** 参数仅包含特定集群配置所需的清单。
2. 指定 **install-config.yaml** 文件的位置。
3. 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

c. 运行以下命令，使用 **ccoctl** 工具处理所有 **CredentialsRequest** 对象：

```
$ ccoctl aws create-iam-roles \
  --name=<name> \
  --region=<aws_region> \
  --credentials-requests-dir=<path_to_credentials_requests_directory> \
  --identity-provider-arn=arn:aws:iam::<aws_account_id>:oidc-provider/<name>-oidc.s3.
<aws_region>.amazonaws.com
```



注意

对于使用其他 IAM API 端点的 AWS 环境（如 GovCloud），还必须使用 **--region** 参数指定您的区域。

如果您的集群使用 **TechPreviewNoUpgrade** 功能集启用的技术预览功能，则必须包含 **--enable-tech-preview** 参数。

对于每个 **CredentialsRequest** 对象，**ccoctl** 创建一个带有信任策略的 IAM 角色，该角色与指定的 OIDC 身份提供程序相关联，以及来自 OpenShift Container Platform 发行镜像的每个 **CredentialsRequest** 对象中定义的权限策略。

验证

- 要验证 OpenShift Container Platform secret 是否已创建，列出 **<path_to_ccoctl_output_dir>/manifests** 目录中的文件：

```
$ ls <path_to_ccoctl_output_dir>/manifests
```

输出示例

```
cluster-authentication-02-config.yaml
openshift-cloud-credential-operator-cloud-credential-operator-iam-ro-creds-credentials.yaml
openshift-cloud-network-config-controller-cloud-credentials-credentials.yaml
openshift-cluster-api-capa-manager-bootstrap-credentials-credentials.yaml
openshift-cluster-csi-drivers-ebs-cloud-credentials-credentials.yaml
openshift-image-registry-installer-cloud-credentials-credentials.yaml
openshift-ingress-operator-cloud-credentials-credentials.yaml
openshift-machine-api-aws-cloud-credentials-credentials.yaml
```

您可以通过查询 AWS 来验证是否已创建 IAM 角色。如需更多信息，请参阅有关列出 IAM 角色的 AWS 文档。

6.3.4.4.2.3. 整合 Cloud Credential Operator 实用程序清单

要为单个组件在集群外实现短期安全凭证，您必须将创建 Cloud Credential Operator 实用程序 (**ccoctl**) 的清单文件移到安装程序的正确目录中。

先决条件

- 您已使用托管集群的云平台配置了帐户。
- 您已配置了 Cloud Credential Operator 实用程序 (**ccoctl**)。
- 已使用 **ccoctl** 工具创建了集群所需的云供应商资源。

流程

1. 如果您没有将 **install-config.yaml** 配置文件中的 **credentialsMode** 参数设置为 **Manual**，请修改值，如下所示：

   配置文件片段示例

   ```
   apiVersion: v1
   baseDomain: example.com
   credentialsMode: Manual
   # ...
   ```

2. 如果您之前还没有创建安装清单文件，请运行以下命令：

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

   其中 **<installation_directory>** 是安装程序在其中创建文件的目录。

3. 运行以下命令，将 **ccoctl** 工具生成的清单复制到安装程序创建的 **manifests** 目录中：

   ```
   $ cp /<path_to_ccoctl_output_dir>/manifests/* ./manifests/
   ```

4. 运行以下命令，将 **ccoctl** 工具在 **tls** 目录中生成的私钥复制到安装目录中：

   ```
   $ cp -a /<path_to_ccoctl_output_dir>/tls .
   ```

6.3.4.5. Cluster Network Operator 配置

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

CNO 配置在集群安装过程中从 **Network.config.openshift.io** API 组中的 **Network** API 继承以下字段：

clusterNetwork

从中分配 Pod IP 地址的 IP 地址池。

serviceNetwork

服务的 IP 地址池.

defaultNetwork.type

集群网络插件。**OVNKubernetes** 是安装期间唯一支持的插件。

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

6.3.4.5.1. Cluster Network Operator 配置对象

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

表 6.3. Cluster Network Operator 配置对象

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

spec.clusterNetwork	**array**	用于指定从哪些 IP 地址块分配 Pod IP 地址以及集群中每个节点的子网前缀长度的列表。例如：
		
		``
		spec:
		  clusterNetwork:
		  - cidr: 10.128.0.0/19
		    hostPrefix: 23
		  - cidr: 10.128.32.0/19
		    hostPrefix: 23
		``

spec.serviceNetwork	**array**	服务的 IP 地址块。OpenShift SDN 和 OVN-Kubernetes 网络插件只支持服务网络的一个 IP 地址块。例如：
		
		``
		spec:
		  serviceNetwork:
		  - 172.30.0.0/14
		``
		
		您只能在创建清单前在 **install-config.yaml** 文件中自定义此字段。该值在清单文件中是只读的。

spec.defaultNetwork	**object**	为集群网络配置网络插件。
spec.kubeProxyConfig	**object**	此对象的字段指定 kube-proxy 配置。如果使用 OVN-Kubernetes 集群网络供应商，则 kube-proxy 配置不会起作用。

defaultNetwork 对象配置

下表列出了 **defaultNetwork** 对象的值：

表 6.4. defaultNetwork 对象

字段	类型	描述

字段	类型	描述
type	**字符串**	**OVNKubernetes**。Red Hat OpenShift Networking 网络插件在安装过程中被选择。此值在集群安装后无法更改。  **注意** OpenShift Container Platform 默认使用 OVN-Kubernetes 网络插件。OpenShift SDN 不再作为新集群的安装选择提供。
ovnKubernetesConfig	**object**	此对象仅对 OVN-Kubernetes 网络插件有效。

配置 OVN-Kubernetes 网络插件
下表描述了 OVN-Kubernetes 网络插件的配置字段：

表 6.5. ovnKubernetesConfig object

字段	类型	描述
mtu	**integer**	Geneve（通用网络虚拟化封装）覆盖网络的最大传输单元 (MTU)。这根据主网络接口的 MTU 自动探测。您通常不需要覆盖检测到的 MTU。 如果自动探测的值不是您期望的值，请确认节点上主网络接口上的 MTU 是否正确。您不能使用这个选项更改节点上主网络接口的 MTU 值。 如果集群中不同节点需要不同的 MTU 值，则必须将此值设置为**比** 集群中的最低 MTU 值小 100。例如，如果集群中的某些节点的 MTU 为 **9001**，而某些节点的 MTU 为 **1500**，则必须将此值设置为 **1400**。
genevePort	**integer**	用于所有 Geneve 数据包的端口。默认值为 **6081**。此值在集群安装后无法更改。
ipsecConfig	**object**	指定用于自定义 IPsec 配置的配置对象。
ipv4	**object**	为 IPv4 设置指定配置对象。
ipv6	**object**	为 IPv6 设置指定配置对象。
policyAuditConfig	**object**	指定用于自定义网络策略审计日志的配置对象。如果未设置，则使用默认的审计日志设置。

字段	类型	描述
gatewayConfig	**object**	可选：指定一个配置对象来自定义如何将出口流量发送到节点网关。  **注意** 在迁移出口流量时，工作负载和服务流量会受到一定影响，直到 Cluster Network Operator (CNO) 成功推出更改。

表 6.6. **ovnKubernetesConfig.ipv4** object

字段	类型	描述
internalTransitSwitchSubnet	字符串	如果您的现有网络基础架构与 **100.88.0.0/16** IPv4 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。启用东西流量的分布式传输交换机的子网。此子网不能与 OVN-Kubernetes 或主机本身使用的任何其他子网重叠。必须足够大，以适应集群中的每个节点一个 IP 地址。 默认值为 **100.88.0.0/16**。
internalJoinSubnet	字符串	如果您的现有网络基础架构与 **100.64.0.0/16** IPv4 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。您必须确保 IP 地址范围没有与 OpenShift Container Platform 安装使用的任何其他子网重叠。IP 地址范围必须大于可添加到集群的最大节点数。例如，如果 **clusterNetwork.cidr** 值为 **10.128.0.0/14**，并且 **clusterNetwork.hostPrefix** 值为 **/23**，则最大节点数量为 **2^(23-14)=512**。 默认值为 **100.64.0.0/16**。

表 6.7. **ovnKubernetesConfig.ipv6** object

字段	类型	描述
internalTransitSwitchSubnet	字符串	如果您的现有网络基础架构与 **fd97::/64** IPv6 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。启用东西流量的分布式传输交换机的子网。此子网不能与 OVN-Kubernetes 或主机本身使用的任何其他子网重叠。必须足够大，以适应集群中的每个节点一个 IP 地址。 默认值为 **fd97::/64**。

字段	类型	描述
internalJoinSubnet	字符串	如果您的现有网络基础架构与 **fd98::/64** IPv6 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。您必须确保 IP 地址范围没有与 OpenShift Container Platform 安装使用的任何其他子网重叠。IP 地址范围必须大于可添加到集群的最大节点数。 默认值为 **fd98::/64**。

表 6.8. **policyAuditConfig** object

字段	类型	描述
rateLimit	整数	每个节点每秒生成一次的消息数量上限。默认值为每秒 **20** 条消息。
maxFileSize	整数	审计日志的最大大小，以字节为单位。默认值为 **50000000** 或 50 MB。
maxLogFiles	整数	保留的日志文件的最大数量。
目的地	字符串	以下附加审计日志目标之一： **libc** 主机上的 journald 进程的 libc **syslog（）** 函数。 **UDP:<host>:<port>** 一个 syslog 服务器。将 **<host>:<port>** 替换为 **syslog 服务器的主机** 和端口。 **Unix:<file>** 由 **<file>** 指定的 Unix 域套接字文件。 **null** 不要将审计日志发送到任何其他目标。
syslogFacility	字符串	syslog 工具，如 as **kern**，如 RFC5424 定义。默认值为 **local0**。

表 6.9. **gatewayConfig** object

字段	类型	描述

字段	类型	描述
routingViaHost	**布尔值**	将此字段设置为 **true**，将来自 pod 的出口流量发送到主机网络堆栈。对于依赖于在内核路由表中手动配置路由的高级别安装和应用程序，您可能需要将出口流量路由到主机网络堆栈。默认情况下，出口流量在 OVN 中进行处理以退出集群，不受内核路由表中的特殊路由的影响。默认值为 **false**。 此字段与 Open vSwitch 硬件卸载功能有交互。如果将此字段设置为 **true**，则不会获得卸载的性能优势，因为主机网络堆栈会处理出口流量。
ipForwarding	**object**	您可以使用 **Network** 资源中的 **ipForwarding** 规格来控制 OVN-Kubernetes 管理接口上所有流量的 IP 转发。指定 **Restricted** 只允许 Kubernetes 相关流量的 IP 转发。指定 **Global** 以允许转发所有 IP 流量。对于新安装，默认值为 **Restricted**。对于到 OpenShift Container Platform 4.14 或更高版本的更新，默认值为 **Global**。
ipv4	**object**	可选：指定一个对象来为主机配置内部 OVN-Kubernetes 伪装地址，以服务 IPv4 地址的流量。
ipv6	**object**	可选：指定一个对象来为主机配置内部 OVN-Kubernetes 伪装地址，以服务 IPv6 地址的流量。

表 6.10. **gatewayConfig.ipv4** 对象

字段	类型	描述
internalMasqueradeSubnet	**字符串**	内部使用的伪装 IPv4 地址，以启用主机服务流量。主机配置了这些 IP 地址和共享网关网桥接口。默认值为 **169.254.169.0/29**。

表 6.11. **gatewayConfig.ipv6** 对象

字段	类型	描述
internalMasqueradeSubnet	**字符串**	内部使用的伪装 IPv6 地址，以启用主机服务流量。主机配置了这些 IP 地址和共享网关网桥接口。默认值为 **fd69::/125**。

表 6.12. **ipsecConfig** 对象

字段	类型	描述

字段	类型	描述
模式	**字符串**	指定 IPsec 实现的行为。必须是以下值之一： • **Disabled**: 在集群节点上不启用 IPsec。 • **External** ：对于带有外部主机的网络流量，启用 IPsec。 • **Full**: IPsec 为带有外部主机的 pod 流量和网络流量启用 IPsec。

启用 IPSec 的 OVN-Kubernetes 配置示例

```
defaultNetwork:
  type: OVNKubernetes
  ovnKubernetesConfig:
    mtu: 1400
    genevePort: 6081
      ipsecConfig:
        mode: Full
```

重要

使用 OVNKubernetes 可能会导致 IBM Power® 上的堆栈耗尽问题。

kubeProxyConfig 对象配置（仅限 OpenShiftSDN 容器网络接口）
kubeProxyConfig 对象的值在下表中定义：

表 6.13. kubeProxyConfig object

字段	类型	描述
iptablesSyncPeriod	**字符串**	**iptables** 规则的刷新周期。默认值为 **30s**。有效的后缀包括 **s**、**m** 和 **h**，具体参见 Go 时间 包 文档。  **注意** 由于 OpenShift Container Platform 4.3 及更高版本中引进了性能改进，不再需要调整 **iptablesSyncPeriod** 参数。

proxyArguments.iptables-min-sync-period	**array**	刷新 **iptables** 规则前的最短持续时间。此字段确保刷新的频率不会过于频繁。有效的后缀包括 **s**、**m** 和 **h**，具体参见 Go **time** 软件包。默认值为：
		`kubeProxyConfig:`
		`  proxyArguments:`
		`  iptables-min-sync-period:`
		`  - 0s`

6.3.4.6. 指定高级网络配置

您可以使用网络插件的高级网络配置将集群集成到现有网络环境中。您只能在安装集群前指定高级网络配置。

重要

不支持通过修改安装程序创建的 OpenShift Container Platform 清单文件来自定义网络配置。支持应用您创建的清单文件，如以下流程中所示。

先决条件

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

流程

1. 进入包含安装程序的目录并创建清单：

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

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

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

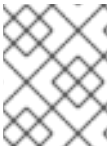
```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
```

3. 在 **cluster-network-03-config.yml** 文件中指定集群的高级网络配置，如下例所示：

   为 OVN-Kubernetes 网络供应商启用 IPsec

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  defaultNetwork:
    ovnKubernetesConfig:
      ipsecConfig:
        mode: Full
```

4. 可选：备份 **manifests/cluster-network-03-config.yml** 文件。在创建 Ignition 配置文件时，安装程序会消耗 **manifests/** 目录。



注意

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

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

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

先决条件

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

流程

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

1. 进入包含安装程序的目录并创建清单：

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

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

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

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

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

创建该文件后，几个网络配置文件位于 **manifests/** 目录中，如下所示：

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

输出示例

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

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

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

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

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

6.3.4.8. 使用 OVN-Kubernetes 配置混合网络

您可以将集群配置为使用 OVN-Kubernetes 网络插件的混合网络。这允许支持不同节点网络配置的混合集群。

先决条件

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

流程

1. 进入包含安装程序的目录并创建清单：

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

其中：

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

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

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

其中：

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

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

指定混合网络配置

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

```
ovnKubernetesConfig:
  hybridOverlayConfig:
    hybridClusterNetwork: 1
    - cidr: 10.132.0.0/14
      hostPrefix: 23
```

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

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

6.3.4.9. 部署集群

您可以在兼容云平台上安装 OpenShift Container Platform。

重要

在初始安装过程中，您只能运行安装程序的 **create cluster** 命令一次。

先决条件

- 您已使用托管集群的云平台配置了帐户。
- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。
- 已确认主机上的云供应商帐户具有部署集群的正确权限。权限不正确的帐户会导致安装过程失败，并显示包括缺失权限的错误消息。

流程

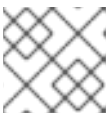
1. 进入包含安装程序的目录并初始化集群部署：

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

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

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

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



注意

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

验证

当集群部署成功完成时：

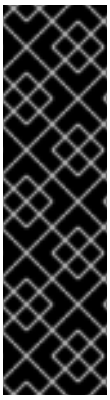
- 终端会显示用于访问集群的说明，包括指向 Web 控制台和 **kubeadmin** 用户的凭证的链接。
- 凭证信息还会输出到 **<installation_directory>/.openshift_install.log**.

重要

不要删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

输出示例

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



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

6.3.4.10. 使用 CLI 登录集群

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

先决条件

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

流程

1. 导出 **kubeadmin** 凭证：

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

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

2. 验证您可以使用导出的配置成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

6.3.4.11. 使用 Web 控制台登录到集群

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

先决条件

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

流程

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

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

注意

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

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

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

注意

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

输出示例

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

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

其他资源

- 如需有关 访问和了解 OpenShift Container Platform Web 控制台的更多详情，请参阅 访问 Web 控制台。

6.3.4.12. 后续步骤

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

6.3.5. 在受限网络中的 AWS 上安装集群

在 OpenShift Container Platform 版本 4.16 中，您可以通过在现有 Amazon Virtual Private Cloud（VPC）上创建安装发行内容的内部镜像在受限网络中的 Amazon Web Services（AWS）上安装集群。

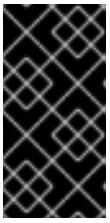
6.3.5.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读有关 选择集群安装方法的文档，并为用户准备它。
- 您已 将断开连接的安装的镜像镜像 到 registry，并获取了 OpenShift Container Platform 版本的 **imageContentSources** 数据。

重要

由于安装介质位于堡垒主机上，因此请使用该计算机完成所有安装步骤。

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



重要

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

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

注意

如果要配置代理，请务必查看此站点列表。

6.3.5.2. 关于在受限网络中安装

在 OpenShift Container Platform 4.16 中，可以执行不需要有效的互联网连接来获取软件组件的安装。受限网络安装可以使用安装程序置备的基础架构或用户置备的基础架构完成，具体取决于您要安装集群的云平台。

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

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

6.3.5.2.1. 其他限制

受限网络中的集群有以下额外限制和限制：

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

6.3.5.3. 关于使用自定义 VPC

在 OpenShift Container Platform 4.16 中，您可以在 Amazon Web Services (AWS)的现有 Amazon Virtual Private Cloud (VPC)中将集群部署到现有子网中。通过将 OpenShift Container Platform 部署到现有的 AWS VPC 中，您可能会避开新帐户中的限制，或者更容易地利用公司所设置的操作限制。如果您无法获得您自己创建 VPC 所需的基础架构创建权限，请使用这个安装选项。

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

6.3.5.3.1. 使用 VPC 的要求

安装程序不再创建以下组件：

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

注意

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

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

安装程序无法：

- 分割网络范围供集群使用。
- 设置子网的路由表。
- 设置 VPC 选项，如 DHCP。

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

您的 VPC 必须满足以下特征：

- VPC 不能使用 **kubernetes.io/cluster/.*: owned**, **Name**, 和 **openshift.io/cluster** 标签。
  安装程序会修改子网以添加 **kubernetes.io/cluster/.*: shared** 标签，因此您的子网必须至少有一个可用的空闲标签插槽。请参阅 AWS 文档中的 标签限制 部分，以确认安装程序可以为您指定的每个子网添加标签。您不能使用 **Name** 标签，因为它与 EC2 **Name** 字段重叠，且安装失败。
- 如果要将 OpenShift Container Platform 集群扩展到 AWS Outpost 并具有现有的 Outpost 子网，现有的子网必须使用 **kubernetes.io/cluster/unmanaged: true** 标签。如果您没有应用此标签，因为 Cloud Controller Manager 在 Outpost 子网中创建服务负载均衡器，安装会失败，这是不受支持的配置。
- 您需要在您的 VPC 中启用 **enableDnsSupport** 和 **enableDnsHostnames** 属性，以便集群可以使用附加到 VPC 中的 Route 53 区来解析集群内部的 DNS 记录。请参阅 AWS 文档中的您的 VPC 中的 DNS 支持部分 。
  如果要使用您自己的 Route 53 托管私有区，您必须在安装集群前将现有托管区与 VPC 相关联。您可以使用 **install-config.yaml** 文件中的 **platform.aws.hostedZone** 和 **platform.aws.hostedZoneRole** 字段定义托管区。您可以通过与安装集群的帐户共享来使用来自另一个帐户的私有托管区。如果使用另一个帐户的私有托管区，则必须使用 **Passthrough** 或 **Manual** 凭证模式。

如果您在断开连接的环境中工作，则无法访问 EC2、ELB 和 S3 端点的公共 IP 地址。根据您要在安装过程中限制互联网流量的级别，有以下配置选项：

选项 1：创建 VPC 端点
创建 VPC 端点，并将其附加到集群使用的子网。将端点命名为如下：

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

通过这个选项，网络流量在 VPC 和所需的 AWS 服务之间保持私有。

选项 2：创建一个没有 VPC 端点的代理
作为安装过程的一部分，您可以配置 HTTP 或 HTTPS 代理。使用此选项时，互联网流量会通过代理访问所需的 AWS 服务。

选项 3：创建带有 VPC 端点的代理
作为安装过程的一部分，您可以使用 VPC 端点配置 HTTP 或 HTTPS 代理。创建 VPC 端点，并将其附加到集群使用的子网。将端点命名为如下：

- **ec2.<aws_region>.amazonaws.com**
- **elasticloadbalancing.<aws_region>.amazonaws.com**

- **s3.<aws_region>.amazonaws.com**

在 **install-config.yaml** 文件中配置代理时，将这些端点添加到 **noProxy** 字段。通过这个选项，代理会阻止集群直接访问互联网。但是，您的 VPC 和所需的 AWS 服务之间网络流量保持私有。

所需的 VPC 组件

您必须提供合适的 VPC 和子网，以便与您的机器通信。

<table>
<tr><th>组件</th><th>AWS 类型</th><th>描述</th></tr>
<tr><td>VPC</td><td>AWS::EC2::VPC
AWS::EC2::VPCEndpoint</td><td>您必须提供一个公共 VPC 供集群使用。VPC 使用引用每个子网的路由表的端点，以改进与托管在 S3 中的 registry 的通信。</td></tr>
<tr><td>公共子网</td><td>AWS::EC2::Subnet
AWS::EC2::SubnetNetworkAclAssociation</td><td>您的 VPC 必须有 1 到 3 个可用区的公共子网，并将其与适当的入口规则关联。</td></tr>
<tr><td>互联网网关</td><td>AWS::EC2::InternetGateway
AWS::EC2::VPCGatewayAttachment
AWS::EC2::RouteTable
AWS::EC2::Route
AWS::EC2::SubnetRouteTableAssociation
AWS::EC2::NatGateway
AWS::EC2::EIP</td><td>您必须有一个公共互联网网关，以及附加到 VPC 的公共路由。在提供的模板中，每个公共子网都有一个具有 EIP 地址的 NAT 网关。这些 NAT 网关允许集群资源（如专用子网实例）访问互联网，而有些受限网络或代理场景则不需要它们。</td></tr>
<tr><td>网络访问控制</td><td>AWS::EC2::NetworkAcl
AWS::EC2::NetworkAclEntry</td><td>您必须允许 VPC 访问下列端口：<table><tr><th>端口</th><th>原因</th></tr><tr><td>80</td><td>入站 HTTP 流量</td></tr><tr><td>443</td><td>入站 HTTPS 流量</td></tr><tr><td>22</td><td>入站 SSH 流量</td></tr><tr><td>1024 - 65535</td><td>入站临时流量</td></tr><tr><td>0 - 65535</td><td>出站临时流量</td></tr></table></td></tr>
</table>

组件	AWS 类型	描述
专用子网	• **AWS::EC2::Subnet** • **AWS::EC2::RouteTable** • **AWS::EC2::SubnetRouteTableAssociation**	您的 VPC 可以具有私有子网。提供的 CloudFormation 模板可为 1 到 3 个可用区创建专用子网。如果您使用专用子网，必须为其提供适当的路由和表。

6.3.5.3.2. VPC 验证

要确保您提供的子网适合您的环境，安装程序会确认以下信息：

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

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

6.3.5.3.3. 权限划分

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

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

6.3.5.3.4. 集群间隔离

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

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

6.3.5.4. 创建安装配置文件

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

先决条件

- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。对于受限网络安装，这些文件位于您的镜像主机上。
- 您有创建镜像 registry 期间生成的 **imageContentSources** 值。
- 您已获取了镜像 registry 的证书内容。

流程

1. 创建 **install-config.yaml** 文件。
   a. 进入包含安装程序的目录并运行以下命令：

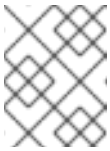
```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

   1 对于 **<installation_directory>**，请指定要存储安装程序创建的文件的目录名称。

   在指定目录时：

   - 验证该目录是否具有**执行**权限。在安装目录中运行 Terraform 二进制文件需要这个权限。
   - 使用空目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

   b. 在提示符处，提供云的配置详情：
      i. 可选： 选择用于访问集群机器的 SSH 密钥。



注意

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

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

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

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

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

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

b. 添加 **additionalTrustBundle** 参数和值。

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

ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
  -----END CERTIFICATE-----
```

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

c. 在以下位置定义 VPC 安装集群的子网：

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

d. 添加镜像内容资源，类似于以下 YAML 摘录：

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

对于这些值，请使用您在创建镜像 registry 时记录的 **imageContentSources**。

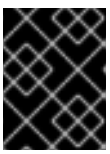
e. 可选：将发布策略设置为 **Internal**：

```
publish: Internal
```

通过设置这个选项，您可以创建一个内部 Ingress Controller 和一个私有负载均衡器。

3. 对您需要的 **install-config.yaml** 文件进行任何其他修改。
有关参数的更多信息，请参阅"安装配置参数"。

4. 备份 **install-config.yaml** 文件，以便您可以使用它安装多个集群。



重要

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

其他资源

- AWS 的安装配置参数

6.3.5.4.1. 集群安装的最低资源要求

每台集群机器都必须满足以下最低要求：

表 6.14. 最低资源要求

机器	操作系统	vCPU [1]	虚拟内存	Storage	每秒输入/输出 (IOPS) [2]
bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane（控制平面）	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS、RHEL 8.6 及更新版本 [3]	2	8 GB	100 GB	300

1. 当未启用并发多线程(SMT)或超线程时，一个 vCPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比例：（每个内核数的线程）× sockets = vCPU。

2. OpenShift Container Platform 和 Kubernetes 对磁盘性能非常敏感，建议使用更快的存储速度，特别是 control plane 节点上需要 10 ms p99 fsync 持续时间的 etcd。请注意，在许多云平台上，存储大小和 IOPS 可一起扩展，因此您可能需要过度分配存储卷来获取足够的性能。

3. 与所有用户置备的安装一样，如果您选择在集群中使用 RHEL 计算机器，则负责所有操作系统生命周期管理和维护，包括执行系统更新、应用补丁和完成所有其他必要的任务。RHEL 7 计算机器的使用已弃用，并已在 OpenShift Container Platform 4.10 及更新的版本中删除。

注意

从 OpenShift Container Platform 版本 4.13 开始，RHCOS 基于 RHEL 版本 9.2，它更新了微架构要求。以下列表包含每个架构需要的最小指令集架构 (ISA)：

- x86-64 体系结构需要 x86-64-v2 ISA
- ARM64 架构需要 ARMv8.0-A ISA
- IBM Power 架构需要 Power 9 ISA
- s390x 架构需要 z14 ISA

如需更多信息，请参阅 RHEL 架构。

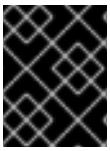
如果平台的实例类型满足集群机器的最低要求，则 OpenShift Container Platform 支持使用它。

其他资源

- 优化存储

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

您可以自定义安装配置文件 (**install-config.yaml**)，以指定有关 OpenShift Container Platform 集群平台的更多详细信息，或修改所需参数的值。



重要

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

```
apiVersion: v1
baseDomain: example.com (1)
credentialsMode: Mint (2)
controlPlane: (3) (4)
  hyperthreading: Enabled (5)
  name: master
  platform:
    aws:
      zones:
      - us-west-2a
      - us-west-2b
      rootVolume:
        iops: 4000
        size: 500
        type: io1 (6)
      metadataService:
        authentication: Optional (7)
      type: m6i.xlarge
  replicas: 3
compute: (8)
- hyperthreading: Enabled (9)
  name: worker
  platform:
    aws:
      rootVolume:
        iops: 2000
        size: 500
        type: io1 (10)
      metadataService:
        authentication: Optional (11)
      type: c5.4xlarge
      zones:
      - us-west-2c
  replicas: 3
metadata:
  name: test-cluster (12)
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
```

```
  - cidr: 10.0.0.0/16
  networkType: OVNKubernetes (13)
  serviceNetwork:
  - 172.30.0.0/16
platform:
  aws:
    region: us-west-2 (14)
    propagateUserTags: true (15)
    userTags:
      adminContact: jdoe
      costCenter: 7536
    subnets: (16)
    - subnet-1
    - subnet-2
    - subnet-3
    amiID: ami-0c5d3e03c0ab9b19a (17)
    serviceEndpoints: (18)
      - name: ec2
        url: https://vpce-id.ec2.us-west-2.vpce.amazonaws.com
    hostedZone: Z3URY6TWQ91KVV (19)
fips: false (20)
sshKey: ssh-ed25519 AAAA... (21)
pullSecret: '{"auths":{"<local_registry>": {"auth": "<credentials>","email": "you@example.com"}}}' (22)
additionalTrustBundle: | (23)
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
imageContentSources: (24)
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-v4.0-art-dev
```

(1) (12) (14) 必需。安装程序会提示您输入这个值。

(2) 可选：添加此参数来强制 Cloud Credential Operator (CCO) 使用指定的模式。默认情况下，CCO 使用 **kube-system** 命名空间中的 root 凭证来动态尝试决定凭证的功能。有关 CCO 模式的详情，请参阅*身份验证和授权*指南中的"About the Cloud Credential Operator"部分。

(3) (8) (15) 如果没有提供这些参数和值，安装程序会提供默认值。

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

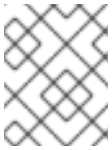
(5) (9) 是否要启用或禁用并发多线程或 **超线程**。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将 参数值设置为 **Disabled** 来禁用它。如果在某些集群机器中禁用并发多线程，则必须在所有集群机器中禁用它。

重要

如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。如果您对机器禁用并发多线程，请使用较大的实例类型，如 **m4.2xlarge** 或 **m5.2xlarge**。

6 10 要为 etcd 配置更快的存储，特别是对于较大的集群，请将存储类型设置为 **io1**，并将 **iops** 设为 **2000**。

7 11 是否需要 Amazon EC2 实例元数据服务 v2 (IMDSv2)。为了要求 IMDSv2，请将参数值设置为 **Required**。要允许使用 IMDSv1 和 IMDSv2，请将参数值设置为 **Optional**。如果没有指定值，则允许 IMDSv1 和 IMDSv2。



注意

在集群安装过程中设置的 control plane 机器的 IMDS 配置只能使用 AWS CLI 更改。可以使用计算机器集来更改计算机器的 IMDS 配置。

13 要安装的集群网络插件。默认值 **OVNKubernetes** 是唯一支持的值。

16 如果您提供自己的 VPC，为集群使用的每个可用区指定子网。

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

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

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

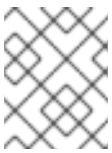
20 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 Red Hat Enterprise Linux (RHEL) 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅在 FIPS 模式中安装该系统。

当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS)时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。

21 您可以选择提供您用来访问集群中机器的 **sshKey** 值。



注意

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

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

23 提供用于镜像 registry 的证书文件内容。

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

6.3.5.4.3. 在安装过程中配置集群范围的代理

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

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 **Proxy** 对象的 **spec.noProxy** 字段中添加站点来绕过代理。

注意

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

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

流程

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

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: ec2.<aws_region>.amazonaws.com,elasticloadbalancing.
<aws_region>.amazonaws.com,s3.<aws_region>.amazonaws.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 **http**。

2 用于创建集群外 HTTPS 连接的代理 URL。

3 要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 **.** 以仅匹配子域。例如，**.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代理。如果您已将 Amazon **EC2**、**Elastic Load Balancing** 和 **S3** VPC 端点添加到 VPC 中，您必须将这些端点添加到 **noProxy** 字段。

4 如果提供，安装程序会在 **openshift-config** 命名空间中生成名为 **user-ca-bundle** 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network

5 可选：决定 **Proxy** 对象的配置以引用 **trustedCA** 字段中 **user-ca-bundle** 配置映射的策略。允许的值是 **Proxyonly** 和 **Always**。仅在配置了 **http/https** 代理时，使用 **Proxyonly** 引用 **user-ca-bundle** 配置映射。使用 **Always** 始终引用 **user-ca-bundle** 配置映射。默认值为 **Proxyonly**。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。

注意

如果安装程序超时，重启并使用安装程序的 **wait-for** 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

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

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

注意

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

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

默认情况下，管理员 secret 存储在 **kube-system** 项目中。如果您在 **install-config.yaml** 文件中将 **credentialsMode** 参数配置为 **Manual**，则必须使用以下替代方案之一：

- 要手动管理长期云凭证，请按照手动创建长期凭证中的步骤操作。
- 要实现在集群外为各个组件管理的短期凭证，请按照配置 A WS 集群以使用短期凭证 中的步骤操作。

6.3.5.5.1. 手动创建长期凭证

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

流程

1. 如果您没有将 **install-config.yaml** 配置文件中的 **credentialsMode** 参数设置为 **Manual**，请修改值，如下所示：

配置文件片段示例

```
apiVersion: v1
baseDomain: example.com
credentialsMode: Manual
```

```
# ...
```

2. 如果您之前还没有创建安装清单文件，请运行以下命令：

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

其中 **<installation_directory>** 是安装程序在其中创建文件的目录。

3. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

4. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 自定义资源 (CR) 列表：

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
  --included \ 1
  --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
  --to=<path_to_directory_for_credentials_requests> 3
```

1. **--included** 参数仅包含特定集群配置所需的清单。
2. 指定 **install-config.yaml** 文件的位置。
3. 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

此命令为每个 **CredentialsRequest** 对象创建一个 YAML 文件。

CredentialsRequest 对象示例

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

5. 在之前生成的 **openshift-install** 清单目录中为 secret 创建 YAML 文件。secret 必须使用在 **spec.secretRef** 中为每个 **CredentialsRequest** 定义的命名空间和 secret 名称存储。

带有 secret 的 CredentialsRequest 对象示例

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component_credentials_request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: AWSProviderSpec
    statementEntries:
    - effect: Allow
      action:
      - s3:CreateBucket
      - s3:DeleteBucket
      resource: "*"
      ...
  secretRef:
    name: <component_secret>
    namespace: <component_namespace>
  ...
```

Secret 对象示例

```
apiVersion: v1
kind: Secret
metadata:
  name: <component_secret>
  namespace: <component_namespace>
data:
  aws_access_key_id: <base64_encoded_aws_access_key_id>
  aws_secret_access_key: <base64_encoded_aws_secret_access_key>
```

重要

在升级使用手动维护凭证的集群前，您必须确保 CCO 处于可升级状态。

6.3.5.5.2. 将 AWS 集群配置为使用短期凭证

要安装配置为使用 AWS 安全令牌服务 (STS) 的集群，您必须配置 CCO 实用程序并为集群创建所需的 AWS 资源。

6.3.5.5.2.1. 配置 Cloud Credential Operator 工具

当 Cloud Credential Operator(CCO)以手动模式运行时，要从集群外部创建和管理云凭证，提取并准备 CCO 实用程序(**ccoctl**)二进制文件。

注意

ccoctl 工具是在 Linux 环境中运行的 Linux 二进制文件。

先决条件

- 您可以访问具有集群管理员权限的 OpenShift Container Platform 帐户。
- 已安装 OpenShift CLI(**oc**)。
- 您已为 **ccoctl** 工具创建了用于以下权限的 AWS 帐户：

  例 6.26. 所需的 AWS 权限

  所需的 iam 权限

  - **iam:CreateOpenIDConnectProvider**
  - **iam:CreateRole**
  - **iam:DeleteOpenIDConnectProvider**
  - **iam:DeleteRole**
  - **iam:DeleteRolePolicy**
  - **iam:GetOpenIDConnectProvider**
  - **iam:GetRole**
  - **iam:GetUser**
  - **iam:ListOpenIDConnectProviders**
  - **iam:ListRolePolicies**
  - **iam:ListRoles**
  - **iam:PutRolePolicy**
  - **iam:TagOpenIDConnectProvider**
  - **iam:TagRole**

  所需的 s3 权限

  - **s3:CreateBucket**
  - **s3:DeleteBucket**
  - **s3:DeleteObject**
  - **s3:GetBucketAcl**
  - **s3:GetBucketTagging**
  - **s3:GetObject**
  - **s3:GetObjectAcl**

- **s3:GetObjectTagging**
- **s3:ListBucket**
- **s3:PutBucketAcl**
- **s3:PutBucketPolicy**
- **s3:PutBucketPublicAccessBlock**
- **s3:PutBucketTagging**
- **s3:PutObject**
- **s3:PutObjectAcl**
- **s3:PutObjectTagging**

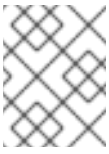
所需的 cloudfront 权限

- **cloudfront:ListCloudFrontOriginAccessIdentities**
- **cloudfront:ListDistributions**
- **cloudfront:ListTagsForResource**

如果您计划通过公共 CloudFront 发行版 URL 将 OIDC 配置存储在 IAM 身份提供程序访问的私有 S3 存储桶中，则运行 **ccoctl** 工具的 AWS 帐户需要以下额外权限：

例 6.27. 使用 CloudFront 私有 S3 存储桶的额外权限

- **cloudfront:CreateCloudFrontOriginAccessIdentity**
- **cloudfront:CreateDistribution**
- **cloudfront:DeleteCloudFrontOriginAccessIdentity**
- **cloudfront:DeleteDistribution**
- **cloudfront:GetCloudFrontOriginAccessIdentity**
- **cloudfront:GetCloudFrontOriginAccessIdentityConfig**
- **cloudfront:GetDistribution**
- **cloudfront:TagResource**
- **cloudfront:UpdateDistribution**



注意

在使用 **ccoctl aws create-all** 命令处理凭证请求时，这些额外权限支持使用 **--create-private-s3-bucket** 选项。

流程

1. 运行以下命令，为 OpenShift Container Platform 发行镜像设置变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. 运行以下命令，从 OpenShift Container Platform 发行镜像获取 CCO 容器镜像：

```
$ CCO_IMAGE=$(oc adm release info --image-for='cloud-credential-operator'
$RELEASE_IMAGE -a ~/.pull-secret)
```

注意

确保 **$RELEASE_IMAGE** 的架构与将使用 **ccoctl** 工具的环境架构相匹配。

3. 运行以下命令，将 CCO 容器镜像中的 **ccoctl** 二进制文件提取到 OpenShift Container Platform 发行镜像中：

```
$ oc image extract $CCO_IMAGE \
  --file="/usr/bin/ccoctl.<rhel_version>" \ 1
  -a ~/.pull-secret
```

1 对于 **<rhel_version>**，请指定与主机使用的 Red Hat Enterprise Linux (RHEL) 版本对应的值。如果没有指定值，则默认使用 **ccoctl.rhel8**。以下值有效：

- **rhel8**: 为使用 RHEL 8 的主机指定这个值。
- **rhel9** ：为使用 RHEL 9 的主机指定这个值。

4. 运行以下命令更改权限以使 **ccoctl** 可执行：

```
$ chmod 775 ccoctl.<rhel_version>
```

验证

- 要验证 **ccoctl** 是否可使用，请运行以下命令来显示帮助文件：

```
$ ccoctl --help
```

ccoctl --help 的输出

```
OpenShift credentials provisioning tool

Usage:
  ccoctl [command]

Available Commands:
  aws         Manage credentials objects for AWS cloud
  azure       Manage credentials objects for Azure
  gcp         Manage credentials objects for Google cloud
  help        Help about any command
  ibmcloud    Manage credentials objects for IBM Cloud
```

```
  nutanix      Manage credentials objects for Nutanix

Flags:
  -h, --help   help for ccoctl

Use "ccoctl [command] --help" for more information about a command.
```

6.3.5.5.2.2. 使用 Cloud Credential Operator 实用程序创建 AWS 资源

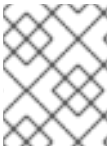
创建 AWS 资源时有以下选项：

- 您可以使用 **ccoctl aws create-all** 命令自动创建 AWS 资源。这是创建资源的最快速方法。请参阅 使用单个命令创建 AWS 资源。
- 如果您需要在修改 AWS 资源前查看 **ccoctl** 工具创建的 JSON 文件，或者 **ccoctl** 工具用于创建 AWS 资源的过程无法自动满足组织的要求，您可以单独创建 AWS 资源。请参阅 单独创建 AWS 资源。

6.3.5.5.2.2.1. 使用单个命令创建 AWS 资源

如果 **ccoctl** 工具用于创建 AWS 资源的过程自动满足机构的要求，您可以使用 **ccoctl aws create-all** 命令自动创建 AWS 资源。

否则，您可以单独创建 AWS 资源。如需更多信息，请参阅"单独创建 AWS 资源"。



注意

默认情况下，**ccoctl** 在运行命令的目录中创建对象。要在其他目录中创建对象，请使用 **--output-dir** 标志。此流程使用 **<path_to_ccoctl_output_dir>** 来引用这个目录。

先决条件

您必须：

- 提取并准备好 **ccoctl** 二进制文件。

流程

1. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 对象列表：

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
  --included \ 1
  --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
  --to=<path_to_directory_for_credentials_requests> 3
```

1 **--included** 参数仅包含特定集群配置所需的清单。

2 指定 **install-config.yaml** 文件的位置。

3 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

注意

此命令可能需要一些时间才能运行。

3. 运行以下命令，使用 **ccoctl** 工具处理所有 **CredentialsRequest** 对象：

```
$ ccoctl aws create-all \
  --name=<name> \ 1
  --region=<aws_region> \ 2
  --credentials-requests-dir=<path_to_credentials_requests_directory> \ 3
  --output-dir=<path_to_ccoctl_output_dir> \ 4
  --create-private-s3-bucket 5
```

1 指定用于标记创建用于跟踪的任何云资源的名称。

2 指定在其中创建云资源的 AWS 区域。

3 指定包含组件 **CredentialsRequest** 对象文件的目录。

4 可选：指定您希望 **ccoctl** 实用程序在其中创建对象的目录。默认情况下，实用程序在运行命令的目录中创建对象。

5 可选：默认情况下，**ccoctl** 实用程序将 OpenID Connect (OIDC) 配置文件存储在公共 S3 存储桶中，并使用 S3 URL 作为公共 OIDC 端点。要将 OIDC 配置存储在 IAM 身份提供程序通过公共 CloudFront 发行版 URL 访问的专用 S3 存储桶中，请使用 **--create-private-s3-bucket** 参数。

注意

如果您的集群使用 **TechPreviewNoUpgrade** 功能集启用的技术预览功能，则必须包含 **--enable-tech-preview** 参数。

验证

- 要验证 OpenShift Container Platform secret 是否已创建，列出 **<path_to_ccoctl_output_dir>/manifests** 目录中的文件：

```
$ ls <path_to_ccoctl_output_dir>/manifests
```

输出示例

```
cluster-authentication-02-config.yaml
openshift-cloud-credential-operator-cloud-credential-operator-iam-ro-creds-credentials.yaml
openshift-cloud-network-config-controller-cloud-credentials-credentials.yaml
openshift-cluster-api-capa-manager-bootstrap-credentials-credentials.yaml
openshift-cluster-csi-drivers-ebs-cloud-credentials-credentials.yaml
```

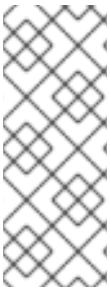
```
openshift-image-registry-installer-cloud-credentials-credentials.yaml
openshift-ingress-operator-cloud-credentials-credentials.yaml
openshift-machine-api-aws-cloud-credentials-credentials.yaml
```

您可以通过查询 AWS 来验证是否已创建 IAM 角色。如需更多信息，请参阅有关列出 IAM 角色的 AWS 文档。

6.3.5.5.2.2.2. 单独创建 AWS 资源

您可以使用 **ccoctl** 工具单独创建 AWS 资源。这个选项对于在不同用户或部门之间创建这些资源的组织可能很有用。

否则，您可以使用 **ccoctl aws create-all** 命令自动创建 AWS 资源。如需更多信息，请参阅"使用单个命令创建 AWS 资源"。



注意

默认情况下，**ccoctl** 在运行命令的目录中创建对象。要在其他目录中创建对象，请使用 **--output-dir** 标志。此流程使用 **<path_to_ccoctl_output_dir>** 来引用这个目录。

有些 **ccoctl** 命令会发出 AWS API 调用来创建或修改 AWS 资源。您可以使用 **--dry-run** 标志来避免 API 调用。使用此标志可在本地文件系统中创建 JSON 文件。您可以使用 **--cli-input-json** 参数查看和修改 JSON 文件，然后使用 AWS CLI 工具应用它们。

先决条件

- 提取并准备 **ccoctl** 二进制文件。

流程

1. 运行以下命令，生成用于为集群设置 OpenID Connect 供应商的公共和私有 RSA 密钥文件：

```
$ ccoctl aws create-key-pair
```

输出示例

```
2021/04/13 11:01:02 Generating RSA keypair
2021/04/13 11:01:03 Writing private key to /<path_to_ccoctl_output_dir>/serviceaccount-signer.private
2021/04/13 11:01:03 Writing public key to /<path_to_ccoctl_output_dir>/serviceaccount-signer.public
2021/04/13 11:01:03 Copying signing key for use by installer
```

其中 **serviceaccount-signer.private** 和 **serviceaccount-signer.public** 是生成的密钥文件。

此命令还会在 **/<path_to_ccoctl_output_dir>/tls/bound-service-account-signing-key.key** 中创建集群在安装过程中所需的私钥。

2. 运行以下命令，在 AWS 上创建 OpenID Connect 身份提供程序和 S3 存储桶：

```
$ ccoctl aws create-identity-provider \
  --name=<name> \ 1
  --region=<aws_region> \ 2
  --public-key-file=<path_to_ccoctl_output_dir>/serviceaccount-signer.public 3
```

■

1. **<name>** 是用于标记为跟踪而创建的云资源的名称。
2. **<aws-region>** 是将要在其中创建云资源的 AWS 区域。
3. **<path_to_ccoctl_output_dir>** 是 **ccoctl aws create-key-pair** 命令生成的公钥文件的路径。

输出示例

```
2021/04/13 11:16:09 Bucket <name>-oidc created
2021/04/13 11:16:10 OpenID Connect discovery document in the S3 bucket <name>-oidc at
.well-known/openid-configuration updated
2021/04/13 11:16:10 Reading public key
2021/04/13 11:16:10 JSON web key set (JWKS) in the S3 bucket <name>-oidc at keys.json
updated
2021/04/13 11:16:18 Identity Provider created with ARN: arn:aws:iam::
<aws_account_id>:oidc-provider/<name>-oidc.s3.<aws_region>.amazonaws.com
```

其中 **openid-configuration** 是发现文档和 **key.json** 是一个 JSON Web 密钥集文件。

此命令还会在 **/<path_to_ccoctl_output_dir>/manifests/cluster-authentication-02-config.yaml** 中创建 YAML 配置文件。此文件为集群生成的服务帐户令牌设置签发者 URL 字段，以便 AWS IAM 身份提供程序信任令牌。

3. 为集群中的每个组件创建 IAM 角色：

   a. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

   ```
   $ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
   ```

   b. 从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 对象列表：

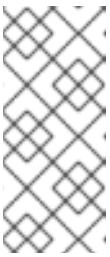
   ```
   $ oc adm release extract \
     --from=$RELEASE_IMAGE \
     --credentials-requests \
     --included \ 1
     --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \
   2
     --to=<path_to_directory_for_credentials_requests> 3
   ```

   1. **--included** 参数仅包含特定集群配置所需的清单。
   2. 指定 **install-config.yaml** 文件的位置。
   3. 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

   c. 运行以下命令，使用 **ccoctl** 工具处理所有 **CredentialsRequest** 对象：

   ```
   $ ccoctl aws create-iam-roles \
     --name=<name> \
     --region=<aws_region> \
   ```

```
  --credentials-requests-dir=<path_to_credentials_requests_directory> \
  --identity-provider-arn=arn:aws:iam::<aws_account_id>:oidc-provider/<name>-oidc.s3.
<aws_region>.amazonaws.com
```



注意

对于使用其他 IAM API 端点的 AWS 环境（如 GovCloud），还必须使用 **--region** 参数指定您的区域。

如果您的集群使用 **TechPreviewNoUpgrade** 功能集启用的技术预览功能，则必须包含 **--enable-tech-preview** 参数。

对于每个 **CredentialsRequest** 对象，**ccoctl** 创建一个带有信任策略的 IAM 角色，该角色与指定的 OIDC 身份提供程序相关联，以及来自 OpenShift Container Platform 发行镜像的每个 **CredentialsRequest** 对象中定义的权限策略。

验证

- 要验证 OpenShift Container Platform secret 是否已创建，列出 **<path_to_ccoctl_output_dir>/manifests** 目录中的文件：

```
$ ls <path_to_ccoctl_output_dir>/manifests
```

输出示例

```
cluster-authentication-02-config.yaml
openshift-cloud-credential-operator-cloud-credential-operator-iam-ro-creds-credentials.yaml
openshift-cloud-network-config-controller-cloud-credentials-credentials.yaml
openshift-cluster-api-capa-manager-bootstrap-credentials-credentials.yaml
openshift-cluster-csi-drivers-ebs-cloud-credentials-credentials.yaml
openshift-image-registry-installer-cloud-credentials-credentials.yaml
openshift-ingress-operator-cloud-credentials-credentials.yaml
openshift-machine-api-aws-cloud-credentials-credentials.yaml
```

您可以通过查询 AWS 来验证是否已创建 IAM 角色。如需更多信息，请参阅有关列出 IAM 角色的 AWS 文档。

6.3.5.5.2.3. 整合 Cloud Credential Operator 实用程序清单

要为单个组件在集群外实现短期安全凭证，您必须将创建 Cloud Credential Operator 实用程序 (**ccoctl**) 的清单文件移到安装程序的正确目录中。

先决条件

- 您已使用托管集群的云平台配置了帐户。
- 您已配置了 Cloud Credential Operator 实用程序 (**ccoctl**)。
- 已使用 **ccoctl** 工具创建了集群所需的云供应商资源。

流程

1. 如果您没有将 **install-config.yaml** 配置文件中的 **credentialsMode** 参数设置为 **Manual**，请修改值，如下所示：

   配置文件片段示例

   ```
   apiVersion: v1
   baseDomain: example.com
   credentialsMode: Manual
   # ...
   ```

2. 如果您之前还没有创建安装清单文件，请运行以下命令：

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

   其中 **<installation_directory>** 是安装程序在其中创建文件的目录。

3. 运行以下命令，将 **ccoctl** 工具生成的清单复制到安装程序创建的 **manifests** 目录中：

   ```
   $ cp /<path_to_ccoctl_output_dir>/manifests/* ./manifests/
   ```

4. 运行以下命令，将 **ccoctl** 工具在 **tls** 目录中生成的私钥复制到安装目录中：

   ```
   $ cp -a /<path_to_ccoctl_output_dir>/tls .
   ```

6.3.5.6. 部署集群

您可以在兼容云平台上安装 OpenShift Container Platform。

重要

在初始安装过程中，您只能运行安装程序的 **create cluster** 命令一次。

先决条件

- 您已使用托管集群的云平台配置了帐户。
- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。
- 已确认主机上的云供应商帐户具有部署集群的正确权限。权限不正确的帐户会导致安装过程失败，并显示包括缺失权限的错误消息。

流程

1. 进入包含安装程序的目录并初始化集群部署：

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

   1. 对于 **<installation_directory>**，请指定自定义 **./install-config.yaml** 文件的位置。
   2. 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不是 **info**。

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

注意

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

验证

当集群部署成功完成时：

- 终端会显示用于访问集群的说明，包括指向 Web 控制台和 **kubeadmin** 用户的凭证的链接。
- 凭证信息还会输出到 **<installation_directory>/.openshift_install.log**.

重要

不要删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

输出示例

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

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

6.3.5.7. 使用 CLI 登录集群

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

先决条件

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

流程

流程

1. 导出 **kubeadmin** 凭证：

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

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

2. 验证您可以使用导出的配置成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

6.3.5.8. 禁用默认的 OperatorHub 目录源

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

流程

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

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

提示

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

6.3.5.9. 后续步骤

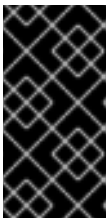
- 验证安装。
- 自定义集群。
- 为 Cluster Samples Operator 和 **must-gather** 工具 配置镜像流。
- 了解如何在 受限网络中使用 Operator Lifecycle Manager(OLM ）。
- 如果您用来安装集群的镜像 registry 具有可信任的 CA，请通过 配置额外的信任存储将其添加到集群中。
- 如果需要，您可以选择不使用远程健康报告。

6.3.6. 在 AWS 上将集群安装到现有的 VPC 中

在 OpenShift Container Platform 版本 4.16 中，您可以在 Amazon Web Services（AWS）上将集群安装到现有 Amazon Virtual Private Cloud（VPC）中。安装程序会置备所需基础架构的其余部分，您可以进一步自定义这些基础架构。要自定义安装，请在安装集群前修改 **install-config.yaml** 文件中的参数。

6.3.6.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读选择集群安装方法并为用户准备它的文档。
- 已将 AWS 帐户配置为托管集群。
- 如果现有的 VPC 由与集群不同的帐户所有，您可以在帐户间共享 VPC。



重要

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

- 如果使用防火墙，将其配置为允许集群需要访问的站点。

6.3.6.2. 关于使用自定义 VPC

在 OpenShift Container Platform 4.16 中，您可以在 Amazon Web Services (AWS)的现有 Amazon Virtual Private Cloud (VPC)中将集群部署到现有子网中。通过将 OpenShift Container Platform 部署到现有的 AWS VPC 中，您可能会避开新帐户中的限制，或者更容易地利用公司所设置的操作限制。如果您无法获得您自己创建 VPC 所需的基础架构创建权限，请使用这个安装选项。

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

6.3.6.2.1. 使用 VPC 的要求

安装程序不再创建以下组件：

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

注意

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

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

安装程序无法：

- 分割网络范围供集群使用。
- 设置子网的路由表。
- 设置 VPC 选项，如 DHCP。

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

您的 VPC 必须满足以下特征：

- 为集群使用的每个可用区创建一个公共和私有子网。每个可用区不能包含多于一个的公共子网和私有子网。有关此类配置的示例，请参阅 AWS 文档中的具有公共和私有子网(NAT)的 VPC。
  记录每个子网 ID。完成安装要求您在 **install-config.yaml** 文件的 **platform** 部分输入这些值。请参阅 AWS 文档中的查找子网 ID。
- VPC 的 CIDR 块必须包含 **Networking.machineCIDR**，它是集群机器的 IP 地址池。子网 CIDR 块必须属于您指定的机器 CIDR。
- VPC 必须附加一个公共互联网网关。对于每个可用区：
  - 公共子网需要路由到互联网网关。
  - 公共子网需要一个具有 EIP 地址的 NAT 网关。
  - 专用子网需要路由到公共子网中的 NAT 网关。
- VPC 不能使用 **kubernetes.io/cluster/.*: owned**, **Name**, 和 **openshift.io/cluster** 标签。
  安装程序会修改子网以添加 **kubernetes.io/cluster/.*: shared** 标签，因此您的子网必须至少有一个可用的空闲标签插槽。请参阅 AWS 文档中的 标签限制 部分，以确认安装程序可以为您指定的每个子网添加标签。您不能使用 **Name** 标签，因为它与 EC2 **Name** 字段重叠，且安装失败。
- 如果要将 OpenShift Container Platform 集群扩展到 AWS Outpost 并具有现有的 Outpost 子网，现有的子网必须使用 **kubernetes.io/cluster/unmanaged: true** 标签。如果您没有应用此标签，因为 Cloud Controller Manager 在 Outpost 子网中创建服务负载均衡器，安装会失败，这是不受支持的配置。
- 您需要在您的 VPC 中启用 **enableDnsSupport** 和 **enableDnsHostnames** 属性，以便集群可以使用附加到 VPC 中的 Route 53 区来解析集群内部的 DNS 记录。请参阅 AWS 文档中的您的 VPC 中的 DNS 支持部分 。
  如果要使用您自己的 Route 53 托管私有区，您必须在安装集群前将现有托管区与 VPC 相关联。您可以使用 **install-config.yaml** 文件中的 **platform.aws.hostedZone** 和 **platform.aws.hostedZoneRole** 字段定义托管区。您可以通过与安装集群的帐户共享来使用来自另一个帐户的私有托管区。如果使用另一个帐户的私有托管区，则必须使用 **Passthrough** 或 **Manual** 凭证模式。

如果您在断开连接的环境中工作，则无法访问 EC2、ELB 和 S3 端点的公共 IP 地址。根据您要在安装过程中限制互联网流量的级别，有以下配置选项：

选项 1：创建 VPC 端点
创建 VPC 端点，并将其附加到集群使用的子网。将端点命名为如下：

- **ec2.<aws_region>.amazonaws.com**
- **elasticloadbalancing.<aws_region>.amazonaws.com**

- **s3.<aws_region>.amazonaws.com**

通过这个选项，网络流量在 VPC 和所需的 AWS 服务之间保持私有。

选项 2：创建一个没有 VPC 端点的代理
作为安装过程的一部分，您可以配置 HTTP 或 HTTPS 代理。使用此选项时，互联网流量会通过代理访问所需的 AWS 服务。

选项 3：创建带有 VPC 端点的代理
作为安装过程的一部分，您可以使用 VPC 端点配置 HTTP 或 HTTPS 代理。创建 VPC 端点，并将其附加到集群使用的子网。将端点命名为如下：

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

在 **install-config.yaml** 文件中配置代理时，将这些端点添加到 **noProxy** 字段。通过这个选项，代理会阻止集群直接访问互联网。但是，您的 VPC 和所需的 AWS 服务之间网络流量保持私有。

所需的 VPC 组件

您必须提供合适的 VPC 和子网，以便与您的机器通信。

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

<table>
<tr><th>组件</th><th>AWS 类型</th><th colspan="2">描述</th></tr>
<tr><td rowspan="6">网络访问控制</td><td rowspan="6">AWS::EC2::NetworkAcl
AWS::EC2::NetworkAclEntry</td><td colspan="2">您必须允许 VPC 访问下列端口：</td></tr>
<tr><td>端口</td><td>原因</td></tr>
<tr><td>80</td><td>入站 HTTP 流量</td></tr>
<tr><td>443</td><td>入站 HTTPS 流量</td></tr>
<tr><td>22</td><td>入站 SSH 流量</td></tr>
<tr><td>1024 - 65535</td><td>入站临时流量</td></tr>
<tr><td>0 - 65535</td><td>出站临时流量</td></tr>
<tr><td>专用子网</td><td>AWS::EC2::Subnet
AWS::EC2::RouteTable
AWS::EC2::SubnetRouteTableAssociation</td><td colspan="2">您的 VPC 可以具有私有子网。提供的 CloudFormation 模板可为 1 到 3 个可用区创建专用子网。如果您使用专用子网，必须为其提供适当的路由和表。</td></tr>
</table>

6.3.6.2.2. VPC 验证

要确保您提供的子网适合您的环境，安装程序会确认以下信息：

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

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

6.3.6.2.3. 权限划分

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

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

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

6.3.6.2.4. 集群间隔离

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

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

6.3.6.2.5. 可选： AWS 安全组

默认情况下，安装程序会创建安全组并将其附加到 control plane 和计算机器。不可修改与默认安全组关联的规则。

但是，您可以将与现有 VPC 关联的其他现有 AWS 安全组应用到 control plane 和计算机器。应用自定义安全组可帮助您满足机构的安全需求，在这种情况下，您需要控制这些机器的传入或传出流量。

作为安装过程的一部分，您可以在部署集群前通过修改 **install-config.yaml** 文件来应用自定义安全组。

如需更多信息，请参阅"将现有 AWS 安全组应用到集群"。

6.3.6.2.6. 在安装到共享 VPC 时修改信任策略

如果使用共享 VPC 安装集群，您可以使用 **Passthrough** 或 **Manual** 凭证模式。您必须在拥有 VPC 的帐户的信任策略中添加用于安装集群的 IAM 角色作为主体。

如果使用 **Passthrough** 模式，请将创建集群的帐户的 Amazon 资源名称 (ARN)（如 **arn:aws:iam::123456789012:user/clustercreator**）添加到信任策略作为主体。

如果使用 **Manual** 模式，请将创建集群的帐户的 ARN，以及集群所有者帐户中的 ingress operator 角色的 ARN，如 **arn:aws:iam::123456789012:role/<cluster-name>-openshift-ingress-operator-cloud-credentials**，添加到信任策略作为主体。

您必须在策略中添加以下操作：

> **例 6.28. 共享 VPC 安装所需的操作**
>
> - **route53:ChangeResourceRecordSets**
> - **route53:ListHostedZones**
> - **route53:ListHostedZonesByName**
> - **route53:ListResourceRecordSets**
> - **route53:ChangeTagsForResource**
> - **route53:GetAccountLimit**

- **route53:GetChange**
- **route53:GetHostedZone**
- **route53:ListTagsForResource**
- **route53:UpdateHostedZoneComment**
- **tag:GetResources**
- **tag:UntagResources**

6.3.6.3. 创建安装配置文件

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

先决条件

- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。

流程

1. 创建 **install-config.yaml** 文件。
   a. 进入包含安装程序的目录并运行以下命令：

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

   1 对于 **<installation_directory>**，请指定要存储安装程序创建的文件的目录名称。

   在指定目录时：

   - 验证该目录是否具有**执行**权限。在安装目录中运行 Terraform 二进制文件需要这个权限。
   - 使用空目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

   b. 在提示符处，提供云的配置详情：
      i. 可选：选择用于访问集群机器的 SSH 密钥。

      注意

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

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

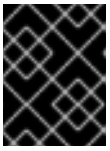
iv. 选择要将集群部署到的 AWS 区域。

v. 选择您为集群配置的 Route 53 服务的基域。

vi. 为集群输入描述性名称。

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

3. 备份 **install-config.yaml** 文件，以便您可以使用它安装多个集群。



重要

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

其他资源

- AWS 的安装配置参数

6.3.6.3.1. 集群安装的最低资源要求

每台集群机器都必须满足以下最低要求：

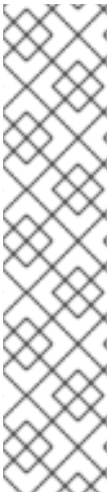
表 6.15. 最低资源要求

机器	操作系统	vCPU [1]	虚拟内存	Storage	每秒输入/输出 (IOPS) [2]
bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane（控制平面）	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS、RHEL 8.6 及更新版本 [3]	2	8 GB	100 GB	300

1. 当未启用并发多线程(SMT)或超线程时，一个 vCPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比例：（每个内核数的线程）× sockets = vCPU。

2. OpenShift Container Platform 和 Kubernetes 对磁盘性能非常敏感，建议使用更快的存储速度，特别是 control plane 节点上需要 10 ms p99 fsync 持续时间的 etcd。请注意，在许多云平台上，存储大小和 IOPS 可一起扩展，因此您可能需要过度分配存储卷来获取足够的性能。

3. 与所有用户置备的安装一样，如果您选择在集群中使用 RHEL 计算机器，则负责所有操作系统生命周期管理和维护，包括执行系统更新、应用补丁和完成所有其他必要的任务。RHEL 7 计算机器的使用已弃用，并已在 OpenShift Container Platform 4.10 及更新的版本中删除。



注意

从 OpenShift Container Platform 版本 4.13 开始，RHCOS 基于 RHEL 版本 9.2，它更新了微架构要求。以下列表包含每个架构需要的最小指令集架构 (ISA)：

- x86-64 体系结构需要 x86-64-v2 ISA
- ARM64 架构需要 ARMv8.0-A ISA
- IBM Power 架构需要 Power 9 ISA
- s390x 架构需要 z14 ISA

如需更多信息，请参阅 RHEL 架构。

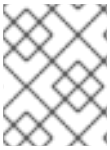
如果平台的实例类型满足集群机器的最低要求，则 OpenShift Container Platform 支持使用它。

其他资源

- 优化存储

6.3.6.3.2. 为 AWS 测试的实例类型

以下 Amazon Web Services(AWS) 实例类型已经过 OpenShift Container Platform 测试。



注意

将以下图中包含的机器类型用于 AWS 实例。如果您使用没有在图表中列出的实例类型，请确保使用的实例大小与名为"最小资源要求"的部分中列出的最少资源要求匹配。

例 6.29. 基于 64 位 x86 架构的机器类型

- **c4.***
- **c5.***
- **c5a.***
- **i3.***
- **m4.***
- **m5.***
- **m5a.***
- **m6a.***
- **m6i.***
- **r4.***
- **r5.***
- **r5a.***

- **r6i.***
- **t3.***
- **t3a.***

6.3.6.3.3. 在 64 位 ARM 基础架构上为 AWS 测试过的实例类型

OpenShift Container Platform 中已经测试了以下 Amazon Web Services (AWS) 64 位 ARM 实例类型。

注意

使用 AWS ARM 实例的以下图中包含的机器类型。如果您使用没有在图中列出的实例类型，请确保使用的实例大小与集群安装"最小资源要求"中列出的最少资源要求匹配。

例 6.30. 基于 64 位 ARM 架构的机器类型

- **c6g.***
- **m6g.***

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

您可以自定义安装配置文件 (**install-config.yaml**)，以指定有关 OpenShift Container Platform 集群平台的更多详细信息，或修改所需参数的值。

重要

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

```
apiVersion: v1
baseDomain: example.com (1)
credentialsMode: Mint (2)
controlPlane: (3) (4)
  hyperthreading: Enabled (5)
  name: master
  platform:
    aws:
      zones:
      - us-west-2a
      - us-west-2b
      rootVolume:
        iops: 4000
        size: 500
        type: io1 (6)
      metadataService:
        authentication: Optional (7)
      type: m6i.xlarge
  replicas: 3
```

```
compute: (8)
- hyperthreading: Enabled (9)
  name: worker
  platform:
    aws:
      rootVolume:
        iops: 2000
        size: 500
        type: io1 (10)
      metadataService:
        authentication: Optional (11)
      type: c5.4xlarge
      zones:
      - us-west-2c
  replicas: 3
metadata:
  name: test-cluster (12)
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OVNKubernetes (13)
  serviceNetwork:
  - 172.30.0.0/16
platform:
  aws:
    region: us-west-2 (14)
    propagateUserTags: true (15)
    userTags:
      adminContact: jdoe
      costCenter: 7536
    subnets: (16)
    - subnet-1
    - subnet-2
    - subnet-3
    amiID: ami-0c5d3e03c0ab9b19a (17)
    serviceEndpoints: (18)
      - name: ec2
        url: https://vpce-id.ec2.us-west-2.vpce.amazonaws.com
    hostedZone: Z3URY6TWQ91KVV (19)
fips: false (20)
sshKey: ssh-ed25519 AAAA... (21)
pullSecret: '{"auths": ...}' (22)
```

(1) (12) (14) (22) 必需。安装程序会提示您输入这个值。

(2) 可选：添加此参数来强制 Cloud Credential Operator (CCO) 使用指定的模式。默认情况下，CCO 使用 **kube-system** 命名空间中的 root 凭证来动态尝试决定凭证的功能。有关 CCO 模式的详情，请参阅*身份验证和授权*指南中的"About the Cloud Credential Operator"部分。

(3) (8) (15) 如果没有提供这些参数和值，安装程序会提供默认值。

4 **controlPlane** 部分是一个单个映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，controlPlane **部分** 的第一行则不以连字符开头。

5 9 是否要启用或禁用并发多线程或 **超线程**。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将 参数值设置为 **Disabled** 来禁用它。如果在某些集群机器中禁用并发多线程，则必须在所有集群机器中禁用它。

重要

如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。如果您对机器禁用并发多线程，请使用较大的实例类型，如 **m4.2xlarge** 或 **m5.2xlarge**。

6 10 要为 etcd 配置更快的存储，特别是对于较大的集群，请将存储类型设置为 **io1**，并将 **iops** 设为 **2000**。

7 11 是否需要 Amazon EC2 实例元数据服务 v2 (IMDSv2)。为了要求 IMDSv2，请将参数值设置为 **Required**。要允许使用 IMDSv1 和 IMDSv2，请将参数值设置为 **Optional**。如果没有指定值，则允许 IMDSv1 和 IMDSv2。

注意

在集群安装过程中设置的 control plane 机器的 IMDS 配置只能使用 AWS CLI 更改。可以使用计算机器集来更改计算机器的 IMDS 配置。

13 要安装的集群网络插件。默认值 **OVNKubernetes** 是唯一支持的值。

16 如果您提供自己的 VPC，为集群使用的每个可用区指定子网。

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

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

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

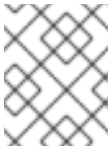
20 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 Red Hat Enterprise Linux (RHEL) 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅在 FIPS 模式中安装该系统。

当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS)时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。

21 您可以选择提供您用来访问集群中机器的 **sshKey** 值。



注意

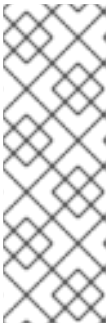
对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

6.3.6.3.5. 在安装过程中配置集群范围的代理

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

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 **Proxy** 对象的 **spec.noProxy** 字段中添加站点来绕过代理。



注意

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

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

流程

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

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: ec2.<aws_region>.amazonaws.com,elasticloadbalancing.
<aws_region>.amazonaws.com,s3.<aws_region>.amazonaws.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 **.** 以仅匹配子域。例如，**.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代理。如果您已将 Amazon **EC2**、**Elastic Load Balancing** 和 **S3** VPC 端点添加到 VPC 中，您必须将这些端点添加到 **noProxy** 字段。

4 如果提供，安装程序会在 **openshift-config** 命名空间中生成名为 **user-ca-bundle** 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network

5 可选：决定 **Proxy** 对象的配置以引用 **trustedCA** 字段中 **user-ca-bundle** 配置映射的策略。允许的值是 **Proxyonly** 和 **Always**。仅在配置了 **http/https** 代理时，使用 **Proxyonly** 引用 **user-ca-bundle** 配置映射。使用 **Always** 始终引用 **user-ca-bundle** 配置映射。默认值为 **Proxyonly**。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。

注意

如果安装程序超时，重启并使用安装程序的 **wait-for** 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

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

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

注意

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

6.3.6.3.6. 将现有 AWS 安全组应用到集群

将现有 AWS 安全组应用到 control plane 和计算机器可帮助您满足机构的安全需求，在这种情况下，您需要控制这些机器的传入或传出流量。

先决条件

- 您已在 AWS 中创建安全组。如需更多信息，请参阅使用 安全组的 AWS 文档。
- 安全组必须与您要将集群部署到的现有 VPC 关联。安全组不能与另一个 VPC 关联。
- 您有一个现有的 **install-config.yaml** 文件。

流程

1. 在 **install-config.yaml** 文件中，编辑 **compute.platform.aws.additionalSecurityGroupIDs** 参数，为您的计算机器指定一个或多个自定义安全组。
2. 编辑 **controlPlane.platform.aws.additionalSecurityGroupIDs** 参数，为您的 control plane 机器指定一个或多个自定义安全组。
3. 保存文件并在部署集群时引用。

指定自定义安全组的 install-config.yaml 文件示例

```
# ...
compute:
- hyperthreading: Enabled
  name: worker
  platform:
    aws:
      additionalSecurityGroupIDs:
        - sg-1 (1)
        - sg-2
  replicas: 3
controlPlane:
  hyperthreading: Enabled
  name: master
  platform:
    aws:
      additionalSecurityGroupIDs:
        - sg-3
        - sg-4
  replicas: 3
platform:
  aws:
    region: us-east-1
    subnets: (2)
      - subnet-1
      - subnet-2
      - subnet-3
```

(1) 在 Amazon EC2 控制台中显示时指定安全组的名称，包括 **sg** 前缀。

(2) 指定集群使用的每个可用区的子网。

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

默认情况下，管理员 secret 存储在 **kube-system** 项目中。如果您在 **install-config.yaml** 文件中将 **credentialsMode** 参数配置为 **Manual**，则必须使用以下替代方案之一：

- 要手动管理长期云凭证，请按照手动创建长期凭证中的步骤操作。
- 要实现在集群外为各个组件管理的短期凭证，请按照配置 A WS 集群以使用短期凭证 中的步骤操作。

6.3.6.4.1. 手动创建长期凭证

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

流程

1. 如果您没有将 **install-config.yaml** 配置文件中的 **credentialsMode** 参数设置为 **Manual**，请修改值，如下所示：

   配置文件片段示例

   ```
   apiVersion: v1
   ```

```
baseDomain: example.com
credentialsMode: Manual
# ...
```

2. 如果您之前还没有创建安装清单文件，请运行以下命令：

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

其中 **<installation_directory>** 是安装程序在其中创建文件的目录。

3. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

4. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 自定义资源 (CR) 列表：

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
  --included \ 1
  --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
  --to=<path_to_directory_for_credentials_requests> 3
```

1. **--included** 参数仅包含特定集群配置所需的清单。
2. 指定 **install-config.yaml** 文件的位置。
3. 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

此命令为每个 **CredentialsRequest** 对象创建一个 YAML 文件。

CredentialsRequest 对象示例

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

5. 在之前生成的 **openshift-install** 清单目录中为 secret 创建 YAML 文件。secret 必须使用在 **spec.secretRef** 中为每个 **CredentialsRequest** 定义的命名空间和 secret 名称存储。

带有 secret 的 CredentialsRequest 对象示例

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component_credentials_request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: AWSProviderSpec
    statementEntries:
    - effect: Allow
      action:
      - s3:CreateBucket
      - s3:DeleteBucket
      resource: "*"
      ...
  secretRef:
    name: <component_secret>
    namespace: <component_namespace>
  ...
```

Secret 对象示例

```
apiVersion: v1
kind: Secret
metadata:
  name: <component_secret>
  namespace: <component_namespace>
data:
  aws_access_key_id: <base64_encoded_aws_access_key_id>
  aws_secret_access_key: <base64_encoded_aws_secret_access_key>
```

重要

在升级使用手动维护凭证的集群前，您必须确保 CCO 处于可升级状态。

6.3.6.4.2. 将 AWS 集群配置为使用短期凭证

要安装配置为使用 AWS 安全令牌服务 (STS) 的集群，您必须配置 CCO 实用程序并为集群创建所需的 AWS 资源。

6.3.6.4.2.1. 配置 Cloud Credential Operator 工具

当 Cloud Credential Operator(CCO)以手动模式运行时，要从集群外部创建和管理云凭证，提取并准备 CCO 实用程序(**ccoctl**)二进制文件。

注意

ccoctl 工具是在 Linux 环境中运行的 Linux 二进制文件。

先决条件

- 您可以访问具有集群管理员权限的 OpenShift Container Platform 帐户。
- 已安装 OpenShift CLI(**oc**)。
- 您已为 **ccoctl** 工具创建了用于以下权限的 AWS 帐户：

  例 6.31. 所需的 AWS 权限

  所需的 iam 权限

  - **iam:CreateOpenIDConnectProvider**
  - **iam:CreateRole**
  - **iam:DeleteOpenIDConnectProvider**
  - **iam:DeleteRole**
  - **iam:DeleteRolePolicy**
  - **iam:GetOpenIDConnectProvider**
  - **iam:GetRole**
  - **iam:GetUser**
  - **iam:ListOpenIDConnectProviders**
  - **iam:ListRolePolicies**
  - **iam:ListRoles**
  - **iam:PutRolePolicy**
  - **iam:TagOpenIDConnectProvider**
  - **iam:TagRole**

  所需的 s3 权限

  - **s3:CreateBucket**
  - **s3:DeleteBucket**
  - **s3:DeleteObject**
  - **s3:GetBucketAcl**
  - **s3:GetBucketTagging**
  - **s3:GetObject**
  - **s3:GetObjectAcl**

- **s3:GetObjectTagging**
- **s3:ListBucket**
- **s3:PutBucketAcl**
- **s3:PutBucketPolicy**
- **s3:PutBucketPublicAccessBlock**
- **s3:PutBucketTagging**
- **s3:PutObject**
- **s3:PutObjectAcl**
- **s3:PutObjectTagging**

所需的 cloudfront 权限

- **cloudfront:ListCloudFrontOriginAccessIdentities**
- **cloudfront:ListDistributions**
- **cloudfront:ListTagsForResource**

如果您计划通过公共 CloudFront 发行版 URL 将 OIDC 配置存储在 IAM 身份提供程序访问的私有 S3 存储桶中，则运行 **ccoctl** 工具的 AWS 帐户需要以下额外权限：

例 6.32. 使用 CloudFront 私有 S3 存储桶的额外权限

- **cloudfront:CreateCloudFrontOriginAccessIdentity**
- **cloudfront:CreateDistribution**
- **cloudfront:DeleteCloudFrontOriginAccessIdentity**
- **cloudfront:DeleteDistribution**
- **cloudfront:GetCloudFrontOriginAccessIdentity**
- **cloudfront:GetCloudFrontOriginAccessIdentityConfig**
- **cloudfront:GetDistribution**
- **cloudfront:TagResource**
- **cloudfront:UpdateDistribution**

注意

在使用 **ccoctl aws create-all** 命令处理凭证请求时，这些额外权限支持使用 **--create-private-s3-bucket** 选项。

流程

1. 运行以下命令，为 OpenShift Container Platform 发行镜像设置变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. 运行以下命令，从 OpenShift Container Platform 发行镜像获取 CCO 容器镜像：

```
$ CCO_IMAGE=$(oc adm release info --image-for='cloud-credential-operator'
$RELEASE_IMAGE -a ~/.pull-secret)
```

注意

确保 **$RELEASE_IMAGE** 的架构与将使用 **ccoctl** 工具的环境架构相匹配。

3. 运行以下命令，将 CCO 容器镜像中的 **ccoctl** 二进制文件提取到 OpenShift Container Platform 发行镜像中：

```
$ oc image extract $CCO_IMAGE \
  --file="/usr/bin/ccoctl.<rhel_version>" \ 1
  -a ~/.pull-secret
```

1 对于 **<rhel_version>**，请指定与主机使用的 Red Hat Enterprise Linux (RHEL) 版本对应的值。如果没有指定值，则默认使用 **ccoctl.rhel8**。以下值有效：

- **rhel8**: 为使用 RHEL 8 的主机指定这个值。
- **rhel9** ：为使用 RHEL 9 的主机指定这个值。

4. 运行以下命令更改权限以使 **ccoctl** 可执行：

```
$ chmod 775 ccoctl.<rhel_version>
```

验证

- 要验证 **ccoctl** 是否可使用，请运行以下命令来显示帮助文件：

```
$ ccoctl --help
```

ccoctl --help 的输出

```
OpenShift credentials provisioning tool

Usage:
  ccoctl [command]

Available Commands:
  aws         Manage credentials objects for AWS cloud
  azure       Manage credentials objects for Azure
  gcp         Manage credentials objects for Google cloud
  help        Help about any command
  ibmcloud    Manage credentials objects for IBM Cloud
```

```
  nutanix      Manage credentials objects for Nutanix

Flags:
  -h, --help   help for ccoctl

Use "ccoctl [command] --help" for more information about a command.
```

6.3.6.4.2.2. 使用 Cloud Credential Operator 实用程序创建 AWS 资源

创建 AWS 资源时有以下选项：

- 您可以使用 **ccoctl aws create-all** 命令自动创建 AWS 资源。这是创建资源的最快速方法。请参阅 使用单个命令创建 AWS 资源。
- 如果您需要在修改 AWS 资源前查看 **ccoctl** 工具创建的 JSON 文件，或者 **ccoctl** 工具用于创建 AWS 资源的过程无法自动满足组织的要求，您可以单独创建 AWS 资源。请参阅 单独创建 AWS 资源。

6.3.6.4.2.2.1. 使用单个命令创建 AWS 资源

如果 **ccoctl** 工具用于创建 AWS 资源的过程自动满足机构的要求，您可以使用 **ccoctl aws create-all** 命令自动创建 AWS 资源。

否则，您可以单独创建 AWS 资源。如需更多信息，请参阅"单独创建 AWS 资源"。

注意

默认情况下，**ccoctl** 在运行命令的目录中创建对象。要在其他目录中创建对象，请使用 **--output-dir** 标志。此流程使用 **<path_to_ccoctl_output_dir>** 来引用这个目录。

先决条件

您必须：

- 提取并准备好 **ccoctl** 二进制文件。

流程

1. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 对象列表：

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
  --included \ 1
  --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
  --to=<path_to_directory_for_credentials_requests> 3
```

1 **--included** 参数仅包含特定集群配置所需的清单。

2 指定 **install-config.yaml** 文件的位置。

3 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

注意

此命令可能需要一些时间才能运行。

3. 运行以下命令，使用 **ccoctl** 工具处理所有 **CredentialsRequest** 对象：

```
$ ccoctl aws create-all \
  --name=<name> \1
  --region=<aws_region> \2
  --credentials-requests-dir=<path_to_credentials_requests_directory> \3
  --output-dir=<path_to_ccoctl_output_dir> \4
  --create-private-s3-bucket 5
```

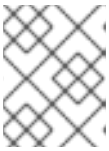
1 指定用于标记创建用于跟踪的任何云资源的名称。

2 指定在其中创建云资源的 AWS 区域。

3 指定包含组件 **CredentialsRequest** 对象文件的目录。

4 可选：指定您希望 **ccoctl** 实用程序在其中创建对象的目录。默认情况下，实用程序在运行命令的目录中创建对象。

5 可选：默认情况下，**ccoctl** 实用程序将 OpenID Connect (OIDC) 配置文件存储在公共 S3 存储桶中，并使用 S3 URL 作为公共 OIDC 端点。要将 OIDC 配置存储在 IAM 身份提供程序通过公共 CloudFront 发行版 URL 访问的专用 S3 存储桶中，请使用 **--create-private-s3-bucket** 参数。



注意

如果您的集群使用 **TechPreviewNoUpgrade** 功能集启用的技术预览功能，则必须包含 **--enable-tech-preview** 参数。

验证

- 要验证 OpenShift Container Platform secret 是否已创建，列出 **<path_to_ccoctl_output_dir>/manifests** 目录中的文件：

```
$ ls <path_to_ccoctl_output_dir>/manifests
```

输出示例

```
cluster-authentication-02-config.yaml
openshift-cloud-credential-operator-cloud-credential-operator-iam-ro-creds-credentials.yaml
openshift-cloud-network-config-controller-cloud-credentials-credentials.yaml
openshift-cluster-api-capa-manager-bootstrap-credentials-credentials.yaml
openshift-cluster-csi-drivers-ebs-cloud-credentials-credentials.yaml
```

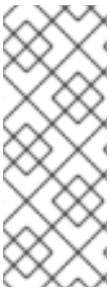
```
openshift-image-registry-installer-cloud-credentials-credentials.yaml
openshift-ingress-operator-cloud-credentials-credentials.yaml
openshift-machine-api-aws-cloud-credentials-credentials.yaml
```

您可以通过查询 AWS 来验证是否已创建 IAM 角色。如需更多信息，请参阅有关列出 IAM 角色的 AWS 文档。

6.3.6.4.2.2.2. 单独创建 AWS 资源

您可以使用 **ccoctl** 工具单独创建 AWS 资源。这个选项对于在不同用户或部门之间创建这些资源的组织可能很有用。

否则，您可以使用 **ccoctl aws create-all** 命令自动创建 AWS 资源。如需更多信息，请参阅"使用单个命令创建 AWS 资源"。



注意

默认情况下，**ccoctl** 在运行命令的目录中创建对象。要在其他目录中创建对象，请使用 **--output-dir** 标志。此流程使用 **<path_to_ccoctl_output_dir>** 来引用这个目录。

有些 **ccoctl** 命令会发出 AWS API 调用来创建或修改 AWS 资源。您可以使用 **--dry-run** 标志来避免 API 调用。使用此标志可在本地文件系统中创建 JSON 文件。您可以使用 **--cli-input-json** 参数查看和修改 JSON 文件，然后使用 AWS CLI 工具应用它们。

先决条件

- 提取并准备 **ccoctl** 二进制文件。

流程

1. 运行以下命令，生成用于为集群设置 OpenID Connect 供应商的公共和私有 RSA 密钥文件：

```
$ ccoctl aws create-key-pair
```

输出示例

```
2021/04/13 11:01:02 Generating RSA keypair
2021/04/13 11:01:03 Writing private key to /<path_to_ccoctl_output_dir>/serviceaccount-signer.private
2021/04/13 11:01:03 Writing public key to /<path_to_ccoctl_output_dir>/serviceaccount-signer.public
2021/04/13 11:01:03 Copying signing key for use by installer
```

其中 **serviceaccount-signer.private** 和 **serviceaccount-signer.public** 是生成的密钥文件。

此命令还会在 **/<path_to_ccoctl_output_dir>/tls/bound-service-account-signing-key.key** 中创建集群在安装过程中所需的私钥。

2. 运行以下命令，在 AWS 上创建 OpenID Connect 身份提供程序和 S3 存储桶：

```
$ ccoctl aws create-identity-provider \
  --name=<name> \ 1
  --region=<aws_region> \ 2
  --public-key-file=<path_to_ccoctl_output_dir>/serviceaccount-signer.public 3
```

-

1. **<name>** 是用于标记为跟踪而创建的云资源的名称。
2. **<aws-region>** 是将要在其中创建云资源的 AWS 区域。
3. **<path_to_ccoctl_output_dir>** 是 **ccoctl aws create-key-pair** 命令生成的公钥文件的路径。

输出示例

```
2021/04/13 11:16:09 Bucket <name>-oidc created
2021/04/13 11:16:10 OpenID Connect discovery document in the S3 bucket <name>-oidc at
.well-known/openid-configuration updated
2021/04/13 11:16:10 Reading public key
2021/04/13 11:16:10 JSON web key set (JWKS) in the S3 bucket <name>-oidc at keys.json
updated
2021/04/13 11:16:18 Identity Provider created with ARN: arn:aws:iam::
<aws_account_id>:oidc-provider/<name>-oidc.s3.<aws_region>.amazonaws.com
```

其中 **openid-configuration** 是发现文档和 **key.json** 是一个 JSON Web 密钥集文件。

此命令还会在 **/<path_to_ccoctl_output_dir>/manifests/cluster-authentication-02-config.yaml** 中创建 YAML 配置文件。此文件为集群生成的服务帐户令牌设置签发者 URL 字段，以便 AWS IAM 身份提供程序信任令牌。

3. 为集群中的每个组件创建 IAM 角色：

   a. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

   ```
   $ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
   ```

   b. 从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 对象列表：

   ```
   $ oc adm release extract \
     --from=$RELEASE_IMAGE \
     --credentials-requests \
     --included \ 1
     --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \
   2
     --to=<path_to_directory_for_credentials_requests> 3
   ```

   1. **--included** 参数仅包含特定集群配置所需的清单。
   2. 指定 **install-config.yaml** 文件的位置。
   3. 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

   c. 运行以下命令，使用 **ccoctl** 工具处理所有 **CredentialsRequest** 对象：

   ```
   $ ccoctl aws create-iam-roles \
     --name=<name> \
     --region=<aws_region> \
   ```

```
  --credentials-requests-dir=<path_to_credentials_requests_directory> \
  --identity-provider-arn=arn:aws:iam::<aws_account_id>:oidc-provider/<name>-oidc.s3.
<aws_region>.amazonaws.com
```

注意

对于使用其他 IAM API 端点的 AWS 环境（如 GovCloud），还必须使用 **--region** 参数指定您的区域。

如果您的集群使用 **TechPreviewNoUpgrade** 功能集启用的技术预览功能，则必须包含 **--enable-tech-preview** 参数。

对于每个 **CredentialsRequest** 对象，**ccoctl** 创建一个带有信任策略的 IAM 角色，该角色与指定的 OIDC 身份提供程序相关联，以及来自 OpenShift Container Platform 发行镜像的每个 **CredentialsRequest** 对象中定义的权限策略。

验证

- 要验证 OpenShift Container Platform secret 是否已创建，列出 **<path_to_ccoctl_output_dir>/manifests** 目录中的文件：

```
$ ls <path_to_ccoctl_output_dir>/manifests
```

输出示例

```
cluster-authentication-02-config.yaml
openshift-cloud-credential-operator-cloud-credential-operator-iam-ro-creds-credentials.yaml
openshift-cloud-network-config-controller-cloud-credentials-credentials.yaml
openshift-cluster-api-capa-manager-bootstrap-credentials-credentials.yaml
openshift-cluster-csi-drivers-ebs-cloud-credentials-credentials.yaml
openshift-image-registry-installer-cloud-credentials-credentials.yaml
openshift-ingress-operator-cloud-credentials-credentials.yaml
openshift-machine-api-aws-cloud-credentials-credentials.yaml
```

您可以通过查询 AWS 来验证是否已创建 IAM 角色。如需更多信息，请参阅有关列出 IAM 角色的 AWS 文档。

6.3.6.4.2.3. 整合 Cloud Credential Operator 实用程序清单

要为单个组件在集群外实现短期安全凭证，您必须将创建 Cloud Credential Operator 实用程序 (**ccoctl**) 的清单文件移到安装程序的正确目录中。

先决条件

- 您已使用托管集群的云平台配置了帐户。
- 您已配置了 Cloud Credential Operator 实用程序 (**ccoctl**)。
- 已使用 **ccoctl** 工具创建了集群所需的云供应商资源。

流程

1. 如果您设有将 **install-config.yaml** 配置文件中的 **credentialsMode** 参数设置为 **Manual**，请修改

1. 如果您没有将 **install-config.yaml** 配置文件中的 **credentialsMode** 参数设置为 **Manual**，请修改值，如下所示：

   配置文件片段示例

   ```
   apiVersion: v1
   baseDomain: example.com
   credentialsMode: Manual
   # ...
   ```

2. 如果您之前还没有创建安装清单文件，请运行以下命令：

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

   其中 **<installation_directory>** 是安装程序在其中创建文件的目录。

3. 运行以下命令，将 **ccoctl** 工具生成的清单复制到安装程序创建的 **manifests** 目录中：

   ```
   $ cp /<path_to_ccoctl_output_dir>/manifests/* ./manifests/
   ```

4. 运行以下命令，将 **ccoctl** 工具在 **tls** 目录中生成的私钥复制到安装目录中：

   ```
   $ cp -a /<path_to_ccoctl_output_dir>/tls .
   ```

6.3.6.5. 部署集群

您可以在兼容云平台上安装 OpenShift Container Platform。

重要

在初始安装过程中，您只能运行安装程序的 **create cluster** 命令一次。

先决条件

- 您已使用托管集群的云平台配置了帐户。
- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。
- 已确认主机上的云供应商帐户具有部署集群的正确权限。权限不正确的帐户会导致安装过程失败，并显示包括缺失权限的错误消息。

流程

1. 进入包含安装程序的目录并初始化集群部署：

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

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

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

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

注意

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

验证

当集群部署成功完成时：

- 终端会显示用于访问集群的说明，包括指向 Web 控制台和 **kubeadmin** 用户的凭证的链接。
- 凭证信息还会输出到 **<installation_directory>/.openshift_install.log**.

重要

不要删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

输出示例

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

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

6.3.6.6. 使用 CLI 登录集群

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

先决条件

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

流程

流程

1. 导出 **kubeadmin** 凭证：

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

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

2. 验证您可以使用导出的配置成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

6.3.6.7. 使用 Web 控制台登录到集群

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

先决条件

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

流程

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

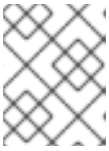
```
$ cat <installation_directory>/auth/kubeadmin-password
```

注意

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

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

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



注意

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

输出示例

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

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

其他资源

- 如需有关 访问和了解 OpenShift Container Platform Web 控制台的更多详情，请参阅 访问 Web 控制台。

6.3.6.8. 后续步骤

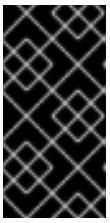
- 验证安装.
- 自定义集群。
- 如果需要，您可以选择 不使用远程健康报告。
- 如果需要，您可以删除云供应商凭证。
- 在 AWS 上将集群安装到现有的 VPC 中后，您可以将 AWS VPC 集群扩展到 AWS Outpost 中 。

6.3.7. 在 AWS 上安装私有集群

在 OpenShift Container Platform 版本 4.16 中，您可以在 Amazon Web Services（AWS）上将私有集群安装到现有的 VPC 中。安装程序会置备所需基础架构的其余部分，您可以进一步自定义这些基础架构。要自定义安装，请在安装集群前修改 **install-config.yaml** 文件中的参数。

6.3.7.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读选择集群安装方法并为用户准备它的文档。
- 已将 AWS 帐户配置为托管集群。



重要

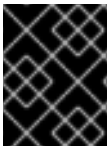
如果您的计算机上存储有 AWS 配置集，则不要在使用多因素验证设备的同时使用您生成的临时会话令牌。在集群的整个生命周期中，集群会持续使用您的当前 AWS 凭证来创建 AWS 资源，因此您必须使用长期凭证。要生成适当的密钥，请参阅 AWS 文档中的管理 IAM 用户的访问密钥。您可在运行安装程序时提供密钥。

- 如果使用防火墙，将其配置为允许集群需要访问的站点。

6.3.7.2. 私有集群

您可以部署不公开外部端点的私有 OpenShift Container Platform 集群。私有集群只能从内部网络访问，且无法在互联网中看到。

默认情况下，OpenShift Container Platform 被置备为使用可公开访问的 DNS 和端点。在部署集群时，私有集群会将 DNS、Ingress Controller 和 API 服务器设置为私有。这意味着集群资源只能从您的内部网络访问，且不能在互联网中看到。



重要

如果集群有任何公共子网，管理员创建的负载均衡器服务可能会公开访问。为确保集群安全性，请验证这些服务是否已明确标注为私有。

要部署私有集群，您必须：

- 使用满足您的要求的现有网络。集群资源可能会在网络上的其他集群间共享。
- 从有权访问的机器中部署：
  - 您置备的云的 API 服务。
  - 您调配的网络上的主机。
  - 用于获取安装介质的互联网。

您可以使用符合这些访问要求的机器，并按照您的公司规定进行操作。例如，该机器可以是云网络中的堡垒主机，也可以是可通过 VPN 访问网络的机器。

6.3.7.2.1. AWS 中的私有集群

要在 Amazon Web Services (AWS) 上创建私有集群，您必须提供一个现有的私有 VPC 和子网来托管集群。安装程序还必须能够解析集群所需的 DNS 记录。安装程序将 Ingress Operator 和 API 服务器配置为只可以从私有网络访问。

集群仍然需要访问互联网来访问 AWS API。

安装私有集群时不需要或创建以下项目：

- 公共子网
- 支持公共入口的公共负载均衡器
- 与集群的 **baseDomain** 匹配的公共 Route 53 区域

安装程序会使用您指定的 **baseDomain** 来创建专用的 Route 53 区域以及集群所需的记录。集群被配置，以便 Operator 不会为集群创建公共记录，且所有集群机器都放置在您指定的私有子网中。

6.3.7.2.1.1. 限制：

为私有集群添加公共功能的能力有限。

- 在安装后，您无法在不进行额外操作的情况下公开 Kubernetes API 端点。这些额外的操作包括为使用中的每个可用区在 VPC 中创建公共子网，创建公共负载均衡器，以及配置 control plane 安全组以便 6443 端口（Kubernetes API 端口）可以接受来自于互联网的网络流量。
- 如果使用公共服务类型负载均衡器，您必须在每个可用区中为公共子网添加 **kubernetes.io/cluster/<cluster-infra-id>: shared** 标签，以便 AWS 可使用它们来创建公共负载均衡器。

6.3.7.3. 关于使用自定义 VPC

在 OpenShift Container Platform 4.16 中，您可以在 Amazon Web Services (AWS)的现有 Amazon Virtual Private Cloud (VPC)中将集群部署到现有子网中。通过将 OpenShift Container Platform 部署到现有的 AWS VPC 中，您可能会避开新帐户中的限制，或者更容易地利用公司所设置的操作限制。如果您无法获

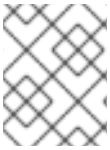
得您自己创建 VPC 所需的基础架构创建权限，请使用这个安装选项。

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

6.3.7.3.1. 使用 VPC 的要求

安装程序不再创建以下组件：

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



注意

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

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

安装程序无法：

- 分割网络范围供集群使用。
- 设置子网的路由表。
- 设置 VPC 选项，如 DHCP。

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

您的 VPC 必须满足以下特征：

- VPC 不能使用 **kubernetes.io/cluster/.*: owned**, **Name**, 和 **openshift.io/cluster** 标签。
  安装程序会修改子网以添加 **kubernetes.io/cluster/.*: shared** 标签，因此您的子网必须至少有一个可用的空闲标签插槽。请参阅 AWS 文档中的 标签限制 部分，以确认安装程序可以为您指定的每个子网添加标签。您不能使用 **Name** 标签，因为它与 EC2 **Name** 字段重叠，且安装失败。
- 如果要将 OpenShift Container Platform 集群扩展到 AWS Outpost 并具有现有的 Outpost 子网，现有的子网必须使用 **kubernetes.io/cluster/unmanaged: true** 标签。如果您没有应用此标签，因为 Cloud Controller Manager 在 Outpost 子网中创建服务负载均衡器，安装会失败，这是不受支持的配置。

- 您需要在您的 VPC 中启用 **enableDnsSupport** 和 **enableDnsHostnames** 属性，以便集群可以使用附加到 VPC 中的 Route 53 区来解析集群内部的 DNS 记录。请参阅 AWS 文档中的您的 VPC 中的 DNS 支持部分 。
  如果要使用您自己的 Route 53 托管私有区，您必须在安装集群前将现有托管区与 VPC 相关联。您可以使用 **install-config.yaml** 文件中的 **platform.aws.hostedZone** 和 **platform.aws.hostedZoneRole** 字段定义托管区。您可以通过与安装集群的帐户共享来使用来自另一个帐户的私有托管区。如果使用另一个帐户的私有托管区，则必须使用 **Passthrough** 或 **Manual** 凭证模式。

如果您在断开连接的环境中工作，则无法访问 EC2、ELB 和 S3 端点的公共 IP 地址。根据您要在安装过程中限制互联网流量的级别，有以下配置选项：

选项 1：创建 VPC 端点
创建 VPC 端点，并将其附加到集群使用的子网。将端点命名为如下：

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

通过这个选项，网络流量在 VPC 和所需的 AWS 服务之间保持私有。

选项 2：创建一个没有 VPC 端点的代理
作为安装过程的一部分，您可以配置 HTTP 或 HTTPS 代理。使用此选项时，互联网流量会通过代理访问所需的 AWS 服务。

选项 3：创建带有 VPC 端点的代理
作为安装过程的一部分，您可以使用 VPC 端点配置 HTTP 或 HTTPS 代理。创建 VPC 端点，并将其附加到集群使用的子网。将端点命名为如下：

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

在 **install-config.yaml** 文件中配置代理时，将这些端点添加到 **noProxy** 字段。通过这个选项，代理会阻止集群直接访问互联网。但是，您的 VPC 和所需的 AWS 服务之间网络流量保持私有。

所需的 VPC 组件

您必须提供合适的 VPC 和子网，以便与您的机器通信。

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

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

6.3.7.3.2. VPC 验证

要确保您提供的子网适合您的环境，安装程序会确认以下信息：

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

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

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

6.3.7.3.3. 权限划分

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

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

6.3.7.3.4. 集群间隔离

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

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

6.3.7.3.5. 可选： AWS 安全组

默认情况下，安装程序会创建安全组并将其附加到 control plane 和计算机器。不可修改与默认安全组关联的规则。

但是，您可以将与现有 VPC 关联的其他现有 AWS 安全组应用到 control plane 和计算机器。应用自定义安全组可帮助您满足机构的安全需求，在这种情况下，您需要控制这些机器的传入或传出流量。

作为安装过程的一部分，您可以在部署集群前通过修改 **install-config.yaml** 文件来应用自定义安全组。

如需更多信息，请参阅"将现有 AWS 安全组应用到集群"。

6.3.7.4. 手动创建安装配置文件

安装集群要求您手动创建安装配置文件。

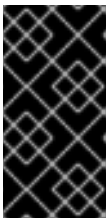
先决条件

- 您在本地机器上有一个 SSH 公钥来提供给安装程序。该密钥将用于在集群节点上进行 SSH 身份验证，以进行调试和灾难恢复。
- 已获取 OpenShift Container Platform 安装程序和集群的 pull secret。

流程

1. 创建一个安装目录来存储所需的安装资产：

```
$ mkdir <installation_directory>
```



重要

您必须创建一个目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

2. 自定义提供的 **install-config.yaml** 文件模板示例，并将其保存在 **<installation_directory>** 中。

注意

此配置文件必须命名为 **install-config.yaml**。

3. 备份 **install-config.yaml** 文件，以便您可以使用它安装多个集群。

重要

install-config.yaml 文件会在安装过程的下一步中使用。现在必须备份它。

其他资源

- AWS 的安装配置参数

6.3.7.4.1. 集群安装的最低资源要求

每台集群机器都必须满足以下最低要求：

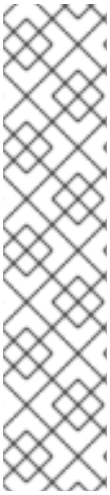
表 6.16. 最低资源要求

机器	操作系统	vCPU [1]	虚拟内存	Storage	每秒输入/输出 (IOPS) [2]
bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane（控制平面）	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS、RHEL 8.6 及更新版本 [3]	2	8 GB	100 GB	300

1. 当未启用并发多线程(SMT)或超线程时，一个 vCPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比例：（每个内核数的线程）× sockets = vCPU。

2. OpenShift Container Platform 和 Kubernetes 对磁盘性能非常敏感，建议使用更快的存储速度，特别是 control plane 节点上需要 10 ms p99 fsync 持续时间的 etcd。请注意，在许多云平台上，存储大小和 IOPS 可一起扩展，因此您可能需要过度分配存储卷来获取足够的性能。

3. 与所有用户置备的安装一样，如果您选择在集群中使用 RHEL 计算机器，则负责所有操作系统生命周期管理和维护，包括执行系统更新、应用补丁和完成所有其他必要的任务。RHEL 7 计算机器的使用已弃用，并已在 OpenShift Container Platform 4.10 及更新的版本中删除。



注意

从 OpenShift Container Platform 版本 4.13 开始，RHCOS 基于 RHEL 版本 9.2，它更新了微架构要求。以下列表包含每个架构需要的最小指令集架构 (ISA)：

- x86-64 体系结构需要 x86-64-v2 ISA
- ARM64 架构需要 ARMv8.0-A ISA
- IBM Power 架构需要 Power 9 ISA
- s390x 架构需要 z14 ISA

如需更多信息，请参阅 RHEL 架构。

如果平台的实例类型满足集群机器的最低要求，则 OpenShift Container Platform 支持使用它。

其他资源

- 优化存储

6.3.7.4.2. 为 AWS 测试的实例类型

以下 Amazon Web Services(AWS) 实例类型已经过 OpenShift Container Platform 测试。

注意

将以下图中包含的机器类型用于 AWS 实例。如果您使用没有在图表中列出的实例类型，请确保使用的实例大小与名为"最小资源要求"的部分中列出的最少资源要求匹配。

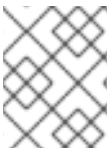
例 6.33. 基于 64 位 x86 架构的机器类型

- **c4.***
- **c5.***
- **c5a.***
- **i3.***
- **m4.***
- **m5.***
- **m5a.***
- **m6a.***

- **m6i.***
- **r4.***
- **r5.***
- **r5a.***
- **r6i.***
- **t3.***
- **t3a.***

6.3.7.4.3. 在 64 位 ARM 基础架构上为 AWS 测试过的实例类型

OpenShift Container Platform 中已经测试了以下 Amazon Web Services (AWS) 64 位 ARM 实例类型。



注意

使用 AWS ARM 实例的以下图中包含的机器类型。如果您使用没有在图中列出的实例类型，请确保使用的实例大小与集群安装"最小资源要求"中列出的最少资源要求匹配。

例 6.34. 基于 64 位 ARM 架构的机器类型

- **c6g.***
- **m6g.***

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

您可以自定义安装配置文件 (**install-config.yaml**)，以指定有关 OpenShift Container Platform 集群平台的更多详细信息，或修改所需参数的值。

重要

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

```
apiVersion: v1
baseDomain: example.com 1
credentialsMode: Mint 2
controlPlane: 3 4
  hyperthreading: Enabled 5
  name: master
  platform:
    aws:
      zones:
      - us-west-2a
      - us-west-2b
      rootVolume:
```

```
        iops: 4000
        size: 500
        type: io1 6
      metadataService:
        authentication: Optional 7
      type: m6i.xlarge
  replicas: 3
compute: 8
- hyperthreading: Enabled 9
  name: worker
  platform:
    aws:
      rootVolume:
        iops: 2000
        size: 500
        type: io1 10
      metadataService:
        authentication: Optional 11
      type: c5.4xlarge
      zones:
      - us-west-2c
  replicas: 3
metadata:
  name: test-cluster 12
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 13
  serviceNetwork:
  - 172.30.0.0/16
platform:
  aws:
    region: us-west-2 14
    propagateUserTags: true 15
    userTags:
      adminContact: jdoe
      costCenter: 7536
    subnets: 16
    - subnet-1
    - subnet-2
    - subnet-3
    amiID: ami-0c5d3e03c0ab9b19a 17
    serviceEndpoints: 18
      - name: ec2
        url: https://vpce-id.ec2.us-west-2.vpce.amazonaws.com
    hostedZone: Z3URY6TWQ91KVV 19
fips: false 20
sshKey: ssh-ed25519 AAAA... 21
publish: Internal 22
pullSecret: '{"auths": ...}' 23
```

1 12 14 23 必需。安装程序会提示您输入这个值。

2 可选：添加此参数来强制 Cloud Credential Operator (CCO) 使用指定的模式。默认情况下，CCO 使用 **kube-system** 命名空间中的 root 凭证来动态尝试决定凭证的功能。有关 CCO 模式的详情，请参阅*身份验证和授权*指南中的"About the Cloud Credential Operator"部分。

3 8 15 如果没有提供这些参数和值，安装程序会提供默认值。

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

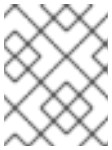
5 9 是否要启用或禁用并发多线程或 **超线程**。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将 参数值设置为 **Disabled** 来禁用它。如果在某些集群机器中禁用并发多线程，则必须在所有集群机器中禁用它。

重要

如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。如果您对机器禁用并发多线程，请使用较大的实例类型，如 **m4.2xlarge** 或 **m5.2xlarge**。

6 10 要为 etcd 配置更快的存储，特别是对于较大的集群，请将存储类型设置为 **io1**，并将 **iops** 设为 **2000**。

7 11 是否需要 Amazon EC2 实例元数据服务 v2 (IMDSv2)。为了要求 IMDSv2，请将参数值设置为 **Required**。要允许使用 IMDSv1 和 IMDSv2，请将参数值设置为 **Optional**。如果没有指定值，则允许 IMDSv1 和 IMDSv2。



注意

在集群安装过程中设置的 control plane 机器的 IMDS 配置只能使用 AWS CLI 更改。可以使用计算机器集来更改计算机器的 IMDS 配置。

13 要安装的集群网络插件。默认值 **OVNKubernetes** 是唯一支持的值。

16 如果您提供自己的 VPC，为集群使用的每个可用区指定子网。

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

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

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

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

重要

要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 Red Hat Enterprise Linux (RHEL) 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅在 FIPS 模式中安装该系统。

当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS)时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。

21 您可以选择提供您用来访问集群中机器的 **sshKey** 值。

注意

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

22 如何发布集群的面向用户的端点。将 **publish** 设置为 **Internal** 以部署一个私有集群，它不能被互联网访问。默认值为 **External**。

6.3.7.4.5. 在安装过程中配置集群范围的代理

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

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 **Proxy** 对象的 **spec.noProxy** 字段中添加站点来绕过代理。

注意

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

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

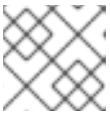
流程

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

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: ec2.<aws_region>.amazonaws.com,elasticloadbalancing.
```

```
<aws_region>.amazonaws.com,s3.<aws_region>.amazonaws.com 3
additionalTrustBundle: | 4
   -----BEGIN CERTIFICATE-----
   <MY_TRUSTED_CA_CERT>
   -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 **.** 以仅匹配子域。例如，**.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代理。如果您已将 Amazon **EC2**、**Elastic Load Balancing** 和 **S3** VPC 端点添加到 VPC 中，您必须将这些端点添加到 **noProxy** 字段。
4. 如果提供，安装程序会在 **openshift-config** 命名空间中生成名为 **user-ca-bundle** 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 **trusted-ca-bundle** 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， **Proxy** 对象的 **trustedCA** 字段中也会引用此配置映射。**additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。
5. 可选：决定 **Proxy** 对象的配置以引用 **trustedCA** 字段中 **user-ca-bundle** 配置映射的策略。允许的值是 **Proxyonly** 和 **Always**。仅在配置了 **http/https** 代理时，使用 **Proxyonly** 引用 **user-ca-bundle** 配置映射。使用 **Always** 始终引用 **user-ca-bundle** 配置映射。默认值为 **Proxyonly**。



注意

安装程序不支持代理的 **readinessEndpoints** 字段。

注意

如果安装程序超时，重启并使用安装程序的 **wait-for** 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

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

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

注意

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

6.3.7.4.6. 将现有 AWS 安全组应用到集群

将现有 AWS 安全组应用到 control plane 和计算机器可帮助您满足机构的安全需求，在这种情况下，您需要控制这些机器的传入或传出流量。

先决条件

- 您已在 AWS 中创建安全组。如需更多信息，请参阅使用 安全组的 AWS 文档。
- 安全组必须与您要将集群部署到的现有 VPC 关联。安全组不能与另一个 VPC 关联。
- 您有一个现有的 **install-config.yaml** 文件。

流程

1. 在 **install-config.yaml** 文件中，编辑 **compute.platform.aws.additionalSecurityGroupIDs** 参数，为您的计算机器指定一个或多个自定义安全组。
2. 编辑 **controlPlane.platform.aws.additionalSecurityGroupIDs** 参数，为您的 control plane 机器指定一个或多个自定义安全组。
3. 保存文件并在部署集群时引用。

指定自定义安全组的 install-config.yaml 文件示例

```
# ...
compute:
- hyperthreading: Enabled
  name: worker
  platform:
    aws:
      additionalSecurityGroupIDs:
        - sg-1 (1)
        - sg-2
  replicas: 3
controlPlane:
  hyperthreading: Enabled
  name: master
  platform:
    aws:
      additionalSecurityGroupIDs:
        - sg-3
        - sg-4
  replicas: 3
platform:
  aws:
    region: us-east-1
    subnets: (2)
      - subnet-1
      - subnet-2
      - subnet-3
```

(1) 在 Amazon EC2 控制台中显示时指定安全组的名称，包括 **sg** 前缀。

(2) 指定集群使用的每个可用区的子网。

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

默认情况下，管理员 secret 存储在 **kube-system** 项目中。如果您在 **install-config.yaml** 文件中将 **credentialsMode** 参数配置为 **Manual**，则必须使用以下替代方案之一：

- 要手动管理长期云凭证，请按照手动创建长期凭证中的步骤操作。
- 要实现在集群外为各个组件管理的短期凭证，请按照配置 A WS 集群以使用短期凭证 中的步骤操作。

6.3.7.5.1. 手动创建长期凭证

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

流程

1. 如果您没有将 **install-config.yaml** 配置文件中的 **credentialsMode** 参数设置为 **Manual**，请修改值，如下所示：

   配置文件片段示例

   ```
   apiVersion: v1
   baseDomain: example.com
   credentialsMode: Manual
   # ...
   ```

2. 如果您之前还没有创建安装清单文件，请运行以下命令：

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

   其中 **<installation_directory>** 是安装程序在其中创建文件的目录。

3. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

   ```
   $ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
   ```

4. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 自定义资源 (CR) 列表：

   ```
   $ oc adm release extract \
     --from=$RELEASE_IMAGE \
     --credentials-requests \
     --included \ 1
     --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
     --to=<path_to_directory_for_credentials_requests> 3
   ```

   1. **--included** 参数仅包含特定集群配置所需的清单。
   2. 指定 **install-config.yaml** 文件的位置。
   3. 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

   此命令为每个 **CredentialsRequest** 对象创建一个 YAML 文件。

CredentialsRequest 对象示例

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

5. 在之前生成的 **openshift-install** 清单目录中为 secret 创建 YAML 文件。secret 必须使用在 **spec.secretRef** 中为每个 **CredentialsRequest** 定义的命名空间和 secret 名称存储。

带有 secret 的 CredentialsRequest 对象示例

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component_credentials_request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: AWSProviderSpec
    statementEntries:
    - effect: Allow
      action:
      - s3:CreateBucket
      - s3:DeleteBucket
      resource: "*"
      ...
  secretRef:
    name: <component_secret>
    namespace: <component_namespace>
  ...
```

Secret 对象示例

```
apiVersion: v1
kind: Secret
metadata:
  name: <component_secret>
```

```
  namespace: <component_namespace>
data:
  aws_access_key_id: <base64_encoded_aws_access_key_id>
  aws_secret_access_key: <base64_encoded_aws_secret_access_key>
```

重要

在升级使用手动维护凭证的集群前，您必须确保 CCO 处于可升级状态。

6.3.7.5.2. 将 AWS 集群配置为使用短期凭证

要安装配置为使用 AWS 安全令牌服务 (STS) 的集群，您必须配置 CCO 实用程序并为集群创建所需的 AWS 资源。

6.3.7.5.2.1. 配置 Cloud Credential Operator 工具

当 Cloud Credential Operator(CCO)以手动模式运行时，要从集群外部创建和管理云凭证，提取并准备 CCO 实用程序(**ccoctl**)二进制文件。

注意

ccoctl 工具是在 Linux 环境中运行的 Linux 二进制文件。

先决条件

- 您可以访问具有集群管理员权限的 OpenShift Container Platform 帐户。
- 已安装 OpenShift CLI(**oc**)。
- 您已为 **ccoctl** 工具创建了用于以下权限的 AWS 帐户：

  例 6.35. 所需的 AWS 权限

  所需的 iam 权限

  - **iam:CreateOpenIDConnectProvider**
  - **iam:CreateRole**
  - **iam:DeleteOpenIDConnectProvider**
  - **iam:DeleteRole**
  - **iam:DeleteRolePolicy**
  - **iam:GetOpenIDConnectProvider**
  - **iam:GetRole**
  - **iam:GetUser**
  - **iam:ListOpenIDConnectProviders**
  - **iam:ListRolePolicies**
  - **iam:ListRoles**

- **iam:PutRolePolicy**
- **iam:TagOpenIDConnectProvider**
- **iam:TagRole**

所需的 s3 权限

- **s3:CreateBucket**
- **s3:DeleteBucket**
- **s3:DeleteObject**
- **s3:GetBucketAcl**
- **s3:GetBucketTagging**
- **s3:GetObject**
- **s3:GetObjectAcl**
- **s3:GetObjectTagging**
- **s3:ListBucket**
- **s3:PutBucketAcl**
- **s3:PutBucketPolicy**
- **s3:PutBucketPublicAccessBlock**
- **s3:PutBucketTagging**
- **s3:PutObject**
- **s3:PutObjectAcl**
- **s3:PutObjectTagging**

所需的 cloudfront 权限

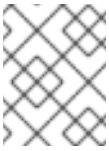
- **cloudfront:ListCloudFrontOriginAccessIdentities**
- **cloudfront:ListDistributions**
- **cloudfront:ListTagsForResource**

如果您计划通过公共 CloudFront 发行版 URL 将 OIDC 配置存储在 IAM 身份提供程序访问的私有 S3 存储桶中，则运行 **ccoctl** 工具的 AWS 帐户需要以下额外权限：

例 6.36. 使用 CloudFront 私有 S3 存储桶的额外权限

- **cloudfront:CreateCloudFrontOriginAccessIdentity**
- **cloudfront:CreateDistribution**

- **cloudfront:DeleteCloudFrontOriginAccessIdentity**
- **cloudfront:DeleteDistribution**
- **cloudfront:GetCloudFrontOriginAccessIdentity**
- **cloudfront:GetCloudFrontOriginAccessIdentityConfig**
- **cloudfront:GetDistribution**
- **cloudfront:TagResource**
- **cloudfront:UpdateDistribution**



注意

在使用 **ccoctl aws create-all** 命令处理凭证请求时，这些额外权限支持使用 **--create-private-s3-bucket** 选项。

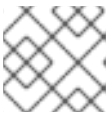
流程

1. 运行以下命令，为 OpenShift Container Platform 发行镜像设置变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. 运行以下命令，从 OpenShift Container Platform 发行镜像获取 CCO 容器镜像：

```
$ CCO_IMAGE=$(oc adm release info --image-for='cloud-credential-operator'
$RELEASE_IMAGE -a ~/.pull-secret)
```



注意

确保 **$RELEASE_IMAGE** 的架构与将使用 **ccoctl** 工具的环境架构相匹配。

3. 运行以下命令，将 CCO 容器镜像中的 **ccoctl** 二进制文件提取到 OpenShift Container Platform 发行镜像中：

```
$ oc image extract $CCO_IMAGE \
  --file="/usr/bin/ccoctl.<rhel_version>" \ 1
  -a ~/.pull-secret
```

1 对于 **<rhel_version>**，请指定与主机使用的 Red Hat Enterprise Linux (RHEL) 版本对应的值。如果没有指定值，则默认使用 **ccoctl.rhel8**。以下值有效：

- **rhel8**: 为使用 RHEL 8 的主机指定这个值。
- **rhel9** ：为使用 RHEL 9 的主机指定这个值。

4. 运行以下命令更改权限以使 **ccoctl** 可执行：

```
$ chmod 775 ccoctl.<rhel_version>
```

验证

- 要验证 **ccoctl** 是否可使用，请运行以下命令来显示帮助文件：

```
$ ccoctl --help
```

ccoctl --help 的输出

```
OpenShift credentials provisioning tool

Usage:
  ccoctl [command]

Available Commands:
  aws         Manage credentials objects for AWS cloud
  azure       Manage credentials objects for Azure
  gcp         Manage credentials objects for Google cloud
  help        Help about any command
  ibmcloud    Manage credentials objects for IBM Cloud
  nutanix     Manage credentials objects for Nutanix

Flags:
  -h, --help   help for ccoctl

Use "ccoctl [command] --help" for more information about a command.
```

6.3.7.5.2.2. 使用 Cloud Credential Operator 实用程序创建 AWS 资源

创建 AWS 资源时有以下选项：

- 您可以使用 **ccoctl aws create-all** 命令自动创建 AWS 资源。这是创建资源的最快速方法。请参阅 使用单个命令创建 AWS 资源。
- 如果您需要在修改 AWS 资源前查看 **ccoctl** 工具创建的 JSON 文件，或者 **ccoctl** 工具用于创建 AWS 资源的过程无法自动满足组织的要求，您可以单独创建 AWS 资源。请参阅 单独创建 AWS 资源。

6.3.7.5.2.2.1. 使用单个命令创建 AWS 资源

如果 **ccoctl** 工具用于创建 AWS 资源的过程自动满足机构的要求，您可以使用 **ccoctl aws create-all** 命令自动创建 AWS 资源。

否则，您可以单独创建 AWS 资源。如需更多信息，请参阅"单独创建 AWS 资源"。

注意

默认情况下，**ccoctl** 在运行命令的目录中创建对象。要在其他目录中创建对象，请使用 **--output-dir** 标志。此流程使用 **<path_to_ccoctl_output_dir>** 来引用这个目录。

先决条件

您必须：

- 提取并准备好 **ccoctl** 二进制文件。

流程

1. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 对象列表：

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
  --included \ 1
  --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
  --to=<path_to_directory_for_credentials_requests> 3
```

1. **--included** 参数仅包含特定集群配置所需的清单。
2. 指定 **install-config.yaml** 文件的位置。
3. 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

注意

此命令可能需要一些时间才能运行。

3. 运行以下命令，使用 **ccoctl** 工具处理所有 **CredentialsRequest** 对象：

```
$ ccoctl aws create-all \
  --name=<name> \ 1
  --region=<aws_region> \ 2
  --credentials-requests-dir=<path_to_credentials_requests_directory> \ 3
  --output-dir=<path_to_ccoctl_output_dir> \ 4
  --create-private-s3-bucket 5
```

1. 指定用于标记创建用于跟踪的任何云资源的名称。
2. 指定在其中创建云资源的 AWS 区域。
3. 指定包含组件 **CredentialsRequest** 对象文件的目录。
4. 可选：指定您希望 **ccoctl** 实用程序在其中创建对象的目录。默认情况下，实用程序在运行命令的目录中创建对象。
5. 可选：默认情况下，**ccoctl** 实用程序将 OpenID Connect (OIDC) 配置文件存储在公共 S3 存储桶中，并使用 S3 URL 作为公共 OIDC 端点。要将 OIDC 配置存储在 IAM 身份提供程序通过公共 CloudFront 发行版 URL 访问的专用 S3 存储桶中，请使用 **--create-private-s3-bucket** 参数。

注意

如果您的集群使用 **TechPreviewNoUpgrade** 功能集启用的技术预览功能，则必须包含 **--enable-tech-preview** 参数。

验证

- 要验证 OpenShift Container Platform secret 是否已创建，列出 **<path_to_ccoctl_output_dir>/manifests** 目录中的文件：

```
$ ls <path_to_ccoctl_output_dir>/manifests
```

输出示例

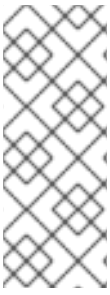
```
cluster-authentication-02-config.yaml
openshift-cloud-credential-operator-cloud-credential-operator-iam-ro-creds-credentials.yaml
openshift-cloud-network-config-controller-cloud-credentials-credentials.yaml
openshift-cluster-api-capa-manager-bootstrap-credentials-credentials.yaml
openshift-cluster-csi-drivers-ebs-cloud-credentials-credentials.yaml
openshift-image-registry-installer-cloud-credentials-credentials.yaml
openshift-ingress-operator-cloud-credentials-credentials.yaml
openshift-machine-api-aws-cloud-credentials-credentials.yaml
```

您可以通过查询 AWS 来验证是否已创建 IAM 角色。如需更多信息，请参阅有关列出 IAM 角色的 AWS 文档。

6.3.7.5.2.2.2. 单独创建 AWS 资源

您可以使用 **ccoctl** 工具单独创建 AWS 资源。这个选项对于在不同用户或部门之间创建这些资源的组织可能很有用。

否则，您可以使用 **ccoctl aws create-all** 命令自动创建 AWS 资源。如需更多信息，请参阅"使用单个命令创建 AWS 资源"。



注意

默认情况下，**ccoctl** 在运行命令的目录中创建对象。要在其他目录中创建对象，请使用 **--output-dir** 标志。此流程使用 **<path_to_ccoctl_output_dir>** 来引用这个目录。

有些 **ccoctl** 命令会发出 AWS API 调用来创建或修改 AWS 资源。您可以使用 **--dry-run** 标志来避免 API 调用。使用此标志可在本地文件系统中创建 JSON 文件。您可以使用 **--cli-input-json** 参数查看和修改 JSON 文件，然后使用 AWS CLI 工具应用它们。

先决条件

- 提取并准备 **ccoctl** 二进制文件。

流程

1. 运行以下命令，生成用于为集群设置 OpenID Connect 供应商的公共和私有 RSA 密钥文件：

```
$ ccoctl aws create-key-pair
```

输出示例

```
2021/04/13 11:01:02 Generating RSA keypair
2021/04/13 11:01:03 Writing private key to /<path_to_ccoctl_output_dir>/serviceaccount-signer.private
2021/04/13 11:01:03 Writing public key to /<path_to_ccoctl_output_dir>/serviceaccount-signer.public
2021/04/13 11:01:03 Copying signing key for use by installer
```

其中 **serviceaccount-signer.private** 和 **serviceaccount-signer.public** 是生成的密钥文件。

此命令还会在 **/<path_to_ccoctl_output_dir>/tls/bound-service-account-signing-key.key** 中创建集群在安装过程中所需的私钥。

2. 运行以下命令，在 AWS 上创建 OpenID Connect 身份提供程序和 S3 存储桶：

```
$ ccoctl aws create-identity-provider \
  --name=<name> \1
  --region=<aws_region> \2
  --public-key-file=<path_to_ccoctl_output_dir>/serviceaccount-signer.public 3
```

1. **<name>** 是用于标记为跟踪而创建的云资源的名称。
2. **<aws-region>** 是将要在其中创建云资源的 AWS 区域。
3. **<path_to_ccoctl_output_dir>** 是 **ccoctl aws create-key-pair** 命令生成的公钥文件的路径。

输出示例

```
2021/04/13 11:16:09 Bucket <name>-oidc created
2021/04/13 11:16:10 OpenID Connect discovery document in the S3 bucket <name>-oidc at .well-known/openid-configuration updated
2021/04/13 11:16:10 Reading public key
2021/04/13 11:16:10 JSON web key set (JWKS) in the S3 bucket <name>-oidc at keys.json updated
2021/04/13 11:16:18 Identity Provider created with ARN: arn:aws:iam::<aws_account_id>:oidc-provider/<name>-oidc.s3.<aws_region>.amazonaws.com
```

其中 **openid-configuration** 是发现文档和 **key.json** 是一个 JSON Web 密钥集文件。

此命令还会在 **/<path_to_ccoctl_output_dir>/manifests/cluster-authentication-02-config.yaml** 中创建 YAML 配置文件。此文件为集群生成的服务帐户令牌设置签发者 URL 字段，以便 AWS IAM 身份提供程序信任令牌。

3. 为集群中的每个组件创建 IAM 角色：

   a. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

   ```
   $ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
   ```

   b. 从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 对象列表：

   ```
   $ oc adm release extract \
     --from=$RELEASE_IMAGE \
   ```

```
  --credentials-requests \
  --included \ 1
  --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \
2
  --to=<path_to_directory_for_credentials_requests> 3
```

1. **--included** 参数仅包含特定集群配置所需的清单。
2. 指定 **install-config.yaml** 文件的位置。
3. 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

c. 运行以下命令，使用 **ccoctl** 工具处理所有 **CredentialsRequest** 对象：

```
$ ccoctl aws create-iam-roles \
  --name=<name> \
  --region=<aws_region> \
  --credentials-requests-dir=<path_to_credentials_requests_directory> \
  --identity-provider-arn=arn:aws:iam::<aws_account_id>:oidc-provider/<name>-oidc.s3.
<aws_region>.amazonaws.com
```

注意

对于使用其他 IAM API 端点的 AWS 环境（如 GovCloud），还必须使用 **--region** 参数指定您的区域。

如果您的集群使用 **TechPreviewNoUpgrade** 功能集启用的技术预览功能，则必须包含 **--enable-tech-preview** 参数。

对于每个 **CredentialsRequest** 对象，**ccoctl** 创建一个带有信任策略的 IAM 角色，该角色与指定的 OIDC 身份提供程序相关联，以及来自 OpenShift Container Platform 发行镜像的每个 **CredentialsRequest** 对象中定义的权限策略。

验证

- 要验证 OpenShift Container Platform secret 是否已创建，列出 **<path_to_ccoctl_output_dir>/manifests** 目录中的文件：

```
$ ls <path_to_ccoctl_output_dir>/manifests
```

输出示例

```
cluster-authentication-02-config.yaml
openshift-cloud-credential-operator-cloud-credential-operator-iam-ro-creds-credentials.yaml
openshift-cloud-network-config-controller-cloud-credentials-credentials.yaml
openshift-cluster-api-capa-manager-bootstrap-credentials-credentials.yaml
openshift-cluster-csi-drivers-ebs-cloud-credentials-credentials.yaml
openshift-image-registry-installer-cloud-credentials-credentials.yaml
openshift-ingress-operator-cloud-credentials-credentials.yaml
openshift-machine-api-aws-cloud-credentials-credentials.yaml
```

您可以通过查询 AWS 来验证是否已创建 IAM 角色。如需更多信息，请参阅有关列出 IAM 角色的 AWS 文档。

6.3.7.5.2.3. 整合 Cloud Credential Operator 实用程序清单

要为单个组件在集群外实现短期安全凭证，您必须将创建 Cloud Credential Operator 实用程序 (**ccoctl**) 的清单文件移到安装程序的正确目录中。

先决条件

- 您已使用托管集群的云平台配置了帐户。
- 您已配置了 Cloud Credential Operator 实用程序 (**ccoctl**)。
- 已使用 **ccoctl** 工具创建了集群所需的云供应商资源。

流程

1. 如果您没有将 **install-config.yaml** 配置文件中的 **credentialsMode** 参数设置为 **Manual**，请修改值，如下所示：

   配置文件片段示例

   ```
   apiVersion: v1
   baseDomain: example.com
   credentialsMode: Manual
   # ...
   ```

2. 如果您之前还没有创建安装清单文件，请运行以下命令：

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

   其中 **<installation_directory>** 是安装程序在其中创建文件的目录。

3. 运行以下命令，将 **ccoctl** 工具生成的清单复制到安装程序创建的 **manifests** 目录中：

   ```
   $ cp /<path_to_ccoctl_output_dir>/manifests/* ./manifests/
   ```

4. 运行以下命令，将 **ccoctl** 工具在 **tls** 目录中生成的私钥复制到安装目录中：

   ```
   $ cp -a /<path_to_ccoctl_output_dir>/tls .
   ```

6.3.7.6. 部署集群

您可以在兼容云平台上安装 OpenShift Container Platform。

重要

在初始安装过程中，您只能运行安装程序的 **create cluster** 命令一次。

先决条件

- 您已使用托管集群的云平台配置了帐户。

- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。
- 已确认主机上的云供应商帐户具有部署集群的正确权限。权限不正确的帐户会导致安装过程失败，并显示包括缺失权限的错误消息。

流程

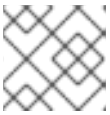
1. 进入包含安装程序的目录并初始化集群部署：

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

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

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

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



注意

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

验证

当集群部署成功完成时：

- 终端会显示用于访问集群的说明，包括指向 Web 控制台和 **kubeadmin** 用户的凭证的链接。
- 凭证信息还会输出到 **<installation_directory>/.openshift_install.log**.

重要

不要删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

输出示例

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

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

6.3.7.7. 使用 CLI 登录集群

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

先决条件

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

流程

1. 导出 **kubeadmin** 凭证：

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

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

2. 验证您可以使用导出的配置成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

6.3.7.8. 使用 Web 控制台登录到集群

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

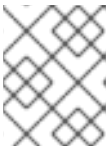
先决条件

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

流程

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

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



注意

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

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

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

注意

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

输出示例

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

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

其他资源

- 如需有关 访问和了解 OpenShift Container Platform Web 控制台的更多详情，请参阅 访问 Web 控制台。

6.3.7.9. 后续步骤

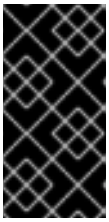
- 验证安装.
- 自定义集群。
- 如果需要，您可以选择 不使用远程健康报告。
- 如果需要，您可以删除云供应商凭证。

6.3.8. 在 AWS 上将集群安装到一个政府区域

在 OpenShift Container Platform 版本 4.16 中，您可以在 Amazon Web Services（AWS）上将集群安装到一个政府区域。要配置区域，在安装集群前修改 **install-config.yaml** 文件中的参数。

6.3.8.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读选择集群安装方法并为用户准备它的文档。
- 已将 AWS 帐户配置为托管集群。



重要

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

- 如果使用防火墙，将其配置为允许集群需要访问的站点。

6.3.8.2. AWS 政府区域

OpenShift Container Platform 支持将集群部署到 AWS GovCloud(US) 区域。

支持以下 AWS GovCloud 分区：

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

6.3.8.3. 安装要求

在安装集群前，您必须：

- 提供托管集群的现有的私有 AWS VPC 和子网。
  AWS GovCloud 的 Route 53 不支持公共区。因此，当您部署到 AWS 政府区域时，集群必须是私有的。
- 手动创建安装配置文件 (**install-config.yaml**)。

6.3.8.4. 私有集群

您可以部署不公开外部端点的私有 OpenShift Container Platform 集群。私有集群只能从内部网络访问，且无法在互联网中看到。

注意

AWS GovCloud 区域的 Route 53 不支持公共区。因此，如果集群部署到 AWS GovCloud 区域，则集群必须是私有的。

默认情况下，OpenShift Container Platform 被置备为使用可公开访问的 DNS 和端点。在部署集群时，私有集群会将 DNS、Ingress Controller 和 API 服务器设置为私有。这意味着集群资源只能从您的内部网络访问，且不能在互联网中看到。

重要

如果集群有任何公共子网，管理员创建的负载均衡器服务可能会公开访问。为确保集群安全性，请验证这些服务是否已明确标注为私有。

要部署私有集群，您必须：

- 使用满足您的要求的现有网络。集群资源可能会在网络上的其他集群间共享。
- 从有权访问的机器中部署：

- 您置备的云的 API 服务。
- 您调配的网络上的主机。
- 用于获取安装介质的互联网。

您可以使用符合这些访问要求的机器，并按照您的公司规定进行操作。例如，该机器可以是云网络中的堡垒主机，也可以是可通过 VPN 访问网络的机器。

6.3.8.4.1. AWS 中的私有集群

要在 Amazon Web Services (AWS) 上创建私有集群，您必须提供一个现有的私有 VPC 和子网来托管集群。安装程序还必须能够解析集群所需的 DNS 记录。安装程序将 Ingress Operator 和 API 服务器配置为只可以从私有网络访问。

集群仍然需要访问互联网来访问 AWS API。

安装私有集群时不需要或创建以下项目：

- 公共子网
- 支持公共入口的公共负载均衡器
- 与集群的 **baseDomain** 匹配的公共 Route 53 区域

安装程序会使用您指定的 **baseDomain** 来创建专用的 Route 53 区域以及集群所需的记录。集群被配置，以便 Operator 不会为集群创建公共记录，且所有集群机器都放置在您指定的私有子网中。

6.3.8.4.1.1. 限制：

为私有集群添加公共功能的能力有限。

- 在安装后，您无法在不进行额外操作的情况下公开 Kubernetes API 端点。这些额外的操作包括为使用中的每个可用区在 VPC 中创建公共子网，创建公共负载均衡器，以及配置 control plane 安全组以便 6443 端口（Kubernetes API 端口）可以接受来自于互联网的网络流量。
- 如果使用公共服务类型负载均衡器，您必须在每个可用区中为公共子网添加 **kubernetes.io/cluster/<cluster-infra-id>: shared** 标签，以便 AWS 可使用它们来创建公共负载均衡器。

6.3.8.5. 关于使用自定义 VPC

在 OpenShift Container Platform 4.16 中，您可以在 Amazon Web Services (AWS)的现有 Amazon Virtual Private Cloud (VPC)中将集群部署到现有子网中。通过将 OpenShift Container Platform 部署到现有的 AWS VPC 中，您可能会避开新帐户中的限制，或者更容易地利用公司所设置的操作限制。如果您无法获得您自己创建 VPC 所需的基础架构创建权限，请使用这个安装选项。

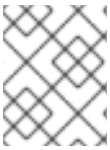
因为安装程序无法了解您现有子网中还有哪些其他组件，所以无法选择子网 CIDR 。您必须为安装集群的子网配置网络。

6.3.8.5.1. 使用 VPC 的要求

安装程序不再创建以下组件：

- 互联网网关

- NAT 网关
- 子网
- 路由表
- VPCs
- VPC DHCP 选项
- VPC 端点



注意

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

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

安装程序无法：

- 分割网络范围供集群使用。
- 设置子网的路由表。
- 设置 VPC 选项，如 DHCP。

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

您的 VPC 必须满足以下特征：

- VPC 不能使用 **kubernetes.io/cluster/.*: owned**, **Name**, 和 **openshift.io/cluster** 标签。
  安装程序会修改子网以添加 **kubernetes.io/cluster/.*: shared** 标签，因此您的子网必须至少有一个可用的空闲标签插槽。请参阅 AWS 文档中的 标签限制 部分，以确认安装程序可以为您指定的每个子网添加标签。您不能使用 **Name** 标签，因为它与 EC2 **Name** 字段重叠，且安装失败。
- 如果要将 OpenShift Container Platform 集群扩展到 AWS Outpost 并具有现有的 Outpost 子网，现有的子网必须使用 **kubernetes.io/cluster/unmanaged: true** 标签。如果您没有应用此标签，因为 Cloud Controller Manager 在 Outpost 子网中创建服务负载均衡器，安装会失败，这是不受支持的配置。
- 您需要在您的 VPC 中启用 **enableDnsSupport** 和 **enableDnsHostnames** 属性，以便集群可以使用附加到 VPC 中的 Route 53 区来解析集群内部的 DNS 记录。请参阅 AWS 文档中的您的 VPC 中的 DNS 支持部分 。
  如果要使用您自己的 Route 53 托管私有区，您必须在安装集群前将现有托管区与 VPC 相关联。您可以使用 **install-config.yaml** 文件中的 **platform.aws.hostedZone** 和 **platform.aws.hostedZoneRole** 字段定义托管区。您可以通过与安装集群的帐户共享来使用来自另一个帐户的私有托管区。如果使用另一个帐户的私有托管区，则必须使用 **Passthrough** 或 **Manual** 凭证模式。

如果您在断开连接的环境中工作，则无法访问 EC2、ELB 和 S3 端点的公共 IP 地址。根据您要在安装过程中限制互联网流量的级别，有以下配置选项：

选项 1：创建 VPC 端点

创建 VPC 端点，并将其附加到集群使用的子网。将端点命名为如下：

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

通过这个选项，网络流量在 VPC 和所需的 AWS 服务之间保持私有。

选项 2：创建一个没有 VPC 端点的代理
作为安装过程的一部分，您可以配置 HTTP 或 HTTPS 代理。使用此选项时，互联网流量会通过代理访问所需的 AWS 服务。

选项 3：创建带有 VPC 端点的代理
作为安装过程的一部分，您可以使用 VPC 端点配置 HTTP 或 HTTPS 代理。创建 VPC 端点，并将其附加到集群使用的子网。将端点命名为如下：

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

在 **install-config.yaml** 文件中配置代理时，将这些端点添加到 **noProxy** 字段。通过这个选项，代理会阻止集群直接访问互联网。但是，您的 VPC 和所需的 AWS 服务之间网络流量保持私有。

所需的 VPC 组件

您必须提供合适的 VPC 和子网，以便与您的机器通信。

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

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

6.3.8.5.2. VPC 验证

要确保您提供的子网适合您的环境，安装程序会确认以下信息：

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

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

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

6.3.8.5.3. 权限划分

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

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

6.3.8.5.4. 集群间隔离

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

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

6.3.8.5.5. 可选： AWS 安全组

默认情况下，安装程序会创建安全组并将其附加到 control plane 和计算机器。不可修改与默认安全组关联的规则。

但是，您可以将与现有 VPC 关联的其他现有 AWS 安全组应用到 control plane 和计算机器。应用自定义安全组可帮助您满足机构的安全需求，在这种情况下，您需要控制这些机器的传入或传出流量。

作为安装过程的一部分，您可以在部署集群前通过修改 **install-config.yaml** 文件来应用自定义安全组。

如需更多信息，请参阅"将现有 AWS 安全组应用到集群"。

6.3.8.6. 获取 AWS Marketplace 镜像

如果要使用 AWS Marketplace 镜像部署 OpenShift Container Platform 集群，您必须首先通过 AWS 订阅。订阅提供的提供的 AMI ID 可让您使用安装程序用来部署计算节点的 AMI ID。

先决条件

- 有 AWS 账户购买的产品。此帐户不必与用于安装集群的帐户相同。

流程

1. 从 AWS Marketplace 完成 OpenShift Container Platform 订阅。

2. 记录特定 AWS 区域的 AMI ID。作为安装过程的一部分，您必须在部署集群前使用这个值更新 **install-config.yaml** 文件。

使用 AWS Marketplace 计算节点的 install-config.yaml 文件示例

```
apiVersion: v1
baseDomain: example.com
compute:
- hyperthreading: Enabled
  name: worker
  platform:
    aws:
      amiID: ami-06c4d345f7c207239 (1)
      type: m5.4xlarge
  replicas: 3
metadata:
  name: test-cluster
platform:
  aws:
    region: us-east-2 (2)
sshKey: ssh-ed25519 AAAA...
pullSecret: '{"auths": ...}'
```

(1) 来自 AWS Marketplace 订阅的 AMI ID。

(2) 您的 AMI ID 与特定的 AWS 区域相关联。在创建安装配置文件时，请确保选择配置订阅时指定的相同 AWS 区域。

6.3.8.7. 手动创建安装配置文件

安装集群要求您手动创建安装配置文件。

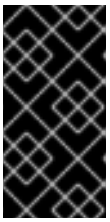
先决条件

- 您在本地机器上有一个 SSH 公钥来提供给安装程序。该密钥将用于在集群节点上进行 SSH 身份验证，以进行调试和灾难恢复。
- 已获取 OpenShift Container Platform 安装程序和集群的 pull secret。

流程

1. 创建一个安装目录来存储所需的安装资产：

```
$ mkdir <installation_directory>
```



重要

您必须创建一个目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

2. 自定义提供的 **install-config.yaml** 文件模板示例，并将其保存在 **<installation_directory>** 中。

注意

此配置文件必须命名为 **install-config.yaml**。

3. 备份 **install-config.yaml** 文件，以便您可以使用它安装多个集群。

重要

install-config.yaml 文件会在安装过程的下一步中使用。现在必须备份它。

其他资源

- AWS 的安装配置参数

6.3.8.7.1. 集群安装的最低资源要求

每台集群机器都必须满足以下最低要求：

表 6.17. 最低资源要求

机器	操作系统	vCPU [1]	虚拟内存	Storage	每秒输入/输出 (IOPS) [2]
bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane（控制平面）	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS、RHEL 8.6 及更新版本 [3]	2	8 GB	100 GB	300

1. 当未启用并发多线程(SMT)或超线程时，一个 vCPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比例：（每个内核数的线程）× sockets = vCPU。

2. OpenShift Container Platform 和 Kubernetes 对磁盘性能非常敏感，建议使用更快的存储速度，特别是 control plane 节点上需要 10 ms p99 fsync 持续时间的 etcd。请注意，在许多云平台上，存储大小和 IOPS 可一起扩展，因此您可能需要过度分配存储卷来获取足够的性能。

3. 与所有用户置备的安装一样，如果您选择在集群中使用 RHEL 计算机器，则负责所有操作系统生命周期管理和维护，包括执行系统更新、应用补丁和完成所有其他必要的任务。RHEL 7 计算机器的使用已弃用，并已在 OpenShift Container Platform 4.10 及更新的版本中删除。

注意

从 OpenShift Container Platform 版本 4.13 开始，RHCOS 基于 RHEL 版本 9.2，它更新了微架构要求。以下列表包含每个架构需要的最小指令集架构 (ISA)：

- x86-64 体系结构需要 x86-64-v2 ISA
- ARM64 架构需要 ARMv8.0-A ISA
- IBM Power 架构需要 Power 9 ISA
- s390x 架构需要 z14 ISA

如需更多信息，请参阅 RHEL 架构。

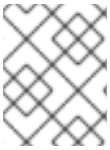
如果平台的实例类型满足集群机器的最低要求，则 OpenShift Container Platform 支持使用它。

其他资源

- 优化存储

6.3.8.7.2. 为 AWS 测试的实例类型

以下 Amazon Web Services(AWS) 实例类型已经过 OpenShift Container Platform 测试。



注意

将以下图中包含的机器类型用于 AWS 实例。如果您使用没有在图表中列出的实例类型，请确保使用的实例大小与名为"最小资源要求"的部分中列出的最少资源要求匹配。

例 6.37. 基于 64 位 x86 架构的机器类型

- **c4.***
- **c5.***
- **c5a.***
- **i3.***
- **m4.***
- **m5.***
- **m5a.***
- **m6a.***
- **m6i.***
- **r4.***
- **r5.***
- **r5a.***

- **r6i.***
- **t3.***
- **t3a.***

6.3.8.7.3. 在 64 位 ARM 基础架构上为 AWS 测试过的实例类型

OpenShift Container Platform 中已经测试了以下 Amazon Web Services (AWS) 64 位 ARM 实例类型。

注意

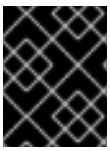
使用 AWS ARM 实例的以下图中包含的机器类型。如果您使用没有在图中列出的实例类型，请确保使用的实例大小与集群安装"最小资源要求"中列出的最少资源要求匹配。

例 6.38. 基于 64 位 ARM 架构的机器类型

- **c6g.***
- **m6g.***

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

您可以自定义安装配置文件 (**install-config.yaml**)，以指定有关 OpenShift Container Platform 集群平台的更多详细信息，或修改所需参数的值。



重要

此示例 YAML 文件仅供参考。使用它作为资源，在您手动创建的安装配置文件中输入参数值。

```
apiVersion: v1
baseDomain: example.com 1
credentialsMode: Mint 2
controlPlane: 3 4
  hyperthreading: Enabled 5
  name: master
  platform:
    aws:
      zones:
      - us-gov-west-1a
      - us-gov-west-1b
      rootVolume:
        iops: 4000
        size: 500
        type: io1 6
      metadataService:
        authentication: Optional 7
      type: m6i.xlarge
  replicas: 3
```

```
compute: 8
- hyperthreading: Enabled 9
  name: worker
  platform:
    aws:
      rootVolume:
        iops: 2000
        size: 500
        type: io1 10
      metadataService:
        authentication: Optional 11
      type: c5.4xlarge
      zones:
      - us-gov-west-1c
  replicas: 3
metadata:
  name: test-cluster 12
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 13
  serviceNetwork:
  - 172.30.0.0/16
platform:
  aws:
    region: us-gov-west-1 14
    propagateUserTags: true 15
    userTags:
      adminContact: jdoe
      costCenter: 7536
    subnets: 16
    - subnet-1
    - subnet-2
    - subnet-3
    amiID: ami-0c5d3e03c0ab9b19a 17
    serviceEndpoints: 18
      - name: ec2
        url: https://vpce-id.ec2.us-west-2.vpce.amazonaws.com
    hostedZone: Z3URY6TWQ91KVV 19
fips: false 20
sshKey: ssh-ed25519 AAAA... 21
publish: Internal 22
pullSecret: '{"auths": ...}' 23
```

1 12 14 23 必需。

2 可选：添加此参数来强制 Cloud Credential Operator (CCO) 使用指定的模式。默认情况下，CCO 使用 **kube-system** 命名空间中的 root 凭证来动态尝试决定凭证的功能。有关 CCO 模式的详情，请参阅*身份验证和授权*指南中的"About the Cloud Credential Operator"部分。

3 8 15 如果没有提供这些参数和值，安装程序会提供默认值。

4 **controlPlane** 部分是一个单个映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，controlPlane **部分** 的第一行则不以连字符开头。

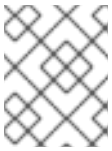
5 9 是否要启用或禁用并发多线程或 **超线程**。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将 参数值设置为 **Disabled** 来禁用它。如果在某些集群机器中禁用并发多线程，则必须在所有集群机器中禁用它。

重要

如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。如果您对机器禁用并发多线程，请使用较大的实例类型，如 **m4.2xlarge** 或 **m5.2xlarge**。

6 10 要为 etcd 配置更快的存储，特别是对于较大的集群，请将存储类型设置为 **io1**，并将 **iops** 设为 **2000**。

7 11 是否需要 Amazon EC2 实例元数据服务 v2 (IMDSv2)。为了要求 IMDSv2，请将参数值设置为 **Required**。要允许使用 IMDSv1 和 IMDSv2，请将参数值设置为 **Optional**。如果没有指定值，则允许 IMDSv1 和 IMDSv2。



注意

在集群安装过程中设置的 control plane 机器的 IMDS 配置只能使用 AWS CLI 更改。可以使用计算机器集来更改计算机器的 IMDS 配置。

13 要安装的集群网络插件。默认值 **OVNKubernetes** 是唯一支持的值。

16 如果您提供自己的 VPC，为集群使用的每个可用区指定子网。

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

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

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

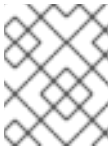
20 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 Red Hat Enterprise Linux (RHEL) 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅在 FIPS 模式中安装该系统。

当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS)时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。

21 您可以选择提供您用来访问集群中机器的 **sshKey** 值。



注意

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

22 如何发布集群的面向用户的端点。将 **publish** 设置为 **Internal** 以部署一个私有集群，它不能被互联网访问。默认值为 **External**。

6.3.8.7.5. 在安装过程中配置集群范围的代理

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

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 **Proxy** 对象的 **spec.noProxy** 字段中添加站点来绕过代理。

注意

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

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

流程

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

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: ec2.<aws_region>.amazonaws.com,elasticloadbalancing.
<aws_region>.amazonaws.com,s3.<aws_region>.amazonaws.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 **http**。

2 用于创建集群外 HTTPS 连接的代理 URL。

3

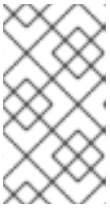
要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 **.** 以仅匹配子域。例如，**.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代

4 如果提供，安装程序会在 **openshift-config** 命名空间中生成名为 **user-ca-bundle** 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 **trusted-ca-bundle** 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， **Proxy** 对象的 **trustedCA** 字段中也会引用此配置映射。**additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

5 可选：决定 **Proxy** 对象的配置以引用 **trustedCA** 字段中 **user-ca-bundle** 配置映射的策略。允许的值是 **Proxyonly** 和 **Always**。仅在配置了 **http/https** 代理时，使用 **Proxyonly** 引用 **user-ca-bundle** 配置映射。使用 **Always** 始终引用 **user-ca-bundle** 配置映射。默认值为 **Proxyonly**。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。



注意

如果安装程序超时，重启并使用安装程序的 **wait-for** 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

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

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

注意

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

6.3.8.7.6. 将现有 AWS 安全组应用到集群

将现有 AWS 安全组应用到 control plane 和计算机器可帮助您满足机构的安全需求，在这种情况下，您需要控制这些机器的传入或传出流量。

先决条件

- 您已在 AWS 中创建安全组。如需更多信息，请参阅使用 安全组的 AWS 文档。
- 安全组必须与您要将集群部署到的现有 VPC 关联。安全组不能与另一个 VPC 关联。
- 您有一个现有的 **install-config.yaml** 文件。

流程

1. 在 **install-config.yaml** 文件中，编辑 **compute.platform.aws.additionalSecurityGroupIDs** 参数，为您的计算机器指定一个或多个自定义安全组。

2. 编辑 **controlPlane.platform.aws.additionalSecurityGroupIDs** 参数，为您的 control plane 机器指定一个或多个自定义安全组。

3. 保存文件并在部署集群时引用。

指定自定义安全组的 install-config.yaml 文件示例

```
# ...
compute:
- hyperthreading: Enabled
  name: worker
  platform:
    aws:
      additionalSecurityGroupIDs:
        - sg-1 (1)
        - sg-2
  replicas: 3
controlPlane:
  hyperthreading: Enabled
  name: master
  platform:
    aws:
      additionalSecurityGroupIDs:
        - sg-3
        - sg-4
  replicas: 3
platform:
  aws:
    region: us-east-1
    subnets: (2)
      - subnet-1
      - subnet-2
      - subnet-3
```

(1) 在 Amazon EC2 控制台中显示时指定安全组的名称，包括 **sg** 前缀。

(2) 指定集群使用的每个可用区的子网。

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

默认情况下，管理员 secret 存储在 **kube-system** 项目中。如果您在 **install-config.yaml** 文件中将 **credentialsMode** 参数配置为 **Manual**，则必须使用以下替代方案之一：

- 要手动管理长期云凭证，请按照手动创建长期凭证中的步骤操作。
- 要实现在集群外为各个组件管理的短期凭证，请按照 Incorporat ing the Cloud Credential Operator 实用程序清单中的步骤操作。

6.3.8.8.1. 手动创建长期凭证

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

流程

1. 如果您没有将 **install-config.yaml** 配置文件中的 **credentialsMode** 参数设置为 **Manual**，请修改值，如下所示：

   配置文件片段示例

   ```
   apiVersion: v1
   baseDomain: example.com
   credentialsMode: Manual
   # ...
   ```

2. 如果您之前还没有创建安装清单文件，请运行以下命令：

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

   其中 **<installation_directory>** 是安装程序在其中创建文件的目录。

3. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

   ```
   $ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
   ```

4. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 自定义资源 (CR) 列表：

   ```
   $ oc adm release extract \
     --from=$RELEASE_IMAGE \
     --credentials-requests \
     --included \ 1
     --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
     --to=<path_to_directory_for_credentials_requests> 3
   ```

   1. **--included** 参数仅包含特定集群配置所需的清单。
   2. 指定 **install-config.yaml** 文件的位置。
   3. 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

   此命令为每个 **CredentialsRequest** 对象创建一个 YAML 文件。

   CredentialsRequest 对象示例

   ```
   apiVersion: cloudcredential.openshift.io/v1
   kind: CredentialsRequest
   metadata:
     name: <component_credentials_request>
     namespace: openshift-cloud-credential-operator
     ...
   spec:
     providerSpec:
       apiVersion: cloudcredential.openshift.io/v1
       kind: AWSProviderSpec
       statementEntries:
       - effect: Allow
   ```

```
    action:
    - iam:GetUser
    - iam:GetUserPolicy
    - iam:ListAccessKeys
    resource: "*"
...
```

5. 在之前生成的 **openshift-install** 清单目录中为 secret 创建 YAML 文件。secret 必须使用在 **spec.secretRef** 中为每个 **CredentialsRequest** 定义的命名空间和 secret 名称存储。

带有 secret 的 CredentialsRequest 对象示例

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component_credentials_request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: AWSProviderSpec
    statementEntries:
    - effect: Allow
      action:
      - s3:CreateBucket
      - s3:DeleteBucket
      resource: "*"
      ...
  secretRef:
    name: <component_secret>
    namespace: <component_namespace>
  ...
```

Secret 对象示例

```
apiVersion: v1
kind: Secret
metadata:
  name: <component_secret>
  namespace: <component_namespace>
data:
  aws_access_key_id: <base64_encoded_aws_access_key_id>
  aws_secret_access_key: <base64_encoded_aws_secret_access_key>
```

重要

在升级使用手动维护凭证的集群前，您必须确保 CCO 处于可升级状态。

6.3.8.8.2. 将 AWS 集群配置为使用短期凭证

要安装配置为使用 AWS 安全令牌服务 (STS) 的集群，您必须配置 CCO 实用程序并为集群创建所需的 AWS 资源。

6.3.8.8.2.1. 配置 Cloud Credential Operator 工具

当 Cloud Credential Operator(CCO)以手动模式运行时，要从集群外部创建和管理云凭证，提取并准备 CCO 实用程序(**ccoctl**)二进制文件。

注意

ccoctl 工具是在 Linux 环境中运行的 Linux 二进制文件。

先决条件

- 您可以访问具有集群管理员权限的 OpenShift Container Platform 帐户。
- 已安装 OpenShift CLI(**oc**)。
- 您已为 **ccoctl** 工具创建了用于以下权限的 AWS 帐户：

  例 6.39. 所需的 AWS 权限

  所需的 iam 权限

  - **iam:CreateOpenIDConnectProvider**
  - **iam:CreateRole**
  - **iam:DeleteOpenIDConnectProvider**
  - **iam:DeleteRole**
  - **iam:DeleteRolePolicy**
  - **iam:GetOpenIDConnectProvider**
  - **iam:GetRole**
  - **iam:GetUser**
  - **iam:ListOpenIDConnectProviders**
  - **iam:ListRolePolicies**
  - **iam:ListRoles**
  - **iam:PutRolePolicy**
  - **iam:TagOpenIDConnectProvider**
  - **iam:TagRole**

  所需的 s3 权限

  - **s3:CreateBucket**
  - **s3:DeleteBucket**
  - **s3:DeleteObject**
  - **s3:GetBucketAcl**

- **s3:GetBucketTagging**
- **s3:GetObject**
- **s3:GetObjectAcl**
- **s3:GetObjectTagging**
- **s3:ListBucket**
- **s3:PutBucketAcl**
- **s3:PutBucketPolicy**
- **s3:PutBucketPublicAccessBlock**
- **s3:PutBucketTagging**
- **s3:PutObject**
- **s3:PutObjectAcl**
- **s3:PutObjectTagging**

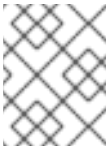
所需的 cloudfront 权限

- **cloudfront:ListCloudFrontOriginAccessIdentities**
- **cloudfront:ListDistributions**
- **cloudfront:ListTagsForResource**

如果您计划通过公共 CloudFront 发行版 URL 将 OIDC 配置存储在 IAM 身份提供程序访问的私有 S3 存储桶中，则运行 **ccoctl** 工具的 AWS 帐户需要以下额外权限：

例 6.40. 使用 CloudFront 私有 S3 存储桶的额外权限

- **cloudfront:CreateCloudFrontOriginAccessIdentity**
- **cloudfront:CreateDistribution**
- **cloudfront:DeleteCloudFrontOriginAccessIdentity**
- **cloudfront:DeleteDistribution**
- **cloudfront:GetCloudFrontOriginAccessIdentity**
- **cloudfront:GetCloudFrontOriginAccessIdentityConfig**
- **cloudfront:GetDistribution**
- **cloudfront:TagResource**
- **cloudfront:UpdateDistribution**



注意

在使用 **ccoctl aws create-all** 命令处理凭证请求时，这些额外权限支持使用 **--create-private-s3-bucket** 选项。

流程

1. 运行以下命令，为 OpenShift Container Platform 发行镜像设置变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. 运行以下命令，从 OpenShift Container Platform 发行镜像获取 CCO 容器镜像：

```
$ CCO_IMAGE=$(oc adm release info --image-for='cloud-credential-operator'
$RELEASE_IMAGE -a ~/.pull-secret)
```

注意

确保 **$RELEASE_IMAGE** 的架构与将使用 **ccoctl** 工具的环境架构相匹配。

3. 运行以下命令，将 CCO 容器镜像中的 **ccoctl** 二进制文件提取到 OpenShift Container Platform 发行镜像中：

```
$ oc image extract $CCO_IMAGE \
  --file="/usr/bin/ccoctl.<rhel_version>" \ 1
  -a ~/.pull-secret
```

1 对于 **<rhel_version>**，请指定与主机使用的 Red Hat Enterprise Linux (RHEL) 版本对应的值。如果没有指定值，则默认使用 **ccoctl.rhel8**。以下值有效：

- **rhel8**: 为使用 RHEL 8 的主机指定这个值。
- **rhel9** ：为使用 RHEL 9 的主机指定这个值。

4. 运行以下命令更改权限以使 **ccoctl** 可执行：

```
$ chmod 775 ccoctl.<rhel_version>
```

验证

- 要验证 **ccoctl** 是否可使用，请运行以下命令来显示帮助文件：

```
$ ccoctl --help
```

ccoctl --help 的输出

```
OpenShift credentials provisioning tool

Usage:
  ccoctl [command]
```

```
Available Commands:
  aws         Manage credentials objects for AWS cloud
  azure       Manage credentials objects for Azure
  gcp         Manage credentials objects for Google cloud
  help        Help about any command
  ibmcloud    Manage credentials objects for IBM Cloud
  nutanix     Manage credentials objects for Nutanix

Flags:
  -h, --help   help for ccoctl

Use "ccoctl [command] --help" for more information about a command.
```

6.3.8.8.2.2. 使用 Cloud Credential Operator 实用程序创建 AWS 资源

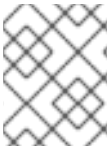
创建 AWS 资源时有以下选项：

- 您可以使用 **ccoctl aws create-all** 命令自动创建 AWS 资源。这是创建资源的最快速方法。请参阅使用单个命令创建 AWS 资源。
- 如果您需要在修改 AWS 资源前查看 **ccoctl** 工具创建的 JSON 文件，或者 **ccoctl** 工具用于创建 AWS 资源的过程无法自动满足组织的要求，您可以单独创建 AWS 资源。请参阅单独创建 AWS 资源。

6.3.8.8.2.2.1. 使用单个命令创建 AWS 资源

如果 **ccoctl** 工具用于创建 AWS 资源的过程自动满足机构的要求，您可以使用 **ccoctl aws create-all** 命令自动创建 AWS 资源。

否则，您可以单独创建 AWS 资源。如需更多信息，请参阅"单独创建 AWS 资源"。



注意

默认情况下，**ccoctl** 在运行命令的目录中创建对象。要在其他目录中创建对象，请使用 **--output-dir** 标志。此流程使用 **<path_to_ccoctl_output_dir>** 来引用这个目录。

先决条件

您必须：

- 提取并准备好 **ccoctl** 二进制文件。

流程

1. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 对象列表：

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
```

```
--credentials-requests \
--included \ 1
--install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
--to=<path_to_directory_for_credentials_requests> 3
```

1. **--included** 参数仅包含特定集群配置所需的清单。
2. 指定 **install-config.yaml** 文件的位置。
3. 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

注意

此命令可能需要一些时间才能运行。

3. 运行以下命令，使用 **ccoctl** 工具处理所有 **CredentialsRequest** 对象：

```
$ ccoctl aws create-all \
  --name=<name> \ 1
  --region=<aws_region> \ 2
  --credentials-requests-dir=<path_to_credentials_requests_directory> \ 3
  --output-dir=<path_to_ccoctl_output_dir> \ 4
  --create-private-s3-bucket 5
```

1. 指定用于标记创建用于跟踪的任何云资源的名称。
2. 指定在其中创建云资源的 AWS 区域。
3. 指定包含组件 **CredentialsRequest** 对象文件的目录。
4. 可选：指定您希望 **ccoctl** 实用程序在其中创建对象的目录。默认情况下，实用程序在运行命令的目录中创建对象。
5. 可选：默认情况下，**ccoctl** 实用程序将 OpenID Connect (OIDC) 配置文件存储在公共 S3 存储桶中，并使用 S3 URL 作为公共 OIDC 端点。要将 OIDC 配置存储在 IAM 身份提供程序通过公共 CloudFront 发行版 URL 访问的专用 S3 存储桶中，请使用 **--create-private-s3-bucket** 参数。

注意

如果您的集群使用 **TechPreviewNoUpgrade** 功能集启用的技术预览功能，则必须包含 **--enable-tech-preview** 参数。

验证

- 要验证 OpenShift Container Platform secret 是否已创建，列出 **<path_to_ccoctl_output_dir>/manifests** 目录中的文件：

```
$ ls <path_to_ccoctl_output_dir>/manifests
```

输出示例

```
cluster-authentication-02-config.yaml
openshift-cloud-credential-operator-cloud-credential-operator-iam-ro-creds-credentials.yaml
openshift-cloud-network-config-controller-cloud-credentials-credentials.yaml
openshift-cluster-api-capa-manager-bootstrap-credentials-credentials.yaml
openshift-cluster-csi-drivers-ebs-cloud-credentials-credentials.yaml
openshift-image-registry-installer-cloud-credentials-credentials.yaml
openshift-ingress-operator-cloud-credentials-credentials.yaml
openshift-machine-api-aws-cloud-credentials-credentials.yaml
```

您可以通过查询 AWS 来验证是否已创建 IAM 角色。如需更多信息，请参阅有关列出 IAM 角色的 AWS 文档。

6.3.8.8.2.2.2. 单独创建 AWS 资源

您可以使用 **ccoctl** 工具单独创建 AWS 资源。这个选项对于在不同用户或部门之间创建这些资源的组织可能很有用。

否则，您可以使用 **ccoctl aws create-all** 命令自动创建 AWS 资源。如需更多信息，请参阅"使用单个命令创建 AWS 资源"。

注意

默认情况下，**ccoctl** 在运行命令的目录中创建对象。要在其他目录中创建对象，请使用 **--output-dir** 标志。此流程使用 **<path_to_ccoctl_output_dir>** 来引用这个目录。

有些 **ccoctl** 命令会发出 AWS API 调用来创建或修改 AWS 资源。您可以使用 **--dry-run** 标志来避免 API 调用。使用此标志可在本地文件系统中创建 JSON 文件。您可以使用 **--cli-input-json** 参数查看和修改 JSON 文件，然后使用 AWS CLI 工具应用它们。

先决条件

- 提取并准备 **ccoctl** 二进制文件。

流程

1. 运行以下命令，生成用于为集群设置 OpenID Connect 供应商的公共和私有 RSA 密钥文件：

```
$ ccoctl aws create-key-pair
```

输出示例

```
2021/04/13 11:01:02 Generating RSA keypair
2021/04/13 11:01:03 Writing private key to /<path_to_ccoctl_output_dir>/serviceaccount-signer.private
2021/04/13 11:01:03 Writing public key to /<path_to_ccoctl_output_dir>/serviceaccount-signer.public
2021/04/13 11:01:03 Copying signing key for use by installer
```

其中 **serviceaccount-signer.private** 和 **serviceaccount-signer.public** 是生成的密钥文件。

此命令还会在 **/<path_to_ccoctl_output_dir>/tls/bound-service-account-signing-key.key** 中创建集群在安装过程中所需的私钥。

2. 运行以下命令，在 AWS 上创建 OpenID Connect 身份提供程序和 S3 存储桶：

```
$ ccoctl aws create-identity-provider \
  --name=<name> \1
  --region=<aws_region> \2
  --public-key-file=<path_to_ccoctl_output_dir>/serviceaccount-signer.public 3
```

1 **<name>** 是用于标记为跟踪而创建的云资源的名称。

2 **<aws-region>** 是将要在其中创建云资源的 AWS 区域。

3 **<path_to_ccoctl_output_dir>** 是 **ccoctl aws create-key-pair** 命令生成的公钥文件的路径。

输出示例

```
2021/04/13 11:16:09 Bucket <name>-oidc created
2021/04/13 11:16:10 OpenID Connect discovery document in the S3 bucket <name>-oidc at
.well-known/openid-configuration updated
2021/04/13 11:16:10 Reading public key
2021/04/13 11:16:10 JSON web key set (JWKS) in the S3 bucket <name>-oidc at keys.json
updated
2021/04/13 11:16:18 Identity Provider created with ARN: arn:aws:iam::
<aws_account_id>:oidc-provider/<name>-oidc.s3.<aws_region>.amazonaws.com
```

其中 **openid-configuration** 是发现文档和 **key.json** 是一个 JSON Web 密钥集文件。

此命令还会在 **/<path_to_ccoctl_output_dir>/manifests/cluster-authentication-02-config.yaml** 中创建 YAML 配置文件。此文件为集群生成的服务帐户令牌设置签发者 URL 字段，以便 AWS IAM 身份提供程序信任令牌。

3. 为集群中的每个组件创建 IAM 角色：

   a. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

   b. 从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 对象列表：

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
  --included \1
  --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \
2
  --to=<path_to_directory_for_credentials_requests> 3
```

1 **--included** 参数仅包含特定集群配置所需的清单。

2 指定 **install-config.yaml** 文件的位置。

3 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

c. 运行以下命令，使用 **ccoctl** 工具处理所有 **CredentialsRequest** 对象：

```
$ ccoctl aws create-iam-roles \
  --name=<name> \
  --region=<aws_region> \
  --credentials-requests-dir=<path_to_credentials_requests_directory> \
  --identity-provider-arn=arn:aws:iam::<aws_account_id>:oidc-provider/<name>-oidc.s3.
<aws_region>.amazonaws.com
```

注意

对于使用其他 IAM API 端点的 AWS 环境（如 GovCloud），还必须使用 **--region** 参数指定您的区域。

如果您的集群使用 **TechPreviewNoUpgrade** 功能集启用的技术预览功能，则必须包含 **--enable-tech-preview** 参数。

对于每个 **CredentialsRequest** 对象，**ccoctl** 创建一个带有信任策略的 IAM 角色，该角色与指定的 OIDC 身份提供程序相关联，以及来自 OpenShift Container Platform 发行镜像的每个 **CredentialsRequest** 对象中定义的权限策略。

验证

- 要验证 OpenShift Container Platform secret 是否已创建，列出 **<path_to_ccoctl_output_dir>/manifests** 目录中的文件：

```
$ ls <path_to_ccoctl_output_dir>/manifests
```

输出示例

```
cluster-authentication-02-config.yaml
openshift-cloud-credential-operator-cloud-credential-operator-iam-ro-creds-credentials.yaml
openshift-cloud-network-config-controller-cloud-credentials-credentials.yaml
openshift-cluster-api-capa-manager-bootstrap-credentials-credentials.yaml
openshift-cluster-csi-drivers-ebs-cloud-credentials-credentials.yaml
openshift-image-registry-installer-cloud-credentials-credentials.yaml
openshift-ingress-operator-cloud-credentials-credentials.yaml
openshift-machine-api-aws-cloud-credentials-credentials.yaml
```

您可以通过查询 AWS 来验证是否已创建 IAM 角色。如需更多信息，请参阅有关列出 IAM 角色的 AWS 文档。

6.3.8.8.2.3. 整合 Cloud Credential Operator 实用程序清单

要为单个组件在集群外实现短期安全凭证，您必须将创建 Cloud Credential Operator 实用程序 (**ccoctl**) 的清单文件移到安装程序的正确目录中。

先决条件

- 您已使用托管集群的云平台配置了帐户。
- 您已配置了 Cloud Credential Operator 实用程序 (**ccoctl**)。

- 已使用 **ccoctl** 工具创建了集群所需的云供应商资源。

流程

1. 如果您没有将 **install-config.yaml** 配置文件中的 **credentialsMode** 参数设置为 **Manual**，请修改值，如下所示：

   配置文件片段示例

   ```
   apiVersion: v1
   baseDomain: example.com
   credentialsMode: Manual
   # ...
   ```

2. 如果您之前还没有创建安装清单文件，请运行以下命令：

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

   其中 **<installation_directory>** 是安装程序在其中创建文件的目录。

3. 运行以下命令，将 **ccoctl** 工具生成的清单复制到安装程序创建的 **manifests** 目录中：

   ```
   $ cp /<path_to_ccoctl_output_dir>/manifests/* ./manifests/
   ```

4. 运行以下命令，将 **ccoctl** 工具在 **tls** 目录中生成的私钥复制到安装目录中：

   ```
   $ cp -a /<path_to_ccoctl_output_dir>/tls .
   ```

6.3.8.9. 部署集群

您可以在兼容云平台上安装 OpenShift Container Platform。

重要

在初始安装过程中，您只能运行安装程序的 **create cluster** 命令一次。

先决条件

- 您已使用托管集群的云平台配置了帐户。
- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。
- 已确认主机上的云供应商帐户具有部署集群的正确权限。权限不正确的帐户会导致安装过程失败，并显示包括缺失权限的错误消息。

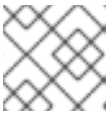
流程

1. 进入包含安装程序的目录并初始化集群部署：

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

1. 对于 **<installation_directory>**，请指定自定义 **./install-config.yaml** 文件的位置。
2. 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不是 **info**。

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



注意

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

验证

当集群部署成功完成时：

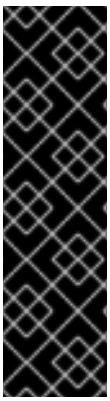
- 终端会显示用于访问集群的说明，包括指向 Web 控制台和 **kubeadmin** 用户的凭证的链接。
- 凭证信息还会输出到 **<installation_directory>/.openshift_install.log**.

重要

不要删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

输出示例

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



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

6.3.8.10. 使用 CLI 登录集群

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

先决条件

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

流程

1. 导出 **kubeadmin** 凭证：

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

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

2. 验证您可以使用导出的配置成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

6.3.8.11. 使用 Web 控制台登录到集群

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

先决条件

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

流程

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

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

注意

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

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

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

注意

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

输出示例

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

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

其他资源

- 如需有关 访问和了解 OpenShift Container Platform Web 控制台的更多详情，请参阅 访问 Web 控制台。

6.3.8.12. 后续步骤

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

6.3.9. 在 AWS 上将集群安装到 Secret 或 Top Secret 区域

在 OpenShift Container Platform 版本 4.16 中，您可以在 Amazon Web Services (AWS) 上将集群安装到以下 secret 区域：

- Secret Commercial Cloud Services (SC2S)
- 商业云服务 (C2S)

要在任一区域中配置集群，请在安装集群前更改 **install config.yaml** 文件中的参数。

警告

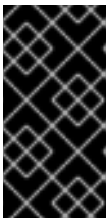
在 OpenShift Container Platform 4.16 中，安装程序使用 Cluster API 而不是 Terraform 在 AWS 上安装时置备集群基础架构。在 OpenShift Container Platform 4.16 发布时，使用 Cluster API 在 AWS 中将集群安装到 secret 或 top-secret 区域还没有被测试。在测试了安装到 secret 区域后，本文档会做相应的更新。

Network Load Balancers 对 secret 或 top secret 区域中的安全组的支持存在一个已知的问题，这会导致在这些区域中安装失败。如需更多信息，请参阅 OCPBUGS-33311。

6.3.9.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读选择集群安装方法并为用户准备它的文档。

- 已将 AWS 帐户配置为托管集群。



重要

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

- 如果使用防火墙，将其配置为允许集群需要访问的站点。

6.3.9.2. AWS secret 区域

支持以下 AWS secret 分区：

- **us-isob-east-1** (SC2S)
- **us-iso-east-1** (C2S)

注意

AWS SC2S 和 C2S 区域的最大支持 MTU 与 AWS 商业区域不同。有关在安装过程中配置 MTU 的更多信息，请参阅 *使用网络自定义在 AWS 中安装集群*中的 *Cluster Network Operator 配置对象*部分。

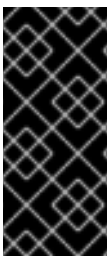
6.3.9.3. 安装要求

红帽不会为 AWS Secret 和 Top Secret 区域发布 Red Hat Enterprise Linux CoreOS (RHCOS) Amzaon Machine Image。

在安装集群前，您必须：

- 上传自定义 RHCOS AMI。
- 手动创建安装配置文件 (**install-config.yaml**)。
- 在安装配置文件中指定 AWS 区域和附带的自定义 AMI。

您不能使用 OpenShift Container Platform 安装程序创建安装配置文件。安装程序不会列出没有原生支持 RHCOS AMI 的 AWS 区域。



重要

您还必须在 **install-config.yaml** 文件的 **additionalTrustBundle** 字段中定义自定义 CA 证书，因为 AWS API 需要自定义 CA 信任捆绑包。要允许安装程序访问 AWS API，还必须在运行安装程序的机器上定义 CA 证书。您必须将 CA 捆绑包添加到机器上的信任存储中，使用 **AWS_CA_BUNDLE** 环境变量，或者在 AWS 配置文件的 **ca_bundle** 字段中定义 CA 捆绑包。

6.3.9.4. 私有集群

您可以部署不公开外部端点的私有 OpenShift Container Platform 集群。私有集群只能从内部网络访问，且无法在互联网中看到。

注意

AWS 顶级 Secret 区域的 Route 53 不支持公共区。因此，如果集群部署到 AWS 顶级 Secret 区域，则集群必须是私有的。

默认情况下，OpenShift Container Platform 被置备为使用可公开访问的 DNS 和端点。在部署集群时，私有集群会将 DNS、Ingress Controller 和 API 服务器设置为私有。这意味着集群资源只能从您的内部网络访问，且不能在互联网中看到。

重要

如果集群有任何公共子网，管理员创建的负载均衡器服务可能会公开访问。为确保集群安全性，请验证这些服务是否已明确标注为私有。

要部署私有集群，您必须：

- 使用满足您的要求的现有网络。集群资源可能会在网络上的其他集群间共享。
- 从有权访问的机器中部署：
  - 您置备的云的 API 服务。
  - 您调配的网络上的主机。
  - 用于获取安装介质的互联网。

您可以使用符合这些访问要求的机器，并按照您的公司规定进行操作。例如，该机器可以是云网络中的堡垒主机，也可以是可通过 VPN 访问网络的机器。

6.3.9.4.1. AWS 中的私有集群

要在 Amazon Web Services (AWS) 上创建私有集群，您必须提供一个现有的私有 VPC 和子网来托管集群。安装程序还必须能够解析集群所需的 DNS 记录。安装程序将 Ingress Operator 和 API 服务器配置为只可以从私有网络访问。

集群仍然需要访问互联网来访问 AWS API。

安装私有集群时不需要或创建以下项目：

- 公共子网
- 支持公共入口的公共负载均衡器
- 与集群的 **baseDomain** 匹配的公共 Route 53 区域

安装程序会使用您指定的 **baseDomain** 来创建专用的 Route 53 区域以及集群所需的记录。集群被配置，以便 Operator 不会为集群创建公共记录，且所有集群机器都放置在您指定的私有子网中。

6.3.9.4.1.1. 限制：

为私有集群添加公共功能的能力有限。

- 在安装后，您无法在不进行额外操作的情况下公开 Kubernetes API 端点。这些额外的操作包括为使用中的每个可用区在 VPC 中创建公共子网，创建公共负载均衡器，以及配置 control plane 安全组以便 6443 端口（Kubernetes API 端口）可以接受来自于互联网的网络流量。

- 如果使用公共服务类型负载均衡器，您必须在每个可用区中为公共子网添加 **kubernetes.io/cluster/<cluster-infra-id>: shared** 标签，以便 AWS 可使用它们来创建公共负载均衡器。

6.3.9.5. 关于使用自定义 VPC

在 OpenShift Container Platform 4.16 中，您可以在 Amazon Web Services (AWS)的现有 Amazon Virtual Private Cloud (VPC)中将集群部署到现有子网中。通过将 OpenShift Container Platform 部署到现有的 AWS VPC 中，您可能会避开新帐户中的限制，或者更容易地利用公司所设置的操作限制。如果您无法获得您自己创建 VPC 所需的基础架构创建权限，请使用这个安装选项。

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

6.3.9.5.1. 使用 VPC 的要求

安装程序不再创建以下组件：

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

注意

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

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

安装程序无法：

- 分割网络范围供集群使用。
- 设置子网的路由表。
- 设置 VPC 选项，如 DHCP。

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

您的 VPC 必须满足以下特征：

- VPC 不能使用 **kubernetes.io/cluster/.*: owned**, **Name**, 和 **openshift.io/cluster** 标签。

安装程序会修改子网以添加 **kubernetes.io/cluster/.*: shared** 标签，因此您的子网必须至少有一个可用的空闲标签插槽。请参阅 AWS 文档中的 标签限制 部分，以确认安装程序可以为您指定的每个子网添加标签。您不能使用 **Name** 标签，因为它与 EC2 **Name** 字段重叠，且安装失败。

- 如果要将 OpenShift Container Platform 集群扩展到 AWS Outpost 并具有现有的 Outpost 子网，现有的子网必须使用 **kubernetes.io/cluster/unmanaged: true** 标签。如果您没有应用此标签，因为 Cloud Controller Manager 在 Outpost 子网中创建服务负载均衡器，安装会失败，这是不受支持的配置。

- 您需要在您的 VPC 中启用 **enableDnsSupport** 和 **enableDnsHostnames** 属性，以便集群可以使用附加到 VPC 中的 Route 53 区来解析集群内部的 DNS 记录。请参阅 AWS 文档中的您的 VPC 中的 DNS 支持部分 。
  如果要使用您自己的 Route 53 托管私有区，您必须在安装集群前将现有托管区与 VPC 相关联。您可以使用 **install-config.yaml** 文件中的 **platform.aws.hostedZone** 和 **platform.aws.hostedZoneRole** 字段定义托管区。您可以通过与安装集群的帐户共享来使用来自另一个帐户的私有托管区。如果使用另一个帐户的私有托管区，则必须使用 **Passthrough** 或 **Manual** 凭证模式。

SC2S 或 C2S 区域中的一个集群无法访问 EC2、ELB 和 S3 端点的公共 IP 地址。根据您要在安装过程中限制互联网流量的级别，有以下配置选项：

选项 1：创建 VPC 端点
创建 VPC 端点，并将其附加到集群使用的子网。将端点命名为如下：

SC2S

- **elasticloadbalancing.<aws_region>.sc2s.sgov.gov**
- **ec2.<aws_region>.sc2s.sgov.gov**
- **s3.<aws_region>.sc2s.sgov.gov**

C2S

- **elasticloadbalancing.<aws_region>.c2s.ic.gov**
- **ec2.<aws_region>.c2s.ic.gov**
- **s3.<aws_region>.c2s.ic.gov**

通过这个选项，网络流量在 VPC 和所需的 AWS 服务之间保持私有。

选项 2：创建一个没有 VPC 端点的代理
作为安装过程的一部分，您可以配置 HTTP 或 HTTPS 代理。使用此选项时，互联网流量会通过代理访问所需的 AWS 服务。

选项 3：创建带有 VPC 端点的代理
作为安装过程的一部分，您可以使用 VPC 端点配置 HTTP 或 HTTPS 代理。创建 VPC 端点，并将其附加到集群使用的子网。将端点命名为如下：

SC2S

- **elasticloadbalancing.<aws_region>.sc2s.sgov.gov**
- **ec2.<aws_region>.sc2s.sgov.gov**
- **s3.<aws_region>.sc2s.sgov.gov**

C2S

- **elasticloadbalancing.<aws_region>.c2s.ic.gov**
- **ec2.<aws_region>.c2s.ic.gov**
- **s3.<aws_region>.c2s.ic.gov**

在 **install-config.yaml** 文件中配置代理时，将这些端点添加到 **noProxy** 字段。通过这个选项，代理会阻止集群直接访问互联网。但是，您的 VPC 和所需的 AWS 服务之间网络流量保持私有。

所需的 VPC 组件

您必须提供合适的 VPC 和子网，以便与您的机器通信。

组件	AWS 类型	描述
VPC	• **AWS::EC2::VPC** • **AWS::EC2::VPCEndpoint**	您必须提供一个公共 VPC 供集群使用。VPC 使用引用每个子网的路由表的端点，以改进与托管在 S3 中的 registry 的通信。
公共子网	• **AWS::EC2::Subnet** • **AWS::EC2::SubnetNetworkAclAssociation**	您的 VPC 必须有 1 到 3 个可用区的公共子网，并将其与适当的入口规则关联。
互联网网关	• **AWS::EC2::InternetGateway** • **AWS::EC2::VPCGatewayAttachment** • **AWS::EC2::RouteTable** • **AWS::EC2::Route** • **AWS::EC2::SubnetRouteTableAssociation** • **AWS::EC2::NatGateway** • **AWS::EC2::EIP**	您必须有一个公共互联网网关，以及附加到 VPC 的公共路由。在提供的模板中，每个公共子网都有一个具有 EIP 地址的 NAT 网关。这些 NAT 网关允许集群资源（如专用子网实例）访问互联网，而有些受限网络或代理场景则不需要它们。
网络访问控制	• **AWS::EC2::NetworkAcl** • **AWS::EC2::NetworkAclEntry**	您必须允许 VPC 访问下列端口： **端口** — **原因** **80** — 入站 HTTP 流量 **443** — 入站 HTTPS 流量 **22** — 入站 SSH 流量

组件	AWS 类型	描述	
		1024 - 65535	入站临时流量
		0 - 65535	出站临时流量
专用子网	• **AWS::EC2::Subnet** • **AWS::EC2::RouteTable** • **AWS::EC2::SubnetRouteTableAssociation**	您的 VPC 可以具有私有子网。提供的 CloudFormation 模板可为 1 到 3 个可用区创建专用子网。如果您使用专用子网，必须为其提供适当的路由和表。	

6.3.9.5.2. VPC 验证

要确保您提供的子网适合您的环境，安装程序会确认以下信息：

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

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

6.3.9.5.3. 权限划分

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

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

6.3.9.5.4. 集群间隔离

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

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

- 整个网络都允许 control plane TCP 6443 入站流量 (Kubernetes API)。
- 整个网络都允许 control plane TCP 22623 入站流量 (MCS) 。

6.3.9.5.5. 可选： AWS 安全组

默认情况下，安装程序会创建安全组并将其附加到 control plane 和计算机器。不可修改与默认安全组关联的规则。

但是，您可以将与现有 VPC 关联的其他现有 AWS 安全组应用到 control plane 和计算机器。应用自定义安全组可帮助您满足机构的安全需求，在这种情况下，您需要控制这些机器的传入或传出流量。

作为安装过程的一部分，您可以在部署集群前通过修改 **install-config.yaml** 文件来应用自定义安全组。

如需更多信息，请参阅"将现有 AWS 安全组应用到集群"。

6.3.9.6. 在 AWS 中上传自定义 RHCOS AMI

如果要部署到自定义 Amazon Web Services（AWS）区域，您必须上传属于该区域的自定义 Red Hat Enterprise Linux CoreOS（RHCOS）Amazon Machine Image（AMI）。

先决条件

- 已配置了一个 AWS 帐户。
- 已使用所需的 IAM 服务角色创建 Amazon S3 存储桶。
- 将 RHCOS VMDK 文件上传到 Amazon S3。RHCOS VMDK 文件必须是小于或等于您要安装的 OpenShift Container Platform 版本的最高版本。
- 您下载了 AWS CLI 并安装到您的计算机上。请参阅使用捆绑安装程序安装 AWS CLI。

流程

1. 将 AWS 配置集导出为环境变量：

```
$ export AWS_PROFILE=<aws_profile> 1
```

2. 将与自定义 AMI 关联的区域导出为环境变量：

```
$ export AWS_DEFAULT_REGION=<aws_region> 1
```

3. 将上传至 Amazon S3 的 RHCOS 版本导出为环境变量：

```
$ export RHCOS_VERSION=<version> 1
```

1 1 1 RHCOS VMDK 版本，如 **4.16.0**。

4. 将 Amazon S3 存储桶名称导出为环境变量：

```
$ export VMIMPORT_BUCKET_NAME=<s3_bucket_name>
```

5. 创建 **containers.json** 文件并定义 RHCOS VMDK 文件：

```
$ cat <<EOF > containers.json
{
  "Description": "rhcos-${RHCOS_VERSION}-x86_64-aws.x86_64",
  "Format": "vmdk",
  "UserBucket": {
    "S3Bucket": "${VMIMPORT_BUCKET_NAME}",
    "S3Key": "rhcos-${RHCOS_VERSION}-x86_64-aws.x86_64.vmdk"
  }
}
EOF
```

6. 将 RHCOS 磁盘导入为 Amazon EBS 快照：

```
$ aws ec2 import-snapshot --region ${AWS_DEFAULT_REGION} \
    --description "<description>" \ 1
    --disk-container "file://<file_path>/containers.json" 2
```

1 导入 RHCOS 磁盘的描述，如 **rhcos-${RHCOS_VERSION}-x86_64-aws.x86_64**。

2 描述 RHCOS 磁盘的 JSON 文件的文件路径。JSON 文件应包含您的 Amazon S3 存储桶名称和密钥。

7. 检查镜像导入的状态：

```
$ watch -n 5 aws ec2 describe-import-snapshot-tasks --region ${AWS_DEFAULT_REGION}
```

输出示例

```
{
    "ImportSnapshotTasks": [
        {
            "Description": "rhcos-4.7.0-x86_64-aws.x86_64",
            "ImportTaskId": "import-snap-fh6i8uil",
            "SnapshotTaskDetail": {
                "Description": "rhcos-4.7.0-x86_64-aws.x86_64",
                "DiskImageSize": 819056640.0,
                "Format": "VMDK",
                "SnapshotId": "snap-06331325870076318",
                "Status": "completed",
                "UserBucket": {
                    "S3Bucket": "external-images",
                    "S3Key": "rhcos-4.7.0-x86_64-aws.x86_64.vmdk"
                }
            }
        }
    ]
}
```

复制 **SnapshotId** 以注册镜像。

8. 从 RHCOS 快照创建自定义 RHCOS AMI:

```
$ aws ec2 register-image \
```

```
    --region ${AWS_DEFAULT_REGION} \
    --architecture x86_64 \ 1
    --description "rhcos-${RHCOS_VERSION}-x86_64-aws.x86_64" \ 2
    --ena-support \
    --name "rhcos-${RHCOS_VERSION}-x86_64-aws.x86_64" \ 3
    --virtualization-type hvm \
    --root-device-name '/dev/xvda' \
    --block-device-mappings 'DeviceName=/dev/xvda,Ebs=
{DeleteOnTermination=true,SnapshotId=<snapshot_ID>}' 4
```

1. RHCOS VMDK 架构类型，如 **x86_64**, **aarch64**, **s390x**, 或 **ppc64le**。
2. 来自导入快照的 **Description**。
3. RHCOS AMI 的名称。
4. 导入的快照中的 **SnapshotID**。

如需了解更多有关这些 API 的信息，请参阅 AWS 文档 导入快照 和 创建由 EBS 支持的 AMI。

6.3.9.7. 手动创建安装配置文件

安装集群要求您手动创建安装配置文件。

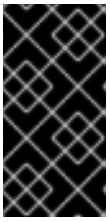
先决条件

- 您上传了一个自定义 RHCOS AMI。
- 您的本地机器上有一个 SSH 公钥供安装程序使用。该密钥将用于在集群节点上进行 SSH 身份验证，以进行调试和灾难恢复。
- 已获取 OpenShift Container Platform 安装程序和集群的 pull secret。

流程

1. 创建一个安装目录来存储所需的安装资产：

```
$ mkdir <installation_directory>
```



重要

您必须创建一个目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

2. 自定义提供的 **install-config.yaml** 文件模板示例，并将其保存在 **<installation_directory>** 中。

注意

此配置文件必须命名为 **install-config.yaml**。

3. 备份 **install-config.yaml** 文件，以便您可以使用它安装多个集群。

重要

install-config.yaml 文件会在安装过程的下一步中使用。现在必须备份它。

其他资源

- AWS 的安装配置参数

6.3.9.7.1. 为 AWS 测试的实例类型

以下 Amazon Web Services(AWS) 实例类型已经过 OpenShift Container Platform 测试。

注意

将以下图中包含的机器类型用于 AWS 实例。如果您使用没有在图表中列出的实例类型，请确保使用的实例大小与名为"最小资源要求"的部分中列出的最少资源要求匹配。

例 6.41. 基于 64 位 x86 架构的机器类型用于 secret 区域

- **c4.***
- **c5.***
- **i3.***
- **m4.***
- **m5.***
- **r4.***
- **r5.***
- **t3.***

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

您可以自定义安装配置文件 (**install-config.yaml**)，以指定有关 OpenShift Container Platform 集群平台的更多详细信息，或修改所需参数的值。

重要

此示例 YAML 文件仅供参考。使用它作为资源，在您手动创建的安装配置文件中输入参数值。

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

```
  name: master
  platform:
    aws:
      zones:
      - us-iso-east-1a
      - us-iso-east-1b
      rootVolume:
        iops: 4000
        size: 500
        type: io1 6
      metadataService:
        authentication: Optional 7
      type: m6i.xlarge
  replicas: 3
compute: 8
- hyperthreading: Enabled 9
  name: worker
  platform:
    aws:
      rootVolume:
        iops: 2000
        size: 500
        type: io1 10
      metadataService:
        authentication: Optional 11
      type: c5.4xlarge
      zones:
      - us-iso-east-1a
      - us-iso-east-1b
  replicas: 3
metadata:
  name: test-cluster 12
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 13
  serviceNetwork:
  - 172.30.0.0/16
platform:
  aws:
    region: us-iso-east-1 14
    propagateUserTags: true 15
    userTags:
      adminContact: jdoe
      costCenter: 7536
    subnets: 16
    - subnet-1
    - subnet-2
    - subnet-3
    amiID: ami-96c6f8f7 17 18
    serviceEndpoints: 19
```

```
    - name: ec2
      url: https://vpce-id.ec2.us-west-2.vpce.amazonaws.com
  hostedZone: Z3URY6TWQ91KVV (20)
fips: false (21)
sshKey: ssh-ed25519 AAAA... (22)
publish: Internal (23)
pullSecret: '{"auths": ...}' (24)
additionalTrustBundle: | (25)
  -----BEGIN CERTIFICATE-----
  <MY_TRUSTED_CA_CERT>
  -----END CERTIFICATE-----
```

(1) (12) (14) (17) (24) 必需。

(2) 可选：添加此参数来强制 Cloud Credential Operator (CCO) 使用指定的模式。默认情况下，CCO 使用 **kube-system** 命名空间中的 root 凭证来动态尝试决定凭证的功能。有关 CCO 模式的详情，请参阅*身份验证和授权*指南中的"About the Cloud Credential Operator"部分。

(3) (8) (15) 如果没有提供这些参数和值，安装程序会提供默认值。

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

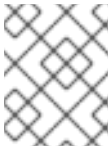
(5) (9) 是否要启用或禁用并发多线程或 **超线程**。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将 参数值设置为 **Disabled** 来禁用它。如果在某些集群机器中禁用并发多线程，则必须在所有集群机器中禁用它。

重要

如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。如果您对机器禁用并发多线程，请使用较大的实例类型，如 **m4.2xlarge** 或 **m5.2xlarge**。

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

(7) (11) 是否需要 Amazon EC2 实例元数据服务 v2 (IMDSv2)。为了要求 IMDSv2，请将参数值设置为 **Required**。要允许使用 IMDSv1 和 IMDSv2，请将参数值设置为 **Optional**。如果没有指定值，则允许 IMDSv1 和 IMDSv2。



注意

在集群安装过程中设置的 control plane 机器的 IMDS 配置只能使用 AWS CLI 更改。可以使用计算机器集来更改计算机器的 IMDS 配置。

(13) 要安装的集群网络插件。默认值 **OVNKubernetes** 是唯一支持的值。

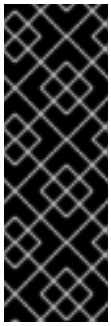
(16) 如果您提供自己的 VPC，为集群使用的每个可用区指定子网。

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

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

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

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

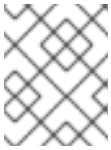


重要

要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 Red Hat Enterprise Linux (RHEL) 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅在 FIPS 模式中安装该系统。

当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS)时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。

22 您可以选择提供您用来访问集群中机器的 **sshKey** 值。



注意

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

23 如何发布集群的面向用户的端点。将 **publish** 设置为 **Internal** 以部署一个私有集群，它不能被互联网访问。默认值为 **External**。

25 自定义 CA 证书。当部署到 SC2S 或 C2S 区域时这是必需的，因为 AWS API 需要自定义 CA 信任捆绑包。

6.3.9.7.3. 在安装过程中配置集群范围的代理

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

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 **Proxy** 对象的 **spec.noProxy** 字段中添加站点来绕过代理。

注意

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

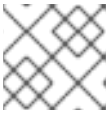
对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，**Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

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

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: ec2.<aws_region>.amazonaws.com,elasticloadbalancing.
<aws_region>.amazonaws.com,s3.<aws_region>.amazonaws.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 **.** 以仅匹配子域。例如，**.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 ***** 绕过所有目的地的代理。如果您已将 Amazon **EC2**、**Elastic Load Balancing** 和 **S3** VPC 端点添加到 VPC 中，您必须将这些端点添加到 **noProxy** 字段。
4. 如果提供，安装程序会在 **openshift-config** 命名空间中生成名为 **user-ca-bundle** 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 **trusted-ca-bundle** 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， **Proxy** 对象的 **trustedCA** 字段中也会引用此配置映射。**additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。
5. 可选：决定 **Proxy** 对象的配置以引用 **trustedCA** 字段中 **user-ca-bundle** 配置映射的策略。允许的值是 **Proxyonly** 和 **Always**。仅在配置了 **http/https** 代理时，使用 **Proxyonly** 引用 **user-ca-bundle** 配置映射。使用 **Always** 始终引用 **user-ca-bundle** 配置映射。默认值为 **Proxyonly**。



注意

安装程序不支持代理的 **readinessEndpoints** 字段。

注意

如果安装程序超时，重启并使用安装程序的 **wait-for** 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

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

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

注意

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

6.3.9.7.4. 将现有 AWS 安全组应用到集群

将现有 AWS 安全组应用到 control plane 和计算机器可帮助您满足机构的安全需求，在这种情况下，您需要控制这些机器的传入或传出流量。

先决条件

- 您已在 AWS 中创建安全组。如需更多信息，请参阅使用 安全组的 AWS 文档。
- 安全组必须与您要将集群部署到的现有 VPC 关联。安全组不能与另一个 VPC 关联。
- 您有一个现有的 **install-config.yaml** 文件。

流程

1. 在 **install-config.yaml** 文件中，编辑 **compute.platform.aws.additionalSecurityGroupIDs** 参数，为您的计算机器指定一个或多个自定义安全组。
2. 编辑 **controlPlane.platform.aws.additionalSecurityGroupIDs** 参数，为您的 control plane 机器指定一个或多个自定义安全组。
3. 保存文件并在部署集群时引用。

指定自定义安全组的 install-config.yaml 文件示例

```
# ...
compute:
- hyperthreading: Enabled
  name: worker
  platform:
    aws:
      additionalSecurityGroupIDs:
        - sg-1 (1)
        - sg-2
  replicas: 3
controlPlane:
  hyperthreading: Enabled
  name: master
  platform:
    aws:
      additionalSecurityGroupIDs:
        - sg-3
        - sg-4
  replicas: 3
platform:
  aws:
    region: us-east-1
    subnets: (2)
      - subnet-1
      - subnet-2
      - subnet-3
```

1. 在 Amazon EC2 控制台中显示时指定安全组的名称，包括 **sg** 前缀。
2. 指定集群使用的每个可用区的子网。

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

默认情况下，管理员 secret 存储在 **kube-system** 项目中。如果您在 **install-config.yaml** 文件中将 **credentialsMode** 参数配置为 **Manual**，则必须使用以下替代方案之一：

- 要手动管理长期云凭证，请按照手动创建长期凭证中的步骤操作。
- 要实现在集群外为各个组件管理的短期凭证，请按照配置 A WS 集群以使用短期凭证 中的步骤操作。

6.3.9.8.1. 手动创建长期凭证

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

流程

1. 如果您没有将 **install-config.yaml** 配置文件中的 **credentialsMode** 参数设置为 **Manual**，请修改值，如下所示：

   配置文件片段示例

   ```
   apiVersion: v1
   baseDomain: example.com
   credentialsMode: Manual
   # ...
   ```

2. 如果您之前还没有创建安装清单文件，请运行以下命令：

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

   其中 **<installation_directory>** 是安装程序在其中创建文件的目录。

3. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

   ```
   $ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
   ```

4. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 自定义资源 (CR) 列表：

   ```
   $ oc adm release extract \
     --from=$RELEASE_IMAGE \
     --credentials-requests \
     --included \ 1
     --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
     --to=<path_to_directory_for_credentials_requests> 3
   ```

   1. **--included** 参数仅包含特定集群配置所需的清单。

2 指定 **install-config.yaml** 文件的位置。

3 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

此命令为每个 **CredentialsRequest** 对象创建一个 YAML 文件。

CredentialsRequest 对象示例

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

5. 在之前生成的 **openshift-install** 清单目录中为 secret 创建 YAML 文件。secret 必须使用在 **spec.secretRef** 中为每个 **CredentialsRequest** 定义的命名空间和 secret 名称存储。

带有 secret 的 CredentialsRequest 对象示例

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component_credentials_request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: AWSProviderSpec
    statementEntries:
    - effect: Allow
      action:
      - s3:CreateBucket
      - s3:DeleteBucket
      resource: "*"
      ...
  secretRef:
    name: <component_secret>
    namespace: <component_namespace>
  ...
```

Secret 对象示例

```
apiVersion: v1
kind: Secret
metadata:
  name: <component_secret>
  namespace: <component_namespace>
data:
  aws_access_key_id: <base64_encoded_aws_access_key_id>
  aws_secret_access_key: <base64_encoded_aws_secret_access_key>
```

重要

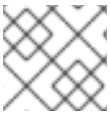
在升级使用手动维护凭证的集群前，您必须确保 CCO 处于可升级状态。

6.3.9.8.2. 将 AWS 集群配置为使用短期凭证

要安装配置为使用 AWS 安全令牌服务 (STS) 的集群，您必须配置 CCO 实用程序并为集群创建所需的 AWS 资源。

6.3.9.8.2.1. 配置 Cloud Credential Operator 工具

当 Cloud Credential Operator(CCO)以手动模式运行时，要从集群外部创建和管理云凭证，提取并准备 CCO 实用程序(**ccoctl**)二进制文件。



注意

ccoctl 工具是在 Linux 环境中运行的 Linux 二进制文件。

先决条件

- 您可以访问具有集群管理员权限的 OpenShift Container Platform 帐户。
- 已安装 OpenShift CLI(**oc**)。
- 您已为 **ccoctl** 工具创建了用于以下权限的 AWS 帐户：

  例 6.42. 所需的 AWS 权限

  所需的 iam 权限

  - **iam:CreateOpenIDConnectProvider**
  - **iam:CreateRole**
  - **iam:DeleteOpenIDConnectProvider**
  - **iam:DeleteRole**
  - **iam:DeleteRolePolicy**
  - **iam:GetOpenIDConnectProvider**
  - **iam:GetRole**

- **iam:GetUser**
- **iam:ListOpenIDConnectProviders**
- **iam:ListRolePolicies**
- **iam:ListRoles**
- **iam:PutRolePolicy**
- **iam:TagOpenIDConnectProvider**
- **iam:TagRole**

所需的 s3 权限

- **s3:CreateBucket**
- **s3:DeleteBucket**
- **s3:DeleteObject**
- **s3:GetBucketAcl**
- **s3:GetBucketTagging**
- **s3:GetObject**
- **s3:GetObjectAcl**
- **s3:GetObjectTagging**
- **s3:ListBucket**
- **s3:PutBucketAcl**
- **s3:PutBucketPolicy**
- **s3:PutBucketPublicAccessBlock**
- **s3:PutBucketTagging**
- **s3:PutObject**
- **s3:PutObjectAcl**
- **s3:PutObjectTagging**

所需的 cloudfront 权限

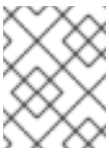
- **cloudfront:ListCloudFrontOriginAccessIdentities**
- **cloudfront:ListDistributions**
- **cloudfront:ListTagsForResource**

如果您计划通过公共 CloudFront 发行版 URL 将 OIDC 配置存储在 IAM 身份提供程序记录的私有

如果您计划通过公共 CloudFront 发行版 URL 将 OIDC 配置存储在 IAM 身份提供程序访问的私有 S3 存储桶中，则运行 **ccoctl** 工具的 AWS 帐户需要以下额外权限：

例 6.43. 使用 CloudFront 私有 S3 存储桶的额外权限

- **cloudfront:CreateCloudFrontOriginAccessIdentity**
- **cloudfront:CreateDistribution**
- **cloudfront:DeleteCloudFrontOriginAccessIdentity**
- **cloudfront:DeleteDistribution**
- **cloudfront:GetCloudFrontOriginAccessIdentity**
- **cloudfront:GetCloudFrontOriginAccessIdentityConfig**
- **cloudfront:GetDistribution**
- **cloudfront:TagResource**
- **cloudfront:UpdateDistribution**



注意

在使用 **ccoctl aws create-all** 命令处理凭证请求时，这些额外权限支持使用 **--create-private-s3-bucket** 选项。

流程

1. 运行以下命令，为 OpenShift Container Platform 发行镜像设置变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. 运行以下命令，从 OpenShift Container Platform 发行镜像获取 CCO 容器镜像：

```
$ CCO_IMAGE=$(oc adm release info --image-for='cloud-credential-operator'
$RELEASE_IMAGE -a ~/.pull-secret)
```

注意

确保 **$RELEASE_IMAGE** 的架构与将使用 **ccoctl** 工具的环境架构相匹配。

3. 运行以下命令，将 CCO 容器镜像中的 **ccoctl** 二进制文件提取到 OpenShift Container Platform 发行镜像中：

```
$ oc image extract $CCO_IMAGE \
  --file="/usr/bin/ccoctl.<rhel_version>" \ 1
  -a ~/.pull-secret
```

1 对于 **<rhel_version>**，请指定与主机使用的 Red Hat Enterprise Linux (RHEL) 版本对应的值。如果没有指定值，则默认使用 **ccoctl.rhel8**。以下值有效：

- **rhel8**: 为使用 RHEL 8 的主机指定这个值。
- **rhel9** ：为使用 RHEL 9 的主机指定这个值。

4. 运行以下命令更改权限以使 **ccoctl** 可执行：

```
$ chmod 775 ccoctl.<rhel_version>
```

验证

- 要验证 **ccoctl** 是否可使用，请运行以下命令来显示帮助文件：

```
$ ccoctl --help
```

ccoctl --help 的输出

```
OpenShift credentials provisioning tool

Usage:
  ccoctl [command]

Available Commands:
  aws         Manage credentials objects for AWS cloud
  azure       Manage credentials objects for Azure
  gcp         Manage credentials objects for Google cloud
  help        Help about any command
  ibmcloud    Manage credentials objects for IBM Cloud
  nutanix     Manage credentials objects for Nutanix

Flags:
  -h, --help   help for ccoctl

Use "ccoctl [command] --help" for more information about a command.
```

6.3.9.8.2.2. 使用 Cloud Credential Operator 实用程序创建 AWS 资源

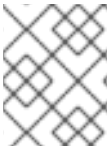
创建 AWS 资源时有以下选项：

- 您可以使用 **ccoctl aws create-all** 命令自动创建 AWS 资源。这是创建资源的最快速方法。请参阅使用单个命令创建 AWS 资源。
- 如果您需要在修改 AWS 资源前查看 **ccoctl** 工具创建的 JSON 文件，或者 **ccoctl** 工具用于创建 AWS 资源的过程无法自动满足组织的要求，您可以单独创建 AWS 资源。请参阅单独创建 AWS 资源。

6.3.9.8.2.2.1. 使用单个命令创建 AWS 资源

如果 **ccoctl** 工具用于创建 AWS 资源的过程自动满足机构的要求，您可以使用 **ccoctl aws create-all** 命令自动创建 AWS 资源。

否则，您可以单独创建 AWS 资源。如需更多信息，请参阅"单独创建 AWS 资源"。



注意

默认情况下，**ccoctl** 在运行命令的目录中创建对象。要在其他目录中创建对象，请使用 **--output-dir** 标志。此流程使用 **<path_to_ccoctl_output_dir>** 来引用这个目录。

先决条件

您必须：

- 提取并准备好 **ccoctl** 二进制文件。

流程

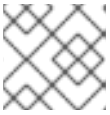
1. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 对象列表：

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
  --included \ 1
  --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
  --to=<path_to_directory_for_credentials_requests> 3
```

1. **--included** 参数仅包含特定集群配置所需的清单。
2. 指定 **install-config.yaml** 文件的位置。
3. 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。



注意

此命令可能需要一些时间才能运行。

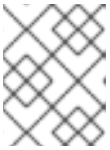
3. 运行以下命令，使用 **ccoctl** 工具处理所有 **CredentialsRequest** 对象：

```
$ ccoctl aws create-all \
  --name=<name> \ 1
  --region=<aws_region> \ 2
  --credentials-requests-dir=<path_to_credentials_requests_directory> \ 3
  --output-dir=<path_to_ccoctl_output_dir> \ 4
  --create-private-s3-bucket 5
```

1. 指定用于标记创建用于跟踪的任何云资源的名称。
2. 指定在其中创建云资源的 AWS 区域。
3. 指定包含组件 **CredentialsRequest** 对象文件的目录。

4. 可选：指定您希望 **ccoctl** 实用程序在其中创建对象的目录。默认情况下，实用程序在运行命令的目录中创建对象。

5. 可选：默认情况下，**ccoctl** 实用程序将 OpenID Connect (OIDC) 配置文件存储在公共 S3 存储桶中，并使用 S3 URL 作为公共 OIDC 端点。要将 OIDC 配置存储在 IAM 身份提供程序通过公共 CloudFront 发行版 URL 访问的专用 S3 存储桶中，请使用 **--create-private-s3-bucket** 参数。



注意

如果您的集群使用 **TechPreviewNoUpgrade** 功能集启用的技术预览功能，则必须包含 **--enable-tech-preview** 参数。

验证

- 要验证 OpenShift Container Platform secret 是否已创建，列出 **<path_to_ccoctl_output_dir>/manifests** 目录中的文件：

```
$ ls <path_to_ccoctl_output_dir>/manifests
```

输出示例

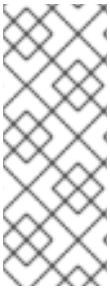
```
cluster-authentication-02-config.yaml
openshift-cloud-credential-operator-cloud-credential-operator-iam-ro-creds-credentials.yaml
openshift-cloud-network-config-controller-cloud-credentials-credentials.yaml
openshift-cluster-api-capa-manager-bootstrap-credentials-credentials.yaml
openshift-cluster-csi-drivers-ebs-cloud-credentials-credentials.yaml
openshift-image-registry-installer-cloud-credentials-credentials.yaml
openshift-ingress-operator-cloud-credentials-credentials.yaml
openshift-machine-api-aws-cloud-credentials-credentials.yaml
```

您可以通过查询 AWS 来验证是否已创建 IAM 角色。如需更多信息，请参阅有关列出 IAM 角色的 AWS 文档。

6.3.9.8.2.2.2. 单独创建 AWS 资源

您可以使用 **ccoctl** 工具单独创建 AWS 资源。这个选项对于在不同用户或部门之间创建这些资源的组织可能很有用。

否则，您可以使用 **ccoctl aws create-all** 命令自动创建 AWS 资源。如需更多信息，请参阅"使用单个命令创建 AWS 资源"。



注意

默认情况下，**ccoctl** 在运行命令的目录中创建对象。要在其他目录中创建对象，请使用 **--output-dir** 标志。此流程使用 **<path_to_ccoctl_output_dir>** 来引用这个目录。

有些 **ccoctl** 命令会发出 AWS API 调用来创建或修改 AWS 资源。您可以使用 **--dry-run** 标志来避免 API 调用。使用此标志可在本地文件系统中创建 JSON 文件。您可以使用 **--cli-input-json** 参数查看和修改 JSON 文件，然后使用 AWS CLI 工具应用它们。

先决条件

- 提取并准备 **ccoctl** 二进制文件。

流程

1. 运行以下命令，生成用于为集群设置 OpenID Connect 供应商的公共和私有 RSA 密钥文件：

```
$ ccoctl aws create-key-pair
```

输出示例

```
2021/04/13 11:01:02 Generating RSA keypair
2021/04/13 11:01:03 Writing private key to /<path_to_ccoctl_output_dir>/serviceaccount-signer.private
2021/04/13 11:01:03 Writing public key to /<path_to_ccoctl_output_dir>/serviceaccount-signer.public
2021/04/13 11:01:03 Copying signing key for use by installer
```

其中 **serviceaccount-signer.private** 和 **serviceaccount-signer.public** 是生成的密钥文件。

此命令还会在 **/<path_to_ccoctl_output_dir>/tls/bound-service-account-signing-key.key** 中创建集群在安装过程中所需的私钥。

2. 运行以下命令，在 AWS 上创建 OpenID Connect 身份提供程序和 S3 存储桶：

```
$ ccoctl aws create-identity-provider \
  --name=<name> \1
  --region=<aws_region> \2
  --public-key-file=<path_to_ccoctl_output_dir>/serviceaccount-signer.public 3
```

1. **<name>** 是用于标记为跟踪而创建的云资源的名称。
2. **<aws-region>** 是将要在其中创建云资源的 AWS 区域。
3. **<path_to_ccoctl_output_dir>** 是 **ccoctl aws create-key-pair** 命令生成的公钥文件的路径。

输出示例

```
2021/04/13 11:16:09 Bucket <name>-oidc created
2021/04/13 11:16:10 OpenID Connect discovery document in the S3 bucket <name>-oidc at .well-known/openid-configuration updated
2021/04/13 11:16:10 Reading public key
2021/04/13 11:16:10 JSON web key set (JWKS) in the S3 bucket <name>-oidc at keys.json updated
2021/04/13 11:16:18 Identity Provider created with ARN: arn:aws:iam::<aws_account_id>:oidc-provider/<name>-oidc.s3.<aws_region>.amazonaws.com
```

其中 **openid-configuration** 是发现文档和 **key.json** 是一个 JSON Web 密钥集文件。

此命令还会在 **/<path_to_ccoctl_output_dir>/manifests/cluster-authentication-02-config.yaml** 中创建 YAML 配置文件。此文件为集群生成的服务帐户令牌设置签发者 URL 字段，以便 AWS IAM 身份提供程序信任令牌。

3. 为集群中的每个组件创建 IAM 角色：

   a. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

   b. 从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 对象列表：

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
  --included \ 1
  --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \
2
  --to=<path_to_directory_for_credentials_requests> 3
```

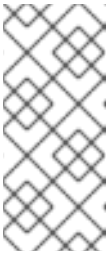
   1 **--included** 参数仅包含特定集群配置所需的清单。

   2 指定 **install-config.yaml** 文件的位置。

   3 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

   c. 运行以下命令，使用 **ccoctl** 工具处理所有 **CredentialsRequest** 对象：

```
$ ccoctl aws create-iam-roles \
  --name=<name> \
  --region=<aws_region> \
  --credentials-requests-dir=<path_to_credentials_requests_directory> \
  --identity-provider-arn=arn:aws:iam::<aws_account_id>:oidc-provider/<name>-oidc.s3.
<aws_region>.amazonaws.com
```



注意

对于使用其他 IAM API 端点的 AWS 环境（如 GovCloud），还必须使用 **--region** 参数指定您的区域。

如果您的集群使用 **TechPreviewNoUpgrade** 功能集启用的技术预览功能，则必须包含 **--enable-tech-preview** 参数。

对于每个 **CredentialsRequest** 对象，**ccoctl** 创建一个带有信任策略的 IAM 角色，该角色与指定的 OIDC 身份提供程序相关联，以及来自 OpenShift Container Platform 发行镜像的每个 **CredentialsRequest** 对象中定义的权限策略。

验证

- 要验证 OpenShift Container Platform secret 是否已创建，列出 **<path_to_ccoctl_output_dir>/manifests** 目录中的文件：

```
$ ls <path_to_ccoctl_output_dir>/manifests
```

输出示例

```
-
```

```
cluster-authentication-02-config.yaml
openshift-cloud-credential-operator-cloud-credential-operator-iam-ro-creds-credentials.yaml
openshift-cloud-network-config-controller-cloud-credentials-credentials.yaml
openshift-cluster-api-capa-manager-bootstrap-credentials-credentials.yaml
openshift-cluster-csi-drivers-ebs-cloud-credentials-credentials.yaml
openshift-image-registry-installer-cloud-credentials-credentials.yaml
openshift-ingress-operator-cloud-credentials-credentials.yaml
openshift-machine-api-aws-cloud-credentials-credentials.yaml
```

您可以通过查询 AWS 来验证是否已创建 IAM 角色。如需更多信息，请参阅有关列出 IAM 角色的 AWS 文档。

6.3.9.8.2.3. 整合 Cloud Credential Operator 实用程序清单

要为单个组件在集群外实现短期安全凭证，您必须将创建 Cloud Credential Operator 实用程序 (**ccoctl**) 的清单文件移到安装程序的正确目录中。

先决条件

- 您已使用托管集群的云平台配置了帐户。
- 您已配置了 Cloud Credential Operator 实用程序 (**ccoctl**)。
- 已使用 **ccoctl** 工具创建了集群所需的云供应商资源。

流程

1. 如果您没有将 **install-config.yaml** 配置文件中的 **credentialsMode** 参数设置为 **Manual**，请修改值，如下所示：

   配置文件片段示例

   ```
   apiVersion: v1
   baseDomain: example.com
   credentialsMode: Manual
   # ...
   ```

2. 如果您之前还没有创建安装清单文件，请运行以下命令：

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

   其中 **<installation_directory>** 是安装程序在其中创建文件的目录。

3. 运行以下命令，将 **ccoctl** 工具生成的清单复制到安装程序创建的 **manifests** 目录中：

   ```
   $ cp /<path_to_ccoctl_output_dir>/manifests/* ./manifests/
   ```

4. 运行以下命令，将 **ccoctl** 工具在 **tls** 目录中生成的私钥复制到安装目录中：

   ```
   $ cp -a /<path_to_ccoctl_output_dir>/tls .
   ```

6.3.9.9. 部署集群

您可以在兼容云平台上安装 OpenShift Container Platform。

重要

在初始安装过程中，您只能运行安装程序的 **create cluster** 命令一次。

先决条件

- 您已使用托管集群的云平台配置了帐户。
- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。
- 已确认主机上的云供应商帐户具有部署集群的正确权限。权限不正确的帐户会导致安装过程失败，并显示包括缺失权限的错误消息。

流程

1. 进入包含安装程序的目录并初始化集群部署：

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

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

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

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

注意

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

验证

当集群部署成功完成时：

- 终端会显示用于访问集群的说明，包括指向 Web 控制台和 **kubeadmin** 用户的凭证的链接。
- 凭证信息还会输出到 **<installation_directory>/.openshift_install.log**.

重要

不要删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

输出示例

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

```
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

6.3.9.10. 使用 CLI 登录集群

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

先决条件

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

流程

1. 导出 **kubeadmin** 凭证：

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

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

2. 验证您可以使用导出的配置成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

6.3.9.11. 使用 Web 控制台登录到集群

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

先决条件

- 有访问安装主机的访问权限。

- 您完成了集群安装，所有集群 Operator 都可用。

流程

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

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

注意

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

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

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

注意

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

输出示例

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

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

其他资源

- 访问Web控制台

6.3.9.12. 后续步骤

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

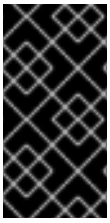
6.3.10. 在 AWS China 上安装集群

在 OpenShift Container Platform 版本 4.16 中，您可以将集群安装到以下 Amazon Web Services (AWS) 中国区域：

- **cn-north-1** (Beijing)
- **cn-northwest-1** (Ningxia)

6.3.10.1. 先决条件

- 您有一个 Internet Content Provider (ICP) 许可证。
- 您可以参阅有关 OpenShift Container Platform 安装和更新流程的详细信息。
- 您可以阅读选择集群安装方法并为用户准备它的文档。
- 已将 AWS 帐户配置为托管集群。
- 如果使用防火墙，将其配置为允许集群需要访问的站点。



重要

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

6.3.10.2. 安装要求

红帽没有发布 AWS China 区域的 Red Hat Enterprise Linux CoreOS (RHCOS) Amzaon 机器镜像。

在安装集群前，您必须：

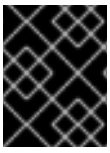
- 上传自定义 RHCOS AMI。
- 手动创建安装配置文件 (**install-config.yaml**)。
- 在安装配置文件中指定 AWS 区域和附带的自定义 AMI。

您不能使用 OpenShift Container Platform 安装程序创建安装配置文件。安装程序不会列出没有原生支持 RHCOS AMI 的 AWS 区域。

6.3.10.3. 私有集群

您可以部署不公开外部端点的私有 OpenShift Container Platform 集群。私有集群只能从内部网络访问，且无法在互联网中看到。

默认情况下，OpenShift Container Platform 被置备为使用可公开访问的 DNS 和端点。在部署集群时，私有集群会将 DNS、Ingress Controller 和 API 服务器设置为私有。这意味着集群资源只能从您的内部网络访问，且不能在互联网中看到。



重要

如果集群有任何公共子网，管理员创建的负载均衡器服务可能会公开访问。为确保集群安全性，请验证这些服务是否已明确标注为私有。

要部署私有集群，您必须：

- 使用满足您的要求的现有网络。集群资源可能会在网络上的其他集群间共享。
- 从有权访问的机器中部署：
  - 您置备的云的 API 服务。

- 您调配的网络上的主机。
- 用于获取安装介质的互联网。

您可以使用符合这些访问要求的机器，并按照您的公司规定进行操作。例如，此计算机可以是云网络上的堡垒主机。

注意

AWS China 不支持 VPC 和您的网络之间的 VPN 连接。有关 Beijing 和 Ningxia 地区的 Amazon VPC 服务的更多信息，请参阅 AWS China 的 Amazon Virtual Private Cloud 文档。

6.3.10.3.1. AWS 中的私有集群

要在 Amazon Web Services (AWS) 上创建私有集群，您必须提供一个现有的私有 VPC 和子网来托管集群。安装程序还必须能够解析集群所需的 DNS 记录。安装程序将 Ingress Operator 和 API 服务器配置为只可以从私有网络访问。

集群仍然需要访问互联网来访问 AWS API。

安装私有集群时不需要或创建以下项目：

- 公共子网
- 支持公共入口的公共负载均衡器
- 与集群的 **baseDomain** 匹配的公共 Route 53 区域

安装程序会使用您指定的 **baseDomain** 来创建专用的 Route 53 区域以及集群所需的记录。集群被配置，以便 Operator 不会为集群创建公共记录，且所有集群机器都放置在您指定的私有子网中。

6.3.10.3.1.1. 限制：

为私有集群添加公共功能的能力有限。

- 在安装后，您无法在不进行额外操作的情况下公开 Kubernetes API 端点。这些额外的操作包括为使用中的每个可用区在 VPC 中创建公共子网，创建公共负载均衡器，以及配置 control plane 安全组以便 6443 端口（Kubernetes API 端口）可以接受来自于互联网的网络流量。
- 如果使用公共服务类型负载均衡器，您必须在每个可用区中为公共子网添加 **kubernetes.io/cluster/<cluster-infra-id>: shared** 标签，以便 AWS 可使用它们来创建公共负载均衡器。

6.3.10.4. 关于使用自定义 VPC

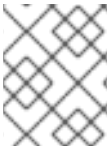
在 OpenShift Container Platform 4.16 中，您可以在 Amazon Web Services (AWS)的现有 Amazon Virtual Private Cloud (VPC)中将集群部署到现有子网中。通过将 OpenShift Container Platform 部署到现有的 AWS VPC 中，您可能会避开新帐户中的限制，或者更容易地利用公司所设置的操作限制。如果您无法获得您自己创建 VPC 所需的基础架构创建权限，请使用这个安装选项。

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

6.3.10.4.1. 使用 VPC 的要求

安装程序不再创建以下组件：

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



注意

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

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

安装程序无法：

- 分割网络范围供集群使用。
- 设置子网的路由表。
- 设置 VPC 选项，如 DHCP。

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

您的 VPC 必须满足以下特征：

- VPC 不能使用 **kubernetes.io/cluster/.*: owned**, **Name**, 和 **openshift.io/cluster** 标签。
  安装程序会修改子网以添加 **kubernetes.io/cluster/.*: shared** 标签，因此您的子网必须至少有一个可用的空闲标签插槽。请参阅 AWS 文档中的 标签限制 部分，以确认安装程序可以为您指定的每个子网添加标签。您不能使用 **Name** 标签，因为它与 EC2 **Name** 字段重叠，且安装失败。

- 如果要将 OpenShift Container Platform 集群扩展到 AWS Outpost 并具有现有的 Outpost 子网，现有的子网必须使用 **kubernetes.io/cluster/unmanaged: true** 标签。如果您没有应用此标签，因为 Cloud Controller Manager 在 Outpost 子网中创建服务负载均衡器，安装会失败，这是不受支持的配置。

- 您需要在您的 VPC 中启用 **enableDnsSupport** 和 **enableDnsHostnames** 属性，以便集群可以使用附加到 VPC 中的 Route 53 区来解析集群内部的 DNS 记录。请参阅 AWS 文档中的您的 VPC 中的 DNS 支持部分 。
  如果要使用您自己的 Route 53 托管私有区，您必须在安装集群前将现有托管区与 VPC 相关联。您可以使用 **install-config.yaml** 文件中的 **platform.aws.hostedZone** 和 **platform.aws.hostedZoneRole** 字段定义托管区。您可以通过与安装集群的帐户共享来使用来自另一个帐户的私有托管区。如果使用另一个帐户的私有托管区，则必须使用 **Passthrough** 或 **Manual** 凭证模式。

如果您在断开连接的环境中工作，则无法访问 EC2、ELB 和 S3 端点的公共 IP 地址。根据您要在安装过程中限制互联网流量的级别，有以下配置选项：

选项 1：创建 VPC 端点
创建 VPC 端点，并将其附加到集群使用的子网。将端点命名为如下：

- **ec2.<aws_region>.amazonaws.com.cn**
- **elasticloadbalancing.<aws_region>.amazonaws.com**
- **s3.<aws_region>.amazonaws.com**

通过这个选项，网络流量在 VPC 和所需的 AWS 服务之间保持私有。

选项 2：创建一个没有 VPC 端点的代理
作为安装过程的一部分，您可以配置 HTTP 或 HTTPS 代理。使用此选项时，互联网流量会通过代理访问所需的 AWS 服务。

选项 3：创建带有 VPC 端点的代理
作为安装过程的一部分，您可以使用 VPC 端点配置 HTTP 或 HTTPS 代理。创建 VPC 端点，并将其附加到集群使用的子网。将端点命名为如下：

- **ec2.<aws_region>.amazonaws.com.cn**
- **elasticloadbalancing.<aws_region>.amazonaws.com**
- **s3.<aws_region>.amazonaws.com**

在 **install-config.yaml** 文件中配置代理时，将这些端点添加到 **noProxy** 字段。通过这个选项，代理会阻止集群直接访问互联网。但是，您的 VPC 和所需的 AWS 服务之间网络流量保持私有。

所需的 VPC 组件

您必须提供合适的 VPC 和子网，以便与您的机器通信。

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

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

6.3.10.4.2. VPC 验证

要确保您提供的子网适合您的环境，安装程序会确认以下信息：

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

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

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

6.3.10.4.3. 权限划分

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

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

6.3.10.4.4. 集群间隔离

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

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

6.3.10.4.5. 可选： AWS 安全组

默认情况下，安装程序会创建安全组并将其附加到 control plane 和计算机器。不可修改与默认安全组关联的规则。

但是，您可以将与现有 VPC 关联的其他现有 AWS 安全组应用到 control plane 和计算机器。应用自定义安全组可帮助您满足机构的安全需求，在这种情况下，您需要控制这些机器的传入或传出流量。

作为安装过程的一部分，您可以在部署集群前通过修改 **install-config.yaml** 文件来应用自定义安全组。

如需更多信息，请参阅"将现有 AWS 安全组应用到集群"。

6.3.10.5. 在 AWS 中上传自定义 RHCOS AMI

如果要部署到自定义 Amazon Web Services（AWS）区域，您必须上传属于该区域的自定义 Red Hat Enterprise Linux CoreOS（RHCOS）Amazon Machine Image（AMI）。

先决条件

- 已配置了一个 AWS 帐户。
- 已使用所需的 IAM 服务角色创建 Amazon S3 存储桶。
- 将 RHCOS VMDK 文件上传到 Amazon S3。RHCOS VMDK 文件必须是小于或等于您要安装的 OpenShift Container Platform 版本的最高版本。

- 您下载了 AWS CLI 并安装到您的计算机上。请参阅使用捆绑安装程序安装 AWS CLI。

流程

1. 将 AWS 配置集导出为环境变量：

```
$ export AWS_PROFILE=<aws_profile> 1
```

1 包含 AWS 凭证的 AWS 配置集名称，如 **beijingadmin**。

2. 将与自定义 AMI 关联的区域导出为环境变量：

```
$ export AWS_DEFAULT_REGION=<aws_region> 1
```

1 AWS 区域，如 **cn-north-1**。

3. 将上传至 Amazon S3 的 RHCOS 版本导出为环境变量：

```
$ export RHCOS_VERSION=<version> 1
```

1 RHCOS VMDK 版本，如 **4.16.0**。

4. 将 Amazon S3 存储桶名称导出为环境变量：

```
$ export VMIMPORT_BUCKET_NAME=<s3_bucket_name>
```

5. 创建 **containers.json** 文件并定义 RHCOS VMDK 文件：

```
$ cat <<EOF > containers.json
{
  "Description": "rhcos-${RHCOS_VERSION}-x86_64-aws.x86_64",
  "Format": "vmdk",
  "UserBucket": {
    "S3Bucket": "${VMIMPORT_BUCKET_NAME}",
    "S3Key": "rhcos-${RHCOS_VERSION}-x86_64-aws.x86_64.vmdk"
  }
}
EOF
```

6. 将 RHCOS 磁盘导入为 Amazon EBS 快照：

```
$ aws ec2 import-snapshot --region ${AWS_DEFAULT_REGION} \
    --description "<description>" \ 1
    --disk-container "file://<file_path>/containers.json" 2
```

1 导入 RHCOS 磁盘的描述，如 **rhcos-${RHCOS_VERSION}-x86_64-aws.x86_64**。

2 描述 RHCOS 磁盘的 JSON 文件的文件路径。JSON 文件应包含您的 Amazon S3 存储桶名称和密钥。

7. 检查镜像导入的状态：

```
$ watch -n 5 aws ec2 describe-import-snapshot-tasks --region ${AWS_DEFAULT_REGION}
```

输出示例

```
{
    "ImportSnapshotTasks": [
        {
            "Description": "rhcos-4.7.0-x86_64-aws.x86_64",
            "ImportTaskId": "import-snap-fh6i8uil",
            "SnapshotTaskDetail": {
                "Description": "rhcos-4.7.0-x86_64-aws.x86_64",
                "DiskImageSize": 819056640.0,
                "Format": "VMDK",
                "SnapshotId": "snap-06331325870076318",
                "Status": "completed",
                "UserBucket": {
                    "S3Bucket": "external-images",
                    "S3Key": "rhcos-4.7.0-x86_64-aws.x86_64.vmdk"
                }
            }
        }
    ]
}
```

复制 **SnapshotId** 以注册镜像。

8. 从 RHCOS 快照创建自定义 RHCOS AMI:

```
$ aws ec2 register-image \
  --region ${AWS_DEFAULT_REGION} \
  --architecture x86_64 \ 1
  --description "rhcos-${RHCOS_VERSION}-x86_64-aws.x86_64" \ 2
  --ena-support \
  --name "rhcos-${RHCOS_VERSION}-x86_64-aws.x86_64" \ 3
  --virtualization-type hvm \
  --root-device-name '/dev/xvda' \
  --block-device-mappings 'DeviceName=/dev/xvda,Ebs=
{DeleteOnTermination=true,SnapshotId=<snapshot_ID>}' 4
```

1. RHCOS VMDK 架构类型，如 **x86_64**, **aarch64**, **s390x**, 或 **ppc64le**。
2. 来自导入快照的 **Description**。
3. RHCOS AMI 的名称。
4. 导入的快照中的 **SnapshotID**。

如需了解更多有关这些 API 的信息，请参阅 AWS 文档 导入快照 和 创建由 EBS 支持的 AMI。

6.3.10.6. 手动创建安装配置文件

安装集群要求您手动创建安装配置文件。

先决条件

- 您上传了一个自定义 RHCOS AMI。
- 您的本地机器上有一个 SSH 公钥供安装程序使用。该密钥将用于在集群节点上进行 SSH 身份验证，以进行调试和灾难恢复。
- 已获取 OpenShift Container Platform 安装程序和集群的 pull secret。

流程

1. 创建一个安装目录来存储所需的安装资产：

```
$ mkdir <installation_directory>
```

重要

您必须创建一个目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

2. 自定义提供的 **install-config.yaml** 文件模板示例，并将其保存在 **<installation_directory>** 中。

注意

此配置文件必须命名为 **install-config.yaml**。

3. 备份 **install-config.yaml** 文件，以便您可以使用它安装多个集群。

重要

install-config.yaml 文件会在安装过程的下一步中使用。现在必须备份它。

其他资源

- AWS 的安装配置参数

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

您可以自定义安装配置文件 (**install-config.yaml**)，以指定有关 OpenShift Container Platform 集群平台的更多详细信息，或修改所需参数的值。

重要

此示例 YAML 文件仅供参考。使用它作为资源，在您手动创建的安装配置文件中输入参数值。

```
apiVersion: v1
baseDomain: example.com (1)
```

```
credentialsMode: Mint (2)
controlPlane: (3) (4)
  hyperthreading: Enabled (5)
  name: master
  platform:
    aws:
      zones:
      - cn-north-1a
      - cn-north-1b
      rootVolume:
        iops: 4000
        size: 500
        type: io1 (6)
      metadataService:
        authentication: Optional (7)
      type: m6i.xlarge
  replicas: 3
compute: (8)
- hyperthreading: Enabled (9)
  name: worker
  platform:
    aws:
      rootVolume:
        iops: 2000
        size: 500
        type: io1 (10)
      metadataService:
        authentication: Optional (11)
      type: c5.4xlarge
      zones:
      - cn-north-1a
  replicas: 3
metadata:
  name: test-cluster (12)
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OVNKubernetes (13)
  serviceNetwork:
  - 172.30.0.0/16
platform:
  aws:
    region: cn-north-1 (14)
    propagateUserTags: true (15)
    userTags:
      adminContact: jdoe
      costCenter: 7536
    subnets: (16)
    - subnet-1
    - subnet-2
    - subnet-3
```

```
    amiID: ami-96c6f8f7 17 18
    serviceEndpoints: 19
      - name: ec2
        url: https://vpce-id.ec2.cn-north-1.vpce.amazonaws.com.cn
    hostedZone: Z3URY6TWQ91KVV 20
fips: false 21
sshKey: ssh-ed25519 AAAA... 22
publish: Internal 23
pullSecret: '{"auths": ...}' 24
```

1 12 14 17 24 必需。

2 可选：添加此参数来强制 Cloud Credential Operator (CCO) 使用指定的模式。默认情况下，CCO 使用 **kube-system** 命名空间中的 root 凭证来动态尝试决定凭证的功能。有关 CCO 模式的详情，请参阅*身份验证和授权*指南中的"About the Cloud Credential Operator"部分。

3 8 15 如果没有提供这些参数和值，安装程序会提供默认值。

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

5 9 是否要启用或禁用并发多线程或 **超线程**。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将 参数值设置为 **Disabled** 来禁用它。如果在某些集群机器中禁用并发多线程，则必须在所有集群机器中禁用它。

重要

如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。如果您对机器禁用并发多线程，请使用较大的实例类型，如 **m4.2xlarge** 或 **m5.2xlarge**。

6 10 要为 etcd 配置更快的存储，特别是对于较大的集群，请将存储类型设置为 **io1**，并将 **iops** 设为 **2000**。

7 11 是否需要 Amazon EC2 实例元数据服务 v2 (IMDSv2)。为了要求 IMDSv2，请将参数值设置为 **Required**。要允许使用 IMDSv1 和 IMDSv2，请将参数值设置为 **Optional**。如果没有指定值，则允许 IMDSv1 和 IMDSv2。

注意

在集群安装过程中设置的 control plane 机器的 IMDS 配置只能使用 AWS CLI 更改。可以使用计算机器集来更改计算机器的 IMDS 配置。

13 要安装的集群网络插件。默认值 **OVNKubernetes** 是唯一支持的值。

16 如果您提供自己的 VPC，为集群使用的每个可用区指定子网。

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

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

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

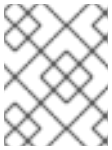
21 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 Red Hat Enterprise Linux (RHEL) 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅在 FIPS 模式中安装该系统。

当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS)时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。

22 您可以选择提供您用来访问集群中机器的 **sshKey** 值。



注意

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

23 如何发布集群的面向用户的端点。将 **publish** 设置为 **Internal** 以部署一个私有集群，它不能被互联网访问。默认值为 **External**。

6.3.10.6.2. 集群安装的最低资源要求

每台集群机器都必须满足以下最低要求：

表 6.18. 最低资源要求

机器	操作系统	vCPU [1]	虚拟内存	Storage	每秒输入/输出 (IOPS) [2]
bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane（控制平面）	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS、RHEL 8.6 及更新版本 [3]	2	8 GB	100 GB	300

1. 当未启用并发多线程(SMT)或超线程时，一个 vCPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比例：（每个内核数的线程）× sockets = vCPU。

2. OpenShift Container Platform 和 Kubernetes 对磁盘性能非常敏感，建议使用更快的存储速度，特别是 control plane 节点上需要 10 ms p99 fsync 持续时间的 etcd。请注意，在许多云平台上，存储大小和 IOPS 可一起扩展，因此您可能需要过度分配存储卷来获取足够的性能。

3. 与所有用户置备的安装一样，如果您选择在集群中使用 RHEL 计算机器，则负责所有操作系统生命周期管理和维护，包括执行系统更新、应用补丁和完成所有其他必要的任务。RHEL 7 计算机器的使用已弃用，并已在 OpenShift Container Platform 4.10 及更新的版本中删除。

注意

从 OpenShift Container Platform 版本 4.13 开始，RHCOS 基于 RHEL 版本 9.2，它更新了微架构要求。以下列表包含每个架构需要的最小指令集架构 (ISA)：

- x86-64 体系结构需要 x86-64-v2 ISA
- ARM64 架构需要 ARMv8.0-A ISA
- IBM Power 架构需要 Power 9 ISA
- s390x 架构需要 z14 ISA

如需更多信息，请参阅 RHEL 架构。

如果平台的实例类型满足集群机器的最低要求，则 OpenShift Container Platform 支持使用它。

其他资源

- 优化存储

6.3.10.6.3. 为 AWS 测试的实例类型

以下 Amazon Web Services(AWS) 实例类型已经过 OpenShift Container Platform 测试。

注意

将以下图中包含的机器类型用于 AWS 实例。如果您使用没有在图表中列出的实例类型，请确保使用的实例大小与名为"最小资源要求"的部分中列出的最少资源要求匹配。

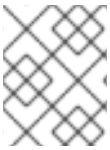
例 6.44. 基于 64 位 x86 架构的机器类型

- **c4.***
- **c5.***
- **c5a.***
- **i3.***
- **m4.***
- **m5.***
- **m5a.***
- **m6a.***
- **m6i.***
- **r4.***

- **r5.***
- **r5a.***
- **r6i.***
- **t3.***
- **t3a.***

6.3.10.6.4. 在 64 位 ARM 基础架构上为 AWS 测试过的实例类型

OpenShift Container Platform 中已经测试了以下 Amazon Web Services (AWS) 64 位 ARM 实例类型。



注意

使用 AWS ARM 实例的以下图中包含的机器类型。如果您使用没有在图中列出的实例类型，请确保使用的实例大小与集群安装"最小资源要求"中列出的最少资源要求匹配。

例 6.45. 基于 64 位 ARM 架构的机器类型

- **c6g.***
- **m6g.***

6.3.10.6.5. 在安装过程中配置集群范围代理

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

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 **Proxy** 对象的 **spec.noProxy** 字段中添加站点来绕过代理。

注意

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

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

流程

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

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: ec2.<aws_region>.amazonaws.com,elasticloadbalancing.
<aws_region>.amazonaws.com,s3.<aws_region>.amazonaws.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 **.** 以仅匹配子域。例如，**.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代理。如果您已将 Amazon **EC2**、**Elastic Load Balancing** 和 **S3** VPC 端点添加到 VPC 中，您必须将这些端点添加到 **noProxy** 字段。
4. 如果提供，安装程序会在 **openshift-config** 命名空间中生成名为 **user-ca-bundle** 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 **trusted-ca-bundle** 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， **Proxy** 对象的 **trustedCA** 字段中也会引用此配置映射。**additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。
5. 可选：决定 **Proxy** 对象的配置以引用 **trustedCA** 字段中 **user-ca-bundle** 配置映射的策略。允许的值是 **Proxyonly** 和 **Always**。仅在配置了 **http/https** 代理时，使用 **Proxyonly** 引用 **user-ca-bundle** 配置映射。使用 **Always** 始终引用 **user-ca-bundle** 配置映射。默认值为 **Proxyonly**。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。

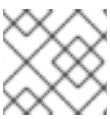
注意

如果安装程序超时，重启并使用安装程序的 **wait-for** 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

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

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



注意

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

6.3.10.6.6. 将现有 AWS 安全组应用到集群

将现有 AWS 安全组应用到 control plane 和计算机器可帮助您满足机构的安全需求，在这种情况下，您需要控制这些机器的传入或传出流量。

先决条件

- 您已在 AWS 中创建安全组。如需更多信息，请参阅使用 安全组的 AWS 文档。
- 安全组必须与您要将集群部署到的现有 VPC 关联。安全组不能与另一个 VPC 关联。
- 您有一个现有的 **install-config.yaml** 文件。

流程

1. 在 **install-config.yaml** 文件中，编辑 **compute.platform.aws.additionalSecurityGroupIDs** 参数，为您的计算机器指定一个或多个自定义安全组。
2. 编辑 **controlPlane.platform.aws.additionalSecurityGroupIDs** 参数，为您的 control plane 机器指定一个或多个自定义安全组。
3. 保存文件并在部署集群时引用。

指定自定义安全组的 install-config.yaml 文件示例

```
# ...
compute:
- hyperthreading: Enabled
  name: worker
  platform:
    aws:
      additionalSecurityGroupIDs:
        - sg-1 (1)
        - sg-2
  replicas: 3
controlPlane:
  hyperthreading: Enabled
  name: master
  platform:
    aws:
      additionalSecurityGroupIDs:
        - sg-3
        - sg-4
  replicas: 3
platform:
  aws:
    region: us-east-1
    subnets: (2)
      - subnet-1
      - subnet-2
      - subnet-3
```

(1) 在 Amazon EC2 控制台中显示时指定安全组的名称，包括 **sg** 前缀。

(2) 指定集群使用的每个可用区的子网。

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

默认情况下，管理员 secret 存储在 **kube-system** 项目中。如果您在 **install-config.yaml** 文件中将 **credentialsMode** 参数配置为 **Manual**，则必须使用以下替代方案之一：

- 要手动管理长期云凭证，请按照手动创建长期凭证中的步骤操作。
- 要实现在集群外为各个组件管理的短期凭证，请按照配置 A WS 集群以使用短期凭证 中的步骤操作。

6.3.10.7.1. 手动创建长期凭证

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

流程

1. 如果您没有将 **install-config.yaml** 配置文件中的 **credentialsMode** 参数设置为 **Manual**，请修改值，如下所示：

   配置文件片段示例

   ```
   apiVersion: v1
   baseDomain: example.com
   credentialsMode: Manual
   # ...
   ```

2. 如果您之前还没有创建安装清单文件，请运行以下命令：

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

   其中 **<installation_directory>** 是安装程序在其中创建文件的目录。

3. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

   ```
   $ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
   ```

4. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 自定义资源 (CR) 列表：

   ```
   $ oc adm release extract \
     --from=$RELEASE_IMAGE \
     --credentials-requests \
     --included \ 1
     --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
     --to=<path_to_directory_for_credentials_requests> 3
   ```

   1. **--included** 参数仅包含特定集群配置所需的清单。
   2. 指定 **install-config.yaml** 文件的位置。
   3. 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

此命令为每个 **CredentialsRequest** 对象创建一个 YAML 文件。

CredentialsRequest 对象示例

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

5. 在之前生成的 **openshift-install** 清单目录中为 secret 创建 YAML 文件。secret 必须使用在 **spec.secretRef** 中为每个 **CredentialsRequest** 定义的命名空间和 secret 名称存储。

带有 secret 的 CredentialsRequest 对象示例

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component_credentials_request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: AWSProviderSpec
    statementEntries:
    - effect: Allow
      action:
      - s3:CreateBucket
      - s3:DeleteBucket
      resource: "*"
      ...
  secretRef:
    name: <component_secret>
    namespace: <component_namespace>
  ...
```

Secret 对象示例

```
apiVersion: v1
kind: Secret
```

```
metadata:
  name: <component_secret>
  namespace: <component_namespace>
data:
  aws_access_key_id: <base64_encoded_aws_access_key_id>
  aws_secret_access_key: <base64_encoded_aws_secret_access_key>
```

重要

在升级使用手动维护凭证的集群前，您必须确保 CCO 处于可升级状态。

6.3.10.7.2. 将 AWS 集群配置为使用短期凭证

要安装配置为使用 AWS 安全令牌服务 (STS) 的集群，您必须配置 CCO 实用程序并为集群创建所需的 AWS 资源。

6.3.10.7.2.1. 配置 Cloud Credential Operator 工具

当 Cloud Credential Operator(CCO)以手动模式运行时，要从集群外部创建和管理云凭证，提取并准备 CCO 实用程序(**ccoctl**)二进制文件。

注意

ccoctl 工具是在 Linux 环境中运行的 Linux 二进制文件。

先决条件

- 您可以访问具有集群管理员权限的 OpenShift Container Platform 帐户。
- 已安装 OpenShift CLI(**oc**)。
- 您已为 **ccoctl** 工具创建了用于以下权限的 AWS 帐户：

  例 6.46. 所需的 AWS 权限

  所需的 iam 权限

  - **iam:CreateOpenIDConnectProvider**
  - **iam:CreateRole**
  - **iam:DeleteOpenIDConnectProvider**
  - **iam:DeleteRole**
  - **iam:DeleteRolePolicy**
  - **iam:GetOpenIDConnectProvider**
  - **iam:GetRole**
  - **iam:GetUser**
  - **iam:ListOpenIDConnectProviders**
  - **iam:ListRolePolicies**

- **iam:ListRoles**
- **iam:PutRolePolicy**
- **iam:TagOpenIDConnectProvider**
- **iam:TagRole**

所需的 s3 权限

- **s3:CreateBucket**
- **s3:DeleteBucket**
- **s3:DeleteObject**
- **s3:GetBucketAcl**
- **s3:GetBucketTagging**
- **s3:GetObject**
- **s3:GetObjectAcl**
- **s3:GetObjectTagging**
- **s3:ListBucket**
- **s3:PutBucketAcl**
- **s3:PutBucketPolicy**
- **s3:PutBucketPublicAccessBlock**
- **s3:PutBucketTagging**
- **s3:PutObject**
- **s3:PutObjectAcl**
- **s3:PutObjectTagging**

所需的 cloudfront 权限

- **cloudfront:ListCloudFrontOriginAccessIdentities**
- **cloudfront:ListDistributions**
- **cloudfront:ListTagsForResource**

如果您计划通过公共 CloudFront 发行版 URL 将 OIDC 配置存储在 IAM 身份提供程序访问的私有 S3 存储桶中，则运行 **ccoctl** 工具的 AWS 帐户需要以下额外权限：

例 6.47. 使用 CloudFront 私有 S3 存储桶的额外权限

- **cloudfront:CreateCloudFrontOriginAccessIdentity**

- **cloudfront:CreateDistribution**
- **cloudfront:DeleteCloudFrontOriginAccessIdentity**
- **cloudfront:DeleteDistribution**
- **cloudfront:GetCloudFrontOriginAccessIdentity**
- **cloudfront:GetCloudFrontOriginAccessIdentityConfig**
- **cloudfront:GetDistribution**
- **cloudfront:TagResource**
- **cloudfront:UpdateDistribution**

注意

在使用 **ccoctl aws create-all** 命令处理凭证请求时，这些额外权限支持使用 **--create-private-s3-bucket** 选项。

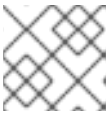
流程

1. 运行以下命令，为 OpenShift Container Platform 发行镜像设置变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. 运行以下命令，从 OpenShift Container Platform 发行镜像获取 CCO 容器镜像：

```
$ CCO_IMAGE=$(oc adm release info --image-for='cloud-credential-operator'
$RELEASE_IMAGE -a ~/.pull-secret)
```



注意

确保 **$RELEASE_IMAGE** 的架构与将使用 **ccoctl** 工具的环境架构相匹配。

3. 运行以下命令，将 CCO 容器镜像中的 **ccoctl** 二进制文件提取到 OpenShift Container Platform 发行镜像中：

```
$ oc image extract $CCO_IMAGE \
  --file="/usr/bin/ccoctl.<rhel_version>" \ 1
  -a ~/.pull-secret
```

1 对于 **<rhel_version>**，请指定与主机使用的 Red Hat Enterprise Linux (RHEL) 版本对应的值。如果没有指定值，则默认使用 **ccoctl.rhel8**。以下值有效：

- **rhel8**: 为使用 RHEL 8 的主机指定这个值。
- **rhel9** ：为使用 RHEL 9 的主机指定这个值。

4. 运行以下命令更改权限以使 **ccoctl** 可执行：

-

```
$ chmod 775 ccoctl.<rhel_version>
```

验证

- 要验证 **ccoctl** 是否可使用，请运行以下命令来显示帮助文件：

```
$ ccoctl --help
```

ccoctl --help 的输出

```
OpenShift credentials provisioning tool

Usage:
  ccoctl [command]

Available Commands:
  aws         Manage credentials objects for AWS cloud
  azure       Manage credentials objects for Azure
  gcp         Manage credentials objects for Google cloud
  help        Help about any command
  ibmcloud    Manage credentials objects for IBM Cloud
  nutanix     Manage credentials objects for Nutanix

Flags:
  -h, --help   help for ccoctl

Use "ccoctl [command] --help" for more information about a command.
```

6.3.10.7.2.2. 使用 Cloud Credential Operator 实用程序创建 AWS 资源

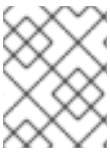
创建 AWS 资源时有以下选项：

- 您可以使用 **ccoctl aws create-all** 命令自动创建 AWS 资源。这是创建资源的最快速方法。请参阅 使用单个命令创建 AWS 资源。
- 如果您需要在修改 AWS 资源前查看 **ccoctl** 工具创建的 JSON 文件，或者 **ccoctl** 工具用于创建 AWS 资源的过程无法自动满足组织的要求，您可以单独创建 AWS 资源。请参阅 单独创建 AWS 资源。

6.3.10.7.2.2.1. 使用单个命令创建 AWS 资源

如果 **ccoctl** 工具用于创建 AWS 资源的过程自动满足机构的要求，您可以使用 **ccoctl aws create-all** 命令自动创建 AWS 资源。

否则，您可以单独创建 AWS 资源。如需更多信息，请参阅"单独创建 AWS 资源"。



注意

默认情况下，**ccoctl** 在运行命令的目录中创建对象。要在其他目录中创建对象，请使用 **--output-dir** 标志。此流程使用 **<path_to_ccoctl_output_dir>** 来引用这个目录。

先决条件

您必须：

- 提取并准备好 **ccoctl** 二进制文件。

流程

1. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 对象列表：

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
  --included \ 1
  --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
  --to=<path_to_directory_for_credentials_requests> 3
```

1. **--included** 参数仅包含特定集群配置所需的清单。
2. 指定 **install-config.yaml** 文件的位置。
3. 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

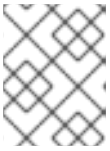
注意

此命令可能需要一些时间才能运行。

3. 运行以下命令，使用 **ccoctl** 工具处理所有 **CredentialsRequest** 对象：

```
$ ccoctl aws create-all \
  --name=<name> \ 1
  --region=<aws_region> \ 2
  --credentials-requests-dir=<path_to_credentials_requests_directory> \ 3
  --output-dir=<path_to_ccoctl_output_dir> \ 4
  --create-private-s3-bucket 5
```

1. 指定用于标记创建用于跟踪的任何云资源的名称。
2. 指定在其中创建云资源的 AWS 区域。
3. 指定包含组件 **CredentialsRequest** 对象文件的目录。
4. 可选：指定您希望 **ccoctl** 实用程序在其中创建对象的目录。默认情况下，实用程序在运行命令的目录中创建对象。
5. 可选：默认情况下，**ccoctl** 实用程序将 OpenID Connect (OIDC) 配置文件存储在公共 S3 存储桶中，并使用 S3 URL 作为公共 OIDC 端点。要将 OIDC 配置存储在 IAM 身份提供程序通过公共 CloudFront 发行版 URL 访问的专用 S3 存储桶中，请使用 **--create-private-s3-bucket** 参数。



注意

如果您的集群使用 **TechPreviewNoUpgrade** 功能集启用的技术预览功能，则必须包含 **--enable-tech-preview** 参数。

验证

- 要验证 OpenShift Container Platform secret 是否已创建，列出 **<path_to_ccoctl_output_dir>/manifests** 目录中的文件：

```
$ ls <path_to_ccoctl_output_dir>/manifests
```

输出示例

```
cluster-authentication-02-config.yaml
openshift-cloud-credential-operator-cloud-credential-operator-iam-ro-creds-credentials.yaml
openshift-cloud-network-config-controller-cloud-credentials-credentials.yaml
openshift-cluster-api-capa-manager-bootstrap-credentials-credentials.yaml
openshift-cluster-csi-drivers-ebs-cloud-credentials-credentials.yaml
openshift-image-registry-installer-cloud-credentials-credentials.yaml
openshift-ingress-operator-cloud-credentials-credentials.yaml
openshift-machine-api-aws-cloud-credentials-credentials.yaml
```

您可以通过查询 AWS 来验证是否已创建 IAM 角色。如需更多信息，请参阅有关列出 IAM 角色的 AWS 文档。

6.3.10.7.2.2.2. 单独创建 AWS 资源

您可以使用 **ccoctl** 工具单独创建 AWS 资源。这个选项对于在不同用户或部门之间创建这些资源的组织可能很有用。

否则，您可以使用 **ccoctl aws create-all** 命令自动创建 AWS 资源。如需更多信息，请参阅"使用单个命令创建 AWS 资源"。

注意

默认情况下，**ccoctl** 在运行命令的目录中创建对象。要在其他目录中创建对象，请使用 **--output-dir** 标志。此流程使用 **<path_to_ccoctl_output_dir>** 来引用这个目录。

有些 **ccoctl** 命令会发出 AWS API 调用来创建或修改 AWS 资源。您可以使用 **--dry-run** 标志来避免 API 调用。使用此标志可在本地文件系统中创建 JSON 文件。您可以使用 **--cli-input-json** 参数查看和修改 JSON 文件，然后使用 AWS CLI 工具应用它们。

先决条件

- 提取并准备 **ccoctl** 二进制文件。

流程

1. 运行以下命令，生成用于为集群设置 OpenID Connect 供应商的公共和私有 RSA 密钥文件：

```
$ ccoctl aws create-key-pair
```

输出示例

```
2021/04/13 11:01:02 Generating RSA keypair
2021/04/13 11:01:03 Writing private key to /<path_to_ccoctl_output_dir>/serviceaccount-signer.private
2021/04/13 11:01:03 Writing public key to /<path_to_ccoctl_output_dir>/serviceaccount-signer.public
2021/04/13 11:01:03 Copying signing key for use by installer
```

其中 **serviceaccount-signer.private** 和 **serviceaccount-signer.public** 是生成的密钥文件。

此命令还会在 **/<path_to_ccoctl_output_dir>/tls/bound-service-account-signing-key.key** 中创建集群在安装过程中所需的私钥。

2. 运行以下命令，在 AWS 上创建 OpenID Connect 身份提供程序和 S3 存储桶：

```
$ ccoctl aws create-identity-provider \
  --name=<name> \1
  --region=<aws_region> \2
  --public-key-file=<path_to_ccoctl_output_dir>/serviceaccount-signer.public 3
```

1. **<name>** 是用于标记为跟踪而创建的云资源的名称。
2. **<aws-region>** 是将要在其中创建云资源的 AWS 区域。
3. **<path_to_ccoctl_output_dir>** 是 **ccoctl aws create-key-pair** 命令生成的公钥文件的路径。

输出示例

```
2021/04/13 11:16:09 Bucket <name>-oidc created
2021/04/13 11:16:10 OpenID Connect discovery document in the S3 bucket <name>-oidc at .well-known/openid-configuration updated
2021/04/13 11:16:10 Reading public key
2021/04/13 11:16:10 JSON web key set (JWKS) in the S3 bucket <name>-oidc at keys.json updated
2021/04/13 11:16:18 Identity Provider created with ARN: arn:aws:iam::<aws_account_id>:oidc-provider/<name>-oidc.s3.<aws_region>.amazonaws.com
```

其中 **openid-configuration** 是发现文档和 **key.json** 是一个 JSON Web 密钥集文件。

此命令还会在 **/<path_to_ccoctl_output_dir>/manifests/cluster-authentication-02-config.yaml** 中创建 YAML 配置文件。此文件为集群生成的服务帐户令牌设置签发者 URL 字段，以便 AWS IAM 身份提供程序信任令牌。

3. 为集群中的每个组件创建 IAM 角色：

   a. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

   ```
   $ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
   ```

   b. 从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 对象列表：

   ```
   $ oc adm release extract \
     --from=$RELEASE_IMAGE \
   ```

```
  --credentials-requests \
  --included \ 1
  --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \
2
  --to=<path_to_directory_for_credentials_requests> 3
```

1. **--included** 参数仅包含特定集群配置所需的清单。
2. 指定 **install-config.yaml** 文件的位置。
3. 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

c. 运行以下命令，使用 **ccoctl** 工具处理所有 **CredentialsRequest** 对象：

```
$ ccoctl aws create-iam-roles \
  --name=<name> \
  --region=<aws_region> \
  --credentials-requests-dir=<path_to_credentials_requests_directory> \
  --identity-provider-arn=arn:aws:iam::<aws_account_id>:oidc-provider/<name>-oidc.s3.
<aws_region>.amazonaws.com
```

注意

对于使用其他 IAM API 端点的 AWS 环境（如 GovCloud），还必须使用 **--region** 参数指定您的区域。

如果您的集群使用 **TechPreviewNoUpgrade** 功能集启用的技术预览功能，则必须包含 **--enable-tech-preview** 参数。

对于每个 **CredentialsRequest** 对象，**ccoctl** 创建一个带有信任策略的 IAM 角色，该角色与指定的 OIDC 身份提供程序相关联，以及来自 OpenShift Container Platform 发行镜像的每个 **CredentialsRequest** 对象中定义的权限策略。

验证

- 要验证 OpenShift Container Platform secret 是否已创建，列出 **<path_to_ccoctl_output_dir>/manifests** 目录中的文件：

```
$ ls <path_to_ccoctl_output_dir>/manifests
```

输出示例

```
cluster-authentication-02-config.yaml
openshift-cloud-credential-operator-cloud-credential-operator-iam-ro-creds-credentials.yaml
openshift-cloud-network-config-controller-cloud-credentials-credentials.yaml
openshift-cluster-api-capa-manager-bootstrap-credentials-credentials.yaml
openshift-cluster-csi-drivers-ebs-cloud-credentials-credentials.yaml
openshift-image-registry-installer-cloud-credentials-credentials.yaml
openshift-ingress-operator-cloud-credentials-credentials.yaml
openshift-machine-api-aws-cloud-credentials-credentials.yaml
```

您可以通过查询 AWS 来验证是否已创建 IAM 角色。如需更多信息，请参阅有关列出 IAM 角色的 AWS 文档。

6.3.10.7.2.3. 整合 Cloud Credential Operator 实用程序清单

要为单个组件在集群外实现短期安全凭证，您必须将创建 Cloud Credential Operator 实用程序 (**ccoctl**) 的清单文件移到安装程序的正确目录中。

先决条件

- 您已使用托管集群的云平台配置了帐户。
- 您已配置了 Cloud Credential Operator 实用程序 (**ccoctl**)。
- 已使用 **ccoctl** 工具创建了集群所需的云供应商资源。

流程

1. 如果您没有将 **install-config.yaml** 配置文件中的 **credentialsMode** 参数设置为 **Manual**，请修改值，如下所示：

   配置文件片段示例

   ```
   apiVersion: v1
   baseDomain: example.com
   credentialsMode: Manual
   # ...
   ```

2. 如果您之前还没有创建安装清单文件，请运行以下命令：

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

   其中 **<installation_directory>** 是安装程序在其中创建文件的目录。

3. 运行以下命令，将 **ccoctl** 工具生成的清单复制到安装程序创建的 **manifests** 目录中：

   ```
   $ cp /<path_to_ccoctl_output_dir>/manifests/* ./manifests/
   ```

4. 运行以下命令，将 **ccoctl** 工具在 **tls** 目录中生成的私钥复制到安装目录中：

   ```
   $ cp -a /<path_to_ccoctl_output_dir>/tls .
   ```

6.3.10.8. 部署集群

您可以在兼容云平台上安装 OpenShift Container Platform。

重要

在初始安装过程中，您只能运行安装程序的 **create cluster** 命令一次。

先决条件

- 您已使用托管集群的云平台配置了帐户。

- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。
- 已确认主机上的云供应商帐户具有部署集群的正确权限。权限不正确的帐户会导致安装过程失败，并显示包括缺失权限的错误消息。

流程

1. 进入包含安装程序的目录并初始化集群部署：

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

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

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

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

注意

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

验证

当集群部署成功完成时：

- 终端会显示用于访问集群的说明，包括指向 Web 控制台和 **kubeadmin** 用户的凭证的链接。
- 凭证信息还会输出到 **<installation_directory>/.openshift_install.log**.

重要

不要删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

输出示例

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

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

6.3.10.9. 使用 CLI 登录集群

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

先决条件

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

流程

1. 导出 **kubeadmin** 凭证：

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

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

2. 验证您可以使用导出的配置成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

6.3.10.10. 使用 Web 控制台登录到集群

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

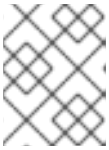
先决条件

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

流程

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

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



注意

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

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

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

注意

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

输出示例

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

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

其他资源

- 如需有关 访问和了解 OpenShift Container Platform Web 控制台的更多详情，请参阅 访问 Web 控制台。

6.3.10.11. 后续步骤

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

6.3.11. 在 AWS Local Zones 上使用计算节点安装集群

您可以通过在 **install-config.yaml** 文件的边缘计算池中设置区域名称，或使用 Local Zone 子网在现有 Amazon Virtual Private Cloud (VPC)上安装集群，在 Amazon Web Services (AWS) Local Zones 上快速安装 OpenShift Container Platform 集群。

AWS Local Zones (AWS Local Zones) 是一个基础架构，它放置了接近满足的云资源的区域。如需更多信息，请参阅 AWS 区域文档。

6.3.11.1. 基础架构先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新流程 的详细信息。

- 熟悉 选择集群安装方法并为用户准备它。
- 已将 AWS 帐户配置为托管集群。

警告

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

- 您下载了 AWS CLI 并安装到您的计算机上。请参阅 AWS 文档中的使用捆绑安装程序（Linux、macOS 或 UNIX）安装 AWS CLI。
- 如果使用防火墙，则会 将其配置为允许集群需要访问的站点。
- 您记下了区域和支持的 AWS 区域位置，在其中创建网络资源。
- 您可以参阅 AWS 文档中的 AWS Local Zones 功能。
- 您已将支持 AWS Local Zones 的网络资源添加到 Identity and Access Management (IAM) 用户或角色的权限。以下示例启用了一个 zone group，它为创建支持 AWS Local Zones 的网络资源提供用户或团队访问权限。

附加到 IAM 用户或角色的 ec2:ModifyAvailabilityZoneGroup 权限的额外 IAM 策略示例。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "ec2:ModifyAvailabilityZoneGroup"
      ],
      "Effect": "Allow",
      "Resource": "*"
    }
  ]
}
```

6.3.11.2. 关于 AWS 本地区域和边缘计算池

阅读以下部分以了解 AWS Local Zones 环境中的基础架构行为和集群限制。

6.3.11.2.1. AWS 本地区中的集群限制

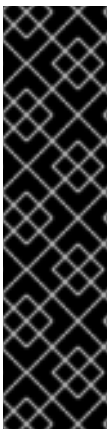
当您试图在 Amazon Web Services (AWS) Local Zone 中使用默认安装配置部署集群时，有一些限制。

重要

以下列表在预先配置的 AWS 区域中部署集群时的详情限制：

- 区域中的 Amazon EC2 实例和 Region 中的 Amazon EC2 实例之间的最大传输单元 (MTU) 为 **1300**。这会导致集群范围的网络 MTU 根据与部署一起使用的网络插件而改变。
- 不支持 Network Load Balancer (NLB)、Classic Load Balancer 和网络地址转换 (NAT)网关等网络资源。
- 对于 AWS 上的 OpenShift Container Platform 集群，AWS Elastic Block Storage (EBS) **gp3** 类型卷是节点卷和存储类的默认设置。这个卷类型在区域位置没有全局可用。默认情况下，在区中运行的节点使用 **gp2** EBS 卷进行部署。在区节点上创建工作负载时，必须设置 **gp2-csi StorageClass** 参数。

如果您希望安装程序为 OpenShift Container Platform 集群自动创建 Local Zone 子网，则使用此方法会有针对特定配置的限制。



重要

当您将安装程序设置为自动为 OpenShift Container Platform 集群创建子网时，会有以下配置限制：

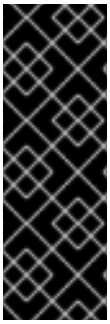
- 当安装程序在 AWS Local Zones 中创建专用子网时，安装程序会将每个子网与其父区的路由表相关联。此操作通过 AWS 区域中的 NAT 网关的方式确保每个专用子网都可以将出口流量路由到互联网。
- 如果集群安装过程中不存在 parent-zone 路由表，安装程序会将任何专用子网与 Amazon Virtual Private Cloud (VPC)中的第一个可用私有路由表相关联。此方法仅对 OpenShift Container Platform 集群中的 AWS Local Zones 子网有效。

6.3.11.2.2. 关于边缘计算池

边缘计算节点是在 AWS Local Zones 位置中运行的污点计算节点。

在部署使用 Local Zones 的集群时，请考虑以下点：

- 本地区域中的 Amazon EC2 实例比可用区中的 Amazon EC2 实例的成本更高。
- 在 AWS Local Zones 和最终用户中运行的应用程序间延迟较低。例如，在 Local Zones 和 Availability Zones 混合了入口流量时，一些工作负载会有一个延迟影响。



重要

通常，本地区域中的 Amazon EC2 实例和 Region 中的 Amazon EC2 实例之间的最大传输单元 (MTU) 为 1300。对于开销，集群网络 MTU 必须总是小于 EC2 MTU。具体开销由网络插件决定。例如： OVN-Kubernetes 的开销为 **100 字节**。

网络插件可以提供额外的功能，如 IPsec，它们也会影响 MTU 大小。

如需更多信息，请参阅 AWS 文档中的 Local Zones 如何工作。

OpenShift Container Platform 4.12 引入了一个新的计算池 *edge*，用于在远程区中使用。边缘计算池配置在 AWS Local Zones 位置之间很常见。由于 Local Zones 资源上的 EC2 和 EBS 等资源的类型和大小限制，默认的实例类型可能与传统的计算池不同。

Local Zones 位置的默认 Elastic Block Store (EBS) 是 **gp2**，它与非边缘计算池不同。根据区域上的实例产品，边缘计算池中每个本地区域使用的实例类型可能与其他计算池不同。

边缘计算池创建新的标签，供开发人员用来将应用程序部署到 AWS Local Zones 节点上。新标签包括：

- **node-role.kubernetes.io/edge=''**
- **machine.openshift.io/zone-type=local-zone**
- **machine.openshift.io/zone-group=$ZONE_GROUP_NAME**

默认情况下，边缘计算池的机器集定义 **NoSchedule** 污点，以防止其他工作负载分散到 Local Zones 实例。只有用户在 pod 规格中定义容限时，用户才能运行用户工作负载。

其他资源

- MTU 值选择
- 更改集群网络的 MTU
- 了解污点和容限
- 存储类
- Ingress Controller 分片

6.3.11.3. 安装先决条件

在 AWS Local Zones 环境中安装集群前，您必须配置基础架构，以便它可以使用 Local Zone 功能。

6.3.11.3.1. 选择 AWS 本地区域

如果您计划在 AWS Local Zones 中创建子网，则必须单独选择每个 zone group。

先决条件

- 已安装 AWS CLI。
- 您已决定要部署 OpenShift Container Platform 集群的 AWS 区域。
- 您已将 permissive IAM 策略附加到选择 zone 组的用户或组帐户。

流程

1. 运行以下命令，列出 AWS 区域中可用的区域：

   在 AWS 区域中列出可用 AWS 区域的命令示例

   ```
   $ aws --region "<value_of_AWS_Region>" ec2 describe-availability-zones \
       --query 'AvailabilityZones[].[{ZoneName: ZoneName, GroupName: GroupName, Status:
   OptInStatus}]' \
   ```

```
    --filters Name=zone-type,Values=local-zone \
    --all-availability-zones
```

根据 AWS 区域，可用区列表可能比较长。该命令返回以下字段：

ZoneName

本地区域的名称。

GroupName

组成区域的组。要选择 Region，保存名称。

Status

Local Zones 组的状态。如果状态是 **not-opted-in**，则需要选择 **GroupName**，如下一步所述。

2. 运行以下命令，选择 AWS 帐户上的 zone 组：

```
$ aws ec2 modify-availability-zone-group \
    --group-name "<value_of_GroupName>" \ 1
    --opt-in-status opted-in
```

1 将 **<value_of_GroupName>** 替换为您要创建子网的 Local Zones 的组名称。例如，指定 **us-east-1-nyc-1** 以使用区域 **us-east-1-nyc-1a** (US East New York)。

6.3.11.3.2. 获取 AWS Marketplace 镜像

如果要使用 AWS Marketplace 镜像部署 OpenShift Container Platform 集群，您必须首先通过 AWS 订阅。订阅提供的提供的 AMI ID 可让您使用安装程序用来部署计算节点的 AMI ID。

先决条件

- 有 AWS 账户购买的产品。此帐户不必与用于安装集群的帐户相同。

流程

1. 从 AWS Marketplace 完成 OpenShift Container Platform 订阅。
2. 记录特定 AWS 区域的 AMI ID。作为安装过程的一部分，您必须在部署集群前使用这个值更新 **install-config.yaml** 文件。

使用 AWS Marketplace 计算节点的 install-config.yaml 文件示例

```
apiVersion: v1
baseDomain: example.com
compute:
- hyperthreading: Enabled
  name: worker
  platform:
    aws:
      amiID: ami-06c4d345f7c207239 1
      type: m5.4xlarge
  replicas: 3
metadata:
  name: test-cluster
```

```
platform:
  aws:
    region: us-east-2 2
sshKey: ssh-ed25519 AAAA...
pullSecret: '{"auths": ...}'
```

1 来自 AWS Marketplace 订阅的 AMI ID。

2 您的 AMI ID 与特定的 AWS 区域相关联。在创建安装配置文件时，请确保选择配置订阅时指定的相同 AWS 区域。

6.3.11.4. 准备安装

在将节点扩展到 Local Zones 之前，您必须为集群安装环境准备某些资源。

6.3.11.4.1. 集群安装的最低资源要求

每台集群机器都必须满足以下最低要求：

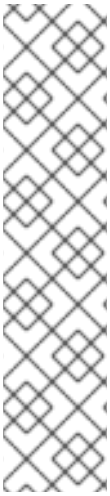
表 6.19. 最低资源要求

机器	操作系统	vCPU [1]	虚拟内存	Storage	每秒输入/输出 (IOPS) [2]
bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane（控制平面）	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS、RHEL 8.6 及更新版本 [3]	2	8 GB	100 GB	300

1. 当未启用并发多线程(SMT)或超线程时，一个 vCPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比例：（每个内核数的线程）× sockets = vCPU。

2. OpenShift Container Platform 和 Kubernetes 对磁盘性能非常敏感，建议使用更快的存储速度，特别是 control plane 节点上需要 10 ms p99 fsync 持续时间的 etcd。请注意，在许多云平台上，存储大小和 IOPS 可一起扩展，因此您可能需要过度分配存储卷来获取足够的性能。

3. 与所有用户置备的安装一样，如果您选择在集群中使用 RHEL 计算机器，则负责所有操作系统生命周期管理和维护，包括执行系统更新、应用补丁和完成所有其他必要的任务。RHEL 7 计算机器的使用已弃用，并已在 OpenShift Container Platform 4.10 及更新的版本中删除。



注意

从 OpenShift Container Platform 版本 4.13 开始，RHCOS 基于 RHEL 版本 9.2，它更新了微架构要求。以下列表包含每个架构需要的最小指令集架构 (ISA)：

- x86-64 体系结构需要 x86-64-v2 ISA
- ARM64 架构需要 ARMv8.0-A ISA
- IBM Power 架构需要 Power 9 ISA
- s390x 架构需要 z14 ISA

如需更多信息，请参阅 RHEL 架构。

如果平台的实例类型满足集群机器的最低要求，则 OpenShift Container Platform 支持使用它。

6.3.11.4.2. 为 AWS 测试的实例类型

以下 Amazon Web Services (AWS) 实例类型已使用 OpenShift Container Platform 测试，以用于 AWS Local Zones。

注意

将以下图中包含的机器类型用于 AWS 实例。如果您使用没有在图表中列出的实例类型，请确保使用的实例大小与名为"最小资源要求"的部分中列出的最少资源要求匹配。

例 6.48. 基于 AWS 本地区的 64 位 x86 架构的机器类型

- **c5.***
- **c5d.***
- **m6i.***
- **m5.***
- **r5.***
- **t3.***

其他资源

- 请参阅 AWS 文档中的 AWS Local Zones 功能。

6.3.11.4.3. 创建安装配置文件

生成并自定义安装程序部署集群所需的安装配置文件。

先决条件

- 已获取 OpenShift Container Platform 安装程序用于用户置备的基础架构和集群的 pull secret。

- 使用红帽发布的附带 Red Hat Enterprise Linux CoreOS (RHCOS) AMI 检查您是否将集群部署到 AWS 区域。如果要部署到需要自定义 AMI 的 AWS 区域，如 AWS GovCloud 区域，您必须手动创建 **install-config.yaml** 文件。

流程

1. 创建 **install-config.yaml** 文件。

   a. 进入包含安装程序的目录并运行以下命令：

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

   1 对于 **<installation_directory>**，请指定要存储安装程序创建的文件的目录名称。

   重要

   指定一个空目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

   b. 在提示符处，提供云的配置详情：

      i. 可选： 选择用于访问集群机器的 SSH 密钥。

      注意

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

      ii. 选择 **aws** 作为目标平台。

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

      注意

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

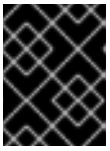
      iv. 选择要将集群部署到的 AWS Region。

      v. 选择您为集群配置的 Route 53 服务的基域。

      vi. 为集群输入描述性名称。

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

2. 可选：备份 **install-config.yaml** 文件。



重要

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

6.3.11.4.4. 使用边缘计算池安装配置文件示例

以下示例显示了包含边缘机器池配置的 **install-config.yaml** 文件。

使用自定义实例类型使用边缘池的配置

```
apiVersion: v1
baseDomain: devcluster.openshift.com
metadata:
  name: ipi-edgezone
compute:
- name: edge
  platform:
    aws:
      type: r5.2xlarge
platform:
  aws:
    region: us-west-2
pullSecret: '{"auths": ...}'
sshKey: ssh-ed25519 AAAA...
```

实例类型因位置而异。要验证集群运行的本地区中的可用性，请参阅 AWS 文档。

使用带有自定义 Amazon Elastic Block Store (EBS) 类型的边缘池的配置

```
apiVersion: v1
baseDomain: devcluster.openshift.com
metadata:
  name: ipi-edgezone
compute:
- name: edge
  platform:
    aws:
      zones:
      - us-west-2-lax-1a
      - us-west-2-lax-1b
      - us-west-2-phx-2a
      rootVolume:
        type: gp3
        size: 120
platform:
  aws:
    region: us-west-2
pullSecret: '{"auths": ...}'
sshKey: ssh-ed25519 AAAA...
```

Elastic Block Storage (EBS) 类型因位置而异。检查 AWS 文档，以验证集群运行的本地区域中的可用性。

使用自定义安全组使用边缘池的配置

```
apiVersion: v1
baseDomain: devcluster.openshift.com
metadata:
  name: ipi-edgezone
compute:
- name: edge
  platform:
    aws:
      additionalSecurityGroupIDs:
        - sg-1 (1)
        - sg-2
platform:
  aws:
    region: us-west-2
pullSecret: '{"auths": ...}'
sshKey: ssh-ed25519 AAAA...
```

(1) 指定安全组的名称，因为它在 Amazon EC2 控制台中显示。确保包含 **sg** 前缀。

6.3.11.4.5. 自定义集群网络 MTU

在 AWS 上部署集群前，您可以自定义集群网络的集群网络最大传输单元 (MTU) 来满足基础架构的需求。

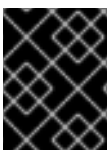
默认情况下，当使用支持的 Local Zones 功能安装集群时，集群网络的 MTU 值会自动调整为网络插件接受的最低值。

重要

为 Local Zones 基础架构中运行的 EC2 实例设置不受支持的 MTU 值可能会导致 OpenShift Container Platform 集群出现问题。

如果 Local Zone 在 Local Zone 和 AWS 区域中的 EC2 实例之间支持更高的 MTU 值，您可以手动配置较高的值来提高集群网络的网络性能。

您可以通过在 **install-config.yaml** 配置文件中指定 **networking.clusterNetworkMTU** 参数来自定义集群的 MTU。



重要

Local Zones 中的所有子网都必须支持更高的 MTU 值，以便该区中的每个节点都可以成功与 AWS 区域中的服务通信并部署您的工作负载。

覆盖默认 MTU 值的示例

```
apiVersion: v1
baseDomain: devcluster.openshift.com
metadata:
  name: edge-zone
networking:
  clusterNetworkMTU: 8901
compute:
- name: edge
```

```
  platform:
    aws:
      zones:
      - us-west-2-lax-1a
      - us-west-2-lax-1b
platform:
  aws:
    region: us-west-2
pullSecret: '{"auths": ...}'
sshKey: ssh-ed25519 AAAA...
```

其他资源

- 有关最大支持的最大传输单元(MTU)值的更多信息，请参阅 AWS 文档中的 Local Zones 支持的 AWS 资源。

6.3.11.5. AWS 本地区域的集群安装选项

选择以下安装选项之一，使用 Local Zones 中定义的边缘计算节点在 AWS 上安装 OpenShift Container Platform 集群：

- 完全自动化选项：安装集群将计算节点快速扩展到边缘计算池，其中安装程序会自动为 OpenShift Container Platform 集群创建基础架构资源。
- 现有 VPC 选项： 在 AWS 上安装集群到现有的 VPC 中，您可以为 **install-config.yaml** 文件提供 Local Zones 子网。

后续步骤

选择以下选项之一在 AWS Local Zones 环境中安装 OpenShift Container Platform 集群：

- 在 AWS 本地区中快速安装集群
- 在带有定义的 AWS Local Zone 子网的现有 VPC 上安装集群

6.3.11.6. 在 AWS 本地区中快速安装集群

对于 OpenShift Container Platform 4.16，您可以在 Amazon Web Services (AWS) 上快速安装集群，以将计算节点扩展到 Local Zones 位置。通过使用此安装路由，安装程序会为您在配置文件中定义的每个区自动创建网络资源和区域子网。要自定义安装，您必须在部署集群前修改 **install-config.yaml** 文件中的参数。

6.3.11.6.1. 修改安装配置文件以使用 AWS 本地区域

修改 **install-config.yaml** 文件，使其包含 AWS Local Zones。

先决条件

- 您已配置了 AWS 帐户。
- 您可以通过运行 **aws configure**，将 AWS 密钥和 AWS 区域添加到本地 AWS 配置集中。
- 熟悉在指定安装程序为 OpenShift Container Platform 集群自动创建子网时应用的配置限制。
- 您可以选择每个区的 Local Zones 组。

- 您使用"创建安装配置文件"流程创建了 **install-config.yaml** 文件。

流程

1. 通过在边缘计算池的 **platform.aws.zones** 属性中指定 Local Zones 名称来修改 **install-config.yaml** 文件。

```
# ...
platform:
  aws:
    region: <region_name> 1
compute:
- name: edge
  platform:
    aws:
      zones: 2
      - <local_zone_name>
#...
```

1 AWS 区域名称。

2 您使用的 Local Zones 名称列表必须存在于 **platform.aws.region** 字段中指定的同一 AWS 区域。

在 us-west-2 AWS 区域上安装集群的配置示例，将边缘节点扩展到 Los Angeles 和 Las Vegas 位置中的 Local Zones

```
apiVersion: v1
baseDomain: example.com
metadata:
  name: cluster-name
platform:
  aws:
    region: us-west-2
compute:
- name: edge
  platform:
    aws:
      zones:
      - us-west-2-lax-1a
      - us-west-2-lax-1b
      - us-west-2-las-1a
pullSecret: '{"auths": ...}'
sshKey: 'ssh-ed25519 AAAA...'
#...
```

2. 部署集群。

其他资源

- 创建安装配置文件
- AWS 本地区中的集群限制

后续步骤

- 部署集群

6.3.11.7. 在带有 Local Zone 子网的现有 VPC 上安装集群

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

在 AWS 上安装集群到现有的 VPC 中，需要使用 AWS Local Zones 将计算节点扩展到 Cloud Infrastructure 的边缘。

Local Zone 子网将常规计算节点扩展到边缘网络。每个边缘计算节点都运行用户工作负载。创建 Amazon Web Service (AWS) Local Zone 环境并部署了集群后，您可以使用边缘计算节点在 Local Zone 子网中创建用户工作负载。

注意

如果要创建专用子网，您必须修改提供的 CloudFormation 模板或创建自己的模板。

您可以使用提供的 CloudFormation 模板来创建网络资源。另外，您可以修改模板来自定义模板，或使用其包含的信息根据公司的策略创建 AWS 资源。

重要

执行安装程序置备的基础架构安装的步骤仅作为示例。在现有 VPC 上安装集群需要了解云供应商和 OpenShift Container Platform 安装过程。您可以使用 CloudFormation 模板来帮助您完成这些步骤，或者帮助您建模您自己的集群安装。您可以决定使用其他方法生成这些资源，而不使用 CloudFormation 模板来创建资源。

6.3.11.7.1. 在 AWS 中创建 VPC

您可以在 OpenShift Container Platform 集群的 Amazon Web Services (AWS) 中创建一个 Virtual Private Cloud (VPC) 和子网，以将计算节点扩展到边缘位置。您可以进一步自定义 VPC 以满足您的要求，包括 VPN 和路由表。您还可以添加新的 Local Zones 子网，不包含在初始部署中。

您可以使用提供的 CloudFormation 模板和自定义参数文件创建代表 VPC 的 AWS 资源堆栈。

注意

如果不使用提供的 CloudFormation 模板来创建 AWS 基础架构，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 已配置了一个 AWS 帐户。
- 您可以通过运行 **aws configure**，将 AWS 密钥和 AWS 区域添加到本地 AWS 配置集中。
- 您可以选择 AWS 帐户上的 AWS 区域区域。

流程

1. 创建一个 JSON 文件，其包含 CloudFormation 模板需要的参数值：

```
[
  {
    "ParameterKey": "VpcCidr", 1
    "ParameterValue": "10.0.0.0/16" 2
  },
  {
    "ParameterKey": "AvailabilityZoneCount", 3
    "ParameterValue": "3" 4
  },
  {
    "ParameterKey": "SubnetBits", 5
    "ParameterValue": "12" 6
  }
]
```

1. VPC 的 CIDR 块。
2. 以 **x.x.x.x/16-24** 格式指定 CIDR 块。
3. 在其中部署 VPC 的可用区的数量。
4. 指定一个 **1** 到 **3** 之间的整数。
5. 各个可用区中每个子网的大小。
6. 指定 **5** 到 **13** 之间的整数，其中 **5** 为 **/27**，**13** 为 **/19**。

2. 进入名为 "CloudFormation template for the VPC" 的文档部分，然后从提供的模板中复制语法。将复制的模板语法保存为本地系统中的 YAML 文件。此模板描述了集群所需的 VPC。

3. 运行以下命令，启动 CloudFormation 模板以创建代表 VPC 的 AWS 资源堆栈：

重要

您必须在一行内输入命令。

```
$ aws cloudformation create-stack --stack-name <name> \ 1
    --template-body file://<template>.yaml \ 2
    --parameters file://<parameters>.json 3
```

1. **<name>** 是 CloudFormation 堆栈的名称，如 **cluster-VPC**。如果您删除集群，则需要此堆栈的名称。
2. **<template>** 是您保存的 CloudFormation 模板 YAML 文件的相对路径和名称。
3. **<parameters>** 是相对路径和 CloudFormation 参数 JSON 文件的名称。

输出示例

```
arn:aws:cloudformation:us-east-1:123456789012:stack/cluster-vpc/dbedae40-2fd3-11eb-820e-12a48460849f
```

4. 运行以下命令确认模板组件已存在：

```
$ aws cloudformation describe-stacks --stack-name <name>
```

在 **StackStatus** 显示 **CREATE_COMPLETE** 后，输出会显示以下参数的值。您必须为创建集群运行的其他 CloudFormation 模板提供这些参数值。

VpcId	您的 VPC ID。
PublicSubnetIds	新公共子网的 ID。
PrivateSubnetIds	新专用子网的 ID。
PublicRouteTableId	新公共路由表 ID 的 ID。

6.3.11.7.2. VPC 的 CloudFormation 模板

您可以使用以下 CloudFormation 模板来部署 OpenShift Container Platform 集群所需的 VPC。

例 6.49. VPC 的 CloudFormation 模板

```
AWSTemplateFormatVersion: 2010-09-09
Description: Template for Best Practice VPC with 1-3 AZs

Parameters:
  VpcCidr:
    AllowedPattern: ^(([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])\.){3}([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])(\/(1[6-9]|2[0-4]))$
    ConstraintDescription: CIDR block parameter must be in the form x.x.x.x/16-24.
    Default: 10.0.0.0/16
    Description: CIDR block for VPC.
    Type: String
  AvailabilityZoneCount:
    ConstraintDescription: "The number of availability zones. (Min: 1, Max: 3)"
    MinValue: 1
    MaxValue: 3
    Default: 1
    Description: "How many AZs to create VPC subnets for. (Min: 1, Max: 3)"
    Type: Number
  SubnetBits:
    ConstraintDescription: CIDR block parameter must be in the form x.x.x.x/19-27.
    MinValue: 5
    MaxValue: 13
    Default: 12
    Description: "Size of each subnet to create within the availability zones. (Min: 5 = /27, Max: 13 = /19)"
```

```
    Type: Number

Metadata:
  AWS::CloudFormation::Interface:
    ParameterGroups:
    - Label:
        default: "Network Configuration"
      Parameters:
      - VpcCidr
      - SubnetBits
    - Label:
        default: "Availability Zones"
      Parameters:
      - AvailabilityZoneCount
    ParameterLabels:
      AvailabilityZoneCount:
        default: "Availability Zone Count"
      VpcCidr:
        default: "VPC CIDR"
      SubnetBits:
        default: "Bits Per Subnet"

Conditions:
  DoAz3: !Equals [3, !Ref AvailabilityZoneCount]
  DoAz2: !Or [!Equals [2, !Ref AvailabilityZoneCount], Condition: DoAz3]

Resources:
  VPC:
    Type: "AWS::EC2::VPC"
    Properties:
      EnableDnsSupport: "true"
      EnableDnsHostnames: "true"
      CidrBlock: !Ref VpcCidr
  PublicSubnet:
    Type: "AWS::EC2::Subnet"
    Properties:
      VpcId: !Ref VPC
      CidrBlock: !Select [0, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
      AvailabilityZone: !Select
      - 0
      - Fn::GetAZs: !Ref "AWS::Region"
  PublicSubnet2:
    Type: "AWS::EC2::Subnet"
    Condition: DoAz2
    Properties:
      VpcId: !Ref VPC
      CidrBlock: !Select [1, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
      AvailabilityZone: !Select
      - 1
      - Fn::GetAZs: !Ref "AWS::Region"
  PublicSubnet3:
    Type: "AWS::EC2::Subnet"
    Condition: DoAz3
    Properties:
      VpcId: !Ref VPC
      CidrBlock: !Select [2, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
```

```
    AvailabilityZone: !Select
    - 2
    - Fn::GetAZs: !Ref "AWS::Region"
InternetGateway:
  Type: "AWS::EC2::InternetGateway"
GatewayToInternet:
  Type: "AWS::EC2::VPCGatewayAttachment"
  Properties:
    VpcId: !Ref VPC
    InternetGatewayId: !Ref InternetGateway
PublicRouteTable:
  Type: "AWS::EC2::RouteTable"
  Properties:
    VpcId: !Ref VPC
PublicRoute:
  Type: "AWS::EC2::Route"
  DependsOn: GatewayToInternet
  Properties:
    RouteTableId: !Ref PublicRouteTable
    DestinationCidrBlock: 0.0.0.0/0
    GatewayId: !Ref InternetGateway
PublicSubnetRouteTableAssociation:
  Type: "AWS::EC2::SubnetRouteTableAssociation"
  Properties:
    SubnetId: !Ref PublicSubnet
    RouteTableId: !Ref PublicRouteTable
PublicSubnetRouteTableAssociation2:
  Type: "AWS::EC2::SubnetRouteTableAssociation"
  Condition: DoAz2
  Properties:
    SubnetId: !Ref PublicSubnet2
    RouteTableId: !Ref PublicRouteTable
PublicSubnetRouteTableAssociation3:
  Condition: DoAz3
  Type: "AWS::EC2::SubnetRouteTableAssociation"
  Properties:
    SubnetId: !Ref PublicSubnet3
    RouteTableId: !Ref PublicRouteTable
PrivateSubnet:
  Type: "AWS::EC2::Subnet"
  Properties:
    VpcId: !Ref VPC
    CidrBlock: !Select [3, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
    AvailabilityZone: !Select
    - 0
    - Fn::GetAZs: !Ref "AWS::Region"
PrivateRouteTable:
  Type: "AWS::EC2::RouteTable"
  Properties:
    VpcId: !Ref VPC
PrivateSubnetRouteTableAssociation:
  Type: "AWS::EC2::SubnetRouteTableAssociation"
  Properties:
    SubnetId: !Ref PrivateSubnet
    RouteTableId: !Ref PrivateRouteTable
NAT:
```

```
  DependsOn:
  - GatewayToInternet
  Type: "AWS::EC2::NatGateway"
  Properties:
    AllocationId:
      "Fn::GetAtt":
      - EIP
      - AllocationId
    SubnetId: !Ref PublicSubnet
EIP:
  Type: "AWS::EC2::EIP"
  Properties:
    Domain: vpc
Route:
  Type: "AWS::EC2::Route"
  Properties:
    RouteTableId:
      Ref: PrivateRouteTable
    DestinationCidrBlock: 0.0.0.0/0
    NatGatewayId:
      Ref: NAT
PrivateSubnet2:
  Type: "AWS::EC2::Subnet"
  Condition: DoAz2
  Properties:
    VpcId: !Ref VPC
    CidrBlock: !Select [4, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
    AvailabilityZone: !Select
    - 1
    - Fn::GetAZs: !Ref "AWS::Region"
PrivateRouteTable2:
  Type: "AWS::EC2::RouteTable"
  Condition: DoAz2
  Properties:
    VpcId: !Ref VPC
PrivateSubnetRouteTableAssociation2:
  Type: "AWS::EC2::SubnetRouteTableAssociation"
  Condition: DoAz2
  Properties:
    SubnetId: !Ref PrivateSubnet2
    RouteTableId: !Ref PrivateRouteTable2
NAT2:
  DependsOn:
  - GatewayToInternet
  Type: "AWS::EC2::NatGateway"
  Condition: DoAz2
  Properties:
    AllocationId:
      "Fn::GetAtt":
      - EIP2
      - AllocationId
    SubnetId: !Ref PublicSubnet2
EIP2:
  Type: "AWS::EC2::EIP"
  Condition: DoAz2
  Properties:
```

```
    Domain: vpc
Route2:
  Type: "AWS::EC2::Route"
  Condition: DoAz2
  Properties:
    RouteTableId:
      Ref: PrivateRouteTable2
    DestinationCidrBlock: 0.0.0.0/0
    NatGatewayId:
      Ref: NAT2
PrivateSubnet3:
  Type: "AWS::EC2::Subnet"
  Condition: DoAz3
  Properties:
    VpcId: !Ref VPC
    CidrBlock: !Select [5, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
    AvailabilityZone: !Select
    - 2
    - Fn::GetAZs: !Ref "AWS::Region"
PrivateRouteTable3:
  Type: "AWS::EC2::RouteTable"
  Condition: DoAz3
  Properties:
    VpcId: !Ref VPC
PrivateSubnetRouteTableAssociation3:
  Type: "AWS::EC2::SubnetRouteTableAssociation"
  Condition: DoAz3
  Properties:
    SubnetId: !Ref PrivateSubnet3
    RouteTableId: !Ref PrivateRouteTable3
NAT3:
  DependsOn:
  - GatewayToInternet
  Type: "AWS::EC2::NatGateway"
  Condition: DoAz3
  Properties:
    AllocationId:
      "Fn::GetAtt":
      - EIP3
      - AllocationId
    SubnetId: !Ref PublicSubnet3
EIP3:
  Type: "AWS::EC2::EIP"
  Condition: DoAz3
  Properties:
    Domain: vpc
Route3:
  Type: "AWS::EC2::Route"
  Condition: DoAz3
  Properties:
    RouteTableId:
      Ref: PrivateRouteTable3
    DestinationCidrBlock: 0.0.0.0/0
    NatGatewayId:
      Ref: NAT3
S3Endpoint:
```

```
    Type: AWS::EC2::VPCEndpoint
    Properties:
      PolicyDocument:
        Version: 2012-10-17
        Statement:
        - Effect: Allow
          Principal: '*'
          Action:
          - '*'
          Resource:
          - '*'
      RouteTableIds:
      - !Ref PublicRouteTable
      - !Ref PrivateRouteTable
      - !If [DoAz2, !Ref PrivateRouteTable2, !Ref "AWS::NoValue"]
      - !If [DoAz3, !Ref PrivateRouteTable3, !Ref "AWS::NoValue"]
      ServiceName: !Join
      - ''
      - - com.amazonaws.
        - !Ref 'AWS::Region'
        - .s3
      VpcId: !Ref VPC

Outputs:
  VpcId:
    Description: ID of the new VPC.
    Value: !Ref VPC
  PublicSubnetIds:
    Description: Subnet IDs of the public subnets.
    Value:
      !Join [
        ",",
        [!Ref PublicSubnet, !If [DoAz2, !Ref PublicSubnet2, !Ref "AWS::NoValue"], !If [DoAz3, !Ref
PublicSubnet3, !Ref "AWS::NoValue"]]
      ]
  PrivateSubnetIds:
    Description: Subnet IDs of the private subnets.
    Value:
      !Join [
        ",",
        [!Ref PrivateSubnet, !If [DoAz2, !Ref PrivateSubnet2, !Ref "AWS::NoValue"], !If [DoAz3, !Ref
PrivateSubnet3, !Ref "AWS::NoValue"]]
      ]
  PublicRouteTableId:
    Description: Public Route table ID
    Value: !Ref PublicRouteTable
  PrivateRouteTableIds:
    Description: Private Route table IDs
    Value:
      !Join [
        ",",
        [
          !Join ["=", [
            !Select [0, "Fn::GetAZs": !Ref "AWS::Region"],
            !Ref PrivateRouteTable
          ]],
```

```
        !If [DoAz2,
            !Join ["=", [!Select [1, "Fn::GetAZs": !Ref "AWS::Region"], !Ref PrivateRouteTable2]],
            !Ref "AWS::NoValue"
        ],
        !If [DoAz3,
            !Join ["=", [!Select [2, "Fn::GetAZs": !Ref "AWS::Region"], !Ref PrivateRouteTable3]],
            !Ref "AWS::NoValue"
        ]
      ]
    ]
```

6.3.11.7.3. 在本地区中创建子网

在 OpenShift Container Platform 集群中为边缘计算节点配置机器集前，您必须在 Local Zones 中创建子网。对您要将计算节点部署到的每个 Local Zone 完成以下步骤。

您可以使用提供的 CloudFormation 模板并创建 CloudFormation 堆栈。然后，您可以使用此堆栈自定义置备子网。

注意

如果不使用提供的 CloudFormation 模板来创建 AWS 基础架构，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 已配置了一个 AWS 帐户。
- 您可以通过运行 **aws configure**，将 AWS 密钥和区域添加到本地 AWS 配置集中。
- 您可以选择 Local Zones 组。

流程

1. 进入名为"CloudFormation template for the VPC 子网"的文档部分，并从模板中复制语法。将复制的模板语法保存为本地系统中的 YAML 文件。此模板描述了集群所需的 VPC。
2. 运行以下命令来部署 CloudFormation 模板，它会创建一个代表 VPC 的 AWS 资源堆栈：

```
$ aws cloudformation create-stack --stack-name <stack_name> \ (1)
  --region ${CLUSTER_REGION} \
  --template-body file://<template>.yaml \ (2)
  --parameters \
    ParameterKey=VpcId,ParameterValue="${VPC_ID}" \ (3)
    ParameterKey=ClusterName,ParameterValue="${CLUSTER_NAME}" \ (4)
    ParameterKey=ZoneName,ParameterValue="${ZONE_NAME}" \ (5)
    ParameterKey=PublicRouteTableId,ParameterValue="${ROUTE_TABLE_PUB}" \ (6)
    ParameterKey=PublicSubnetCidr,ParameterValue="${SUBNET_CIDR_PUB}" \ (7)
    ParameterKey=PrivateRouteTableId,ParameterValue="${ROUTE_TABLE_PVT}" \ (8)
    ParameterKey=PrivateSubnetCidr,ParameterValue="${SUBNET_CIDR_PVT}" (9)
```

1. **<stack_name>** 是 CloudFormation 堆栈的名称，如 **cluster-wl-<local_zone_shortname>**。如果您删除集群，则需要此堆栈的名称。
2. **<template>** 是相对路径，以及保存的 CloudFormation 模板 YAML 文件的名称。
3. **${VPC_ID}** 是 VPC ID，它是 VPC 模板输出中的 **VpcID** 值。
4. **${ZONE_NAME}** 是创建子网的 Local Zones 名称的值。
5. **${CLUSTER_NAME}** 是 **ClusterName** 的值，用作新 AWS 资源名称的前缀。
6. **${SUBNET_CIDR_PUB}** 是一个有效的 CIDR 块，用于创建公共子网。这个块必须是 VPC CIDR 块 **VpcCidr** 的一部分。
7. **${ROUTE_TABLE_PVT}** 是从 VPC 的 CloudFormation 堆栈的输出中提取的 **PrivateRouteTableId**。
8. **${SUBNET_CIDR_PVT}** 是一个有效的 CIDR 块，用于创建专用子网。这个块必须是 VPC CIDR 块 **VpcCidr** 的一部分。

输出示例

```
arn:aws:cloudformation:us-east-1:123456789012:stack/<stack_name>/dbedae40-820e-11eb-2fd3-12a48460849f
```

验证

- 运行以下命令确认模板组件已存在：

```
$ aws cloudformation describe-stacks --stack-name <stack_name>
```

在 **StackStatus** 显示 **CREATE_COMPLETE** 后，输出会显示以下参数的值。确保将这些参数值提供给您为集群创建的其他 CloudFormation 模板。

PublicSubnetId	由 CloudFormation 堆栈创建的公共子网的 ID。
PrivateSubnetId	由 CloudFormation 堆栈创建的专用子网的 ID。

6.3.11.7.4. VPC 子网的 CloudFormation 模板

您可以使用以下 CloudFormation 模板，在 Local Zones 基础架构的区域中部署私有和公共子网。

例 6.50. VPC 子网的 CloudFormation 模板

```
AWSTemplateFormatVersion: 2010-09-09
Description: Template for Best Practice Subnets (Public and Private)

Parameters:
  VpcId:
    Description: VPC ID that comprises all the target subnets.
```

```
    Type: String
    AllowedPattern: ^(?:(?:vpc)(?:-[a-zA-Z0-9]+)?\b|(?:[0-9]{1,3}\.){3}[0-9]{1,3})$
    ConstraintDescription: VPC ID must be with valid name, starting with vpc-.*.
  ClusterName:
    Description: Cluster name or prefix name to prepend the Name tag for each subnet.
    Type: String
    AllowedPattern: ".+"
    ConstraintDescription: ClusterName parameter must be specified.
  ZoneName:
    Description: Zone Name to create the subnets, such as us-west-2-lax-1a.
    Type: String
    AllowedPattern: ".+"
    ConstraintDescription: ZoneName parameter must be specified.
  PublicRouteTableId:
    Description: Public Route Table ID to associate the public subnet.
    Type: String
    AllowedPattern: ".+"
    ConstraintDescription: PublicRouteTableId parameter must be specified.
  PublicSubnetCidr:
    AllowedPattern: ^(([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])\.){3}([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-
4][0-9]|25[0-5])(\/(1[6-9]|2[0-4]))$
    ConstraintDescription: CIDR block parameter must be in the form x.x.x.x/16-24.
    Default: 10.0.128.0/20
    Description: CIDR block for public subnet.
    Type: String
  PrivateRouteTableId:
    Description: Private Route Table ID to associate the private subnet.
    Type: String
    AllowedPattern: ".+"
    ConstraintDescription: PrivateRouteTableId parameter must be specified.
  PrivateSubnetCidr:
    AllowedPattern: ^(([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])\.){3}([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-
4][0-9]|25[0-5])(\/(1[6-9]|2[0-4]))$
    ConstraintDescription: CIDR block parameter must be in the form x.x.x.x/16-24.
    Default: 10.0.128.0/20
    Description: CIDR block for private subnet.
    Type: String


Resources:
  PublicSubnet:
    Type: "AWS::EC2::Subnet"
    Properties:
      VpcId: !Ref VpcId
      CidrBlock: !Ref PublicSubnetCidr
      AvailabilityZone: !Ref ZoneName
      Tags:
      - Key: Name
        Value: !Join ['-', [!Ref ClusterName, "public", !Ref ZoneName]]

  PublicSubnetRouteTableAssociation:
    Type: "AWS::EC2::SubnetRouteTableAssociation"
    Properties:
      SubnetId: !Ref PublicSubnet
      RouteTableId: !Ref PublicRouteTableId
```

```
  PrivateSubnet:
    Type: "AWS::EC2::Subnet"
    Properties:
      VpcId: !Ref VpcId
      CidrBlock: !Ref PrivateSubnetCidr
      AvailabilityZone: !Ref ZoneName
      Tags:
      - Key: Name
        Value: !Join ['-', [!Ref ClusterName, "private", !Ref ZoneName]]

  PrivateSubnetRouteTableAssociation:
    Type: "AWS::EC2::SubnetRouteTableAssociation"
    Properties:
      SubnetId: !Ref PrivateSubnet
      RouteTableId: !Ref PrivateRouteTableId

Outputs:
  PublicSubnetId:
    Description: Subnet ID of the public subnets.
    Value:
      !Join ["", [!Ref PublicSubnet]]

  PrivateSubnetId:
    Description: Subnet ID of the private subnets.
    Value:
      !Join ["", [!Ref PrivateSubnet]]
```

其他资源

- 您可以通过导航 AWS CloudFormation 控制台来查看您创建的 CloudFormation 堆栈的详情。

6.3.11.7.5. 修改安装配置文件以使用 AWS 本地区域子网

修改 **install-config.yaml** 文件，使其包含 Local Zones 子网。

先决条件

- 您使用"在 Local Zones 中创建子网"流程创建子网。
- 您使用"创建安装配置文件"流程创建了 **install-config.yaml** 文件。

流程

- 通过在 **platform.aws.subnets** 参数中指定 Local Zones 子网来修改 **install-config.yaml** 配置文件。

  带有本地区域子网的安装配置文件示例

```
# ...
platform:
  aws:
    region: us-west-2
    subnets: 1
```

```
    - publicSubnetId-1
    - publicSubnetId-2
    - publicSubnetId-3
    - privateSubnetId-1
    - privateSubnetId-2
    - privateSubnetId-3
    - publicSubnetId-LocalZone-1
# ...
```

1 区域中创建的子网 ID 列表： 可用性和本地区域。

其他资源

- 有关查看您创建的 CloudFormation 堆栈的更多信息，请参阅 AWS CloudFormation 控制台。
- 如需有关 AWS 配置集和凭证配置的更多信息，请参阅 AWS 文档中的配置和凭证文件设置。

后续步骤

- 部署集群

6.3.11.8. 可选： AWS 安全组

默认情况下，安装程序会创建安全组并将其附加到 control plane 和计算机器。不可修改与默认安全组关联的规则。

但是，您可以将与现有 VPC 关联的其他现有 AWS 安全组应用到 control plane 和计算机器。应用自定义安全组可帮助您满足机构的安全需求，在这种情况下，您需要控制这些机器的传入或传出流量。

作为安装过程的一部分，您可以在部署集群前通过修改 **install-config.yaml** 文件来应用自定义安全组。

如需更多信息，请参阅"边缘计算池和 AWS 本地区域"。

6.3.11.9. 可选：将公共 IP 地址分配给边缘计算节点

如果您的工作负载需要在 Local Zones 基础架构上公共子网中部署边缘计算节点，您可以在安装集群时配置机器集清单。

AWS Local Zones 基础架构访问指定区中的网络流量，因此应用程序可在提供更接近该区的最终用户时利用较低延迟。

在私有子网中部署计算节点的默认设置可能无法满足您的需求，因此当您想要将更多自定义应用到基础架构时，请考虑在公共子网中创建边缘计算节点。

重要

默认情况下，OpenShift Container Platform 在私有子网中部署计算节点。为获得最佳性能，请考虑将计算节点放在附加了其公共 IP 地址的子网中。

您必须创建额外的安全组，但请确保仅在需要时通过互联网打开组规则。

流程

1. 进入包含安装程序的目录并生成清单文件。确保安装清单在 **openshift** 和 **manifests** 目录级别创建。

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

2. 编辑安装程序为 Local Zones 生成的机器集清单，以便清单部署在公共子网中。为 **spec.template.spec.providerSpec.value.publicIP** 参数指定 **true**。

在 Local Zones 中快速安装集群的机器集清单配置示例

```
spec:
  template:
    spec:
      providerSpec:
        value:
          publicIp: true
          subnet:
            filters:
              - name: tag:Name
                values:
                  - ${INFRA_ID}-public-${ZONE_NAME}
```

在具有 Local Zones 子网的现有 VPC 上安装集群的机器集清单配置示例

```
apiVersion: machine.openshift.io/v1beta1
kind: MachineSet
metadata:
  name: <infrastructure_id>-edge-<zone>
  namespace: openshift-machine-api
spec:
  template:
    spec:
      providerSpec:
        value:
          publicIp: true
```

6.3.11.10. 部署集群

您可以在兼容云平台上安装 OpenShift Container Platform。

重要

在初始安装过程中，您只能运行安装程序的 **create cluster** 命令一次。

先决条件

- 您已使用托管集群的云平台配置了帐户。
- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。
- 已确认主机上的云供应商帐户具有部署集群的正确权限。权限不正确的帐户会导致安装过程失败，并显示包括缺失权限的错误消息。

流程

1. 进入包含安装程序的目录并初始化集群部署：

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

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

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

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

注意

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

验证

当集群部署成功完成时：

- 终端会显示用于访问集群的说明，包括指向 Web 控制台和 **kubeadmin** 用户的凭证的链接。
- 凭证信息还会输出到 **<installation_directory>/.openshift_install.log**.

重要

不要删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

输出示例

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

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

6.3.11.11. 验证部署集群的状态

验证 OpenShift Container Platform 是否已在 AWS Local Zones 上部署。

6.3.11.11.1. 使用 CLI 登录到集群

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

先决条件

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

流程

1. 导出 **kubeadmin** 凭证：

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

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

2. 验证您可以使用导出的配置成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

6.3.11.11.2. 使用 Web 控制台登录到集群

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

先决条件

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

流程

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

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

注意

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

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

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

注意

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

输出示例

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

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

其他资源

- 访问Web控制台

6.3.11.11.3. 验证使用边缘计算池创建的节点

安装使用 AWS Local Zones 基础架构的集群后，检查安装过程中由机器集清单创建的机器状态。

1. 要检查从添加到 **install-config.yaml** 文件中的子网中创建的机器集，请运行以下命令：

```
$ oc get machineset -n openshift-machine-api
```

输出示例

```
NAME                              DESIRED   CURRENT   READY   AVAILABLE   AGE
cluster-7xw5g-edge-us-east-1-nyc-1a   1         1         1       1           3h4m
cluster-7xw5g-worker-us-east-1a       1         1         1       1           3h4m
cluster-7xw5g-worker-us-east-1b       1         1         1       1           3h4m
cluster-7xw5g-worker-us-east-1c       1         1         1       1           3h4m
```

2. 要检查从机器集创建的机器，请运行以下命令：

```
$ oc get machines -n openshift-machine-api
```

输出示例

```
NAME                                    PHASE     TYPE        REGION      ZONE           AGE
cluster-7xw5g-edge-us-east-1-nyc-1a-wbclh   Running   c5d.2xlarge   us-east-1   us-east-1-
nyc-1a   3h
cluster-7xw5g-master-0                  Running   m6i.xlarge    us-east-1   us-east-1a     3h4m
cluster-7xw5g-master-1                  Running   m6i.xlarge    us-east-1   us-east-1b     3h4m
cluster-7xw5g-master-2                  Running   m6i.xlarge    us-east-1   us-east-1c     3h4m
cluster-7xw5g-worker-us-east-1a-rtp45       Running   m6i.xlarge    us-east-1   us-east-1a
3h
cluster-7xw5g-worker-us-east-1b-glm7c       Running   m6i.xlarge    us-east-1   us-east-1b
```

```
3h
cluster-7xw5g-worker-us-east-1c-qfvz4      Running   m6i.xlarge    us-east-1   us-east-1c
3h
```

3. 要检查具有边缘角色的节点，请运行以下命令：

```
$ oc get nodes -l node-role.kubernetes.io/edge
```

输出示例

```
NAME                          STATUS   ROLES         AGE    VERSION
ip-10-0-207-188.ec2.internal   Ready    edge,worker   172m   v1.25.2+d2e245f
```

后续步骤

- 验证安装.
- 如果需要，您可以选择 不使用远程健康。

6.3.12. 在 AWS Wavelength Zones 上使用计算节点安装集群

您可以通过在 **install-config.yaml** 文件的边缘计算池中设置区域名称，或者在带有 Wavelength Zone 子网的现有 Amazon Virtual Private Cloud (VPC)上安装集群，在 Amazon Web Services (AWS)上快速安装 OpenShift Container Platform 集群。

AWS Wavelength Zones 是 AWS 为移动边缘计算(MEC)应用程序配置的基础架构。

Wavelength 区域在通信服务提供商(CSP)的 5G 网络中嵌入 AWS 计算和存储服务。通过将应用服务器放在 Wavelength Zone 中，您的 5G 设备的应用程序流量可以保留在 5G 网络中。设备的应用程序流量直接到达目标服务器，使延迟成为非问题。

其他资源

- 请参阅 AWS 文档中的 Wavelength Zones。

6.3.12.1. 基础架构先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新流程 的详细信息。
- 熟悉 选择集群安装方法并为用户准备它。
- 已将 AWS 帐户配置为托管集群。

警告

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

- 您下载了 AWS CLI 并安装到您的计算机上。请参阅 AWS 文档中的使用捆绑安装程序（Linux、macOS 或 UNIX）安装 AWS CLI。
- 如果使用防火墙，则会 将其配置为允许集群需要访问的站点。
- 您记下了区域和支持的 AWS Wavelength Zone 位置，在其中创建网络资源。
- 请参阅 AWS 文档中的 AWS Wavelength 功能。
- 您可以在 AWS 文档中阅读 Wavelength 区域的配额和注意事项。
- 您已将创建支持 AWS Wavelength Zones 的网络资源添加到 Identity and Access Management (IAM) 用户或角色的权限。例如：

附加 ec2:ModifyAvailabilityZoneGroup,ec2:CreateCarrierGateway, ec2:DeleteCarrierGateway 权限的额外 IAM 策略示例

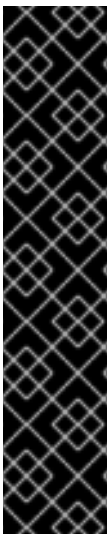
```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ec2:DeleteCarrierGateway",
        "ec2:CreateCarrierGateway"
      ],
      "Resource": "*"
    },
    {
      "Action": [
        "ec2:ModifyAvailabilityZoneGroup"
      ],
      "Effect": "Allow",
      "Resource": "*"
    }
  ]
}
```

6.3.12.2. 关于 AWS Wavelength Zones 和 edge 计算池

阅读以下部分以了解 AWS Wavelength 区域环境中的基础架构行为和集群限制。

6.3.12.2.1. AWS Wavelength 区域中的集群限制

当您试图在 Amazon Web Services (AWS) Wavelength Zone 中使用默认安装配置部署集群时，有一些限制。



重要

以下列表在预先配置的 AWS 区域中部署集群时的详情限制：

- 区域中的 Amazon EC2 实例和 Region 中的 Amazon EC2 实例之间的最大传输单元 (MTU) 为 **1300**。这会导致集群范围的网络 MTU 根据与部署一起使用的网络插件而改变。
- 不支持 Network Load Balancer (NLB)、Classic Load Balancer 和网络地址转换 (NAT)网关等网络资源。
- 对于 AWS 上的 OpenShift Container Platform 集群，AWS Elastic Block Storage (EBS) **gp3** 类型卷是节点卷和存储类的默认设置。这个卷类型在区域位置没有全局可用。默认情况下，在区中运行的节点使用 **gp2** EBS 卷进行部署。在区节点上创建工作负载时，必须设置 **gp2-csi StorageClass** 参数。

如果您希望安装程序为 OpenShift Container Platform 集群自动创建 Wavelength Zone 子网，则使用此方法应用特定的配置限制。以下详细介绍了其中的一些限制。对于其他限制，请确保阅读红帽在"Infrastructure prerequisites"部分中提供的 Wavelength Zones 的"Quotas and considerations"文档。

重要

当您将安装程序设置为自动为 OpenShift Container Platform 集群创建子网时，会有以下配置限制：

- 当安装程序在 AWS Wavelength 区域中创建专用子网时，安装程序会将每个子网与其父区的路由表相关联。此操作通过 AWS 区域中的 NAT 网关的方式确保每个专用子网都可以将出口流量路由到互联网。
- 如果集群安装过程中不存在 parent-zone 路由表，安装程序会将任何专用子网与 Amazon Virtual Private Cloud (VPC)中的第一个可用私有路由表相关联。这个方法只适用于 OpenShift Container Platform 集群中的 AWS Wavelength Zones 子网。

6.3.12.2.2. 关于边缘计算池

边缘计算节点是在 AWS Wavelength 区域位置中运行的污点计算节点。

在部署使用 Wavelength 区域的集群时，请考虑以下点：

- Wavelength 区域中的 Amazon EC2 实例比可用区中的 Amazon EC2 实例的成本更高。
- 在 AWS Wavelength 区域和最终用户中运行的应用程序间延迟较低。例如，当一些工作负载在 Wavelength 区域和可用区之间混合了入口流量，则对一些工作负载会有一个延迟影响。

重要

通常，Wavelength 区域中的 Amazon EC2 实例和 Region 中的 Amazon EC2 实例之间的最大传输单元(MTU)为 1300。对于开销，集群网络 MTU 必须总是小于 EC2 MTU。具体开销由网络插件决定。例如： OVN-Kubernetes 的开销为 **100 字节**。

网络插件可以提供额外的功能，如 IPsec，它们也会影响 MTU 大小。

如需更多信息，请参阅 AWS 文档中的 AWS Wavelength 如何工作 。

OpenShift Container Platform 4.12 引入了一个新的计算池 *edge*，用于在远程区中使用。边缘计算池配置在 AWS Wavelength 区域位置之间很常见。由于 Wavelength Zones 资源上的 EC2 和 EBS 等资源的类型和大小限制，默认的实例类型可能与传统的计算池不同。

Wavelength Zones 位置的默认 Elastic Block Store (EBS) 是 **gp2**，它与非边缘计算池不同。根据区域上的实例产品，边缘计算池中每个 Wavelength 区域使用的实例类型可能与其他计算池不同。

边缘计算池创建新的标签，供开发人员用来将应用程序部署到 AWS Wavelength 区域节点上。新标签包括：

- **node-role.kubernetes.io/edge=''**
- **machine.openshift.io/zone-type=wavelength-zone**
- **machine.openshift.io/zone-group=$ZONE_GROUP_NAME**

默认情况下，边缘计算池的机器集定义 **NoSchedule** 污点，以防止其他工作负载分散到 Wavelength Zones 实例。只有用户在 pod 规格中定义容限时，用户才能运行用户工作负载。

其他资源

- MTU 值选择
- 更改集群网络的 MTU
- 了解污点和容限
- 存储类
- Ingress Controller 分片

6.3.12.3. 安装先决条件

在 AWS Wavelength Zones 环境中安装集群前，您必须配置基础架构，以便它可以使用 Wavelength Zone 功能。

6.3.12.3.1. 选择 AWS Wavelength 区域

如果您计划在 AWS Wavelength 区域中创建子网，则必须单独选择每个 zone group。

先决条件

- 已安装 AWS CLI。
- 您已决定要部署 OpenShift Container Platform 集群的 AWS 区域。

- 您已将 permissive IAM 策略附加到选择 zone 组的用户或组帐户。

流程

1. 运行以下命令，列出 AWS 区域中可用的区域：

   在 AWS 区域中列出可用 AWS Wavelength 区域的命令示例

   ```
   $ aws --region "<value_of_AWS_Region>" ec2 describe-availability-zones \
       --query 'AvailabilityZones[].[{ZoneName: ZoneName, GroupName: GroupName, Status:
   OptInStatus}]' \
       --filters Name=zone-type,Values=wavelength-zone \
       --all-availability-zones
   ```

   根据 AWS 区域，可用区列表可能比较长。该命令返回以下字段：

   ZoneName

   Wavelength 区域的名称。

   GroupName

   组成区域的组。要选择 Region，保存名称。

   Status

   Wavelength Zones 组的状态。如果状态是 **not-opted-in**，则需要选择 **GroupName**，如下一步所述。

2. 运行以下命令，选择 AWS 帐户上的 zone 组：

   ```
   $ aws ec2 modify-availability-zone-group \
       --group-name "<value_of_GroupName>" \ 1
       --opt-in-status opted-in
   ```

   1 将 **<value_of_GroupName>** 替换为您要创建子网的 Wavelength 区域组的名称。对于 Wavelength 区域的示例，指定 **us-east-1-wl1** 使用区域 **us-east-1-wl1-nyc-wlz-1** (US East New York)。

6.3.12.3.2. 获取 AWS Marketplace 镜像

如果要使用 AWS Marketplace 镜像部署 OpenShift Container Platform 集群，您必须首先通过 AWS 订阅。订阅提供的提供的 AMI ID 可让您使用安装程序用来部署计算节点的 AMI ID。

先决条件

- 有 AWS 账户购买的产品。此帐户不必与用于安装集群的帐户相同。

流程

1. 从 AWS Marketplace 完成 OpenShift Container Platform 订阅。
2. 记录特定 AWS 区域的 AMI ID。作为安装过程的一部分，您必须在部署集群前使用这个值更新 **install-config.yaml** 文件。

   使用 AWS Marketplace 计算节点的 install-config.yaml 文件示例

   ■

```
apiVersion: v1
baseDomain: example.com
compute:
- hyperthreading: Enabled
  name: worker
  platform:
    aws:
      amiID: ami-06c4d345f7c207239 (1)
      type: m5.4xlarge
  replicas: 3
metadata:
  name: test-cluster
platform:
  aws:
    region: us-east-2 (2)
sshKey: ssh-ed25519 AAAA...
pullSecret: '{"auths": ...}'
```

1. 来自 AWS Marketplace 订阅的 AMI ID。
2. 您的 AMI ID 与特定的 AWS 区域相关联。在创建安装配置文件时，请确保选择配置订阅时指定的相同 AWS 区域。

6.3.12.4. 准备安装

在将节点扩展到 Wavelength 区域前，您必须为集群安装环境准备某些资源。

6.3.12.4.1. 集群安装的最低资源要求

每台集群机器都必须满足以下最低要求：

表 6.20. 最低资源要求

机器	操作系统	vCPU [1]	虚拟内存	Storage	每秒输入/输出 (IOPS) [2]
bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane（控制平面）	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS、RHEL 8.6 及更新版本 [3]	2	8 GB	100 GB	300

1. 当未启用并发多线程(SMT)或超线程时，一个 vCPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比例：（每个内核数的线程）× sockets = vCPU。
2. OpenShift Container Platform 和 Kubernetes 对磁盘性能非常敏感，建议使用更快的存储速度，特别是 control plane 节点上需要 10 ms p99 fsync 持续时间的 etcd。请注意，在许多云平台上，存储大小和 IOPS 可一起扩展，因此您可能需要过度分配存储卷来获取足够的性能。

3. 与所有用户置备的安装一样，如果您选择在集群中使用 RHEL 计算机器，则负责所有操作系统生命周期管理和维护，包括执行系统更新、应用补丁和完成所有其他必要的任务。RHEL 7 计算机器的使用已弃用，并已在 OpenShift Container Platform 4.10 及更新的版本中删除。

注意

从 OpenShift Container Platform 版本 4.13 开始，RHCOS 基于 RHEL 版本 9.2，它更新了微架构要求。以下列表包含每个架构需要的最小指令集架构 (ISA)：

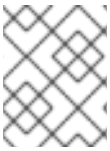
- x86-64 体系结构需要 x86-64-v2 ISA
- ARM64 架构需要 ARMv8.0-A ISA
- IBM Power 架构需要 Power 9 ISA
- s390x 架构需要 z14 ISA

如需更多信息，请参阅 RHEL 架构。

如果平台的实例类型满足集群机器的最低要求，则 OpenShift Container Platform 支持使用它。

6.3.12.4.2. 为 AWS 测试的实例类型

以下 Amazon Web Services (AWS) 实例类型已与 OpenShift Container Platform 测试，用于 AWS Wavelength Zones。



注意

将以下图中包含的机器类型用于 AWS 实例。如果您使用没有在图表中列出的实例类型，请确保使用的实例大小与名为"最小资源要求"的部分中列出的最少资源要求匹配。

例 6.51. 基于 AWS Wavelength 区域的 64 位 x86 架构的机器类型

- **r5.***
- **t3.***

其他资源

- 请参阅 AWS 文档中的 AWS Wavelength 功能。

6.3.12.4.3. 创建安装配置文件

生成并自定义安装程序部署集群所需的安装配置文件。

先决条件

- 已获取 OpenShift Container Platform 安装程序和集群的 pull secret。
- 使用红帽发布的附带 Red Hat Enterprise Linux CoreOS (RHCOS) AMI 检查您是否将集群部署到 AWS 区域。如果要部署到需要自定义 AMI 的 AWS 区域，如 AWS GovCloud 区域，您必须手动创建 **install-config.yaml** 文件。

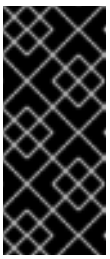
流程

1. 创建 **install-config.yaml** 文件。

   a. 进入包含安装程序的目录并运行以下命令：

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

   1 对于 **<installation_directory>**，请指定要存储安装程序创建的文件的目录名称。

   重要

   指定一个空目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

   b. 在提示符处，提供云的配置详情：

      i. 可选： 选择用于访问集群机器的 SSH 密钥。

      注意

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

      ii. 选择 **aws** 作为目标平台。

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

      注意

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

      iv. 选择要将集群部署到的 AWS Region。

      v. 选择您为集群配置的 Route 53 服务的基域。

      vi. 为集群输入描述性名称。

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

2. 可选：备份 **install-config.yaml** 文件。

   重要

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

6.3.12.4.4. 使用边缘计算池安装配置文件示例

以下示例显示了包含边缘机器池配置的 **install-config.yaml** 文件。

使用自定义实例类型使用边缘池的配置

```
apiVersion: v1
baseDomain: devcluster.openshift.com
metadata:
  name: ipi-edgezone
compute:
- name: edge
  platform:
    aws:
      type: r5.2xlarge
platform:
  aws:
    region: us-west-2
pullSecret: '{"auths": ...}'
sshKey: ssh-ed25519 AAAA...
```

实例类型因位置而异。要验证集群运行的 Wavelength 区域中的可用性，请参阅 AWS 文档。

使用自定义安全组使用边缘池的配置

```
apiVersion: v1
baseDomain: devcluster.openshift.com
metadata:
  name: ipi-edgezone
compute:
- name: edge
  platform:
    aws:
      additionalSecurityGroupIDs:
        - sg-1 (1)
        - sg-2
platform:
  aws:
    region: us-west-2
pullSecret: '{"auths": ...}'
sshKey: ssh-ed25519 AAAA...
```

(1) 指定安全组的名称，因为它在 Amazon EC2 控制台中显示。确保包含 **sg** 前缀。

6.3.12.5. AWS Wavelength Zones 环境的集群安装选项

选择以下安装选项之一，在 AWS 上安装带有在 Wavelength Zones 中定义的边缘计算节点的 OpenShift Container Platform 集群：

- 完全自动化选项：安装集群将计算节点快速扩展到边缘计算池，其中安装程序会自动为 OpenShift Container Platform 集群创建基础架构资源。
- 现有 VPC 选项：在 AWS 上安装集群到现有的 VPC 中，您可以为 **install-config.yaml** 文件提供 Wavelength Zones 子网。

后续步骤

选择以下选项之一在 AWS Wavelength Zones 环境中安装 OpenShift Container Platform 集群：

- 在 AWS Wavelength 区中快速安装集群
- 修改安装配置文件以使用 AWS Wavelength 区域

6.3.12.6. 在 AWS Wavelength 区中快速安装集群

对于 OpenShift Container Platform 4.16，您可以在 Amazon Web Services (AWS) 上快速安装集群，以将计算节点扩展到 Wavelength 区域位置。通过使用此安装路由，安装程序会为您在配置文件中定义的每个区自动创建网络资源和 Wavelength 区域子网。要自定义安装，您必须在部署集群前修改 **install-config.yaml** 文件中的参数。

6.3.12.6.1. 修改安装配置文件以使用 AWS Wavelength 区域

修改 **install-config.yaml** 文件，使其包含 AWS Wavelength 区域。

先决条件

- 您已配置了 AWS 帐户。
- 您可以通过运行 **aws configure**，将 AWS 密钥和 AWS 区域添加到本地 AWS 配置集中。
- 熟悉在指定安装程序为 OpenShift Container Platform 集群自动创建子网时应用的配置限制。
- 您选择每个区的 Wavelength Zones 组。
- 您使用"创建安装配置文件"流程创建了 **install-config.yaml** 文件。

流程

1. 通过在边缘计算池的 **platform.aws.zones** 属性中指定 Wavelength Zones 名称来修改 **install-config.yaml** 文件。

```
# ...
platform:
  aws:
    region: <region_name> 1
compute:
- name: edge
  platform:
    aws:
      zones: 2
      - <wavelength_zone_name>
#...
```

1 AWS 区域名称。

2 您使用的 Wavelength 区域名称列表必须存在于 **platform.aws.region** 字段中指定的同一 AWS 区域。

在 us-west-2 AWS 区域上安装集群的配置示例，将边缘节点扩展到 Los Angeles 和 Las Vegas 位置中的 Wavelength Zones

```
apiVersion: v1
baseDomain: example.com
metadata:
  name: cluster-name
platform:
  aws:
    region: us-west-2
compute:
- name: edge
  platform:
    aws:
      zones:
      - us-west-2-wl1-lax-wlz-1
      - us-west-2-wl1-las-wlz-1
pullSecret: '{"auths": ...}'
sshKey: 'ssh-ed25519 AAAA...'
#...
```

2. 部署集群。

其他资源

- 创建安装配置文件
- AWS Wavelength 区域中的集群限制

后续步骤

- 部署集群

6.3.12.7. 在具有 Wavelength Zone 子网的现有 VPC 上安装集群

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

在 AWS 上安装集群到现有的 VPC 中，需要使用 AWS Wavelength 区域将计算节点扩展到 Cloud Infrastructure 的边缘。

您可以使用提供的 CloudFormation 模板来创建网络资源。另外，您可以修改模板来自定义模板，或使用其包含的信息根据公司的策略创建 AWS 资源。

重要

执行安装程序置备的基础架构安装的步骤仅作为示例。在现有 VPC 上安装集群需要了解云供应商和 OpenShift Container Platform 安装过程。您可以使用 CloudFormation 模板来帮助您完成这些步骤，或者帮助您建模您自己的集群安装。您可以决定使用其他方法生成这些资源，而不使用 CloudFormation 模板来创建资源。

6.3.12.7.1. 在 AWS 中创建 VPC

您可以在 OpenShift Container Platform 集群的 Amazon Web Services (AWS) 中为所有 Wavelength 区域位置创建一个 Virtual Private Cloud (VPC) 和子网，以将计算节点扩展到边缘位置。您可以进一步自定义 VPC 以满足您的要求，包括 VPN 和路由表。您还可以添加新的 Wavelength 区域子网，不包含在初始部署中。

您可以使用提供的 CloudFormation 模板和自定义参数文件创建代表 VPC 的 AWS 资源堆栈。

注意

如果不使用提供的 CloudFormation 模板来创建 AWS 基础架构，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 已配置了一个 AWS 帐户。
- 您可以通过运行 **aws configure**，将 AWS 密钥和 AWS 区域添加到本地 AWS 配置集中。
- 您可以选择 AWS 帐户上的 AWS Wavelength 区域。

流程

1. 创建一个 JSON 文件，其包含 CloudFormation 模板需要的参数值：

```
[
  {
    "ParameterKey": "VpcCidr", 1
    "ParameterValue": "10.0.0.0/16" 2
  },
  {
    "ParameterKey": "AvailabilityZoneCount", 3
    "ParameterValue": "3" 4
  },
  {
    "ParameterKey": "SubnetBits", 5
    "ParameterValue": "12" 6
  }
]
```

1. VPC 的 CIDR 块。
2. 以 **x.x.x.x/16-24** 格式指定 CIDR 块。
3. 在其中部署 VPC 的可用区的数量。
4. 指定一个 **1** 到 **3** 之间的整数。
5. 各个可用区中每个子网的大小。
6. 指定 **5** 到 **13** 之间的整数，其中 **5** 为 **/27**，**13** 为 **/19**。

2. 进入名为 "CloudFormation template for the VPC" 的文档部分，然后从提供的模板中复制语法。将复制的模板语法保存为本地系统中的 YAML 文件。此模板描述了集群所需的 VPC。

3. 运行以下命令，启动 CloudFormation 模板以创建代表 VPC 的 AWS 资源堆栈：

重要

您必须在一行内输入命令。

```
$ aws cloudformation create-stack --stack-name <name> \ 1
    --template-body file://<template>.yaml \ 2
    --parameters file://<parameters>.json 3
```

1. **<name>** 是 CloudFormation 堆栈的名称，如 **cluster-VPC**。如果您删除集群，则需要此堆栈的名称。
2. **<template>** 是您保存的 CloudFormation 模板 YAML 文件的相对路径和名称。
3. **<parameters>** 是相对路径和 CloudFormation 参数 JSON 文件的名称。

输出示例

```
arn:aws:cloudformation:us-east-1:123456789012:stack/cluster-vpc/dbedae40-2fd3-11eb-820e-12a48460849f
```

4. 运行以下命令确认模板组件已存在：

```
$ aws cloudformation describe-stacks --stack-name <name>
```

在 **StackStatus** 显示 **CREATE_COMPLETE** 后，输出会显示以下参数的值。您必须为创建集群运行的其他 CloudFormation 模板提供这些参数值。

VpcId	您的 VPC ID。
PublicSubnetIds	新公共子网的 ID。
PrivateSubnetIds	新专用子网的 ID。
PublicRouteTableId	新公共路由表 ID 的 ID。

6.3.12.7.2. VPC 的 CloudFormation 模板

您可以使用以下 CloudFormation 模板来部署 OpenShift Container Platform 集群所需的 VPC。

例 6.52. VPC 的 CloudFormation 模板

```
AWSTemplateFormatVersion: 2010-09-09
Description: Template for Best Practice VPC with 1-3 AZs

Parameters:
```

```
  VpcCidr:
    AllowedPattern: ^(([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])\.){3}([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-
4][0-9]|25[0-5])(\/(1[6-9]|2[0-4]))$
    ConstraintDescription: CIDR block parameter must be in the form x.x.x.x/16-24.
    Default: 10.0.0.0/16
    Description: CIDR block for VPC.
    Type: String
  AvailabilityZoneCount:
    ConstraintDescription: "The number of availability zones. (Min: 1, Max: 3)"
    MinValue: 1
    MaxValue: 3
    Default: 1
    Description: "How many AZs to create VPC subnets for. (Min: 1, Max: 3)"
    Type: Number
  SubnetBits:
    ConstraintDescription: CIDR block parameter must be in the form x.x.x.x/19-27.
    MinValue: 5
    MaxValue: 13
    Default: 12
    Description: "Size of each subnet to create within the availability zones. (Min: 5 = /27, Max: 13 =
/19)"
    Type: Number

Metadata:
  AWS::CloudFormation::Interface:
    ParameterGroups:
    - Label:
        default: "Network Configuration"
      Parameters:
      - VpcCidr
      - SubnetBits
    - Label:
        default: "Availability Zones"
      Parameters:
      - AvailabilityZoneCount
    ParameterLabels:
      AvailabilityZoneCount:
        default: "Availability Zone Count"
      VpcCidr:
        default: "VPC CIDR"
      SubnetBits:
        default: "Bits Per Subnet"

Conditions:
  DoAz3: !Equals [3, !Ref AvailabilityZoneCount]
  DoAz2: !Or [!Equals [2, !Ref AvailabilityZoneCount], Condition: DoAz3]

Resources:
  VPC:
    Type: "AWS::EC2::VPC"
    Properties:
      EnableDnsSupport: "true"
      EnableDnsHostnames: "true"
      CidrBlock: !Ref VpcCidr
  PublicSubnet:
    Type: "AWS::EC2::Subnet"
```

```
    Properties:
      VpcId: !Ref VPC
      CidrBlock: !Select [0, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
      AvailabilityZone: !Select
      - 0
      - Fn::GetAZs: !Ref "AWS::Region"
  PublicSubnet2:
    Type: "AWS::EC2::Subnet"
    Condition: DoAz2
    Properties:
      VpcId: !Ref VPC
      CidrBlock: !Select [1, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
      AvailabilityZone: !Select
      - 1
      - Fn::GetAZs: !Ref "AWS::Region"
  PublicSubnet3:
    Type: "AWS::EC2::Subnet"
    Condition: DoAz3
    Properties:
      VpcId: !Ref VPC
      CidrBlock: !Select [2, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
      AvailabilityZone: !Select
      - 2
      - Fn::GetAZs: !Ref "AWS::Region"
  InternetGateway:
    Type: "AWS::EC2::InternetGateway"
  GatewayToInternet:
    Type: "AWS::EC2::VPCGatewayAttachment"
    Properties:
      VpcId: !Ref VPC
      InternetGatewayId: !Ref InternetGateway
  PublicRouteTable:
    Type: "AWS::EC2::RouteTable"
    Properties:
      VpcId: !Ref VPC
  PublicRoute:
    Type: "AWS::EC2::Route"
    DependsOn: GatewayToInternet
    Properties:
      RouteTableId: !Ref PublicRouteTable
      DestinationCidrBlock: 0.0.0.0/0
      GatewayId: !Ref InternetGateway
  PublicSubnetRouteTableAssociation:
    Type: "AWS::EC2::SubnetRouteTableAssociation"
    Properties:
      SubnetId: !Ref PublicSubnet
      RouteTableId: !Ref PublicRouteTable
  PublicSubnetRouteTableAssociation2:
    Type: "AWS::EC2::SubnetRouteTableAssociation"
    Condition: DoAz2
    Properties:
      SubnetId: !Ref PublicSubnet2
      RouteTableId: !Ref PublicRouteTable
  PublicSubnetRouteTableAssociation3:
    Condition: DoAz3
    Type: "AWS::EC2::SubnetRouteTableAssociation"
```

```
  Properties:
    SubnetId: !Ref PublicSubnet3
    RouteTableId: !Ref PublicRouteTable
PrivateSubnet:
  Type: "AWS::EC2::Subnet"
  Properties:
    VpcId: !Ref VPC
    CidrBlock: !Select [3, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
    AvailabilityZone: !Select
    - 0
    - Fn::GetAZs: !Ref "AWS::Region"
PrivateRouteTable:
  Type: "AWS::EC2::RouteTable"
  Properties:
    VpcId: !Ref VPC
PrivateSubnetRouteTableAssociation:
  Type: "AWS::EC2::SubnetRouteTableAssociation"
  Properties:
    SubnetId: !Ref PrivateSubnet
    RouteTableId: !Ref PrivateRouteTable
NAT:
  DependsOn:
  - GatewayToInternet
  Type: "AWS::EC2::NatGateway"
  Properties:
    AllocationId:
      "Fn::GetAtt":
      - EIP
      - AllocationId
    SubnetId: !Ref PublicSubnet
EIP:
  Type: "AWS::EC2::EIP"
  Properties:
    Domain: vpc
Route:
  Type: "AWS::EC2::Route"
  Properties:
    RouteTableId:
      Ref: PrivateRouteTable
    DestinationCidrBlock: 0.0.0.0/0
    NatGatewayId:
      Ref: NAT
PrivateSubnet2:
  Type: "AWS::EC2::Subnet"
  Condition: DoAz2
  Properties:
    VpcId: !Ref VPC
    CidrBlock: !Select [4, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
    AvailabilityZone: !Select
    - 1
    - Fn::GetAZs: !Ref "AWS::Region"
PrivateRouteTable2:
  Type: "AWS::EC2::RouteTable"
  Condition: DoAz2
  Properties:
    VpcId: !Ref VPC
```

```
  PrivateSubnetRouteTableAssociation2:
    Type: "AWS::EC2::SubnetRouteTableAssociation"
    Condition: DoAz2
    Properties:
      SubnetId: !Ref PrivateSubnet2
      RouteTableId: !Ref PrivateRouteTable2
  NAT2:
    DependsOn:
    - GatewayToInternet
    Type: "AWS::EC2::NatGateway"
    Condition: DoAz2
    Properties:
      AllocationId:
        "Fn::GetAtt":
        - EIP2
        - AllocationId
      SubnetId: !Ref PublicSubnet2
  EIP2:
    Type: "AWS::EC2::EIP"
    Condition: DoAz2
    Properties:
      Domain: vpc
  Route2:
    Type: "AWS::EC2::Route"
    Condition: DoAz2
    Properties:
      RouteTableId:
        Ref: PrivateRouteTable2
      DestinationCidrBlock: 0.0.0.0/0
      NatGatewayId:
        Ref: NAT2
  PrivateSubnet3:
    Type: "AWS::EC2::Subnet"
    Condition: DoAz3
    Properties:
      VpcId: !Ref VPC
      CidrBlock: !Select [5, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
      AvailabilityZone: !Select
      - 2
      - Fn::GetAZs: !Ref "AWS::Region"
  PrivateRouteTable3:
    Type: "AWS::EC2::RouteTable"
    Condition: DoAz3
    Properties:
      VpcId: !Ref VPC
  PrivateSubnetRouteTableAssociation3:
    Type: "AWS::EC2::SubnetRouteTableAssociation"
    Condition: DoAz3
    Properties:
      SubnetId: !Ref PrivateSubnet3
      RouteTableId: !Ref PrivateRouteTable3
  NAT3:
    DependsOn:
    - GatewayToInternet
    Type: "AWS::EC2::NatGateway"
    Condition: DoAz3
```

```
    Properties:
      AllocationId:
        "Fn::GetAtt":
        - EIP3
        - AllocationId
      SubnetId: !Ref PublicSubnet3
  EIP3:
    Type: "AWS::EC2::EIP"
    Condition: DoAz3
    Properties:
      Domain: vpc
  Route3:
    Type: "AWS::EC2::Route"
    Condition: DoAz3
    Properties:
      RouteTableId:
        Ref: PrivateRouteTable3
      DestinationCidrBlock: 0.0.0.0/0
      NatGatewayId:
        Ref: NAT3
  S3Endpoint:
    Type: AWS::EC2::VPCEndpoint
    Properties:
      PolicyDocument:
        Version: 2012-10-17
        Statement:
        - Effect: Allow
          Principal: '*'
          Action:
          - '*'
          Resource:
          - '*'
      RouteTableIds:
      - !Ref PublicRouteTable
      - !Ref PrivateRouteTable
      - !If [DoAz2, !Ref PrivateRouteTable2, !Ref "AWS::NoValue"]
      - !If [DoAz3, !Ref PrivateRouteTable3, !Ref "AWS::NoValue"]
      ServiceName: !Join
      - ''
      - - com.amazonaws.
        - !Ref 'AWS::Region'
        - .s3
      VpcId: !Ref VPC

Outputs:
  VpcId:
    Description: ID of the new VPC.
    Value: !Ref VPC
  PublicSubnetIds:
    Description: Subnet IDs of the public subnets.
    Value:
      !Join [
        ",",
        [!Ref PublicSubnet, !If [DoAz2, !Ref PublicSubnet2, !Ref "AWS::NoValue"], !If [DoAz3, !Ref
PublicSubnet3, !Ref "AWS::NoValue"]]
      ]
```

```
  PrivateSubnetIds:
    Description: Subnet IDs of the private subnets.
    Value:
      !Join [
        ",",
        [!Ref PrivateSubnet, !If [DoAz2, !Ref PrivateSubnet2, !Ref "AWS::NoValue"], !If [DoAz3, !Ref
PrivateSubnet3, !Ref "AWS::NoValue"]]
      ]
  PublicRouteTableId:
    Description: Public Route table ID
    Value: !Ref PublicRouteTable
  PrivateRouteTableIds:
    Description: Private Route table IDs
    Value:
      !Join [
        ",",
        [
          !Join ["=", [
            !Select [0, "Fn::GetAZs": !Ref "AWS::Region"],
            !Ref PrivateRouteTable
          ]],
          !If [DoAz2,
              !Join ["=", [!Select [1, "Fn::GetAZs": !Ref "AWS::Region"], !Ref PrivateRouteTable2]],
              !Ref "AWS::NoValue"
          ],
          !If [DoAz3,
              !Join ["=", [!Select [2, "Fn::GetAZs": !Ref "AWS::Region"], !Ref PrivateRouteTable3]],
              !Ref "AWS::NoValue"
          ]
        ]
      ]
```

6.3.12.7.3. 创建 VPC 载体网关

要在 Wavelength 区域上运行的 OpenShift Container Platform 集群中使用公共子网，您必须创建载体网关，并将载体网关关联到 VPC。子网可用于部署负载均衡器或边缘计算节点。

要在 OpenShift Container Platform 集群的 Wavelength 区域位置创建边缘节点或面向互联网的负载均衡器，您必须创建以下所需的网络组件：

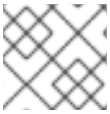
- 与现有 VPC 关联的载体网关。
- 列出路由条目的载波路由表。
- 与载体路由表关联的子网。

对于仅包含 Wavelength 区中的子网的 VPC 存在载体网关。

以下列表解释了 AWS Wavelength 区域位置上下文中的载体网关的功能：

- 提供 Wavelength Zone 和 carrier 网络之间的连接，其中包括来自载体网络的任何可用设备。
- 执行网络地址转换(NAT)功能，如将作为网络边框组的公共 IP 地址（从 Wavelength 区域转换为载体 IP 地址）的 IP 地址转换。这些转换功能适用于入站和出站流量。

- 授权来自位于特定位置的载体网络的入站流量。
- 授权到载波网络和互联网的出站流量。



注意

互联网没有入站连接配置，通过载体网关到 Wavelength 区域。

您可以使用提供的 CloudFormation 模板来创建以下 AWS 资源堆栈：

- 一个载体网关，与模板中的 VPC ID 关联。
- 一个用于 Wavelength Zone 的公共路由表，名为 **<ClusterName>-public-carrier**。
- 以载体网关为目标的新路由表中的默认 IPv4 路由条目。
- AWS Simple Storage Service (S3) 的 VPC 网关端点。

注意

如果不使用提供的 CloudFormation 模板来创建 AWS 基础架构，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 已配置了一个 AWS 帐户。
- 您可以通过运行 **aws configure**，将 AWS 密钥和区域添加到本地 AWS 配置集中。

流程

1. 进入名为"CloudFormation template for the VPC Carrier Gateway"的文档的下一部分，然后复制 **VPC Carrier Gateway 模板的 CloudFormation**模板的语法。将复制的模板语法保存为本地系统中的 YAML 文件。此模板描述了集群所需的 VPC。
2. 运行以下命令来部署 CloudFormation 模板，它会创建一个代表 VPC 的 AWS 资源堆栈：

```
$ aws cloudformation create-stack --stack-name <stack_name> \ 1
  --region ${CLUSTER_REGION} \
  --template-body file://<template>.yaml \ 2
  --parameters \//
    ParameterKey=VpcId,ParameterValue="${VpcId}" \ 3
    ParameterKey=ClusterName,ParameterValue="${ClusterName}" 4
```

1. **<stack_name>** 是 CloudFormation 堆栈的名称，如 **clusterName-vpc-carrier-gw**。如果您删除集群，则需要此堆栈的名称。
2. **<template>** 是相对路径，以及保存的 CloudFormation 模板 YAML 文件的名称。
3. **<VpcId>** 是从名为"Creating a VPC in AWS"部分创建的 CloudFormation 堆栈输出中提取的 VPC ID。
4.

<clusterName> 是一个自定义值，前缀为 CloudFormation 堆栈创建的资源的前缀。您可以使用 **install-config.yaml** 配置文件的 **metadata.name** 部分中定义的相同名称。

输出示例

```
arn:aws:cloudformation:us-east-1:123456789012:stack/<stack_name>/dbedae40-2fd3-11eb-820e-12a48460849f
```

验证

- 运行以下命令确认 CloudFormation 模板组件已存在：

```
$ aws cloudformation describe-stacks --stack-name <stack_name>
```

在 **StackStatus** 显示 **CREATE_COMPLETE** 后，输出显示以下参数的值：确保为集群运行的其他 CloudFormation 模板提供参数值。

PublicRouteTableId	Carrier 基础架构中路由表 ID。

其他资源

- 请参阅 AWS 文档中的 Amazon S3。

6.3.12.7.4. VPC Carrier Gateway 的 CloudFormation 模板

您可以使用以下 CloudFormation 模板，在 AWS Wavelength 基础架构上部署 Carrier Gateway。

例 6.53. VPC Carrier Gateway 的 CloudFormation 模板

```
AWSTemplateFormatVersion: 2010-09-09
Description: Template for Creating Wavelength Zone Gateway (Carrier Gateway).

Parameters:
  VpcId:
    Description: VPC ID to associate the Carrier Gateway.
    Type: String
    AllowedPattern: ^(?:(?:vpc)(?:-[a-zA-Z0-9]+)?\b|(?:[0-9]{1,3}\.){3}[0-9]{1,3})$
    ConstraintDescription: VPC ID must be with valid name, starting with vpc-.*.
  ClusterName:
    Description: Cluster Name or Prefix name to prepend the tag Name for each subnet.
    Type: String
    AllowedPattern: ".+"
    ConstraintDescription: ClusterName parameter must be specified.

Resources:
  CarrierGateway:
    Type: "AWS::EC2::CarrierGateway"
    Properties:
      VpcId: !Ref VpcId
      Tags:
      - Key: Name
```

```
        Value: !Join ['-', [!Ref ClusterName, "cagw"]]

    PublicRouteTable:
      Type: "AWS::EC2::RouteTable"
      Properties:
        VpcId: !Ref VpcId
        Tags:
        - Key: Name
          Value: !Join ['-', [!Ref ClusterName, "public-carrier"]]

    PublicRoute:
      Type: "AWS::EC2::Route"
      DependsOn: CarrierGateway
      Properties:
        RouteTableId: !Ref PublicRouteTable
        DestinationCidrBlock: 0.0.0.0/0
        CarrierGatewayId: !Ref CarrierGateway

    S3Endpoint:
      Type: AWS::EC2::VPCEndpoint
      Properties:
        PolicyDocument:
          Version: 2012-10-17
          Statement:
          - Effect: Allow
            Principal: '*'
            Action:
            - '*'
            Resource:
            - '*'
        RouteTableIds:
        - !Ref PublicRouteTable
        ServiceName: !Join
        - ''
        - - com.amazonaws.
          - !Ref 'AWS::Region'
          - .s3
        VpcId: !Ref VpcId

    Outputs:
      PublicRouteTableId:
        Description: Public Route table ID
        Value: !Ref PublicRouteTable
```

6.3.12.7.5. 在 Wavelength 区域中创建子网

在 OpenShift Container Platform 集群中为边缘计算节点配置机器集前，您必须在 Wavelength Zones 中创建子网。对您要将计算节点部署到的每个 Wavelength 区完成以下步骤。

您可以使用提供的 CloudFormation 模板并创建 CloudFormation 堆栈。然后，您可以使用此堆栈自定义置备子网。

注意

如果不使用提供的 CloudFormation 模板来创建 AWS 基础架构，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 已配置了一个 AWS 帐户。
- 您可以通过运行 **aws configure**，将 AWS 密钥和区域添加到本地 AWS 配置集中。
- 您可以选择 Wavelength Zones 组。

流程

1. 进入名为"CloudFormation template for the VPC 子网"的文档部分，并从模板中复制语法。将复制的模板语法保存为本地系统中的 YAML 文件。此模板描述了集群所需的 VPC。
2. 运行以下命令来部署 CloudFormation 模板，它会创建一个代表 VPC 的 AWS 资源堆栈：

```
$ aws cloudformation create-stack --stack-name <stack_name> \ 1
  --region ${CLUSTER_REGION} \
  --template-body file://<template>.yaml \ 2
  --parameters \
    ParameterKey=VpcId,ParameterValue="${VPC_ID}" \ 3
    ParameterKey=ClusterName,ParameterValue="${CLUSTER_NAME}" \ 4
    ParameterKey=ZoneName,ParameterValue="${ZONE_NAME}" \ 5
    ParameterKey=PublicRouteTableId,ParameterValue="${ROUTE_TABLE_PUB}" \ 6
    ParameterKey=PublicSubnetCidr,ParameterValue="${SUBNET_CIDR_PUB}" \ 7
    ParameterKey=PrivateRouteTableId,ParameterValue="${ROUTE_TABLE_PVT}" \ 8
    ParameterKey=PrivateSubnetCidr,ParameterValue="${SUBNET_CIDR_PVT}" 9
```

1. **<stack_name>** 是 CloudFormation 堆栈的名称，如 **cluster-wl-<wavelength_zone_shortname>**。如果您删除集群，则需要此堆栈的名称。
2. **<template>** 是相对路径，以及保存的 CloudFormation 模板 YAML 文件的名称。
3. **${VPC_ID}** 是 VPC ID，它是 VPC 模板输出中的 **VpcID** 值。
4. **${ZONE_NAME}** 是用于创建子网的 Wavelength Zones 名称的值。
5. **${CLUSTER_NAME}** 是 **ClusterName** 的值，用作新 AWS 资源名称的前缀。
6. **${ROUTE_TABLE_PUB}** 是从 VPC 的载体网关 CloudFormation 堆栈的输出中提取的 **PublicRouteTableId**。
7. **${SUBNET_CIDR_PUB}** 是一个有效的 CIDR 块，用于创建公共子网。这个块必须是 VPC CIDR 块 **VpcCidr** 的一部分。
8. **${ROUTE_TABLE_PVT}** 是从 VPC 的 CloudFormation 堆栈的输出中提取的 **PrivateRouteTableId**。
9. **${SUBNET_CIDR_PVT}** 是一个有效的 CIDR 块，用于创建专用子网。这个块必须是 VPC CIDR 块 **VpcCidr** 的一部分。

输出示例

```
arn:aws:cloudformation:us-east-1:123456789012:stack/<stack_name>/dbedae40-820e-11eb-2fd3-12a48460849f
```

验证

- 运行以下命令确认模板组件已存在：

  ```
  $ aws cloudformation describe-stacks --stack-name <stack_name>
  ```

  在 **StackStatus** 显示 **CREATE_COMPLETE** 后，输出会显示以下参数的值。确保将这些参数值提供给您为集群创建的其他 CloudFormation 模板。

PublicSubnetId	由 CloudFormation 堆栈创建的公共子网的 ID。
PrivateSubnetId	由 CloudFormation 堆栈创建的专用子网的 ID。

6.3.12.7.6. VPC 子网的 CloudFormation 模板

您可以使用以下 CloudFormation 模板，在 Wavelength 区域基础架构上部署私有和公共子网。

例 6.54. VPC 子网的 CloudFormation 模板

```
AWSTemplateFormatVersion: 2010-09-09
Description: Template for Best Practice Subnets (Public and Private)

Parameters:
  VpcId:
    Description: VPC ID that comprises all the target subnets.
    Type: String
    AllowedPattern: ^(?:(?:vpc)(?:-[a-zA-Z0-9]+)?\b|(?:[0-9]{1,3}\.){3}[0-9]{1,3})$
    ConstraintDescription: VPC ID must be with valid name, starting with vpc-.*.
  ClusterName:
    Description: Cluster name or prefix name to prepend the Name tag for each subnet.
    Type: String
    AllowedPattern: ".+"
    ConstraintDescription: ClusterName parameter must be specified.
  ZoneName:
    Description: Zone Name to create the subnets, such as us-west-2-lax-1a.
    Type: String
    AllowedPattern: ".+"
    ConstraintDescription: ZoneName parameter must be specified.
  PublicRouteTableId:
    Description: Public Route Table ID to associate the public subnet.
    Type: String
    AllowedPattern: ".+"
    ConstraintDescription: PublicRouteTableId parameter must be specified.
  PublicSubnetCidr:
    AllowedPattern: ^(([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])\.){3}([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])(\/(1[6-9]|2[0-4]))$
```

```
    ConstraintDescription: CIDR block parameter must be in the form x.x.x.x/16-24.
    Default: 10.0.128.0/20
    Description: CIDR block for public subnet.
    Type: String
  PrivateRouteTableId:
    Description: Private Route Table ID to associate the private subnet.
    Type: String
    AllowedPattern: ".+"
    ConstraintDescription: PrivateRouteTableId parameter must be specified.
  PrivateSubnetCidr:
    AllowedPattern: ^(([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])\.){3}([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-
4][0-9]|25[0-5])(\/(1[6-9]|2[0-4]))$
    ConstraintDescription: CIDR block parameter must be in the form x.x.x.x/16-24.
    Default: 10.0.128.0/20
    Description: CIDR block for private subnet.
    Type: String


Resources:
  PublicSubnet:
    Type: "AWS::EC2::Subnet"
    Properties:
      VpcId: !Ref VpcId
      CidrBlock: !Ref PublicSubnetCidr
      AvailabilityZone: !Ref ZoneName
      Tags:
      - Key: Name
        Value: !Join ['-', [!Ref ClusterName, "public", !Ref ZoneName]]

  PublicSubnetRouteTableAssociation:
    Type: "AWS::EC2::SubnetRouteTableAssociation"
    Properties:
      SubnetId: !Ref PublicSubnet
      RouteTableId: !Ref PublicRouteTableId

  PrivateSubnet:
    Type: "AWS::EC2::Subnet"
    Properties:
      VpcId: !Ref VpcId
      CidrBlock: !Ref PrivateSubnetCidr
      AvailabilityZone: !Ref ZoneName
      Tags:
      - Key: Name
        Value: !Join ['-', [!Ref ClusterName, "private", !Ref ZoneName]]

  PrivateSubnetRouteTableAssociation:
    Type: "AWS::EC2::SubnetRouteTableAssociation"
    Properties:
      SubnetId: !Ref PrivateSubnet
      RouteTableId: !Ref PrivateRouteTableId

Outputs:
  PublicSubnetId:
    Description: Subnet ID of the public subnets.
    Value:
      !Join ["", [!Ref PublicSubnet]]
```

```
PrivateSubnetId:
  Description: Subnet ID of the private subnets.
  Value:
    !Join ["", [!Ref PrivateSubnet]]
```

6.3.12.7.7. 修改安装配置文件以使用 AWS Wavelength 区域子网

修改 **install-config.yaml** 文件，使其包含 Wavelength Zones 子网。

先决条件

- 您使用"在 Wavelength Zones 中创建子网"流程创建子网。
- 您使用"创建安装配置文件"流程创建了 **install-config.yaml** 文件。

流程

- 通过在 **platform.aws.subnets** 参数中指定 Wavelength Zones 子网来修改 **install-config.yaml** 配置文件。

  带有 Wavelength 区域子网的安装配置文件示例

```
# ...
platform:
  aws:
    region: us-west-2
    subnets: 1
    - publicSubnetId-1
    - publicSubnetId-2
    - publicSubnetId-3
    - privateSubnetId-1
    - privateSubnetId-2
    - privateSubnetId-3
    - publicOrPrivateSubnetID-Wavelength-1
# ...
```

1 区域中创建的子网 ID 列表： Availability 和 Wavelength 区域。

其他资源

- 有关查看您创建的 CloudFormation 堆栈的更多信息，请参阅 AWS CloudFormation 控制台。
- 如需有关 AWS 配置集和凭证配置的更多信息，请参阅 AWS 文档中的配置和凭证文件设置。

后续步骤

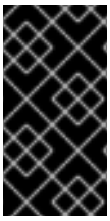
- 部署集群

6.3.12.8. 可选：将公共 IP 地址分配给边缘计算节点

如果您的工作负载需要在 Wavelength 区域基础架构的公共子网中部署边缘计算节点，您可以在安装集群时配置机器集清单。

AWS Wavelength Zones 基础架构访问指定区中的网络流量，因此应用程序可在提供更接近该区的最终用户时利用较低延迟。

在私有子网中部署计算节点的默认设置可能无法满足您的需求，因此当您想要将更多自定义应用到基础架构时，请考虑在公共子网中创建边缘计算节点。



重要

默认情况下，OpenShift Container Platform 在私有子网中部署计算节点。为获得最佳性能，请考虑将计算节点放在附加了其公共 IP 地址的子网中。

您必须创建额外的安全组，但请确保仅在需要时通过互联网打开组规则。

流程

1. 进入包含安装程序的目录并生成清单文件。确保安装清单在 **openshift** 和 **manifests** 目录级别创建。

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

2. 编辑安装程序为 Wavelength Zones 生成的机器集清单，以便清单部署在公共子网中。为 **spec.template.spec.providerSpec.value.publicIP** 参数指定 **true**。

用于快速安装集群的机器集清单配置示例

```
spec:
  template:
    spec:
      providerSpec:
        value:
          publicIp: true
          subnet:
            filters:
              - name: tag:Name
                values:
                  - ${INFRA_ID}-public-${ZONE_NAME}
```

在具有 Wavelength 区域子网的现有 VPC 上安装集群的机器集清单配置示例

```
apiVersion: machine.openshift.io/v1beta1
kind: MachineSet
metadata:
  name: <infrastructure_id>-edge-<zone>
  namespace: openshift-machine-api
spec:
  template:
    spec:
      providerSpec:
        value:
          publicIp: true
```

6.3.12.9. 部署集群

您可以在兼容云平台上安装 OpenShift Container Platform。

重要

在初始安装过程中，您只能运行安装程序的 **create cluster** 命令一次。

先决条件

- 您已使用托管集群的云平台配置了帐户。
- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。
- 已确认主机上的云供应商帐户具有部署集群的正确权限。权限不正确的帐户会导致安装过程失败，并显示包括缺失权限的错误消息。

流程

1. 进入包含安装程序的目录并初始化集群部署：

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

1. 对于 **<installation_directory>**，请指定自定义 **./install-config.yaml** 文件的位置。
2. 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不是 **info**。

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

注意

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

验证

当集群部署成功完成时：

- 终端会显示用于访问集群的说明，包括指向 Web 控制台和 **kubeadmin** 用户的凭证的链接。
- 凭证信息还会输出到 **<installation_directory>/.openshift_install.log**.

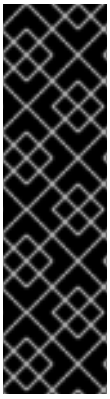
重要

不要删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

输出示例

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

```
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

6.3.12.10. 验证部署集群的状态

验证 OpenShift Container Platform 是否在 AWS Wavelength Zones 上成功部署。

6.3.12.10.1. 使用 CLI 登录集群

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

先决条件

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

流程

1. 导出 **kubeadmin** 凭证：

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

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

2. 验证您可以使用导出的配置成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

6.3.12.10.2. 使用 Web 控制台登录到集群

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

先决条件

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

流程

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

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

注意

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

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

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

注意

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

输出示例

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

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

其他资源

- 访问Web控制台

6.3.12.10.3. 验证使用边缘计算池创建的节点

安装使用 AWS Wavelength Zones 基础架构的集群后，检查安装过程中由机器集清单创建的机器状态。

1. 要检查从添加到 **install-config.yaml** 文件中的子网中创建的机器集，请运行以下命令：

```
$ oc get machineset -n openshift-machine-api
```

输出示例

```
NAME                                 DESIRED   CURRENT   READY   AVAILABLE   AGE
cluster-7xw5g-edge-us-east-1-wl1-nyc-wlz-1   1         1         1       1           3h4m
cluster-7xw5g-worker-us-east-1a              1         1         1       1           3h4m
cluster-7xw5g-worker-us-east-1b              1         1         1       1           3h4m
cluster-7xw5g-worker-us-east-1c              1         1         1       1           3h4m
```

2. 要检查从机器集创建的机器，请运行以下命令：

```
$ oc get machines -n openshift-machine-api
```

输出示例

```
NAME                                      PHASE     TYPE        REGION      ZONE         AGE
cluster-7xw5g-edge-us-east-1-wl1-nyc-wlz-1-wbclh  Running   c5d.2xlarge  us-east-1   us-east-1-wl1-nyc-wlz-1  3h
cluster-7xw5g-master-0                    Running   m6i.xlarge   us-east-1   us-east-1a   3h4m
cluster-7xw5g-master-1                    Running   m6i.xlarge   us-east-1   us-east-1b   3h4m
cluster-7xw5g-master-2                    Running   m6i.xlarge   us-east-1   us-east-1c   3h4m
cluster-7xw5g-worker-us-east-1a-rtp45     Running   m6i.xlarge   us-east-1   us-east-1a   3h
cluster-7xw5g-worker-us-east-1b-glm7c     Running   m6i.xlarge   us-east-1   us-east-1b   3h
cluster-7xw5g-worker-us-east-1c-qfvz4     Running   m6i.xlarge   us-east-1   us-east-1c   3h
```

3. 要检查具有边缘角色的节点，请运行以下命令：

```
$ oc get nodes -l node-role.kubernetes.io/edge
```

输出示例

```
NAME                         STATUS   ROLES        AGE   VERSION
ip-10-0-207-188.ec2.internal   Ready    edge,worker   172m   v1.25.2+d2e245f
```

后续步骤

- 验证安装.
- 如果需要，您可以选择 不使用远程健康。

6.3.13. 使用 AWS Outposts 上的计算节点安装集群

在 OpenShift Container Platform 版本 4.14 中，您可以使用 AWS Outposts 中运行的计算节点在 Amazon Web Services (AWS)上安装集群。自 OpenShift Container Platform 版本 4.15 起，不再支持这个安装方法。

相反，您可以在 AWS 上将集群安装到现有的 VPC 中，并在 AWS Outposts 上置备计算节点作为安装后配置任务。

如需更多信息，请参阅将 AWS VPC 集群扩展到 AWS Outpost

6.4. 用户置备的基础架构

6.4.1. 准备在 AWS 上安装集群

您可以通过完成以下步骤准备在 AWS 上安装 OpenShift Container Platform 集群：

- 为集群验证互联网连接。
- 配置 AWS 帐户.
- 下载安装程序。

注意

如果您要在断开连接的环境中安装，您可以从镜像内容中提取安装程序。如需更多信息，请参阅为断开连接的安装镜像镜像。

- 安装 OpenShift CLI (**oc**)。

注意

如果要在断开连接的环境中安装，请将 **oc** 安装到镜像主机上。

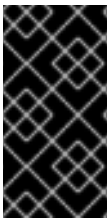
- 生成 SSH 密钥对。您可以在部署后使用此密钥对在 OpenShift Container Platform 集群的节点上进行身份验证。
- 准备用户置备的基础架构。
- 如果环境中无法访问云身份和访问管理(IAM) API，或者不想将管理员级别的凭证 secret 存储在 **kube-system** 命名空间中，请参阅为 AWS 手动创建长期凭证，或将 AWS 集群配置为使用 Amazon Web Services Security Token Service (AWS STS)的短期凭证。

6.4.1.1. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

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



重要

如果您的集群无法直接访问互联网，则可以在置备的某些类型的基础架构上执行受限网络安装。在此过程中，您可以下载所需的内容，并使用它为镜像 registry 填充安装软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群前，您要更新镜像 registry 的内容。

6.4.1.2. 获取安装程序

在安装 OpenShift Container Platform 前，将安装文件下载到您用于安装的主机上。

先决条件

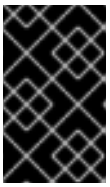
- 您有一台运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB。

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐户，请使用您的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入到安装类型的页面，下载与您的主机操作系统和架构对应的安装程序，并将该文件放在您要存储安装配置文件的目录中。

重要

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



重要

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

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

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

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。此 pull secret 允许您与所含授权机构提供的服务进行身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

6.4.1.3. 安装 OpenShift CLI

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

重要

如果安装了旧版本的 **oc**，则可能无法使用 OpenShift Container Platform 4.16 中的所有命令。下载并安装新版本的 **oc**。

在 Linux 上安装 OpenShift CLI

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

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **产品变体** 下拉列表中选择架构。
3. 从 **版本** 下拉列表中选择适当的版本。
4. 点 **OpenShift v4.16 Linux Client** 条目旁的 **Download Now** 来保存文件。

5. 解包存档：

```
$ tar xvf <file>
```

6. 将 **oc** 二进制文件放到 **PATH 中的目录中**。
要查看您的 **PATH**，请执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

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

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 Windows Client** 条目旁的 **Download Now** 来保存文件。
4. 使用 ZIP 程序解压存档。
5. 将 **oc** 二进制文件移到 **PATH 中的目录中**。
要查看您的 **PATH**，请打开命令提示并执行以下命令：

```
C:\> path
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

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

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

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 macOS Client** 条目旁的 **Download Now** 来保存文件。

注意

对于 macOS arm64，请选择 **OpenShift v4.16 macOS arm64 Client** 条目。

4. 解包和解压存档。

5. 将 **oc** 二进制文件移到 PATH 的目录中。
要查看您的 **PATH**，请打开终端并执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

6.4.1.4. 为集群节点 SSH 访问生成密钥对

在 OpenShift Container Platform 安装过程中，您可以为安装程序提供 SSH 公钥。密钥通过它们的 Ignition 配置文件传递给 Red Hat Enterprise Linux CoreOS(RHCOS)节点，用于验证对节点的 SSH 访问。密钥添加到每个节点上 **core** 用户的 **~/.ssh/authorized_keys** 列表中，这将启用免密码身份验证。

将密钥传递给节点后，您可以使用密钥对作为用户 **核心** 通过 SSH 连接到 RHCOS 节点。若要通过 SSH 访问节点，必须由 SSH 为您的本地用户管理私钥身份。

如果要通过 SSH 连接到集群节点来执行安装调试或灾难恢复，则必须在安装过程中提供 SSH 公钥。**./openshift-install gather** 命令还需要在集群节点上设置 SSH 公钥。

重要

不要在生产环境中跳过这个过程，在生产环境中需要灾难恢复和调试。

注意

您必须使用本地密钥，而不是使用特定平台方法配置 的密钥，如 AWS 密钥对。

流程

1. 如果您在本地计算机上没有可用于在集群节点上进行身份验证的现有 SSH 密钥对，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

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

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_ed25519**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

注意

如果您计划在 **x86_64**、**ppc64le** 和 **s390x** 架构上安装使用 RHEL 加密库（这些加密库已提交给 NIST 用于 FIPS 140-2/140-3 验证）的 OpenShift Container Platform 集群，则不要创建使用 **ed25519** 算法的密钥。相反，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

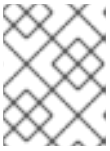
2. 查看公共 SSH 密钥：

```
$ cat <path>/<file_name>.pub
```

例如，运行以下命令来查看 ~/**.ssh/id_ed25519.pub** 公钥：

```
$ cat ~/.ssh/id_ed25519.pub
```

3. 将 SSH 私钥身份添加到本地用户的 SSH 代理（如果尚未添加）。在集群节点上，或者要使用 **./openshift-install gather** 命令，需要对该密钥进行 SSH 代理管理，才能在集群节点上进行免密码 SSH 身份验证。



注意

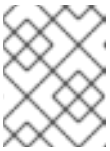
在某些发行版中，自动管理默认 SSH 私钥身份，如 ~/**.ssh/id_rsa** 和 ~/**.ssh/id_dsa**。

a. 如果 **ssh-agent** 进程尚未为您的本地用户运行，请将其作为后台任务启动：

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

输出示例

```
Agent pid 31874
```



注意

如果集群处于 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

4. 将 SSH 私钥添加到 **ssh-agent** ：

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

1 指定 SSH 私钥的路径和文件名，如 ~/**.ssh/id_ed25519.pub**

输出示例

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

后续步骤

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

6.4.1.5. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

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

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

6.4.2. AWS 上用户置备的基础架构安装要求

在您置备的基础架构上开始安装前，请确定您的 AWS 环境满足以下安装要求。

对于包含用户置备的基础架构的集群，您必须部署所有所需的机器。

6.4.2.1. 集群安装所需的机器

最小的 OpenShift Container Platform 集群需要以下主机：

表 6.21. 最低所需的主机

主机	描述
一个临时 bootstrap 机器	集群需要 bootstrap 机器在三台 control plane 机器上部署 OpenShift Container Platform 集群。您可在安装集群后删除 bootstrap 机器。
三台 control plane 机器	control plane 机器运行组成 control plane 的 Kubernetes 和 OpenShift Container Platform 服务。
至少两台计算机器，也称为 worker 机器。	OpenShift Container Platform 用户请求的工作负载在计算机器上运行。

重要

要保持集群的高可用性，请将独立的物理主机用于这些集群机器。

bootstrap 和 control plane 机器必须使用 Red Hat Enterprise Linux CoreOS(RHCOS)作为操作系统。但是，计算机器可以在 Red Hat Enterprise Linux CoreOS(RHCOS)、Red Hat Enterprise Linux(RHEL) 8.6 和更高的版本。

请注意，RHCOS 基于 Red Hat Enterprise Linux(RHEL) 9.2，并继承其所有硬件认证和要求。查看 红帽企业 Linux 技术功能和限制。

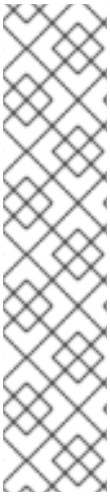
6.4.2.1.1. 集群安装的最低资源要求

每台集群机器都必须满足以下最低要求：

表 6.22. 最低资源要求

机器	操作系统	vCPU [1]	虚拟内存	Storage	每秒输入/输出 (IOPS) [2]
bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane（控制平面）	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS、RHEL 8.6 及更新版本 [3]	2	8 GB	100 GB	300

1. 当未启用并发多线程(SMT)或超线程时，一个 vCPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比例：（每个内核数的线程）× sockets = vCPU。
2. OpenShift Container Platform 和 Kubernetes 对磁盘性能非常敏感，建议使用更快的存储速度，特别是 control plane 节点上需要 10 ms p99 fsync 持续时间的 etcd。请注意，在许多云平台上，存储大小和 IOPS 可一起扩展，因此您可能需要过度分配存储卷来获取足够的性能。
3. 与所有用户置备的安装一样，如果您选择在集群中使用 RHEL 计算机器，则负责所有操作系统生命周期管理和维护，包括执行系统更新、应用补丁和完成所有其他必要的任务。RHEL 7 计算机器的使用已弃用，并已在 OpenShift Container Platform 4.10 及更新的版本中删除。



注意

从 OpenShift Container Platform 版本 4.13 开始，RHCOS 基于 RHEL 版本 9.2，它更新了微架构要求。以下列表包含每个架构需要的最小指令集架构 (ISA)：

- x86-64 体系结构需要 x86-64-v2 ISA
- ARM64 架构需要 ARMv8.0-A ISA
- IBM Power 架构需要 Power 9 ISA
- s390x 架构需要 z14 ISA

如需更多信息，请参阅 RHEL 架构。

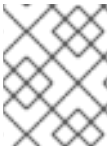
如果平台的实例类型满足集群机器的最低要求，则 OpenShift Container Platform 支持使用它。

其他资源

- 优化存储

6.4.2.1.2. 为 AWS 测试的实例类型

以下 Amazon Web Services(AWS) 实例类型已经过 OpenShift Container Platform 测试。



注意

将以下图中包含的机器类型用于 AWS 实例。如果您使用没有在图表中列出的实例类型，请确保使用的实例大小与名为"最小资源要求"的部分中列出的最少资源要求匹配。

例 6.55. 基于 64 位 x86 架构的机器类型

- **c4.***
- **c5.***
- **c5a.***
- **i3.***
- **m4.***
- **m5.***
- **m5a.***
- **m6a.***
- **m6i.***
- **r4.***
- **r5.***
- **r5a.***
- **r6i.***
- **t3.***
- **t3a.***

6.4.2.1.3. 在 64 位 ARM 基础架构上为 AWS 测试过的实例类型

OpenShift Container Platform 中已经测试了以下 Amazon Web Services (AWS) 64 位 ARM 实例类型。

注意

使用 AWS ARM 实例的以下图中包含的机器类型。如果您使用没有在图中列出的实例类型，请确保使用的实例大小与集群安装"最小资源要求"中列出的最少资源要求匹配。

例 6.56. 基于 64 位 ARM 架构的机器类型

- **c6g.***
- **m6g.***

6.4.2.2. 证书签名请求管理

在使用您置备的基础架构时，集群只能有限地访问自动机器管理，因此您必须提供一种在安装后批准集群证书签名请求 (CSR) 的机制。**kube-controller-manager** 只能批准 kubelet 客户端 CSR。 **machine-approver** 无法保证使用 kubelet 凭证请求的提供证书的有效性，因为它不能确认是正确的机器发出了该请求。您必须决定并实施一种方法，以验证 kubelet 提供证书请求的有效性并进行批准。

6.4.2.3. 所需的 AWS 基础架构组件

要在 Amazon Web Services (AWS) 中用户置备的基础架构上安装 OpenShift Container Platform，您必须手动创建机器及其支持的基础架构。

如需有关不同平台集成测试的更多信息，请参阅 OpenShift Container Platform 4.x Tested Integrations 页面。

通过使用提供的 CloudFormation 模板，您可以创建代表以下组件的 AWS 资源堆栈：

- 一个 AWS Virtual Private Cloud (VPC)
- 网络和负载均衡组件
- 安全组和角色
- 一个 OpenShift Container Platform bootstrap 节点
- OpenShift Container Platform control plane 节点
- 一个 OpenShift Container Platform 计算节点

或者，您可以手动创建组件，也可以重复使用满足集群要求的现有基础架构。查看 CloudFormation 模板，了解组件如何相互连接的更多详情。

6.4.2.3.1. 其他基础架构组件

- VPC
- DNS 条目
- 负载均衡器（典型或网络）和监听器
- 公共和专用路由 53 区域
- 安全组
- IAM 角色
- S3 存储桶

如果您在断开连接的环境中工作，则无法访问 EC2、ELB 和 S3 端点的公共 IP 地址。根据您要在安装过程中限制互联网流量的级别，有以下配置选项：

选项 1：创建 VPC 端点
创建 VPC 端点，并将其附加到集群使用的子网。将端点命名为如下：

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

通过这个选项，网络流量在 VPC 和所需的 AWS 服务之间保持私有。

选项 2：创建一个没有 VPC 端点的代理
作为安装过程的一部分，您可以配置 HTTP 或 HTTPS 代理。使用此选项时，互联网流量会通过代理访问所需的 AWS 服务。

选项 3：创建带有 VPC 端点的代理
作为安装过程的一部分，您可以使用 VPC 端点配置 HTTP 或 HTTPS 代理。创建 VPC 端点，并将其附加到集群使用的子网。将端点命名为如下：

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

在 **install-config.yaml** 文件中配置代理时，将这些端点添加到 **noProxy** 字段。通过这个选项，代理会阻止集群直接访问互联网。但是，您的 VPC 和所需的 AWS 服务之间网络流量保持私有。

所需的 VPC 组件

您必须提供合适的 VPC 和子网，以便与您的机器通信。

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

<table>
<tr><th>组件</th><th>AWS 类型</th><th colspan="2">描述</th></tr>
<tr><td>网络访问控制</td><td rowspan="6">AWS::EC2::NetworkAcl
AWS::EC2::NetworkAclEntry</td><td colspan="2">您必须允许 VPC 访问下列端口：</td></tr>
<tr><td>端口</td><td>原因</td></tr>
<tr><td>80</td><td>入站 HTTP 流量</td></tr>
<tr><td>443</td><td>入站 HTTPS 流量</td></tr>
<tr><td>22</td><td>入站 SSH 流量</td></tr>
<tr><td>1024 - 65535</td><td>入站临时流量</td></tr>
<tr><td>专用子网</td><td>0 - 65535</td><td>出站临时流量</td></tr>
<tr><td>专用子网</td><td>AWS::EC2::Subnet
AWS::EC2::RouteTable
AWS::EC2::SubnetRouteTableAssociation</td><td colspan="2">您的 VPC 可以具有私有子网。提供的 CloudFormation 模板可为 1 到 3 个可用区创建专用子网。如果您使用专用子网，必须为其提供适当的路由和表。</td></tr>
</table>

所需的 DNS 和负载均衡组件

您的 DNS 和负载均衡器配置需要使用公共托管区，并可使用类似安装程序使用的专用托管区（如果安装程序置备了集群的基础架构）。您必须创建一个解析到负载均衡器的 DNS 条目。**api.<cluster_name>.<domain>** 的条目必须指向外部负载均衡器，**api-int.<cluster_name>.<domain>** 的条目则必须指向内部负载均衡器。

集群还需要负载均衡器，以及监听端口 6443（用于 Kubernetes API 及其扩展）和端口 22623（用于新机器的 Ignition 配置文件）的监听程序。目标是 control plane 节点。集群外的客户端和集群内的节点都必须能够访问端口 6443。集群内的节点必须能够访问端口 22623。

组件	AWS 类型	描述
DNS	**AWS::Route53::HostedZone**	内部 DNS 的托管区。
公共负载均衡器	**AWS::ElasticLoadBalancingV2::LoadBalancer**	公共子网的负载均衡器。
外部 API 服务器记录	**AWS::Route53::RecordSetGroup**	外部 API 服务器的别名记录。

组件	AWS 类型	描述
外部监听程序	**AWS::ElasticLoadBalancingV2::Listener**	为外部负载均衡器监听端口 6443 的监听程序。
外部目标组	**AWS::ElasticLoadBalancingV2::TargetGroup**	外部负载均衡器的目标组。
专用负载均衡器	**AWS::ElasticLoadBalancingV2::LoadBalancer**	专用子网的负载均衡器。
内部 API 服务器记录	**AWS::Route53::RecordSetGroup**	内部 API 服务器的别名记录。
内部监听程序	**AWS::ElasticLoadBalancingV2::Listener**	为内部负载均衡器监听端口 22623 的监听程序。
内部目标组	**AWS::ElasticLoadBalancingV2::TargetGroup**	内部负载均衡器的目标组。
内部监听程序	**AWS::ElasticLoadBalancingV2::Listener**	为内部负载均衡器监听端口 6443 的监听程序。
内部目标组	**AWS::ElasticLoadBalancingV2::TargetGroup**	内部负载均衡器的目标组。

安全组

control plane 和 worker 机器需要访问下列端口：

组	类型	IP 协议	端口范围
MasterSecurityGroup	**AWS::EC2::SecurityGroup**	**icmp**	**0**
		tcp	**22**

组	类型	IP 协议	端口范围
		tcp	6443
		tcp	22623
WorkerSecurityGroup	AWS::EC2::SecurityGroup	icmp	0
		tcp	22
BootstrapSecurityGroup	AWS::EC2::SecurityGroup	tcp	22
		tcp	19531

control plane 入口

control plane 机器需要以下入口组。每个入口组都是 **AWS::EC2::SecurityGroupIngress** 资源。

入口组	描述	IP 协议	端口范围
MasterIngressEtcd	etcd	tcp	2379- 2380
MasterIngressVxlan	Vxlan 数据包	udp	4789
MasterIngressWorkerVxlan	Vxlan 数据包	udp	4789
MasterIngressInternal	内部集群通信和 Kubernetes 代理指标	tcp	9000 - 9999
MasterIngressWorkerInternal	内部集群通信	tcp	9000 - 9999
MasterIngressKube	kubernetes kubelet、调度程序和控制器管理器	tcp	10250 - 10259
MasterIngressWorkerKube	kubernetes kubelet、调度程序和控制器管理器	tcp	10250 - 10259
MasterIngressIngressServices	Kubernetes 入口服务	tcp	30000 - 32767

入口组	描述	IP 协议	端口范围
MasterIngressWorkerIngressServices	Kubernetes 入口服务	**tcp**	**30000 - 32767**
MasterIngressGeneve	Geneve 包	**udp**	**6081**
MasterIngressWorkerGeneve	Geneve 包	**udp**	**6081**
MasterIngressIpsecIke	IPsec IKE 数据包	**udp**	**500**
MasterIngressWorkerIpsecIke	IPsec IKE 数据包	**udp**	**500**
MasterIngressIpsecNat	IPsec NAT-T 数据包	**udp**	**4500**
MasterIngressWorkerIpsecNat	IPsec NAT-T 数据包	**udp**	**4500**
MasterIngressIpsecEsp	IPsec ESP 数据包	**50**	**All**
MasterIngressWorkerIpsecEsp	IPsec ESP 数据包	**50**	**All**
MasterIngressInternalUDP	内部集群通信	**udp**	**9000 - 9999**
MasterIngressWorkerInternalUDP	内部集群通信	**udp**	**9000 - 9999**
MasterIngressIngressServicesUDP	Kubernetes 入口服务	**udp**	**30000 - 32767**
MasterIngressWorkerIngressServicesUDP	Kubernetes 入口服务	**udp**	**30000 - 32767**

worker 入口

worker 机器需要以下入口组。每个入口组都是 **AWS::EC2::SecurityGroupIngress** 资源。

入口组	描述	IP 协议	端口范围
WorkerIngressVxlan	Vxlan 数据包	**udp**	**4789**
WorkerIngressWorkerVxlan	Vxlan 数据包	**udp**	**4789**
WorkerIngressInternal	内部集群通信	**tcp**	**9000 - 9999**
WorkerIngressWorkerInternal	内部集群通信	**tcp**	**9000 - 9999**
WorkerIngressKube	Kubernetes kubelet、调度程序和控制器管理器	**tcp**	**10250**
WorkerIngressWorkerKube	Kubernetes kubelet、调度程序和控制器管理器	**tcp**	**10250**
WorkerIngressIngressServices	Kubernetes 入口服务	**tcp**	**30000 - 32767**
WorkerIngressWorkerIngressServices	Kubernetes 入口服务	**tcp**	**30000 - 32767**
WorkerIngressGeneve	Geneve 包	**udp**	**6081**
WorkerIngressMasterGeneve	Geneve 包	**udp**	**6081**
WorkerIngressIpsecIke	IPsec IKE 数据包	**udp**	**500**
WorkerIngressMasterIpsecIke	IPsec IKE 数据包	**udp**	**500**
WorkerIngressIpsecNat	IPsec NAT-T 数据包	**udp**	**4500**
WorkerIngressMasterIpsecNat	IPsec NAT-T 数据包	**udp**	**4500**

入口组	描述	IP 协议	端口范围
WorkerIngressIpsecEsp	IPsec ESP 数据包	**50**	**All**
WorkerIngressMasterIpsecEsp	IPsec ESP 数据包	**50**	**All**
WorkerIngressInternalUDP	内部集群通信	**udp**	**9000 - 9999**
WorkerIngressMasterInternalUDP	内部集群通信	**udp**	**9000 - 9999**
WorkerIngressIngressServicesUDP	Kubernetes 入口服务	**udp**	**30000 - 32767**
WorkerIngressMasterIngressServicesUDP	Kubernetes 入口服务	**udp**	**30000 - 32767**

角色和实例配置集

您必须在 AWS 中为机器授予权限。提供的 CloudFormation 模板为以下 **AWS::IAM::Role** 对象授予机器 **Allow** 权限，并为每一组角色提供一个 **AWS::IAM::InstanceProfile**。如果不使用模板，您可以为机器授予以下宽泛权限或单独权限。

角色	影响	操作	资源
Master	**Allow**	**ec2:***	*
	Allow	**elasticloadbalancing:***	*
	Allow	**iam:PassRole**	*
	Allow	**s3:GetObject**	*
Worker	**Allow**	**ec2:Describe***	*
bootstrap	**Allow**	**ec2:Describe***	*
	Allow	**ec2:AttachVolume**	*
	Allow	**ec2:DetachVolume**	*

6.4.2.3.2. 集群机器

以下机器需要 **AWS::EC2::Instance** 对象：

- bootstrap 机器。安装过程中需要此机器，但可在集群部署后删除。
- 三个 control plane 机器。control plane 机器不受 control plane 机器集的管控。
- 计算机器。在安装过程中创建至少两台计算（compute）机器（也称为 worker 机器）。这些机器不受计算机器集的管控。

6.4.2.4. IAM 用户所需的 AWS 权限

注意

您的 IAM 用户必须在区域 **us-east-1** 中有权限 **tag:GetResources** 来删除基本集群资源。作为 AWS API 的要求的一部分，OpenShift Container Platform 安装程序在此区域中执行各种操作。

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

例 6.57. 安装所需的 EC2 权限

- **ec2:AttachNetworkInterface**
- **ec2:AuthorizeSecurityGroupEgress**
- **ec2:AuthorizeSecurityGroupIngress**
- **ec2:CopyImage**
- **ec2:CreateNetworkInterface**
- **ec2:CreateSecurityGroup**
- **ec2:CreateTags**
- **ec2:CreateVolume**
- **ec2:DeleteSecurityGroup**
- **ec2:DeleteSnapshot**
- **ec2:DeleteTags**
- **ec2:DeregisterImage**
- **ec2:DescribeAccountAttributes**
- **ec2:DescribeAddresses**
- **ec2:DescribeAvailabilityZones**
- **ec2:DescribeDhcpOptions**

- **ec2:DescribeImages**
- **ec2:DescribeInstanceAttribute**
- **ec2:DescribeInstanceCreditSpecifications**
- **ec2:DescribeInstances**
- **ec2:DescribeInstanceTypes**
- **ec2:DescribeInternetGateways**
- **ec2:DescribeKeyPairs**
- **ec2:DescribeNatGateways**
- **ec2:DescribeNetworkAcls**
- **ec2:DescribeNetworkInterfaces**
- **ec2:DescribePrefixLists**
- **ec2:DescribePublicIpv4Pools** （仅在 **install-config.yaml**中指定 **publicIpv4Pool** 时才需要）
- **ec2:DescribeRegions**
- **ec2:DescribeRouteTables**
- **ec2:DescribeSecurityGroupRules**
- **ec2:DescribeSecurityGroups**
- **ec2:DescribeSubnets**
- **ec2:DescribeTags**
- **ec2:DescribeVolumes**
- **ec2:DescribeVpcAttribute**
- **ec2:DescribeVpcClassicLink**
- **ec2:DescribeVpcClassicLinkDnsSupport**
- **ec2:DescribeVpcEndpoints**
- **ec2:DescribeVpcs**
- **ec2:DisassociateAddress** （仅在 **install-config.yaml**中指定 **publicIpv4Pool** 时才需要）
- **ec2:GetEbsDefaultKmsKeyId**
- **ec2:ModifyInstanceAttribute**
- **ec2:ModifyNetworkInterfaceAttribute**
- **ec2:RevokeSecurityGroupEgress**

- **ec2:RevokeSecurityGroupIngress**
- **ec2:RunInstances**
- **ec2:TerminateInstances**

例 6.58. 安装过程中创建网络资源所需的权限

- **ec2:AllocateAddress**
- **ec2:AssociateAddress**
- **ec2:AssociateDhcpOptions**
- **ec2:AssociateRouteTable**
- **ec2:AttachInternetGateway**
- **ec2:CreateDhcpOptions**
- **ec2:CreateInternetGateway**
- **ec2:CreateNatGateway**
- **ec2:CreateRoute**
- **ec2:CreateRouteTable**
- **ec2:CreateSubnet**
- **ec2:CreateVpc**
- **ec2:CreateVpcEndpoint**
- **ec2:ModifySubnetAttribute**
- **ec2:ModifyVpcAttribute**

注意

如果您使用现有的 Virtual Private Cloud (VPC)，您的帐户不需要这些权限来创建网络资源。

例 6.59. 安装所需的 Elastic Load Balancing 权限(ELB)

- **elasticloadbalancing:AddTags**
- **elasticloadbalancing:ApplySecurityGroupsToLoadBalancer**
- **elasticloadbalancing:AttachLoadBalancerToSubnets**
- **elasticloadbalancing:ConfigureHealthCheck**
- **elasticloadbalancing:CreateListener**

- **elasticloadbalancing:CreateLoadBalancer**
- **elasticloadbalancing:CreateLoadBalancerListeners**
- **elasticloadbalancing:CreateTargetGroup**
- **elasticloadbalancing:DeleteLoadBalancer**
- **elasticloadbalancing:DeregisterInstancesFromLoadBalancer**
- **elasticloadbalancing:DeregisterTargets**
- **elasticloadbalancing:DescribeInstanceHealth**
- **elasticloadbalancing:DescribeListeners**
- **elasticloadbalancing:DescribeLoadBalancerAttributes**
- **elasticloadbalancing:DescribeLoadBalancers**
- **elasticloadbalancing:DescribeTags**
- **elasticloadbalancing:DescribeTargetGroupAttributes**
- **elasticloadbalancing:DescribeTargetHealth**
- **elasticloadbalancing:ModifyLoadBalancerAttributes**
- **elasticloadbalancing:ModifyTargetGroup**
- **elasticloadbalancing:ModifyTargetGroupAttributes**
- **elasticloadbalancing:RegisterInstancesWithLoadBalancer**
- **elasticloadbalancing:RegisterTargets**
- **elasticloadbalancing:SetLoadBalancerPoliciesOfListener**
- **elasticloadbalancing:SetSecurityGroups**

重要

OpenShift Container Platform 使用 ELB 和 ELBv2 API 服务来置备负载均衡器。权限列表显示这两个服务所需的权限。AWS web 控制台中存在一个已知问题，其中这两个服务都使用相同的 **elasticloadbalancing** 操作前缀，但无法识别相同的操作。您可以忽略有关服务没有识别某些 **elasticloadbalancing** 操作的警告。

例 6.60. 安装所需的 IAM 权限

- **iam:AddRoleToInstanceProfile**
- **iam:CreateInstanceProfile**
- **iam:CreateRole**

- **iam:DeleteInstanceProfile**
- **iam:DeleteRole**
- **iam:DeleteRolePolicy**
- **iam:GetInstanceProfile**
- **iam:GetRole**
- **iam:GetRolePolicy**
- **iam:GetUser**
- **iam:ListInstanceProfilesForRole**
- **iam:ListRoles**
- **iam:ListUsers**
- **iam:PassRole**
- **iam:PutRolePolicy**
- **iam:RemoveRoleFromInstanceProfile**
- **iam:SimulatePrincipalPolicy**
- **iam:TagInstanceProfile**
- **iam:TagRole**

注意

如果您还没有在 AWS 帐户中创建负载均衡器，IAM 用户还需要 **iam:CreateServiceLinkedRole** 权限。

例 6.61. 安装所需的 Route 53 权限

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

- **route53:ListTagsForResource**
- **route53:UpdateHostedZoneComment**

例 6.62. 安装所需的 Amazon Simple Storage Service (S3) 权限

- **s3:CreateBucket**
- **s3:DeleteBucket**
- **s3:GetAccelerateConfiguration**
- **s3:GetBucketAcl**
- **s3:GetBucketCors**
- **s3:GetBucketLocation**
- **s3:GetBucketLogging**
- **s3:GetBucketObjectLockConfiguration**
- **s3:GetBucketPolicy**
- **s3:GetBucketRequestPayment**
- **s3:GetBucketTagging**
- **s3:GetBucketVersioning**
- **s3:GetBucketWebsite**
- **s3:GetEncryptionConfiguration**
- **s3:GetLifecycleConfiguration**
- **s3:GetReplicationConfiguration**
- **s3:ListBucket**
- **s3:PutBucketAcl**
- **s3:PutBucketPolicy**
- **s3:PutBucketTagging**
- **s3:PutEncryptionConfiguration**

例 6.63. 集群 Operators 所需的 S3 权限

- **s3:DeleteObject**
- **s3:GetObject**
- **s3:GetObjectAcl**

- **s3:GetObjectTagging**
- **s3:GetObjectVersion**
- **s3:PutObject**
- **s3:PutObjectAcl**
- **s3:PutObjectTagging**

例 6.64. 删除基本集群资源所需的权限

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

例 6.65. 删除网络资源所需的权限

- **ec2:DeleteDhcpOptions**
- **ec2:DeleteInternetGateway**
- **ec2:DeleteNatGateway**
- **ec2:DeleteRoute**
- **ec2:DeleteRouteTable**

- **ec2:DeleteSubnet**
- **ec2:DeleteVpc**
- **ec2:DeleteVpcEndpoints**
- **ec2:DetachInternetGateway**
- **ec2:DisassociateRouteTable**
- **ec2:ReleaseAddress**
- **ec2:ReplaceRouteTableAssociation**

注意

如果您使用现有的 VPC，您的帐户不需要这些权限来删除网络资源。您的帐户只需要有 **tag:UntagResources** 权限就能删除网络资源。

例 6.66. 使用自定义密钥管理服务 (KMS) 密钥安装集群的可选权限

- **kms:CreateGrant**
- **kms:Decrypt**
- **kms:DescribeKey**
- **kms:Encrypt**
- **kms:GenerateDataKey**
- **kms:GenerateDataKeyWithoutPlainText**
- **kms:ListGrants**
- **kms:RevokeGrant**

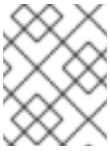
例 6.67. 使用共享实例角色删除集群所需的权限

- **iam:UntagRole**

例 6.68. 创建清单所需的额外 IAM 和 S3 权限

- **iam:GetUserPolicy**
- **iam:ListAccessKeys**
- **iam:PutUserPolicy**
- **iam:TagUser**
- **s3:AbortMultipartUpload**

- **s3:GetBucketPublicAccessBlock**
- **s3:ListBucket**
- **s3:ListBucketMultipartUploads**
- **s3:PutBucketPublicAccessBlock**
- **s3:PutLifecycleConfiguration**



注意

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

例 6.69. 实例的可选权限和安装配额检查

- **ec2:DescribeInstanceTypeOfferings**
- **servicequotas:ListAWSDefaultServiceQuotas**

例 6.70. 在共享 VPC 上安装集群时集群所有者帐户的可选权限

- **sts:AssumeRole**

例 6.71. 为安装启用您自己的公共 IPv4 地址 (BYOIP) 功能所需的权限

- **ec2:DescribePublicIpv4Pools**
- **ec2:DisassociateAddress**

6.4.2.5. 获取 AWS Marketplace 镜像

如果要使用 AWS Marketplace 镜像部署 OpenShift Container Platform 集群，您必须首先通过 AWS 订阅。订阅提供的提供的 AMI ID 可让您使用安装程序用来部署计算节点的 AMI ID。

先决条件

- 有 AWS 账户购买的产品。此帐户不必与用于安装集群的帐户相同。

流程

1. 从 AWS Marketplace 完成 OpenShift Container Platform 订阅。

6.4.3. 使用 CloudFormation 模板在 AWS 中用户置备的基础架构上安装集群

在 OpenShift Container Platform 版本 4.16 中，您可以使用您提供的基础架构在 Amazon Web Services (AWS) 上安装集群。

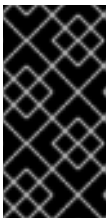
创建此基础架构的一种方法是使用提供的 CloudFormation 模板。您可以修改模板来自定义基础架构，或使用其包含的信息来按照公司策略创建 AWS 对象。

重要

进行用户置备的基础架构安装的步骤仅作为示例。使用您提供的基础架构安装集群需要了解云供应商和 OpenShift Container Platform 安装过程。提供的几个 CloudFormation 模板可帮助完成这些步骤，或者帮助您自行建模。您也可以自由选择通过其他方法创建所需的资源；模板仅作示例之用。

6.4.3.1. 先决条件

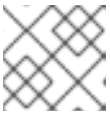
- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读选择集群安装方法并为用户准备它的文档。
- 已将 AWS 帐户配置为托管集群。



重要

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

- 已准备好用户置备的基础架构。
- 您下载了 AWS CLI 并安装到您的计算机上。请参阅 AWS 文档中的使用捆绑安装程序（Linux、macOS 或 UNIX）安装 AWS CLI。
- 如果使用防火墙，将其配置为允许集群需要访问的站点。



注意

如果要配置代理，请务必查看此站点列表。

- 如果环境中无法访问云身份和访问管理(IAM)API，或者不想将管理员级别的凭证 secret 存储在 **kube-system** 命名空间中，您可以 手动创建和维护 IAM 凭证。

6.4.3.2. 创建用于 AWS 的安装文件

要使用用户置备的基础架构在 Amazon Web Services (AWS) 上安装 OpenShift Container Platform，您必须生成并修改安装程序部署集群所需的文件，以便集群只创建要使用的机器。您要生成并自定义 **install-config.yaml** 文件、Kubernetes 清单和 Ignition 配置文件。您还可以选择在安装准备阶段首先设置独立 **var** 分区。

6.4.3.2.1. 可选：创建独立 /var 分区

建议安装程序将 OpenShift Container Platform 的磁盘分区保留给安装程序。然而，在有些情况下您可能需要在文件系统的一部分中创建独立分区。

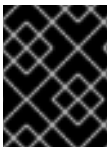
OpenShift Container Platform 支持添加单个分区来将存储附加到 **/var** 分区或 **/var** 的子目录中。例如：

- **/var/lib/containers** ：保存随着系统中添加更多镜像和容器而增长的容器相关内容。

- **/var/lib/etcd** ：保存您可能希望独立保留的数据，比如 etcd 存储的性能优化。
- **/var** ：保存您可能希望独立保留的数据，以满足审计等目的。

通过单独存储 **/var** 目录的内容，可以更轻松地根据需要为区域扩展存储，并在以后重新安装 OpenShift Container Platform，并保持该数据的完整性。使用这个方法，您不必再次拉取所有容器，在更新系统时也不必复制大量日志文件。

因为 **/var** 在进行一个全新的 Red Hat Enterprise Linux CoreOS（RHCOS）安装前必需存在，所以这个流程会在 OpenShift Container Platform 安装过程的 **openshift-install** 准备阶段插入一个创建的机器配置清单的机器配置来设置独立的 **/var** 分区。



重要

如果按照以下步骤在此流程中创建独立 **/var** 分区，则不需要再次创建 Kubernetes 清单和 Ignition 配置文件，如本节所述。

流程

1. 创建存放 OpenShift Container Platform 安装文件的目录：

```
$ mkdir $HOME/clusterconfig
```

2. 运行 **openshift-install**，以在 **manifest** 和 **openshift** 子目录中创建一组文件。在系统提示时回答系统问题：

```
$ openshift-install create manifests --dir $HOME/clusterconfig
```

输出示例

```
? SSH Public Key ...
INFO Credentials loaded from the "myprofile" profile in file "/home/myuser/.aws/credentials"
INFO Consuming Install Config from target directory
INFO Manifests created in: $HOME/clusterconfig/manifests and
$HOME/clusterconfig/openshift
```

3. 可选：确认安装程序在 **clusterconfig/openshift** 目录中创建了清单：

```
$ ls $HOME/clusterconfig/openshift/
```

输出示例

```
99_kubeadmin-password-secret.yaml
99_openshift-cluster-api_master-machines-0.yaml
99_openshift-cluster-api_master-machines-1.yaml
99_openshift-cluster-api_master-machines-2.yaml
...
```

4. 创建用于配置额外分区的 Butane 配置。例如，将文件命名为 **$HOME/clusterconfig/98-var-partition.bu**，将磁盘设备名称改为 **worker** 系统上存储设备的名称，并根据情况设置存储大小。这个示例将 **/var** 目录放在一个单独的分区中：

```
variant: openshift
```

```
version: 4.16.0
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
storage:
  disks:
  - device: /dev/disk/by-id/<device_name> 1
    partitions:
    - label: var
      start_mib: <partition_start_offset> 2
      size_mib: <partition_size> 3
      number: 5
  filesystems:
    - device: /dev/disk/by-partlabel/var
      path: /var
      format: xfs
      mount_options: [defaults, prjquota] 4
      with_mount_unit: true
```

1. 要分区的磁盘的存储设备名称。
2. 在引导磁盘中添加数据分区时，推荐最少使用 25000 MiB(Mebibytes)。root 文件系统会自动调整大小以填充所有可用空间（最多到指定的偏移值）。如果没有指定值，或者指定的值小于推荐的最小值，则生成的 root 文件系统会太小，而在以后进行的 RHCOS 重新安装可能会覆盖数据分区的开始部分。
3. 以兆字节为单位的数据分区大小。
4. 对于用于容器存储的文件系统，必须启用 **prjquota** 挂载选项。

注意

当创建单独的 **/var** 分区时，如果不同的实例类型没有相同的设备名称，则无法为 worker 节点使用不同的实例类型。

5. 从 Butane 配置创建一个清单，并将它保存到 **clusterconfig/openshift** 目录中。例如，运行以下命令：

```
$ butane $HOME/clusterconfig/98-var-partition.bu -o $HOME/clusterconfig/openshift/98-var-partition.yaml
```

6. 再次运行 **openshift-install**，从 **manifest** 和 **openshift** 子目录中的一组文件创建 Ignition 配置：

```
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
auth  bootstrap.ign  master.ign  metadata.json  worker.ign
```

现在，您可以使用 Ignition 配置文件作为安装程序的输入来安装 Red Hat Enterprise Linux CoreOS(RHCOS)系统。

6.4.3.2.2. 创建安装配置文件

生成并自定义安装程序部署集群所需的安装配置文件。

先决条件

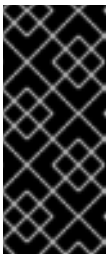
- 已获取 OpenShift Container Platform 安装程序用于用户置备的基础架构和集群的 pull secret。
- 使用红帽发布的附带 Red Hat Enterprise Linux CoreOS (RHCOS) AMI 检查您是否将集群部署到 AWS 区域。如果要部署到需要自定义 AMI 的 AWS 区域，如 AWS GovCloud 区域，您必须手动创建 **install-config.yaml** 文件。

流程

1. 创建 **install-config.yaml** 文件。
   a. 进入包含安装程序的目录并运行以下命令：

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

   1 对于 **<installation_directory>**，请指定要存储安装程序创建的文件的目录名称。



重要

指定一个空目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

   b. 在提示符处，提供云的配置详情：
      i. 可选：选择用于访问集群机器的 SSH 密钥。

注意

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

      ii. 选择 **aws** 作为目标平台。
      iii. 如果计算机上没有保存 AWS 配置集，请为您配置用于运行安装程序的用户输入 AWS 访问密钥 ID 和 secret 访问密钥。

注意

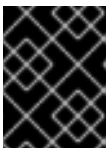
AWS 访问密钥 ID 和 secret 访问密钥存储在安装主机上当前用户主目录中的 **~/.aws/credentials** 中。如果文件中不存在导出的配置集凭证，安装程序会提示您输入凭证。您向安装程序提供的所有凭证都存储在文件中。

      iv. 选择要将集群部署到的 AWS Region。
      v. 选择您为集群配置的 Route 53 服务的基域。
      vi. 为集群输入描述性名称。

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

2. 如果要安装三节点集群，请通过将 **compute.replicas** 参数设置为 **0** 来修改 **install-config.yaml** 文件。这样可确保集群的 control plane 可以调度。如需更多信息，请参阅"在 AWS 上安装三节点集群"。

3. 可选：备份 **install-config.yaml** 文件。



重要

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

其他资源

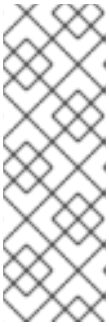
- 如需有关 AWS 配置集和凭证配置的更多信息，请参阅 AWS 文档中的配置和凭证文件设置。

6.4.3.2.3. 在安装过程中配置集群范围代理

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

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 **Proxy** 对象的 **spec.noProxy** 字段中添加站点来绕过代理。



注意

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

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

流程

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

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: ec2.<aws_region>.amazonaws.com,elasticloadbalancing.
<aws_region>.amazonaws.com,s3.<aws_region>.amazonaws.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
```

```
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 **.** 以仅匹配子域。例如，**.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代理。如果您已将 Amazon **EC2**、**Elastic Load Balancing** 和 **S3** VPC 端点添加到 VPC 中，您必须将这些端点添加到 **noProxy** 字段。
4. 如果提供，安装程序会在 **openshift-config** 命名空间中生成名为 **user-ca-bundle** 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 **trusted-ca-bundle** 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， **Proxy** 对象的 **trustedCA** 字段中也会引用此配置映射。**additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。
5. 可选：决定 **Proxy** 对象的配置以引用 **trustedCA** 字段中 **user-ca-bundle** 配置映射的策略。允许的值是 **Proxyonly** 和 **Always**。仅在配置了 **http/https** 代理时，使用 **Proxyonly** 引用 **user-ca-bundle** 配置映射。使用 **Always** 始终引用 **user-ca-bundle** 配置映射。默认值为 **Proxyonly**。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。

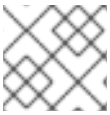
注意

如果安装程序超时，重启并使用安装程序的 **wait-for** 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

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

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



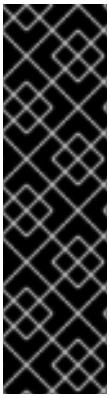
注意

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

6.4.3.2.4. 创建 Kubernetes 清单和 Ignition 配置文件

由于您必须修改一些集群定义文件并手动启动集群机器，因此您必须生成 Kubernetes 清单和 Ignition 配置文件来配置机器。

安装配置文件转换为 Kubernetes 清单。清单嵌套到 Ignition 配置文件中，稍后用于配置集群机器。



重要

- OpenShift Container Platform 安装程序生成的 Ignition 配置文件包含 24 小时后过期的证书，然后在该时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

先决条件

- 已获得 OpenShift Container Platform 安装程序。
- 已创建 **install-config.yaml** 安装配置文件。

流程

1. 进入包含 OpenShift Container Platform 安装程序的目录，并为集群生成 Kubernetes 清单：

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

1 对于 **<installation_directory>**，请指定包含您创建的 **install-config.yaml** 文件的安装目录。

2. 删除定义 control plane 机器的 Kubernetes 清单文件：

```
$ rm -f <installation_directory>/openshift/99_openshift-cluster-api_master-machines-*.yaml
```

通过删除这些文件，您可以防止集群自动生成 control plane 机器。

3. 删除定义 control plane 机器集的 Kubernetes 清单文件：

```
$ rm -f <installation_directory>/openshift/99_openshift-machine-api_master-control-plane-machine-set.yaml
```

```
$ rm -f <installation_directory>/openshift/99_openshift-cluster-api_worker-machineset-*.yaml
```

重要

如果在用户置备的基础架构上安装集群时禁用了 **MachineAPI** 功能，则必须删除定义 worker 机器的 Kubernetes 清单文件。否则，集群将无法安装。

由于您要自行创建和管理 worker 机器，因此不需要初始化这些机器。

警告

如果您要安装一个三节点集群，请跳过以下步骤，以便可以调度 control plane 节点。

重要

当您将 control plane 节点从默认的不可调度配置为可以调度时，需要额外的订阅。这是因为 control plane 节点变为计算节点。

4. 检查 **<installation_directory>/manifests/cluster-scheduler-02-config.yml Kubernetes 清单文件中的 mastersSchedulable** 参数是否已设置为 **false**。此设置可防止在 control plane 机器上调度 pod：

   a. 打开 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** 文件。

   b. 找到 **mastersSchedulable** 参数，并确保它被设置为 **false**。

   c. 保存并退出 文件。

5. 可选：如果您不希望 Ingress Operator 代表您创建 DNS 记录，请删除 **<installation_directory>/manifests/cluster-dns-02-config.yml** DNS 配置文件中的 **privateZone 和 publicZone** 部分：

```
apiVersion: config.openshift.io/v1
kind: DNS
metadata:
  creationTimestamp: null
  name: cluster
spec:
  baseDomain: example.openshift.com
  privateZone: 1
    id: mycluster-100419-private-zone
  publicZone: 2
    id: example.openshift.com
status: {}
```

1 2 完全删除此部分。

   如果这样做，您必须在后续步骤中手动添加入口 DNS 记录。

6. 要创建 Ignition 配置文件，请从包含安装程序的目录运行以下命令：

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

1 对于 **<installation_directory>**，请指定相同的安装目录。

   为安装目录中的 bootstrap、control plane 和计算节点创建 Ignition 配置文件。**kubeadmin-password** 和 **kubeconfig** 文件在 **./<installation_directory>/auth** 目录中创建：

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

6.4.3.3. 提取基础架构名称

Ignition 配置文件包含一个唯一集群标识符，您可以使用它在 Amazon Web Services (AWS) 中唯一地标识您的集群。基础架构名称还用于在 OpenShift Container Platform 安装过程中定位适当的 AWS 资源。提供的 CloudFormation 模板包含对此基础架构名称的引用，因此您必须提取它。

先决条件

- 已获取 OpenShift Container Platform 安装程序和集群的 pull secret。
- 已为集群生成 Ignition 配置文件。
- 已安装 **jq** 软件包。

流程

- 要从 Ignition 配置文件元数据中提取和查看基础架构名称，请运行以下命令：

```
$ jq -r .infraID <installation_directory>/metadata.json 1
```

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

输出示例

```
openshift-vw9j6 1
```

1 此命令的输出是您的集群名称和随机字符串。

6.4.3.4. 在 AWS 中创建 VPC

您必须在 Amazon Web Services（AWS）中创建 Virtual Private Cloud（VPC），供您的 OpenShift Container Platform 集群使用。您可以自定义 VPC 来满足您的要求，包括 VPN 和路由表。

您可以使用提供的 CloudFormation 模板和自定义参数文件创建代表 VPC 的 AWS 资源堆栈。

注意

如果不使用提供的 CloudFormation 模板来创建 AWS 基础架构，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 已配置了一个 AWS 帐户。
- 您可以通过运行 **aws configure**，将 AWS 密钥和区域添加到本地 AWS 配置集中。
- 已为集群生成 Ignition 配置文件。

流程

1. 创建一个 JSON 文件，其包含模板所需的参数值：

```
[
  {
    "ParameterKey": "VpcCidr", 1
    "ParameterValue": "10.0.0.0/16" 2
  },
  {
    "ParameterKey": "AvailabilityZoneCount", 3
    "ParameterValue": "1" 4
  },
  {
    "ParameterKey": "SubnetBits", 5
    "ParameterValue": "12" 6
  }
]
```

1. VPC 的 CIDR 块。
2. 以 **x.x.x.x/16-24** 格式指定 CIDR 块。
3. 在其中部署 VPC 的可用区的数量。
4. 指定一个 **1** 到 **3** 之间的整数。
5. 各个可用区中每个子网的大小。
6. 指定 **5** 到 **13** 之间的整数，其中 **5** 为 **/27**，**13** 为 **/19**。

2. 复制本主题的 **VPC 的 CloudFormation 模板**部分中的模板，并将它以 YAML 文件形式保存到计算机上。此模板描述了集群所需的 VPC。
3. 启动 CloudFormation 模板，以创建代表 VPC 的 AWS 资源堆栈：

重要

您必须在一行内输入命令。

```
$ aws cloudformation create-stack --stack-name <name> 1
    --template-body file://<template>.yaml 2
    --parameters file://<parameters>.json 3
```

1. **<name>** 是 CloudFormation 堆栈的名称，如 **cluster-VPC**。如果您删除集群，则需要此堆栈的名称。

2 **<template>** 是您保存的 CloudFormation 模板 YAML 文件的相对路径和名称。

3 **<parameters>** 是 CloudFormation 参数 JSON 文件的相对路径和名称。

输出示例

```
arn:aws:cloudformation:us-east-1:269333783861:stack/cluster-vpc/dbedae40-2fd3-11eb-820e-12a48460849f
```

4. 确认模板组件已存在：

```
$ aws cloudformation describe-stacks --stack-name <name>
```

在 **StackStatus** 显示 **CREATE_COMPLETE** 后，输出会显示以下参数的值。您必须将这些参数值提供给您在创建集群时要运行的其他 CloudFormation 模板：

VpcId	您的 VPC ID。
PublicSubnetIds	新公共子网的 ID。
PrivateSubnetIds	新专用子网的 ID。

6.4.3.4.1. VPC 的 CloudFormation 模板

您可以使用以下 CloudFormation 模板来部署 OpenShift Container Platform 集群所需的 VPC。

例 6.72. VPC 的 CloudFormation 模板

```
AWSTemplateFormatVersion: 2010-09-09
Description: Template for Best Practice VPC with 1-3 AZs

Parameters:
  VpcCidr:
    AllowedPattern: ^(([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])\.){3}([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])(\/(1[6-9]|2[0-4]))$
    ConstraintDescription: CIDR block parameter must be in the form x.x.x.x/16-24.
    Default: 10.0.0.0/16
    Description: CIDR block for VPC.
    Type: String
  AvailabilityZoneCount:
    ConstraintDescription: "The number of availability zones. (Min: 1, Max: 3)"
    MinValue: 1
    MaxValue: 3
    Default: 1
    Description: "How many AZs to create VPC subnets for. (Min: 1, Max: 3)"
    Type: Number
  SubnetBits:
    ConstraintDescription: CIDR block parameter must be in the form x.x.x.x/19-27.
    MinValue: 5
    MaxValue: 13
```

```
    Default: 12
    Description: "Size of each subnet to create within the availability zones. (Min: 5 = /27, Max: 13 =
/19)"
    Type: Number

Metadata:
  AWS::CloudFormation::Interface:
    ParameterGroups:
    - Label:
        default: "Network Configuration"
      Parameters:
      - VpcCidr
      - SubnetBits
    - Label:
        default: "Availability Zones"
      Parameters:
      - AvailabilityZoneCount
    ParameterLabels:
      AvailabilityZoneCount:
        default: "Availability Zone Count"
      VpcCidr:
        default: "VPC CIDR"
      SubnetBits:
        default: "Bits Per Subnet"

Conditions:
  DoAz3: !Equals [3, !Ref AvailabilityZoneCount]
  DoAz2: !Or [!Equals [2, !Ref AvailabilityZoneCount], Condition: DoAz3]

Resources:
  VPC:
    Type: "AWS::EC2::VPC"
    Properties:
      EnableDnsSupport: "true"
      EnableDnsHostnames: "true"
      CidrBlock: !Ref VpcCidr
  PublicSubnet:
    Type: "AWS::EC2::Subnet"
    Properties:
      VpcId: !Ref VPC
      CidrBlock: !Select [0, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
      AvailabilityZone: !Select
      - 0
      - Fn::GetAZs: !Ref "AWS::Region"
  PublicSubnet2:
    Type: "AWS::EC2::Subnet"
    Condition: DoAz2
    Properties:
      VpcId: !Ref VPC
      CidrBlock: !Select [1, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
      AvailabilityZone: !Select
      - 1
      - Fn::GetAZs: !Ref "AWS::Region"
  PublicSubnet3:
    Type: "AWS::EC2::Subnet"
    Condition: DoAz3
```

```
    Properties:
      VpcId: !Ref VPC
      CidrBlock: !Select [2, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
      AvailabilityZone: !Select
      - 2
      - Fn::GetAZs: !Ref "AWS::Region"
  InternetGateway:
    Type: "AWS::EC2::InternetGateway"
  GatewayToInternet:
    Type: "AWS::EC2::VPCGatewayAttachment"
    Properties:
      VpcId: !Ref VPC
      InternetGatewayId: !Ref InternetGateway
  PublicRouteTable:
    Type: "AWS::EC2::RouteTable"
    Properties:
      VpcId: !Ref VPC
  PublicRoute:
    Type: "AWS::EC2::Route"
    DependsOn: GatewayToInternet
    Properties:
      RouteTableId: !Ref PublicRouteTable
      DestinationCidrBlock: 0.0.0.0/0
      GatewayId: !Ref InternetGateway
  PublicSubnetRouteTableAssociation:
    Type: "AWS::EC2::SubnetRouteTableAssociation"
    Properties:
      SubnetId: !Ref PublicSubnet
      RouteTableId: !Ref PublicRouteTable
  PublicSubnetRouteTableAssociation2:
    Type: "AWS::EC2::SubnetRouteTableAssociation"
    Condition: DoAz2
    Properties:
      SubnetId: !Ref PublicSubnet2
      RouteTableId: !Ref PublicRouteTable
  PublicSubnetRouteTableAssociation3:
    Condition: DoAz3
    Type: "AWS::EC2::SubnetRouteTableAssociation"
    Properties:
      SubnetId: !Ref PublicSubnet3
      RouteTableId: !Ref PublicRouteTable
  PrivateSubnet:
    Type: "AWS::EC2::Subnet"
    Properties:
      VpcId: !Ref VPC
      CidrBlock: !Select [3, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
      AvailabilityZone: !Select
      - 0
      - Fn::GetAZs: !Ref "AWS::Region"
  PrivateRouteTable:
    Type: "AWS::EC2::RouteTable"
    Properties:
      VpcId: !Ref VPC
  PrivateSubnetRouteTableAssociation:
    Type: "AWS::EC2::SubnetRouteTableAssociation"
    Properties:
```

```
      SubnetId: !Ref PrivateSubnet
      RouteTableId: !Ref PrivateRouteTable
  NAT:
    DependsOn:
    - GatewayToInternet
    Type: "AWS::EC2::NatGateway"
    Properties:
      AllocationId:
        "Fn::GetAtt":
        - EIP
        - AllocationId
      SubnetId: !Ref PublicSubnet
  EIP:
    Type: "AWS::EC2::EIP"
    Properties:
      Domain: vpc
  Route:
    Type: "AWS::EC2::Route"
    Properties:
      RouteTableId:
        Ref: PrivateRouteTable
      DestinationCidrBlock: 0.0.0.0/0
      NatGatewayId:
        Ref: NAT
  PrivateSubnet2:
    Type: "AWS::EC2::Subnet"
    Condition: DoAz2
    Properties:
      VpcId: !Ref VPC
      CidrBlock: !Select [4, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
      AvailabilityZone: !Select
      - 1
      - Fn::GetAZs: !Ref "AWS::Region"
  PrivateRouteTable2:
    Type: "AWS::EC2::RouteTable"
    Condition: DoAz2
    Properties:
      VpcId: !Ref VPC
  PrivateSubnetRouteTableAssociation2:
    Type: "AWS::EC2::SubnetRouteTableAssociation"
    Condition: DoAz2
    Properties:
      SubnetId: !Ref PrivateSubnet2
      RouteTableId: !Ref PrivateRouteTable2
  NAT2:
    DependsOn:
    - GatewayToInternet
    Type: "AWS::EC2::NatGateway"
    Condition: DoAz2
    Properties:
      AllocationId:
        "Fn::GetAtt":
        - EIP2
        - AllocationId
      SubnetId: !Ref PublicSubnet2
  EIP2:
```

```
  Type: "AWS::EC2::EIP"
  Condition: DoAz2
  Properties:
    Domain: vpc
Route2:
  Type: "AWS::EC2::Route"
  Condition: DoAz2
  Properties:
    RouteTableId:
      Ref: PrivateRouteTable2
    DestinationCidrBlock: 0.0.0.0/0
    NatGatewayId:
      Ref: NAT2
PrivateSubnet3:
  Type: "AWS::EC2::Subnet"
  Condition: DoAz3
  Properties:
    VpcId: !Ref VPC
    CidrBlock: !Select [5, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
    AvailabilityZone: !Select
    - 2
    - Fn::GetAZs: !Ref "AWS::Region"
PrivateRouteTable3:
  Type: "AWS::EC2::RouteTable"
  Condition: DoAz3
  Properties:
    VpcId: !Ref VPC
PrivateSubnetRouteTableAssociation3:
  Type: "AWS::EC2::SubnetRouteTableAssociation"
  Condition: DoAz3
  Properties:
    SubnetId: !Ref PrivateSubnet3
    RouteTableId: !Ref PrivateRouteTable3
NAT3:
  DependsOn:
  - GatewayToInternet
  Type: "AWS::EC2::NatGateway"
  Condition: DoAz3
  Properties:
    AllocationId:
      "Fn::GetAtt":
      - EIP3
      - AllocationId
    SubnetId: !Ref PublicSubnet3
EIP3:
  Type: "AWS::EC2::EIP"
  Condition: DoAz3
  Properties:
    Domain: vpc
Route3:
  Type: "AWS::EC2::Route"
  Condition: DoAz3
  Properties:
    RouteTableId:
      Ref: PrivateRouteTable3
    DestinationCidrBlock: 0.0.0.0/0
```

```
      NatGatewayId:
        Ref: NAT3
  S3Endpoint:
    Type: AWS::EC2::VPCEndpoint
    Properties:
      PolicyDocument:
        Version: 2012-10-17
        Statement:
        - Effect: Allow
          Principal: '*'
          Action:
          - '*'
          Resource:
          - '*'
      RouteTableIds:
      - !Ref PublicRouteTable
      - !Ref PrivateRouteTable
      - !If [DoAz2, !Ref PrivateRouteTable2, !Ref "AWS::NoValue"]
      - !If [DoAz3, !Ref PrivateRouteTable3, !Ref "AWS::NoValue"]
      ServiceName: !Join
      - ''
      - - com.amazonaws.
        - !Ref 'AWS::Region'
        - .s3
      VpcId: !Ref VPC

Outputs:
  VpcId:
    Description: ID of the new VPC.
    Value: !Ref VPC
  PublicSubnetIds:
    Description: Subnet IDs of the public subnets.
    Value:
      !Join [
        ",",
        [!Ref PublicSubnet, !If [DoAz2, !Ref PublicSubnet2, !Ref "AWS::NoValue"], !If [DoAz3, !Ref
PublicSubnet3, !Ref "AWS::NoValue"]]
      ]
  PrivateSubnetIds:
    Description: Subnet IDs of the private subnets.
    Value:
      !Join [
        ",",
        [!Ref PrivateSubnet, !If [DoAz2, !Ref PrivateSubnet2, !Ref "AWS::NoValue"], !If [DoAz3, !Ref
PrivateSubnet3, !Ref "AWS::NoValue"]]
      ]
  PublicRouteTableId:
    Description: Public Route table ID
    Value: !Ref PublicRouteTable
  PrivateRouteTableIds:
    Description: Private Route table IDs
    Value:
      !Join [
        ",",
        [
          !Join ["=", [
```

```
          !Select [0, "Fn::GetAZs": !Ref "AWS::Region"],
          !Ref PrivateRouteTable
        ]],
        !If [DoAz2,
          !Join ["=", [!Select [1, "Fn::GetAZs": !Ref "AWS::Region"], !Ref PrivateRouteTable2]],
          !Ref "AWS::NoValue"
        ],
        !If [DoAz3,
          !Join ["=", [!Select [2, "Fn::GetAZs": !Ref "AWS::Region"], !Ref PrivateRouteTable3]],
          !Ref "AWS::NoValue"
        ]
      ]
    ]
```

其他资源

- 您可以通过导航 AWS CloudFormation 控制台来查看您创建的 CloudFormation 堆栈的详情。

6.4.3.5. 在 AWS 中创建网络和负载均衡组件

您必须在 OpenShift Container Platform 集群可以使用的 Amazon Web Services（AWS）中配置网络、经典或网络负载均衡。

您可以使用提供的 CloudFormation 模板和自定义参数文件来创建 AWS 资源堆栈。堆栈代表 OpenShift Container Platform 集群所需的网络和负载均衡组件。该模板还创建一个托管区和子网标签。

您可以在单一虚拟私有云(VPC)内多次运行该模板。

注意

如果不使用提供的 CloudFormation 模板来创建 AWS 基础架构，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 已配置了一个 AWS 帐户。
- 您可以通过运行 **aws configure**，将 AWS 密钥和区域添加到本地 AWS 配置集中。
- 已为集群生成 Ignition 配置文件。
- 您在 AWS 中创建并配置了 VPC 及相关子网。

流程

1. 获取您在 **install-config.yaml** 文件中为集群指定的 Route 53 基域的托管区 ID。您可以运行以下命令来获取托管区的详细信息：

```
$ aws route53 list-hosted-zones-by-name --dns-name <route53_domain> 1
```

1 对于 **<route53_domain>**，请指定您为集群生成 **install-config.yaml** 文件时所用的 Route53 基域。

输出示例

```
mycluster.example.com. False 100
HOSTEDZONES 65F8F38E-2268-B835-E15C-AB55336FCBFA
/hostedzone/Z21IXYZABCZ2A4 mycluster.example.com. 10
```

在示例输出中，托管区 ID 为 **Z21IXYZABCZ2A4**。

2. 创建一个 JSON 文件，其包含模板所需的参数值：

```
[
  {
    "ParameterKey": "ClusterName", 1
    "ParameterValue": "mycluster" 2
  },
  {
    "ParameterKey": "InfrastructureName", 3
    "ParameterValue": "mycluster-<random_string>" 4
  },
  {
    "ParameterKey": "HostedZoneId", 5
    "ParameterValue": "<random_string>" 6
  },
  {
    "ParameterKey": "HostedZoneName", 7
    "ParameterValue": "example.com" 8
  },
  {
    "ParameterKey": "PublicSubnets", 9
    "ParameterValue": "subnet-<random_string>" 10
  },
  {
    "ParameterKey": "PrivateSubnets", 11
    "ParameterValue": "subnet-<random_string>" 12
  },
  {
    "ParameterKey": "VpcId", 13
    "ParameterValue": "vpc-<random_string>" 14
  }
]
```

1. 一个简短的、代表集群的名称用于主机名等。
2. 指定您为集群生成 **install-config.yaml** 文件时所用的集群名称。
3. 您的 Ignition 配置文件中为集群编码的集群基础架构名称。
4. 指定从 Ignition 配置文件元数据中提取的基础架构名称，其格式为 **<cluster-name>-<random-string>**。
5. 用来注册目标的 Route 53 公共区 ID。
6. 指定 Route 53 公共区 ID，其格式与 **Z21IXYZABCZ2A4** 类似。您可以从 AWS 控制台获取这个值。

7 用来注册目标的 Route 53 区。

8 指定您为集群生成 **install-config.yaml** 文件时所用的 Route 53 基域。请勿包含 AWS 控制台中显示的结尾句点 (.)。

9 为 VPC 创建的公共子网。

10 指定 VPC 的 CloudFormation 模板输出的 **PublicSubnetIds** 值。

11 为 VPC 创建的专用子网。

12 指定 VPC 的 CloudFormation 模板输出的 **PrivateSubnetIds** 值。

13 为集群创建的 VPC。

14 指定 VPC 的 CloudFormation 模板输出的 **VpcId** 值。

3. 复制本主题的**网络和负载均衡器的 CloudFormation 模板**部分中的模板，并将它以 YAML 文件形式保存到计算机上。此模板描述了集群所需的网络和负载均衡对象。

重要

如果要将集群部署到 AWS 政府或 secret 区域，您必须更新 CloudFormation 模板中的 **InternalApiServerRecord**，以使用 **CNAME** 记录。AWS 政府区不支持 **ALIAS** 类型的记录。

4. 启动 CloudFormation 模板，以创建 AWS 资源堆栈，该堆栈提供网络和负载均衡组件：

重要

您必须在一行内输入命令。

```
$ aws cloudformation create-stack --stack-name <name> 1
    --template-body file://<template>.yaml 2
    --parameters file://<parameters>.json 3
    --capabilities CAPABILITY_NAMED_IAM 4
```

1 **<name>** 是 CloudFormation 堆栈的名称，如 **cluster-dns**。如果您删除集群，则需要此堆栈的名称。

2 **<template>** 是您保存的 CloudFormation 模板 YAML 文件的相对路径和名称。

3 **<parameters>** 是 CloudFormation 参数 JSON 文件的相对路径和名称。

4 您必须明确声明 **CAPABILITY_NAMED_IAM** 功能，因为提供的模板会创建一些 **AWS::IAM::Role** 资源。

输出示例

```
arn:aws:cloudformation:us-east-1:269333783861:stack/cluster-dns/cd3e5de0-2fd4-11eb-5cf0-12be5c33a183
```

5. 确认模板组件已存在：

```
$ aws cloudformation describe-stacks --stack-name <name>
```

在 **StackStatus** 显示 **CREATE_COMPLETE** 后，输出会显示以下参数的值。您必须将这些参数值提供给您在创建集群时要运行的其他 CloudFormation 模板：

PrivateHostedZoneId	专用 DNS 的托管区 ID。
ExternalApiLoadBalancerName	外部 API 负载均衡器的完整名称。
InternalApiLoadBalancerName	内部 API 负载均衡器的完整名称。
ApiServerDnsName	API 服务器的完整主机名。
RegisterNlbIpTargetsLambda	有助于为这些负载均衡器注册/撤销注册 IP 目标的 Lambda ARN。
ExternalApiTargetGroupArn	外部 API 目标组的 ARN。
InternalApiTargetGroupArn	内部 API 目标组的 ARN。
InternalServiceTargetGroupArn	内部服务目标组群的 ARN。

6.4.3.5.1. 网络和负载均衡器的 CloudFormation 模板

您可以使用以下 CloudFormation 模板来部署 OpenShift Container Platform 集群所需的网络对象和负载均衡器。

例 6.73. 网络和负载均衡器的 CloudFormation 模板

```
AWSTemplateFormatVersion: 2010-09-09
Description: Template for OpenShift Cluster Network Elements (Route53 & LBs)

Parameters:
```

```
  ClusterName:
    AllowedPattern: ^([a-zA-Z][a-zA-Z0-9\-]{0,26})$
    MaxLength: 27
    MinLength: 1
    ConstraintDescription: Cluster name must be alphanumeric, start with a letter, and have a
maximum of 27 characters.
    Description: A short, representative cluster name to use for host names and other identifying
names.
    Type: String
  InfrastructureName:
    AllowedPattern: ^([a-zA-Z][a-zA-Z0-9\-]{0,26})$
    MaxLength: 27
    MinLength: 1
    ConstraintDescription: Infrastructure name must be alphanumeric, start with a letter, and have a
maximum of 27 characters.
    Description: A short, unique cluster ID used to tag cloud resources and identify items owned or
used by the cluster.
    Type: String
  HostedZoneId:
    Description: The Route53 public zone ID to register the targets with, such as
Z21IXYZABCZ2A4.
    Type: String
  HostedZoneName:
    Description: The Route53 zone to register the targets with, such as example.com. Omit the
trailing period.
    Type: String
    Default: "example.com"
  PublicSubnets:
    Description: The internet-facing subnets.
    Type: List<AWS::EC2::Subnet::Id>
  PrivateSubnets:
    Description: The internal subnets.
    Type: List<AWS::EC2::Subnet::Id>
  VpcId:
    Description: The VPC-scoped resources will belong to this VPC.
    Type: AWS::EC2::VPC::Id

Metadata:
  AWS::CloudFormation::Interface:
    ParameterGroups:
    - Label:
        default: "Cluster Information"
      Parameters:
      - ClusterName
      - InfrastructureName
    - Label:
        default: "Network Configuration"
      Parameters:
      - VpcId
      - PublicSubnets
      - PrivateSubnets
    - Label:
        default: "DNS"
      Parameters:
      - HostedZoneName
      - HostedZoneId
```

```
    ParameterLabels:
      ClusterName:
        default: "Cluster Name"
      InfrastructureName:
        default: "Infrastructure Name"
      VpcId:
        default: "VPC ID"
      PublicSubnets:
        default: "Public Subnets"
      PrivateSubnets:
        default: "Private Subnets"
      HostedZoneName:
        default: "Public Hosted Zone Name"
      HostedZoneId:
        default: "Public Hosted Zone ID"

Resources:
  ExtApiElb:
    Type: AWS::ElasticLoadBalancingV2::LoadBalancer
    Properties:
      Name: !Join ["-", [!Ref InfrastructureName, "ext"]]
      IpAddressType: ipv4
      Subnets: !Ref PublicSubnets
      Type: network

  IntApiElb:
    Type: AWS::ElasticLoadBalancingV2::LoadBalancer
    Properties:
      Name: !Join ["-", [!Ref InfrastructureName, "int"]]
      Scheme: internal
      IpAddressType: ipv4
      Subnets: !Ref PrivateSubnets
      Type: network

  IntDns:
    Type: "AWS::Route53::HostedZone"
    Properties:
      HostedZoneConfig:
        Comment: "Managed by CloudFormation"
      Name: !Join [".", [!Ref ClusterName, !Ref HostedZoneName]]
      HostedZoneTags:
      - Key: Name
        Value: !Join ["-", [!Ref InfrastructureName, "int"]]
      - Key: !Join ["", ["kubernetes.io/cluster/", !Ref InfrastructureName]]
        Value: "owned"
      VPCs:
      - VPCId: !Ref VpcId
        VPCRegion: !Ref "AWS::Region"

  ExternalApiServerRecord:
    Type: AWS::Route53::RecordSetGroup
    Properties:
      Comment: Alias record for the API server
      HostedZoneId: !Ref HostedZoneId
      RecordSets:
      - Name:
```

```
        !Join [
          ".",
          ["api", !Ref ClusterName, !Join ["", [!Ref HostedZoneName, "."]]],
        ]
      Type: A
      AliasTarget:
        HostedZoneId: !GetAtt ExtApiElb.CanonicalHostedZoneID
        DNSName: !GetAtt ExtApiElb.DNSName

InternalApiServerRecord:
  Type: AWS::Route53::RecordSetGroup
  Properties:
    Comment: Alias record for the API server
    HostedZoneId: !Ref IntDns
    RecordSets:
    - Name:
        !Join [
          ".",
          ["api", !Ref ClusterName, !Join ["", [!Ref HostedZoneName, "."]]],
        ]
      Type: A
      AliasTarget:
        HostedZoneId: !GetAtt IntApiElb.CanonicalHostedZoneID
        DNSName: !GetAtt IntApiElb.DNSName
    - Name:
        !Join [
          ".",
          ["api-int", !Ref ClusterName, !Join ["", [!Ref HostedZoneName, "."]]],
        ]
      Type: A
      AliasTarget:
        HostedZoneId: !GetAtt IntApiElb.CanonicalHostedZoneID
        DNSName: !GetAtt IntApiElb.DNSName

ExternalApiListener:
  Type: AWS::ElasticLoadBalancingV2::Listener
  Properties:
    DefaultActions:
    - Type: forward
      TargetGroupArn:
        Ref: ExternalApiTargetGroup
    LoadBalancerArn:
      Ref: ExtApiElb
    Port: 6443
    Protocol: TCP

ExternalApiTargetGroup:
  Type: AWS::ElasticLoadBalancingV2::TargetGroup
  Properties:
    HealthCheckIntervalSeconds: 10
    HealthCheckPath: "/readyz"
    HealthCheckPort: 6443
    HealthCheckProtocol: HTTPS
    HealthyThresholdCount: 2
    UnhealthyThresholdCount: 2
    Port: 6443
```

```
    Protocol: TCP
    TargetType: ip
    VpcId:
      Ref: VpcId
    TargetGroupAttributes:
    - Key: deregistration_delay.timeout_seconds
      Value: 60

  InternalApiListener:
    Type: AWS::ElasticLoadBalancingV2::Listener
    Properties:
      DefaultActions:
      - Type: forward
        TargetGroupArn:
          Ref: InternalApiTargetGroup
      LoadBalancerArn:
        Ref: IntApiElb
      Port: 6443
      Protocol: TCP

  InternalApiTargetGroup:
    Type: AWS::ElasticLoadBalancingV2::TargetGroup
    Properties:
      HealthCheckIntervalSeconds: 10
      HealthCheckPath: "/readyz"
      HealthCheckPort: 6443
      HealthCheckProtocol: HTTPS
      HealthyThresholdCount: 2
      UnhealthyThresholdCount: 2
      Port: 6443
      Protocol: TCP
      TargetType: ip
      VpcId:
        Ref: VpcId
      TargetGroupAttributes:
      - Key: deregistration_delay.timeout_seconds
        Value: 60

  InternalServiceInternalListener:
    Type: AWS::ElasticLoadBalancingV2::Listener
    Properties:
      DefaultActions:
      - Type: forward
        TargetGroupArn:
          Ref: InternalServiceTargetGroup
      LoadBalancerArn:
        Ref: IntApiElb
      Port: 22623
      Protocol: TCP

  InternalServiceTargetGroup:
    Type: AWS::ElasticLoadBalancingV2::TargetGroup
    Properties:
      HealthCheckIntervalSeconds: 10
      HealthCheckPath: "/healthz"
      HealthCheckPort: 22623
```

```
    HealthCheckProtocol: HTTPS
    HealthyThresholdCount: 2
    UnhealthyThresholdCount: 2
    Port: 22623
    Protocol: TCP
    TargetType: ip
    VpcId:
      Ref: VpcId
    TargetGroupAttributes:
    - Key: deregistration_delay.timeout_seconds
      Value: 60

  RegisterTargetLambdaIamRole:
    Type: AWS::IAM::Role
    Properties:
      RoleName: !Join ["-", [!Ref InfrastructureName, "nlb", "lambda", "role"]]
      AssumeRolePolicyDocument:
        Version: "2012-10-17"
        Statement:
        - Effect: "Allow"
          Principal:
            Service:
            - "lambda.amazonaws.com"
          Action:
          - "sts:AssumeRole"
      Path: "/"
      Policies:
      - PolicyName: !Join ["-", [!Ref InfrastructureName, "master", "policy"]]
        PolicyDocument:
          Version: "2012-10-17"
          Statement:
          - Effect: "Allow"
            Action:
              [
                "elasticloadbalancing:RegisterTargets",
                "elasticloadbalancing:DeregisterTargets",
              ]
            Resource: !Ref InternalApiTargetGroup
          - Effect: "Allow"
            Action:
              [
                "elasticloadbalancing:RegisterTargets",
                "elasticloadbalancing:DeregisterTargets",
              ]
            Resource: !Ref InternalServiceTargetGroup
          - Effect: "Allow"
            Action:
              [
                "elasticloadbalancing:RegisterTargets",
                "elasticloadbalancing:DeregisterTargets",
              ]
            Resource: !Ref ExternalApiTargetGroup

  RegisterNlbIpTargets:
    Type: "AWS::Lambda::Function"
    Properties:
```

```
      Handler: "index.handler"
      Role:
        Fn::GetAtt:
        - "RegisterTargetLambdaIamRole"
        - "Arn"
      Code:
        ZipFile: |
          import json
          import boto3
          import cfnresponse
          def handler(event, context):
            elb = boto3.client('elbv2')
            if event['RequestType'] == 'Delete':
              elb.deregister_targets(TargetGroupArn=event['ResourceProperties']
['TargetArn'],Targets=[{'Id': event['ResourceProperties']['TargetIp']}])
            elif event['RequestType'] == 'Create':
              elb.register_targets(TargetGroupArn=event['ResourceProperties']['TargetArn'],Targets=
[{'Id': event['ResourceProperties']['TargetIp']}])
            responseData = {}
            cfnresponse.send(event, context, cfnresponse.SUCCESS, responseData,
event['ResourceProperties']['TargetArn']+event['ResourceProperties']['TargetIp'])
      Runtime: "python3.8"
      Timeout: 120

  RegisterSubnetTagsLambdaIamRole:
    Type: AWS::IAM::Role
    Properties:
      RoleName: !Join ["-", [!Ref InfrastructureName, "subnet-tags-lambda-role"]]
      AssumeRolePolicyDocument:
        Version: "2012-10-17"
        Statement:
        - Effect: "Allow"
          Principal:
            Service:
            - "lambda.amazonaws.com"
          Action:
          - "sts:AssumeRole"
      Path: "/"
      Policies:
      - PolicyName: !Join ["-", [!Ref InfrastructureName, "subnet-tagging-policy"]]
        PolicyDocument:
          Version: "2012-10-17"
          Statement:
          - Effect: "Allow"
            Action:
              [
                "ec2:DeleteTags",
                "ec2:CreateTags"
              ]
            Resource: "arn:aws:ec2:*:*:subnet/*"
          - Effect: "Allow"
            Action:
              [
                "ec2:DescribeSubnets",
                "ec2:DescribeTags"
              ]
```

```
          Resource: "*"

  RegisterSubnetTags:
    Type: "AWS::Lambda::Function"
    Properties:
      Handler: "index.handler"
      Role:
        Fn::GetAtt:
        - "RegisterSubnetTagsLambdaIamRole"
        - "Arn"
      Code:
        ZipFile: |
          import json
          import boto3
          import cfnresponse
          def handler(event, context):
            ec2_client = boto3.client('ec2')
            if event['RequestType'] == 'Delete':
              for subnet_id in event['ResourceProperties']['Subnets']:
                ec2_client.delete_tags(Resources=[subnet_id], Tags=[{'Key': 'kubernetes.io/cluster/' +
event['ResourceProperties']['InfrastructureName']}]);
            elif event['RequestType'] == 'Create':
              for subnet_id in event['ResourceProperties']['Subnets']:
                ec2_client.create_tags(Resources=[subnet_id], Tags=[{'Key': 'kubernetes.io/cluster/' +
event['ResourceProperties']['InfrastructureName'], 'Value': 'shared'}]);
            responseData = {}
            cfnresponse.send(event, context, cfnresponse.SUCCESS, responseData,
event['ResourceProperties']['InfrastructureName']+event['ResourceProperties']['Subnets'][0])
      Runtime: "python3.8"
      Timeout: 120

  RegisterPublicSubnetTags:
    Type: Custom::SubnetRegister
    Properties:
      ServiceToken: !GetAtt RegisterSubnetTags.Arn
      InfrastructureName: !Ref InfrastructureName
      Subnets: !Ref PublicSubnets

  RegisterPrivateSubnetTags:
    Type: Custom::SubnetRegister
    Properties:
      ServiceToken: !GetAtt RegisterSubnetTags.Arn
      InfrastructureName: !Ref InfrastructureName
      Subnets: !Ref PrivateSubnets

Outputs:
  PrivateHostedZoneId:
    Description: Hosted zone ID for the private DNS, which is required for private records.
    Value: !Ref IntDns
  ExternalApiLoadBalancerName:
    Description: Full name of the external API load balancer.
    Value: !GetAtt ExtApiElb.LoadBalancerFullName
  InternalApiLoadBalancerName:
    Description: Full name of the internal API load balancer.
    Value: !GetAtt IntApiElb.LoadBalancerFullName
  ApiServerDnsName:
```

```
    Description: Full hostname of the API server, which is required for the Ignition config files.
    Value: !Join [".", ["api-int", !Ref ClusterName, !Ref HostedZoneName]]
  RegisterNlbIpTargetsLambda:
    Description: Lambda ARN useful to help register or deregister IP targets for these load
balancers.
    Value: !GetAtt RegisterNlbIpTargets.Arn
  ExternalApiTargetGroupArn:
    Description: ARN of the external API target group.
    Value: !Ref ExternalApiTargetGroup
  InternalApiTargetGroupArn:
    Description: ARN of the internal API target group.
    Value: !Ref InternalApiTargetGroup
  InternalServiceTargetGroupArn:
    Description: ARN of the internal service target group.
    Value: !Ref InternalServiceTargetGroup
```

重要

如果要将集群部署到 AWS 政府或 secret 区域，您必须更新 **InternalApiServerRecord** 以使用 **CNAME** 记录。AWS 政府区不支持 **ALIAS** 类型的记录。例如：

```
Type: CNAME
TTL: 10
ResourceRecords:
- !GetAtt IntApiElb.DNSName
```

其他资源

- 您可以通过导航 AWS CloudFormation 控制台来查看您创建的 CloudFormation 堆栈的详情。
- 您可以通过导航到 AWS Route 53 控制台 来查看托管区的详情。
- 有关列出公共托管区的更多信息，请参阅 AWS 文档中的列出公共托管区。

6.4.3.6. 在 AWS 中创建安全组和角色

您必须在 Amazon Web Services (AWS) 中创建安全组和角色，供您的 OpenShift Container Platform 集群使用。

您可以使用提供的 CloudFormation 模板和自定义参数文件来创建 AWS 资源堆栈。堆栈代表 OpenShift Container Platform 集群所需的安全组和角色。

注意

如果不使用提供的 CloudFormation 模板来创建 AWS 基础架构，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 已配置了一个 AWS 帐户。
- 您可以通过运行 **aws configure**，将 AWS 密钥和区域添加到本地 AWS 配置集中。

- 已为集群生成 Ignition 配置文件。
- 您在 AWS 中创建并配置了 VPC 及相关子网。

流程

1. 创建一个 JSON 文件，其包含模板所需的参数值：

```
[
  {
    "ParameterKey": "InfrastructureName", 1
    "ParameterValue": "mycluster-<random_string>" 2
  },
  {
    "ParameterKey": "VpcCidr", 3
    "ParameterValue": "10.0.0.0/16" 4
  },
  {
    "ParameterKey": "PrivateSubnets", 5
    "ParameterValue": "subnet-<random_string>" 6
  },
  {
    "ParameterKey": "VpcId", 7
    "ParameterValue": "vpc-<random_string>" 8
  }
]
```

1. 您的 Ignition 配置文件中为集群编码的集群基础架构名称。
2. 指定从 Ignition 配置文件元数据中提取的基础架构名称，其格式为 **<cluster-name>-<random-string>**。
3. VPC 的 CIDR 块。
4. 指定以 **x.x.x.x/16-24** 格式定义的用于 VPC 的 CIDR 地址块。
5. 为 VPC 创建的专用子网。
6. 指定 VPC 的 CloudFormation 模板输出的 **PrivateSubnetIds** 值。
7. 为集群创建的 VPC。
8. 指定 VPC 的 CloudFormation 模板输出的 **VpcId** 值。

2. 复制本主题的**安全对象的 CloudFormation 模板**部分中的模板，并将它以 YAML 文件形式保存到计算机上。此模板描述了集群所需的安全组和角色。
3. 启动 CloudFormation 模板，以创建代表安全组和角色的 AWS 资源堆栈：

重要

您必须在一行内输入命令。

```
$ aws cloudformation create-stack --stack-name <name> 1
    --template-body file://<template>.yaml 2
    --parameters file://<parameters>.json 3
    --capabilities CAPABILITY_NAMED_IAM 4
```

1. **<name>** 是 CloudFormation 堆栈的名称，如 **cluster-sec**。如果您删除集群，则需要此堆栈的名称。
2. **<template>** 是您保存的 CloudFormation 模板 YAML 文件的相对路径和名称。
3. **<parameters>** 是 CloudFormation 参数 JSON 文件的相对路径和名称。
4. 您必须明确声明 **CAPABILITY_NAMED_IAM** 功能，因为提供的模板会创建一些 **AWS::IAM::Role** 和 **AWS::IAM::InstanceProfile** 资源。

输出示例

```
arn:aws:cloudformation:us-east-1:269333783861:stack/cluster-sec/03bd4210-2ed7-11eb-6d7a-13fc0b61e9db
```

4. 确认模板组件已存在：

```
$ aws cloudformation describe-stacks --stack-name <name>
```

在 **StackStatus** 显示 **CREATE_COMPLETE** 后，输出会显示以下参数的值。您必须将这些参数值提供给您在创建集群时要运行的其他 CloudFormation 模板：

MasterSecurityGroupId	Master 安全组 ID
WorkerSecurityGroupId	worker 安全组 ID
MasterInstanceProfile	Master IAM 实例配置集
WorkerInstanceProfile	worker IAM 实例配置集

6.4.3.6.1. 安全对象的 CloudFormation 模板

您可以使用以下 CloudFormation 模板来部署 OpenShift Container Platform 集群所需的安全对象。

例 6.74. 安全对象的 CloudFormation 模板

```
AWSTemplateFormatVersion: 2010-09-09
```

```
Description: Template for OpenShift Cluster Security Elements (Security Groups & IAM)

Parameters:
  InfrastructureName:
    AllowedPattern: ^([a-zA-Z][a-zA-Z0-9\-]{0,26})$
    MaxLength: 27
    MinLength: 1
    ConstraintDescription: Infrastructure name must be alphanumeric, start with a letter, and have a
maximum of 27 characters.
    Description: A short, unique cluster ID used to tag cloud resources and identify items owned or
used by the cluster.
    Type: String
  VpcCidr:
    AllowedPattern: ^(([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])\.){3}([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-
4][0-9]|25[0-5])(\/(1[6-9]|2[0-4]))$
    ConstraintDescription: CIDR block parameter must be in the form x.x.x.x/16-24.
    Default: 10.0.0.0/16
    Description: CIDR block for VPC.
    Type: String
  VpcId:
    Description: The VPC-scoped resources will belong to this VPC.
    Type: AWS::EC2::VPC::Id
  PrivateSubnets:
    Description: The internal subnets.
    Type: List<AWS::EC2::Subnet::Id>

Metadata:
  AWS::CloudFormation::Interface:
    ParameterGroups:
    - Label:
        default: "Cluster Information"
      Parameters:
      - InfrastructureName
    - Label:
        default: "Network Configuration"
      Parameters:
      - VpcId
      - VpcCidr
      - PrivateSubnets
    ParameterLabels:
      InfrastructureName:
        default: "Infrastructure Name"
      VpcId:
        default: "VPC ID"
      VpcCidr:
        default: "VPC CIDR"
      PrivateSubnets:
        default: "Private Subnets"

Resources:
  MasterSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: Cluster Master Security Group
      SecurityGroupIngress:
      - IpProtocol: icmp
```

```
      FromPort: 0
      ToPort: 0
      CidrIp: !Ref VpcCidr
    - IpProtocol: tcp
      FromPort: 22
      ToPort: 22
      CidrIp: !Ref VpcCidr
    - IpProtocol: tcp
      ToPort: 6443
      FromPort: 6443
      CidrIp: !Ref VpcCidr
    - IpProtocol: tcp
      FromPort: 22623
      ToPort: 22623
      CidrIp: !Ref VpcCidr
    VpcId: !Ref VpcId

WorkerSecurityGroup:
  Type: AWS::EC2::SecurityGroup
  Properties:
    GroupDescription: Cluster Worker Security Group
    SecurityGroupIngress:
    - IpProtocol: icmp
      FromPort: 0
      ToPort: 0
      CidrIp: !Ref VpcCidr
    - IpProtocol: tcp
      FromPort: 22
      ToPort: 22
      CidrIp: !Ref VpcCidr
    VpcId: !Ref VpcId

MasterIngressEtcd:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: etcd
    FromPort: 2379
    ToPort: 2380
    IpProtocol: tcp

MasterIngressVxlan:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: Vxlan packets
    FromPort: 4789
    ToPort: 4789
    IpProtocol: udp

MasterIngressWorkerVxlan:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
```

```
      SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
      Description: Vxlan packets
      FromPort: 4789
      ToPort: 4789
      IpProtocol: udp

  MasterIngressGeneve:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt MasterSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
      Description: Geneve packets
      FromPort: 6081
      ToPort: 6081
      IpProtocol: udp

  MasterIngressWorkerGeneve:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt MasterSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
      Description: Geneve packets
      FromPort: 6081
      ToPort: 6081
      IpProtocol: udp

  MasterIngressIpsecIke:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt MasterSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
      Description: IPsec IKE packets
      FromPort: 500
      ToPort: 500
      IpProtocol: udp

  MasterIngressIpsecNat:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt MasterSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
      Description: IPsec NAT-T packets
      FromPort: 4500
      ToPort: 4500
      IpProtocol: udp

  MasterIngressIpsecEsp:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt MasterSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
      Description: IPsec ESP packets
      IpProtocol: 50

  MasterIngressWorkerIpsecIke:
    Type: AWS::EC2::SecurityGroupIngress
```

```
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
    Description: IPsec IKE packets
    FromPort: 500
    ToPort: 500
    IpProtocol: udp

MasterIngressWorkerIpsecNat:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
    Description: IPsec NAT-T packets
    FromPort: 4500
    ToPort: 4500
    IpProtocol: udp

MasterIngressWorkerIpsecEsp:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
    Description: IPsec ESP packets
    IpProtocol: 50

MasterIngressInternal:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: Internal cluster communication
    FromPort: 9000
    ToPort: 9999
    IpProtocol: tcp

MasterIngressWorkerInternal:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
    Description: Internal cluster communication
    FromPort: 9000
    ToPort: 9999
    IpProtocol: tcp

MasterIngressInternalUDP:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: Internal cluster communication
    FromPort: 9000
    ToPort: 9999
    IpProtocol: udp
```

```
  MasterIngressWorkerInternalUDP:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt MasterSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
      Description: Internal cluster communication
      FromPort: 9000
      ToPort: 9999
      IpProtocol: udp

  MasterIngressKube:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt MasterSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
      Description: Kubernetes kubelet, scheduler and controller manager
      FromPort: 10250
      ToPort: 10259
      IpProtocol: tcp

  MasterIngressWorkerKube:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt MasterSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
      Description: Kubernetes kubelet, scheduler and controller manager
      FromPort: 10250
      ToPort: 10259
      IpProtocol: tcp

  MasterIngressIngressServices:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt MasterSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
      Description: Kubernetes ingress services
      FromPort: 30000
      ToPort: 32767
      IpProtocol: tcp

  MasterIngressWorkerIngressServices:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt MasterSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
      Description: Kubernetes ingress services
      FromPort: 30000
      ToPort: 32767
      IpProtocol: tcp

  MasterIngressIngressServicesUDP:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt MasterSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
      Description: Kubernetes ingress services
```

```
    FromPort: 30000
    ToPort: 32767
    IpProtocol: udp

MasterIngressWorkerIngressServicesUDP:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
    Description: Kubernetes ingress services
    FromPort: 30000
    ToPort: 32767
    IpProtocol: udp

WorkerIngressVxlan:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
    Description: Vxlan packets
    FromPort: 4789
    ToPort: 4789
    IpProtocol: udp

WorkerIngressMasterVxlan:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: Vxlan packets
    FromPort: 4789
    ToPort: 4789
    IpProtocol: udp

WorkerIngressGeneve:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
    Description: Geneve packets
    FromPort: 6081
    ToPort: 6081
    IpProtocol: udp

WorkerIngressMasterGeneve:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: Geneve packets
    FromPort: 6081
    ToPort: 6081
    IpProtocol: udp

WorkerIngressIpsecIke:
  Type: AWS::EC2::SecurityGroupIngress
```

```
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
    Description: IPsec IKE packets
    FromPort: 500
    ToPort: 500
    IpProtocol: udp

WorkerIngressIpsecNat:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
    Description: IPsec NAT-T packets
    FromPort: 4500
    ToPort: 4500
    IpProtocol: udp

WorkerIngressIpsecEsp:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
    Description: IPsec ESP packets
    IpProtocol: 50

WorkerIngressMasterIpsecIke:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: IPsec IKE packets
    FromPort: 500
    ToPort: 500
    IpProtocol: udp

WorkerIngressMasterIpsecNat:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: IPsec NAT-T packets
    FromPort: 4500
    ToPort: 4500
    IpProtocol: udp

WorkerIngressMasterIpsecEsp:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: IPsec ESP packets
    IpProtocol: 50

WorkerIngressInternal:
  Type: AWS::EC2::SecurityGroupIngress
```

```
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
    Description: Internal cluster communication
    FromPort: 9000
    ToPort: 9999
    IpProtocol: tcp

WorkerIngressMasterInternal:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: Internal cluster communication
    FromPort: 9000
    ToPort: 9999
    IpProtocol: tcp

WorkerIngressInternalUDP:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
    Description: Internal cluster communication
    FromPort: 9000
    ToPort: 9999
    IpProtocol: udp

WorkerIngressMasterInternalUDP:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: Internal cluster communication
    FromPort: 9000
    ToPort: 9999
    IpProtocol: udp

WorkerIngressKube:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
    Description: Kubernetes secure kubelet port
    FromPort: 10250
    ToPort: 10250
    IpProtocol: tcp

WorkerIngressWorkerKube:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: Internal Kubernetes communication
    FromPort: 10250
    ToPort: 10250
```

```
    IpProtocol: tcp

WorkerIngressIngressServices:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
    Description: Kubernetes ingress services
    FromPort: 30000
    ToPort: 32767
    IpProtocol: tcp

WorkerIngressMasterIngressServices:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: Kubernetes ingress services
    FromPort: 30000
    ToPort: 32767
    IpProtocol: tcp

WorkerIngressIngressServicesUDP:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
    Description: Kubernetes ingress services
    FromPort: 30000
    ToPort: 32767
    IpProtocol: udp

WorkerIngressMasterIngressServicesUDP:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: Kubernetes ingress services
    FromPort: 30000
    ToPort: 32767
    IpProtocol: udp

MasterIamRole:
  Type: AWS::IAM::Role
  Properties:
    AssumeRolePolicyDocument:
      Version: "2012-10-17"
      Statement:
      - Effect: "Allow"
        Principal:
          Service:
          - "ec2.amazonaws.com"
        Action:
        - "sts:AssumeRole"
    Policies:
    - PolicyName: !Join ["-", [!Ref InfrastructureName, "master", "policy"]]
```

```
      PolicyDocument:
        Version: "2012-10-17"
        Statement:
        - Effect: "Allow"
          Action:
          - "ec2:AttachVolume"
          - "ec2:AuthorizeSecurityGroupIngress"
          - "ec2:CreateSecurityGroup"
          - "ec2:CreateTags"
          - "ec2:CreateVolume"
          - "ec2:DeleteSecurityGroup"
          - "ec2:DeleteVolume"
          - "ec2:Describe*"
          - "ec2:DetachVolume"
          - "ec2:ModifyInstanceAttribute"
          - "ec2:ModifyVolume"
          - "ec2:RevokeSecurityGroupIngress"
          - "elasticloadbalancing:AddTags"
          - "elasticloadbalancing:AttachLoadBalancerToSubnets"
          - "elasticloadbalancing:ApplySecurityGroupsToLoadBalancer"
          - "elasticloadbalancing:CreateListener"
          - "elasticloadbalancing:CreateLoadBalancer"
          - "elasticloadbalancing:CreateLoadBalancerPolicy"
          - "elasticloadbalancing:CreateLoadBalancerListeners"
          - "elasticloadbalancing:CreateTargetGroup"
          - "elasticloadbalancing:ConfigureHealthCheck"
          - "elasticloadbalancing:DeleteListener"
          - "elasticloadbalancing:DeleteLoadBalancer"
          - "elasticloadbalancing:DeleteLoadBalancerListeners"
          - "elasticloadbalancing:DeleteTargetGroup"
          - "elasticloadbalancing:DeregisterInstancesFromLoadBalancer"
          - "elasticloadbalancing:DeregisterTargets"
          - "elasticloadbalancing:Describe*"
          - "elasticloadbalancing:DetachLoadBalancerFromSubnets"
          - "elasticloadbalancing:ModifyListener"
          - "elasticloadbalancing:ModifyLoadBalancerAttributes"
          - "elasticloadbalancing:ModifyTargetGroup"
          - "elasticloadbalancing:ModifyTargetGroupAttributes"
          - "elasticloadbalancing:RegisterInstancesWithLoadBalancer"
          - "elasticloadbalancing:RegisterTargets"
          - "elasticloadbalancing:SetLoadBalancerPoliciesForBackendServer"
          - "elasticloadbalancing:SetLoadBalancerPoliciesOfListener"
          - "kms:DescribeKey"
          Resource: "*"

  MasterInstanceProfile:
    Type: "AWS::IAM::InstanceProfile"
    Properties:
      Roles:
      - Ref: "MasterIamRole"

  WorkerIamRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: "2012-10-17"
```

```
        Statement:
        - Effect: "Allow"
          Principal:
            Service:
            - "ec2.amazonaws.com"
          Action:
          - "sts:AssumeRole"
      Policies:
      - PolicyName: !Join ["-", [!Ref InfrastructureName, "worker", "policy"]]
        PolicyDocument:
          Version: "2012-10-17"
          Statement:
          - Effect: "Allow"
            Action:
            - "ec2:DescribeInstances"
            - "ec2:DescribeRegions"
            Resource: "*"

  WorkerInstanceProfile:
    Type: "AWS::IAM::InstanceProfile"
    Properties:
      Roles:
      - Ref: "WorkerIamRole"

Outputs:
  MasterSecurityGroupId:
    Description: Master Security Group ID
    Value: !GetAtt MasterSecurityGroup.GroupId

  WorkerSecurityGroupId:
    Description: Worker Security Group ID
    Value: !GetAtt WorkerSecurityGroup.GroupId

  MasterInstanceProfile:
    Description: Master IAM Instance Profile
    Value: !Ref MasterInstanceProfile

  WorkerInstanceProfile:
    Description: Worker IAM Instance Profile
    Value: !Ref WorkerInstanceProfile
```

其他资源

- 您可以通过导航 AWS CloudFormation 控制台来查看您创建的 CloudFormation 堆栈的详情。

6.4.3.7. 使用流元数据访问 RHCOS AMI

在 OpenShift Container Platform 中，*流元数据*以 JSON 格式提供与 RHCOS 相关的标准化元数据，并将元数据注入集群中。流元数据是一种稳定的格式，支持多种架构，旨在自我记录以维护自动化。

您可以使用 **openshift-install** 的 **coreos print-stream-json** 子命令访问流元数据格式的引导镜像的信息。此命令提供了一种以可脚本、机器可读格式打印流元数据的方法。

对于用户置备的安装，**openshift-install** 二进制文件包含对经过测试用于 OpenShift Container Platform 的 RHCOS 引导镜像版本的引用，如 AWS AMI。

流程

要解析流元数据，请使用以下方法之一：

- 在 Go 程序中使用位于 https://github.com/coreos/stream-metadata-go 的正式 **stream-metadata-go** 库。您还可以查看库中的示例代码。
- 在 Python 或 Ruby 等其他编程语言中使用您首选编程语言的 JSON 库。
- 在处理 JSON 数据的命令行工具中，如 **jq**：
  - 为 AWS 区域输出当前的 **x86_64** 或 **aarch64** AMI，如 **us-west-1** ：

    对于 x86_64

    ```
    $ openshift-install coreos print-stream-json | jq -r
    '.architectures.x86_64.images.aws.regions["us-west-1"].image'
    ```

    输出示例

    ```
    ami-0d3e625f84626bbda
    ```

    对于 aarch64

    ```
    $ openshift-install coreos print-stream-json | jq -r
    '.architectures.aarch64.images.aws.regions["us-west-1"].image'
    ```

    输出示例

    ```
    ami-0af1d3b7fa5be2131
    ```

    这个命令的输出是您指定的架构和 **us-west-1** 区域的 AWS AMI ID。AMI 必须与集群属于同一区域。

6.4.3.8. AWS 基础架构的 RHCOS AMI

红帽提供了对可手动为 OpenShift Container Platform 节点指定的各种 AWS 区域和实例架构有效的 Red Hat Enterprise Linux CoreOS(RHCOS)AMI。

注意

通过导入您自己的 AMI，您还可以安装到没有公布的 RHCOS AMI 的区域。

表 6.23. x86_64 RHCOS AMIs

AWS 区	AWS AMI
af-south-1	**ami-018de6b1be470b7e6**

AWS 区	AWS AMI
ap-east-1	ami-092f09a4c8aacf56a
ap-northeast-1	ami-001c9fc85b77505f4
ap-northeast-2	ami-0a5d7f1966d88fba3
ap-northeast-3	ami-085a2726ceb4f5eeb
ap-south-1	ami-03f2c19071fb6eab3
ap-south-2	ami-0c82522890979d94b
ap-southeast-1	ami-06e5b9d16ead6c55c
ap-southeast-2	ami-0ccd429129fbf6bc0
ap-southeast-3	ami-096f9b4d41116d95c
ap-southeast-4	ami-0b511ab55f9f7d052
ca-central-1	ami-0e357287c651be1f2
ca-west-1	ami-0b1692e5776740901
eu-central-1	ami-08b13fb388ba5610d
eu-central-2	ami-04777298b55045ea9
eu-north-1	ami-05421993a4ee567b9
eu-south-1	ami-00959341869d34746
eu-south-2	ami-0a745b610522a87cc
eu-west-1	ami-0e48f85ec57bd7baa
eu-west-2	ami-0c015b1387acddc5e
eu-west-3	ami-01e466af77a95b93d
il-central-1	ami-0a1b706793b54d087
me-central-1	ami-09d58e8b2099ae5bc
me-south-1	ami-0f9c259bc4346599c

AWS 区	AWS AMI
sa-east-1	ami-06277517d966881a3
us-east-1	ami-05483066c3caaccf5
us-east-2	ami-0c704ce516493a479
us-gov-east-1	ami-0695b2cbdd1ec4d48
us-gov-west-1	ami-004404a0eaa427b39
us-west-1	ami-0aaa56637063be772
us-west-2	ami-0870c7a3d5ef4d2b3

表 6.24. aarch64 RHCOS AMI

AWS 区	AWS AMI
af-south-1	ami-0d96d447d3df14f4e
ap-east-1	ami-010236402dfba1717
ap-northeast-1	ami-08feeb6d9068bb12e
ap-northeast-2	ami-0772082c119147205
ap-northeast-3	ami-05bcbb13493d0516f
ap-south-1	ami-0034a539e8adcb817
ap-south-2	ami-0fc56b7c53c38ff70
ap-southeast-1	ami-0b50a0e92a58f3ad8
ap-southeast-2	ami-0697a9174ae206182
ap-southeast-3	ami-05ae309319a7ad504
ap-southeast-4	ami-00500414843baa657
ca-central-1	ami-05e76bf99d3b0d4c8
ca-west-1	ami-099fc953c221ec590

AWS 区	AWS AMI
eu-central-1	**ami-0d2e2698884020b71**
eu-central-2	**ami-0a96ba21ddee01719**
eu-north-1	**ami-0ab8a1070d0b1d9a5**
eu-south-1	**ami-0d0465299a8b196c1**
eu-south-2	**ami-07d5aa3c6d468c738**
eu-west-1	**ami-08e7d209f26c09c80**
eu-west-2	**ami-0c8336d4e2c1f13cb**
eu-west-3	**ami-085d804b98acf0648**
il-central-1	**ami-02ce030e36aeb7643**
me-central-1	**ami-0dea3ac466040b77f**
me-south-1	**ami-02ef2a46887b9786d**
sa-east-1	**ami-0d19817d31b2399f9**
us-east-1	**ami-04859c888566a2c2f**
us-east-2	**ami-02f7e4a5dc66361bb**
us-gov-east-1	**ami-067ef782980ea96ad**
us-gov-west-1	**ami-0ce67540c1bb2319d**
us-west-1	**ami-0a09c5d2f287718a3**
us-west-2	**ami-06b94e64e595f6cee**

6.4.3.8.1. 没有公布的 RHCOS AMI 的 AWS 区域

您可以将 OpenShift Container Platform 集群部署到 Amazon Web Services（AWS）区域，而无需对 Red Hat Enterprise Linux CoreOS（RHCOS）Amazon Machine Image（AMI）或 AWS 软件开发 kit（SDK）的原生支持。如果 AWS 区域没有可用的已公布的 AMI，您可以在安装集群前上传自定义 AMI。

如果您要部署到 AWS SDK 不支持的区域，且您没有指定自定义 AMI，安装程序会自动将 **us-east-1** AMI 复制到用户帐户。然后，安装程序使用默认或用户指定的密钥管理服务（KMS）密钥创建带有加密 EBS 卷的 control plane 机器。这允许 AMI 跟踪与公布的 RHCOS AMI 相同的进程工作流。

在集群创建过程中，无法从终端中选择没有原生支持 RHCOS AMI 的区域，因为它没有发布。但是，您可以通过在 **install-config.yaml** 文件中配置自定义 AMI 来安装到这个区域。

6.4.3.8.2. 在 AWS 中上传自定义 RHCOS AMI

如果要部署到自定义 Amazon Web Services（AWS）区域，您必须上传属于该区域的自定义 Red Hat Enterprise Linux CoreOS（RHCOS）Amazon Machine Image（AMI）。

先决条件

- 已配置了一个 AWS 帐户。
- 已使用所需的 IAM 服务角色创建 Amazon S3 存储桶。
- 将 RHCOS VMDK 文件上传到 Amazon S3。RHCOS VMDK 文件必须是小于或等于您要安装的 OpenShift Container Platform 版本的最高版本。
- 您下载了 AWS CLI 并安装到您的计算机上。请参阅使用捆绑安装程序安装 AWS CLI。

流程

1. 将 AWS 配置集导出为环境变量：

```
$ export AWS_PROFILE=<aws_profile> 1
```

2. 将与自定义 AMI 关联的区域导出为环境变量：

```
$ export AWS_DEFAULT_REGION=<aws_region> 1
```

3. 将上传至 Amazon S3 的 RHCOS 版本导出为环境变量：

```
$ export RHCOS_VERSION=<version> 1
```

1 1 1 RHCOS VMDK 版本，如 **4.16.0**。

4. 将 Amazon S3 存储桶名称导出为环境变量：

```
$ export VMIMPORT_BUCKET_NAME=<s3_bucket_name>
```

5. 创建 **containers.json** 文件并定义 RHCOS VMDK 文件：

```
$ cat <<EOF > containers.json
{
  "Description": "rhcos-${RHCOS_VERSION}-x86_64-aws.x86_64",
  "Format": "vmdk",
  "UserBucket": {
    "S3Bucket": "${VMIMPORT_BUCKET_NAME}",
    "S3Key": "rhcos-${RHCOS_VERSION}-x86_64-aws.x86_64.vmdk"
  }
}
EOF
```

6. 将 RHCOS 磁盘导入为 Amazon EBS 快照：

```
$ aws ec2 import-snapshot --region ${AWS_DEFAULT_REGION} \
    --description "<description>" \ 1
    --disk-container "file://<file_path>/containers.json" 2
```

1 导入 RHCOS 磁盘的描述，如 **rhcos-${RHCOS_VERSION}-x86_64-aws.x86_64**。

2 描述 RHCOS 磁盘的 JSON 文件的文件路径。JSON 文件应包含您的 Amazon S3 存储桶名称和密钥。

7. 检查镜像导入的状态：

```
$ watch -n 5 aws ec2 describe-import-snapshot-tasks --region ${AWS_DEFAULT_REGION}
```

输出示例

```
{
    "ImportSnapshotTasks": [
        {
            "Description": "rhcos-4.7.0-x86_64-aws.x86_64",
            "ImportTaskId": "import-snap-fh6i8uil",
            "SnapshotTaskDetail": {
                "Description": "rhcos-4.7.0-x86_64-aws.x86_64",
                "DiskImageSize": 819056640.0,
                "Format": "VMDK",
                "SnapshotId": "snap-06331325870076318",
                "Status": "completed",
                "UserBucket": {
                    "S3Bucket": "external-images",
                    "S3Key": "rhcos-4.7.0-x86_64-aws.x86_64.vmdk"
                }
            }
        }
    ]
}
```

复制 **SnapshotId** 以注册镜像。

8. 从 RHCOS 快照创建自定义 RHCOS AMI:

```
$ aws ec2 register-image \
  --region ${AWS_DEFAULT_REGION} \
  --architecture x86_64 \ 1
  --description "rhcos-${RHCOS_VERSION}-x86_64-aws.x86_64" \ 2
  --ena-support \
  --name "rhcos-${RHCOS_VERSION}-x86_64-aws.x86_64" \ 3
  --virtualization-type hvm \
  --root-device-name '/dev/xvda' \
  --block-device-mappings 'DeviceName=/dev/xvda,Ebs=
{DeleteOnTermination=true,SnapshotId=<snapshot_ID>}' 4
```

1 RHCOS VMDK 架构类型，如 **x86_64**, **aarch64**, **s390x**, 或 **ppc64le**。

2 来自导入快照的 **Description**。

3 RHCOS AMI 的名称。

4 导入的快照中的 **SnapshotID**。

如需了解更多有关这些 API 的信息，请参阅 AWS 文档 导入快照 和 创建由 EBS 支持的 AMI。

6.4.3.9. 在 AWS 中创建 bootstrap 节点

您必须在 Amazon Web Services (AWS) 中创建 bootstrap 节点，以便在 OpenShift Container Platform 集群初始化过程中使用。您可以按照以下方法：

- 为集群提供 **bootstrap.ign** Ignition 配置文件的位置。此文件位于您的安装目录中。提供的 CloudFormation 模板假定集群的 Ignition 配置文件由 S3 存储桶提供。如果选择从其他位置提供文件，您必须修改模板。
- 使用提供的 CloudFormation 模板和自定义参数文件来创建 AWS 资源堆栈。堆栈代表 OpenShift Container Platform 安装所需的 bootstrap 节点。

注意

如果不使用提供的 CloudFormation 模板来创建 bootstrap 节点，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 已配置了一个 AWS 帐户。
- 您可以通过运行 **aws configure**，将 AWS 密钥和区域添加到本地 AWS 配置集中。
- 已为集群生成 Ignition 配置文件。
- 您在 AWS 中创建并配置了 VPC 及相关子网。
- 您在 AWS 中创建并配置了 DNS、负载均衡器和监听程序。
- 您在 AWS 中创建了集群所需的安全组和角色。

流程

1. 运行以下命令来创建存储桶：

```
$ aws s3 mb s3://<cluster-name>-infra 1
```

1 **<cluster-name>-infra** 是存储桶名称。在创建 **install-config.yaml** 文件时，将 **<cluster-name>** 替换为为集群指定的名称。

如果需要，您必须为 S3 存储桶使用预签名 URL，而不是 **s3://** 模式：

- 部署到具有与 AWS SDK 不同端点的区域。
- 部署代理。

- 提供您自己的自定义端点。

2. 运行以下命令，将 **bootstrap.ign** Ignition 配置文件上传到存储桶：

```
$ aws s3 cp <installation_directory>/bootstrap.ign s3://<cluster-name>-infra/bootstrap.ign 1
```

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

3. 运行以下命令验证文件是否已上传：

```
$ aws s3 ls s3://<cluster-name>-infra/
```

输出示例

```
2019-04-03 16:15:16    314878 bootstrap.ign
```

注意

bootstrap Ignition 配置文件包含 secret，如 X.509 密钥。以下步骤为 S3 存储桶提供基本安全性。若要提供额外的安全性，您可以启用 S3 存储桶策略，仅允许某些用户（如 OpenShift IAM 用户）访问存储桶中包含的对象。您可以完全避开 S3，并从 bootstrap 可访问的任意地址提供 bootstrap Ignition 配置文件。

4. 创建一个 JSON 文件，其包含模板所需的参数值：

```
[
  {
    "ParameterKey": "InfrastructureName", 1
    "ParameterValue": "mycluster-<random_string>" 2
  },
  {
    "ParameterKey": "RhcosAmi", 3
    "ParameterValue": "ami-<random_string>" 4
  },
  {
    "ParameterKey": "AllowedBootstrapSshCidr", 5
    "ParameterValue": "0.0.0.0/0" 6
  },
  {
    "ParameterKey": "PublicSubnet", 7
    "ParameterValue": "subnet-<random_string>" 8
  },
  {
    "ParameterKey": "MasterSecurityGroupId", 9
    "ParameterValue": "sg-<random_string>" 10
  },
  {
    "ParameterKey": "VpcId", 11
    "ParameterValue": "vpc-<random_string>" 12
  },
  {
```

```
    "ParameterKey": "BootstrapIgnitionLocation", 13
    "ParameterValue": "s3://<bucket_name>/bootstrap.ign" 14
  },
  {
    "ParameterKey": "AutoRegisterELB", 15
    "ParameterValue": "yes" 16
  },
  {
    "ParameterKey": "RegisterNlbIpTargetsLambdaArn", 17
    "ParameterValue": "arn:aws:lambda:<aws_region>:<account_number>:function:
<dns_stack_name>-RegisterNlbIpTargets-<random_string>" 18
  },
  {
    "ParameterKey": "ExternalApiTargetGroupArn", 19
    "ParameterValue": "arn:aws:elasticloadbalancing:<aws_region>:
<account_number>:targetgroup/<dns_stack_name>-Exter-<random_string>" 20
  },
  {
    "ParameterKey": "InternalApiTargetGroupArn", 21
    "ParameterValue": "arn:aws:elasticloadbalancing:<aws_region>:
<account_number>:targetgroup/<dns_stack_name>-Inter-<random_string>" 22
  },
  {
    "ParameterKey": "InternalServiceTargetGroupArn", 23
    "ParameterValue": "arn:aws:elasticloadbalancing:<aws_region>:
<account_number>:targetgroup/<dns_stack_name>-Inter-<random_string>" 24
  }
]
```

1. 您的 Ignition 配置文件中为集群编码的集群基础架构名称。
2. 指定从 Ignition 配置文件元数据中提取的基础架构名称，其格式为 **<cluster-name>-<random-string>**。
3. 根据您选择的架构，当前 Red Hat Enterprise Linux CoreOS (RHCOS) AMI 用于 bootstrap 节点。
4. 指定有效的 **AWS::EC2::Image::Id** 值。
5. 允许通过 SSH 访问 bootstrap 节点的 CIDR 块。
6. 以 **x.x.x.x/16-24** 格式指定 CIDR 块。
7. 与 VPC 关联的公共子网，将 bootstrap 节点启动到其中。
8. 指定 VPC 的 CloudFormation 模板输出的 **PublicSubnetIds** 值。
9. master 安全组 ID（用于注册临时规则）
10. 指定安全组和角色的 CloudFormation 模板输出的 **MasterSecurityGroupId** 值。
11. 创建的资源将从属于的 VPC。
12. 指定 VPC 的 CloudFormation 模板输出的 **VpcId** 值。

13 从中获取 bootstrap Ignition 配置文件的位置。

14 指定 S3 存储桶和文件名，格式为 **s3://<bucket_name>/bootstrap.ign**。

15 是否要注册网络负载均衡器 (NLB) 。

16 指定 **yes** 或 **no**。如果指定 **yes**，您必须提供一个 Lambda Amazon Resource Name (ARN) 值。

17 NLB IP 目标注册 lambda 组的 ARN。

18 指定 DNS 和负载均衡的 CloudFormation 模板输出的 **RegisterNlbIpTargetsLambda** 值。如果将集群部署到 AWS GovCloud 区域，请使用 **arn:aws-us-gov**。

19 外部 API 负载均衡器目标组的 ARN。

20 指定 DNS 和负载均衡的 CloudFormation 模板输出的 **ExternalApiTargetGroupArn** 值。如果将集群部署到 AWS GovCloud 区域，请使用 **arn:aws-us-gov**。

21 内部 API 负载均衡器目标组群的 ARN。

22 指定 DNS 和负载均衡的 CloudFormation 模板输出的 **InternalApiTargetGroupArn** 值。如果将集群部署到 AWS GovCloud 区域，请使用 **arn:aws-us-gov**。

23 内部服务负载均衡器目标组群的 ARN。

24 指定 DNS 和负载均衡的 CloudFormation 模板输出的 **InternalServiceTargetGroupArn** 值。如果将集群部署到 AWS GovCloud 区域，请使用 **arn:aws-us-gov**。

5. 复制本主题的 **Bootstrap 机器的 CloudFormation 模板**部分中的模板，并将它以 YAML 文件形式保存到计算机上。此模板描述了集群所需的 bootstrap 机器。

6. 可选：如果要使用代理部署集群，您必须更新模板中的 ignition 以添加 **ignition.config.proxy** 字段。另外，如果您已将 Amazon EC2、Elastic Load Balancing 和 S3 VPC 端点添加到 VPC 中，您必须将这些端点添加到 **noProxy** 字段。

7. 启动 CloudFormation 模板，以创建代表 bootstrap 节点的 AWS 资源堆栈：

重要

您必须在一行内输入命令。

```
$ aws cloudformation create-stack --stack-name <name> 1
    --template-body file://<template>.yaml 2
    --parameters file://<parameters>.json 3
    --capabilities CAPABILITY_NAMED_IAM 4
```

1 **<name>** 是 CloudFormation 堆栈的名称，如 **cluster-bootstrap**。如果您删除集群，则需要此堆栈的名称。

2 **<template>** 是您保存的 CloudFormation 模板 YAML 文件的相对路径和名称。

3 **<parameters>** 是 CloudFormation 参数 JSON 文件的相对路径和名称。

4 您必须明确声明 **CAPABILITY_NAMED_IAM** 功能，因为提供的模板会创建一些 **AWS::IAM::Role** 和 **AWS::IAM::InstanceProfile** 资源。

输出示例

```
arn:aws:cloudformation:us-east-1:269333783861:stack/cluster-bootstrap/12944486-2add-11eb-9dee-12dace8e3a83
```

8. 确认模板组件已存在：

```
$ aws cloudformation describe-stacks --stack-name <name>
```

在 **StackStatus** 显示 **CREATE_COMPLETE** 后，输出会显示以下参数的值。您必须将这些参数值提供给您在创建集群时要运行的其他 CloudFormation 模板：

Bootstrap InstanceId	bootstrap 实例 ID。
Bootstrap PublicIp	bootstrap 节点公共 IP 地址。
Bootstrap PrivateIp	bootstrap 节点专用 IP 地址。

6.4.3.9.1. bootstrap 机器的 CloudFormation 模板

您可以使用以下 CloudFormation 模板来部署 OpenShift Container Platform 集群所需的 bootstrap 机器。

例 6.75. bootstrap 机器的 CloudFormation 模板

```
AWSTemplateFormatVersion: 2010-09-09
Description: Template for OpenShift Cluster Bootstrap (EC2 Instance, Security Groups and IAM)

Parameters:
  InfrastructureName:
    AllowedPattern: ^([a-zA-Z][a-zA-Z0-9\-]{0,26})$
    MaxLength: 27
    MinLength: 1
    ConstraintDescription: Infrastructure name must be alphanumeric, start with a letter, and have a maximum of 27 characters.
    Description: A short, unique cluster ID used to tag cloud resources and identify items owned or used by the cluster.
    Type: String
  RhcosAmi:
    Description: Current Red Hat Enterprise Linux CoreOS AMI to use for bootstrap.
    Type: AWS::EC2::Image::Id
  AllowedBootstrapSshCidr:
    AllowedPattern: ^(([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])\.){3}([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])(\/([0-9]|1[0-9]|2[0-9]|3[0-2]))$
    ConstraintDescription: CIDR block parameter must be in the form x.x.x.x/0-32.
    Default: 0.0.0.0/0
```

```
    Description: CIDR block to allow SSH access to the bootstrap node.
    Type: String
  PublicSubnet:
    Description: The public subnet to launch the bootstrap node into.
    Type: AWS::EC2::Subnet::Id
  MasterSecurityGroupId:
    Description: The master security group ID for registering temporary rules.
    Type: AWS::EC2::SecurityGroup::Id
  VpcId:
    Description: The VPC-scoped resources will belong to this VPC.
    Type: AWS::EC2::VPC::Id
  BootstrapIgnitionLocation:
    Default: s3://my-s3-bucket/bootstrap.ign
    Description: Ignition config file location.
    Type: String
  AutoRegisterELB:
    Default: "yes"
    AllowedValues:
    - "yes"
    - "no"
    Description: Do you want to invoke NLB registration, which requires a Lambda ARN parameter?
    Type: String
  RegisterNlbIpTargetsLambdaArn:
    Description: ARN for NLB IP target registration lambda.
    Type: String
  ExternalApiTargetGroupArn:
    Description: ARN for external API load balancer target group.
    Type: String
  InternalApiTargetGroupArn:
    Description: ARN for internal API load balancer target group.
    Type: String
  InternalServiceTargetGroupArn:
    Description: ARN for internal service load balancer target group.
    Type: String
  BootstrapInstanceType:
    Description: Instance type for the bootstrap EC2 instance
    Default: "i3.large"
    Type: String

Metadata:
  AWS::CloudFormation::Interface:
    ParameterGroups:
    - Label:
        default: "Cluster Information"
      Parameters:
      - InfrastructureName
    - Label:
        default: "Host Information"
      Parameters:
      - RhcosAmi
      - BootstrapIgnitionLocation
      - MasterSecurityGroupId
    - Label:
        default: "Network Configuration"
      Parameters:
      - VpcId
```

```
      - AllowedBootstrapSshCidr
      - PublicSubnet
    - Label:
        default: "Load Balancer Automation"
      Parameters:
      - AutoRegisterELB
      - RegisterNlbIpTargetsLambdaArn
      - ExternalApiTargetGroupArn
      - InternalApiTargetGroupArn
      - InternalServiceTargetGroupArn
    ParameterLabels:
      InfrastructureName:
        default: "Infrastructure Name"
      VpcId:
        default: "VPC ID"
      AllowedBootstrapSshCidr:
        default: "Allowed SSH Source"
      PublicSubnet:
        default: "Public Subnet"
      RhcosAmi:
        default: "Red Hat Enterprise Linux CoreOS AMI ID"
      BootstrapIgnitionLocation:
        default: "Bootstrap Ignition Source"
      MasterSecurityGroupId:
        default: "Master Security Group ID"
      AutoRegisterELB:
        default: "Use Provided ELB Automation"

Conditions:
  DoRegistration: !Equals ["yes", !Ref AutoRegisterELB]

Resources:
  BootstrapIamRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: "2012-10-17"
        Statement:
        - Effect: "Allow"
          Principal:
            Service:
            - "ec2.amazonaws.com"
          Action:
          - "sts:AssumeRole"
      Path: "/"
      Policies:
      - PolicyName: !Join ["-", [!Ref InfrastructureName, "bootstrap", "policy"]]
        PolicyDocument:
          Version: "2012-10-17"
          Statement:
          - Effect: "Allow"
            Action: "ec2:Describe*"
            Resource: "*"
          - Effect: "Allow"
            Action: "ec2:AttachVolume"
            Resource: "*"
```

```
        - Effect: "Allow"
          Action: "ec2:DetachVolume"
          Resource: "*"
        - Effect: "Allow"
          Action: "s3:GetObject"
          Resource: "*"

BootstrapInstanceProfile:
  Type: "AWS::IAM::InstanceProfile"
  Properties:
    Path: "/"
    Roles:
    - Ref: "BootstrapIamRole"

BootstrapSecurityGroup:
  Type: AWS::EC2::SecurityGroup
  Properties:
    GroupDescription: Cluster Bootstrap Security Group
    SecurityGroupIngress:
    - IpProtocol: tcp
      FromPort: 22
      ToPort: 22
      CidrIp: !Ref AllowedBootstrapSshCidr
    - IpProtocol: tcp
      ToPort: 19531
      FromPort: 19531
      CidrIp: 0.0.0.0/0
    VpcId: !Ref VpcId

BootstrapInstance:
  Type: AWS::EC2::Instance
  Properties:
    ImageId: !Ref RhcosAmi
    IamInstanceProfile: !Ref BootstrapInstanceProfile
    InstanceType: !Ref BootstrapInstanceType
    NetworkInterfaces:
    - AssociatePublicIpAddress: "true"
      DeviceIndex: "0"
      GroupSet:
      - !Ref "BootstrapSecurityGroup"
      - !Ref "MasterSecurityGroupId"
      SubnetId: !Ref "PublicSubnet"
    UserData:
      Fn::Base64: !Sub
      - '{"ignition":{"config":{"replace":{"source":"${S3Loc}"}},"version":"3.1.0"}}'
      - {
        S3Loc: !Ref BootstrapIgnitionLocation
      }

RegisterBootstrapApiTarget:
  Condition: DoRegistration
  Type: Custom::NLBRegister
  Properties:
    ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
    TargetArn: !Ref ExternalApiTargetGroupArn
    TargetIp: !GetAtt BootstrapInstance.PrivateIp
```

```
  RegisterBootstrapInternalApiTarget:
    Condition: DoRegistration
    Type: Custom::NLBRegister
    Properties:
      ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
      TargetArn: !Ref InternalApiTargetGroupArn
      TargetIp: !GetAtt BootstrapInstance.PrivateIp

  RegisterBootstrapInternalServiceTarget:
    Condition: DoRegistration
    Type: Custom::NLBRegister
    Properties:
      ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
      TargetArn: !Ref InternalServiceTargetGroupArn
      TargetIp: !GetAtt BootstrapInstance.PrivateIp

Outputs:
  BootstrapInstanceId:
    Description: Bootstrap Instance ID.
    Value: !Ref BootstrapInstance

  BootstrapPublicIp:
    Description: The bootstrap node public IP address.
    Value: !GetAtt BootstrapInstance.PublicIp

  BootstrapPrivateIp:
    Description: The bootstrap node private IP address.
    Value: !GetAtt BootstrapInstance.PrivateIp
```

其他资源

- 您可以通过导航 AWS CloudFormation 控制台来查看您创建的 CloudFormation 堆栈的详情。
- 如需有关 AWS 区的 Red Hat Enterprise Linux CoreOS（RHCOS）AMI 的详细信息，请参阅 AWS 基础架构的 RHCOS AMI。

6.4.3.10. 在 AWS 中创建 control plane 机器

您必须在集群要使用的 Amazon Web Services（AWS）中创建 control plane 机器。

您可以使用提供的 CloudFormation 模板和自定义参数文件，创建代表 control plane 节点的 AWS 资源堆栈。

重要

CloudFormation 模板会创建一个堆栈，它代表三个 control plane 节点。

注意

如果不使用提供的 CloudFormation 模板来创建 control plane 节点，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 已配置了一个 AWS 帐户。
- 您可以通过运行 **aws configure**，将 AWS 密钥和区域添加到本地 AWS 配置集中。
- 已为集群生成 Ignition 配置文件。
- 您在 AWS 中创建并配置了 VPC 及相关子网。
- 您在 AWS 中创建并配置了 DNS、负载均衡器和监听程序。
- 您在 AWS 中创建了集群所需的安全组和角色。
- 已创建 bootstrap 机器。

流程

1. 创建一个 JSON 文件，其包含模板所需的参数值：

```
[
  {
    "ParameterKey": "InfrastructureName", 1
    "ParameterValue": "mycluster-<random_string>" 2
  },
  {
    "ParameterKey": "RhcosAmi", 3
    "ParameterValue": "ami-<random_string>" 4
  },
  {
    "ParameterKey": "AutoRegisterDNS", 5
    "ParameterValue": "yes" 6
  },
  {
    "ParameterKey": "PrivateHostedZoneId", 7
    "ParameterValue": "<random_string>" 8
  },
  {
    "ParameterKey": "PrivateHostedZoneName", 9
    "ParameterValue": "mycluster.example.com" 10
  },
  {
    "ParameterKey": "Master0Subnet", 11
    "ParameterValue": "subnet-<random_string>" 12
  },
  {
    "ParameterKey": "Master1Subnet", 13
    "ParameterValue": "subnet-<random_string>" 14
  },
  {
    "ParameterKey": "Master2Subnet", 15
    "ParameterValue": "subnet-<random_string>" 16
  },
  {
```

```
    "ParameterKey": "MasterSecurityGroupId", 17
    "ParameterValue": "sg-<random_string>" 18
  },
  {
    "ParameterKey": "IgnitionLocation", 19
    "ParameterValue": "https://api-int.<cluster_name>.<domain_name>:22623/config/master"
20
  },
  {
    "ParameterKey": "CertificateAuthorities", 21
    "ParameterValue": "data:text/plain;charset=utf-8;base64,ABC...xYz==" 22
  },
  {
    "ParameterKey": "MasterInstanceProfileName", 23
    "ParameterValue": "<roles_stack>-MasterInstanceProfile-<random_string>" 24
  },
  {
    "ParameterKey": "MasterInstanceType", 25
    "ParameterValue": "" 26
  },
  {
    "ParameterKey": "AutoRegisterELB", 27
    "ParameterValue": "yes" 28
  },
  {
    "ParameterKey": "RegisterNlbIpTargetsLambdaArn", 29
    "ParameterValue": "arn:aws:lambda:<aws_region>:<account_number>:function:
<dns_stack_name>-RegisterNlbIpTargets-<random_string>" 30
  },
  {
    "ParameterKey": "ExternalApiTargetGroupArn", 31
    "ParameterValue": "arn:aws:elasticloadbalancing:<aws_region>:
<account_number>:targetgroup/<dns_stack_name>-Exter-<random_string>" 32
  },
  {
    "ParameterKey": "InternalApiTargetGroupArn", 33
    "ParameterValue": "arn:aws:elasticloadbalancing:<aws_region>:
<account_number>:targetgroup/<dns_stack_name>-Inter-<random_string>" 34
  },
  {
    "ParameterKey": "InternalServiceTargetGroupArn", 35
    "ParameterValue": "arn:aws:elasticloadbalancing:<aws_region>:
<account_number>:targetgroup/<dns_stack_name>-Inter-<random_string>" 36
  }
]
```

1. 您的 Ignition 配置文件中为集群编码的集群基础架构名称。
2. 指定从 Ignition 配置文件元数据中提取的基础架构名称，其格式为 **<cluster-name>-<random-string>**。
3. 根据您选择的架构，当前用于 control plane 机器的 Red Hat Enterprise Linux CoreOS (RHCOS) AMI。

4 指定 **AWS::EC2::Image::Id** 值。

5 是否要执行 DNS etcd 注册。

6 指定 **yes** 或 **no**。如果指定 **yes**，您必须提供托管区信息。

7 用来注册 etcd 目标的 Route 53 专用区 ID。

8 指定 DNS 和负载均衡的 CloudFormation 模板输出的 **PrivateHostedZoneId** 值。

9 用来注册目标的 Route 53 区。

10 指定 **<cluster_name>.<domain_name>**，其中 **<domain_name>** 是您为集群生成 **install-config.yaml** 文件时所用的 Route 53 基域。请勿包含 AWS 控制台中显示的结尾句点 (.)。

11 13 15 在其中启动 control plane 机器的子网，最好是专用子网。

12 14 16 从 DNS 和负载均衡的 CloudFormation 模板输出的 **PrivateSubnets** 值指定子网。

17 与 control plane 节点关联的 master 安全组 ID。

18 指定安全组和角色的 CloudFormation 模板输出的 **MasterSecurityGroupId** 值。

19 从中获取 control plane Ignition 配置文件的位置。

20 指定生成的 Ignition 配置文件的位置，**https://api-int.<cluster_name>.<domain_name>:22623/config/master**。

21 要使用的 base64 编码证书颁发机构字符串。

22 指定安装目录中 **master.ign** 文件中的值。这个值是一个长字符串，格式为 **data:text/plain;charset=utf-8;base64,ABC...xYz==**。

23 与 control plane 节点关联的 IAM 配置集。

24 指定安全组和角色的 CloudFormation 模板输出的 **MasterInstanceProfile** 参数值。

25 根据您选择的架构，用于 control plane 机器的 AWS 实例类型。

26 实例类型值与 control plane 机器的最低资源要求对应。例如，**m6i.xlarge** 是 AMD64 的类型，**m6g.xlarge** 是 ARM64 的类型。

27 是否要注册网络负载均衡器 (NLB) 。

28 指定 **yes** 或 **no**。如果指定 **yes**，您必须提供一个 Lambda Amazon Resource Name (ARN) 值。

29 NLB IP 目标注册 lambda 组的 ARN。

30 指定 DNS 和负载均衡的 CloudFormation 模板输出的 **RegisterNlbIpTargetsLambda** 值。如果将集群部署到 AWS GovCloud 区域，请使用 **arn:aws-us-gov**。

31 外部 API 负载均衡器目标组的 ARN。

32 指定 DNS 和负载均衡的 CloudFormation 模板输出的 **ExternalApiTargetGroupArn** 值。如果将集群部署到 AWS GovCloud 区域，请使用 **arn:aws-us-gov**。

33 内部 API 负载均衡器目标组群的 ARN。

34 指定 DNS 和负载均衡的 CloudFormation 模板输出的 **InternalApiTargetGroupArn** 值。如果将集群部署到 AWS GovCloud 区域，请使用 **arn:aws-us-gov**。

35 内部服务负载均衡器目标组群的 ARN。

36 指定 DNS 和负载均衡的 CloudFormation 模板输出的 **InternalServiceTargetGroupArn** 值。如果将集群部署到 AWS GovCloud 区域，请使用 **arn:aws-us-gov**。

2. 复制**control plane 机器的 CloudFormation 模板**一节中的模板，并将它以 YAML 文件形式保存到计算机上。此模板描述了集群所需的 control plane 机器。

3. 如果您将 **m5** 实例类型指定为 **MasterInstanceType** 的值，请将该实例类型添加到 CloudFormation 模板中的 **MasterInstanceType.AllowedValues** 参数。

4. 启动 CloudFormation 模板，以创建代表 control plane 节点的 AWS 资源堆栈：

重要

您必须在一行内输入命令。

```
$ aws cloudformation create-stack --stack-name <name> 1
    --template-body file://<template>.yaml 2
    --parameters file://<parameters>.json 3
```

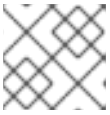
1 **<name>** 是 CloudFormation 堆栈的名称，如 **cluster-control-plane**。如果您删除集群，则需要此堆栈的名称。

2 **<template>** 是您保存的 CloudFormation 模板 YAML 文件的相对路径和名称。

3 **<parameters>** 是 CloudFormation 参数 JSON 文件的相对路径和名称。

输出示例

```
arn:aws:cloudformation:us-east-1:269333783861:stack/cluster-control-plane/21c7e2b0-2ee2-11eb-c6f6-0aa34627df4b
```



注意

CloudFormation 模板会创建一个堆栈，它代表三个 control plane 节点。

5. 确认模板组件已存在：

```
$ aws cloudformation describe-stacks --stack-name <name>
```

6.4.3.10.1. control plane 机器的 CloudFormation 模板

您可以使用以下 CloudFormation 模板来部署 OpenShift Container Platform 集群所需的 control plane 机器。

例 6.76. control plane 机器的 CloudFormation 模板

```
AWSTemplateFormatVersion: 2010-09-09
Description: Template for OpenShift Cluster Node Launch (EC2 master instances)

Parameters:
  InfrastructureName:
    AllowedPattern: ^([a-zA-Z][a-zA-Z0-9\-]{0,26})$
    MaxLength: 27
    MinLength: 1
    ConstraintDescription: Infrastructure name must be alphanumeric, start with a letter, and have a
maximum of 27 characters.
    Description: A short, unique cluster ID used to tag nodes for the kubelet cloud provider.
    Type: String
  RhcosAmi:
    Description: Current Red Hat Enterprise Linux CoreOS AMI to use for bootstrap.
    Type: AWS::EC2::Image::Id
  AutoRegisterDNS:
    Default: ""
    Description: unused
    Type: String
  PrivateHostedZoneId:
    Default: ""
    Description: unused
    Type: String
  PrivateHostedZoneName:
    Default: ""
    Description: unused
    Type: String
  Master0Subnet:
    Description: The subnets, recommend private, to launch the master nodes into.
    Type: AWS::EC2::Subnet::Id
  Master1Subnet:
    Description: The subnets, recommend private, to launch the master nodes into.
    Type: AWS::EC2::Subnet::Id
  Master2Subnet:
    Description: The subnets, recommend private, to launch the master nodes into.
    Type: AWS::EC2::Subnet::Id
  MasterSecurityGroupId:
    Description: The master security group ID to associate with master nodes.
    Type: AWS::EC2::SecurityGroup::Id
  IgnitionLocation:
    Default: https://api-int.$CLUSTER_NAME.$DOMAIN:22623/config/master
    Description: Ignition config file location.
    Type: String
  CertificateAuthorities:
    Default: data:text/plain;charset=utf-8;base64,ABC...xYz==
    Description: Base64 encoded certificate authority string to use.
    Type: String
  MasterInstanceProfileName:
    Description: IAM profile to associate with master nodes.
    Type: String
  MasterInstanceType:
    Default: m5.xlarge
    Type: String

  AutoRegisterELB:
    Default: "yes"
```

```
    AllowedValues:
    - "yes"
    - "no"
    Description: Do you want to invoke NLB registration, which requires a Lambda ARN parameter?
    Type: String
  RegisterNlbIpTargetsLambdaArn:
    Description: ARN for NLB IP target registration lambda. Supply the value from the cluster
infrastructure or select "no" for AutoRegisterELB.
    Type: String
  ExternalApiTargetGroupArn:
    Description: ARN for external API load balancer target group. Supply the value from the cluster
infrastructure or select "no" for AutoRegisterELB.
    Type: String
  InternalApiTargetGroupArn:
    Description: ARN for internal API load balancer target group. Supply the value from the cluster
infrastructure or select "no" for AutoRegisterELB.
    Type: String
  InternalServiceTargetGroupArn:
    Description: ARN for internal service load balancer target group. Supply the value from the
cluster infrastructure or select "no" for AutoRegisterELB.
    Type: String

Metadata:
  AWS::CloudFormation::Interface:
    ParameterGroups:
    - Label:
        default: "Cluster Information"
      Parameters:
      - InfrastructureName
    - Label:
        default: "Host Information"
      Parameters:
      - MasterInstanceType
      - RhcosAmi
      - IgnitionLocation
      - CertificateAuthorities
      - MasterSecurityGroupId
      - MasterInstanceProfileName
    - Label:
        default: "Network Configuration"
      Parameters:
      - VpcId
      - AllowedBootstrapSshCidr
      - Master0Subnet
      - Master1Subnet
      - Master2Subnet
    - Label:
        default: "Load Balancer Automation"
      Parameters:
      - AutoRegisterELB
      - RegisterNlbIpTargetsLambdaArn
      - ExternalApiTargetGroupArn
      - InternalApiTargetGroupArn
      - InternalServiceTargetGroupArn
    ParameterLabels:
      InfrastructureName:
```

```
        default: "Infrastructure Name"
      VpcId:
        default: "VPC ID"
      Master0Subnet:
        default: "Master-0 Subnet"
      Master1Subnet:
        default: "Master-1 Subnet"
      Master2Subnet:
        default: "Master-2 Subnet"
      MasterInstanceType:
        default: "Master Instance Type"
      MasterInstanceProfileName:
        default: "Master Instance Profile Name"
      RhcosAmi:
        default: "Red Hat Enterprise Linux CoreOS AMI ID"
      BootstrapIgnitionLocation:
        default: "Master Ignition Source"
      CertificateAuthorities:
        default: "Ignition CA String"
      MasterSecurityGroupId:
        default: "Master Security Group ID"
      AutoRegisterELB:
        default: "Use Provided ELB Automation"

Conditions:
  DoRegistration: !Equals ["yes", !Ref AutoRegisterELB]

Resources:
  Master0:
    Type: AWS::EC2::Instance
    Properties:
      ImageId: !Ref RhcosAmi
      BlockDeviceMappings:
      - DeviceName: /dev/xvda
        Ebs:
          VolumeSize: "120"
          VolumeType: "gp2"
      IamInstanceProfile: !Ref MasterInstanceProfileName
      InstanceType: !Ref MasterInstanceType
      NetworkInterfaces:
      - AssociatePublicIpAddress: "false"
        DeviceIndex: "0"
        GroupSet:
        - !Ref "MasterSecurityGroupId"
        SubnetId: !Ref "Master0Subnet"
      UserData:
        Fn::Base64: !Sub
        - '{"ignition":{"config":{"merge":[{"source":"${SOURCE}"}]},"security":{"tls":
{"certificateAuthorities":[{"source":"${CA_BUNDLE}"}]}},"version":"3.1.0"}}'
        - {
          SOURCE: !Ref IgnitionLocation,
          CA_BUNDLE: !Ref CertificateAuthorities,
        }
      Tags:
      - Key: !Join ["", ["kubernetes.io/cluster/", !Ref InfrastructureName]]
        Value: "shared"
```

```
  RegisterMaster0:
    Condition: DoRegistration
    Type: Custom::NLBRegister
    Properties:
      ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
      TargetArn: !Ref ExternalApiTargetGroupArn
      TargetIp: !GetAtt Master0.PrivateIp

  RegisterMaster0InternalApiTarget:
    Condition: DoRegistration
    Type: Custom::NLBRegister
    Properties:
      ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
      TargetArn: !Ref InternalApiTargetGroupArn
      TargetIp: !GetAtt Master0.PrivateIp

  RegisterMaster0InternalServiceTarget:
    Condition: DoRegistration
    Type: Custom::NLBRegister
    Properties:
      ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
      TargetArn: !Ref InternalServiceTargetGroupArn
      TargetIp: !GetAtt Master0.PrivateIp

  Master1:
    Type: AWS::EC2::Instance
    Properties:
      ImageId: !Ref RhcosAmi
      BlockDeviceMappings:
      - DeviceName: /dev/xvda
        Ebs:
          VolumeSize: "120"
          VolumeType: "gp2"
      IamInstanceProfile: !Ref MasterInstanceProfileName
      InstanceType: !Ref MasterInstanceType
      NetworkInterfaces:
      - AssociatePublicIpAddress: "false"
        DeviceIndex: "0"
        GroupSet:
        - !Ref "MasterSecurityGroupId"
        SubnetId: !Ref "Master1Subnet"
      UserData:
        Fn::Base64: !Sub
        - '{"ignition":{"config":{"merge":[{"source":"${SOURCE}"}]},"security":{"tls":
{"certificateAuthorities":[{"source":"${CA_BUNDLE}"}]}},"version":"3.1.0"}}'
        - {
          SOURCE: !Ref IgnitionLocation,
          CA_BUNDLE: !Ref CertificateAuthorities,
        }
      Tags:
      - Key: !Join ["", ["kubernetes.io/cluster/", !Ref InfrastructureName]]
        Value: "shared"

  RegisterMaster1:
    Condition: DoRegistration
```

```
    Type: Custom::NLBRegister
    Properties:
      ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
      TargetArn: !Ref ExternalApiTargetGroupArn
      TargetIp: !GetAtt Master1.PrivateIp

  RegisterMaster1InternalApiTarget:
    Condition: DoRegistration
    Type: Custom::NLBRegister
    Properties:
      ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
      TargetArn: !Ref InternalApiTargetGroupArn
      TargetIp: !GetAtt Master1.PrivateIp

  RegisterMaster1InternalServiceTarget:
    Condition: DoRegistration
    Type: Custom::NLBRegister
    Properties:
      ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
      TargetArn: !Ref InternalServiceTargetGroupArn
      TargetIp: !GetAtt Master1.PrivateIp

  Master2:
    Type: AWS::EC2::Instance
    Properties:
      ImageId: !Ref RhcosAmi
      BlockDeviceMappings:
      - DeviceName: /dev/xvda
        Ebs:
          VolumeSize: "120"
          VolumeType: "gp2"
      IamInstanceProfile: !Ref MasterInstanceProfileName
      InstanceType: !Ref MasterInstanceType
      NetworkInterfaces:
      - AssociatePublicIpAddress: "false"
        DeviceIndex: "0"
        GroupSet:
        - !Ref "MasterSecurityGroupId"
        SubnetId: !Ref "Master2Subnet"
      UserData:
        Fn::Base64: !Sub
        - '{"ignition":{"config":{"merge":[{"source":"${SOURCE}"}]},"security":{"tls":
{"certificateAuthorities":[{"source":"${CA_BUNDLE}"}]}},"version":"3.1.0"}}'
        - {
          SOURCE: !Ref IgnitionLocation,
          CA_BUNDLE: !Ref CertificateAuthorities,
        }
      Tags:
      - Key: !Join ["", ["kubernetes.io/cluster/", !Ref InfrastructureName]]
        Value: "shared"

  RegisterMaster2:
    Condition: DoRegistration
    Type: Custom::NLBRegister
    Properties:
      ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
```

```
      TargetArn: !Ref ExternalApiTargetGroupArn
      TargetIp: !GetAtt Master2.PrivateIp

  RegisterMaster2InternalApiTarget:
    Condition: DoRegistration
    Type: Custom::NLBRegister
    Properties:
      ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
      TargetArn: !Ref InternalApiTargetGroupArn
      TargetIp: !GetAtt Master2.PrivateIp

  RegisterMaster2InternalServiceTarget:
    Condition: DoRegistration
    Type: Custom::NLBRegister
    Properties:
      ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
      TargetArn: !Ref InternalServiceTargetGroupArn
      TargetIp: !GetAtt Master2.PrivateIp

Outputs:
  PrivateIPs:
    Description: The control-plane node private IP addresses.
    Value:
      !Join [
        ",",
        [!GetAtt Master0.PrivateIp, !GetAtt Master1.PrivateIp, !GetAtt Master2.PrivateIp]
      ]
```

其他资源

- 您可以通过导航 AWS CloudFormation 控制台来查看您创建的 CloudFormation 堆栈的详情。

6.4.3.11. 在 AWS 中创建 worker 节点

您可以在 Amazon Web Services (AWS) 中创建 worker 节点，供集群使用。

注意

如果您要安装三节点集群，请跳过这一步。三节点集群包含三个 control plane 机器，它们也可以充当计算机器。

您可以使用提供的 CloudFormation 模板和自定义参数文件创建代表 worker 节点的 AWS 资源堆栈。

重要

CloudFormation 模板会创建一个堆栈，它代表一个 worker 节点。您必须为每个 worker 节点创建一个堆栈。

注意

如果不使用提供的 CloudFormation 模板来创建 worker 节点，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 已配置了一个 AWS 帐户。
- 您可以通过运行 **aws configure**，将 AWS 密钥和区域添加到本地 AWS 配置集中。
- 已为集群生成 Ignition 配置文件。
- 您在 AWS 中创建并配置了 VPC 及相关子网。
- 您在 AWS 中创建并配置了 DNS、负载均衡器和监听程序。
- 您在 AWS 中创建了集群所需的安全组和角色。
- 已创建 bootstrap 机器。
- 已创建 control plane 机器。

流程

1. 创建一个 JSON 文件，其包含 CloudFormation 模板需要的参数值：

```
[
  {
    "ParameterKey": "InfrastructureName", 1
    "ParameterValue": "mycluster-<random_string>" 2
  },
  {
    "ParameterKey": "RhcosAmi", 3
    "ParameterValue": "ami-<random_string>" 4
  },
  {
    "ParameterKey": "Subnet", 5
    "ParameterValue": "subnet-<random_string>" 6
  },
  {
    "ParameterKey": "WorkerSecurityGroupId", 7
    "ParameterValue": "sg-<random_string>" 8
  },
  {
    "ParameterKey": "IgnitionLocation", 9
    "ParameterValue": "https://api-int.<cluster_name>.<domain_name>:22623/config/worker"
10
  },
  {
    "ParameterKey": "CertificateAuthorities", 11
    "ParameterValue": "" 12
  },
  {
    "ParameterKey": "WorkerInstanceProfileName", 13
    "ParameterValue": "" 14
  },
  {
    "ParameterKey": "WorkerInstanceType", 15
```

```
    "ParameterValue": "" 16
  }
]
```

1. 您的 Ignition 配置文件中为集群编码的集群基础架构名称。
2. 指定从 Ignition 配置文件元数据中提取的基础架构名称，其格式为 **<cluster-name>-<random-string>**。
3. 根据您选择的架构，当前 Red Hat Enterprise Linux CoreOS (RHCOS) AMI 用于 worker 节点。
4. 指定 **AWS::EC2::Image::Id** 值。
5. 在其中启动 worker 节点的子网，最好是专用子网。
6. 从 DNS 和负载均衡的 CloudFormation 模板输出的 **PrivateSubnets** 值指定子网。
7. 与 worker 节点关联的 worker 安全组 ID。
8. 指定安全组和角色的 CloudFormation 模板输出的 **WorkerSecurityGroupId** 值。
9. 从中获取 bootstrap Ignition 配置文件的位置。
10. 指定生成的 Ignition 配置的位置，**https://api-int.<cluster_name>.<domain_name>:22623/config/worker**。
11. 要使用的 Base64 编码证书颁发机构字符串。
12. 指定安装目录下 **worker.ign** 文件中的值。这个值是一个长字符串，格式为 **data:text/plain;charset=utf-8;base64,ABC...xYz==**。
13. 与 worker 节点关联的 IAM 配置集。
14. 指定安全组和角色的 CloudFormation 模板输出的 **WorkerInstanceProfile** 参数值。
15. 根据您选择的架构，用于计算机器的 AWS 实例类型。
16. 实例类型值对应于计算机器的最低资源要求。例如，**m6i.large** 是 AMD64 的类型，**m6g.large** 是 ARM64 的类型。

2. 复制 **worker 机器的 CloudFormation 模板**一节中的模板，并将它以 YAML 文件形式保存到计算机上。此模板描述了集群所需的网络对象和负载均衡器。
3. 可选：如果将 **m5** 实例类型指定为 **WorkerInstanceType** 的值，请将该实例类型添加到 CloudFormation 模板中的 **WorkerInstanceType.AllowedValues** 参数。
4. 可选：如果您使用 AWS Marketplace 镜像部署，请使用从订阅获取的 AMI ID 更新 **Worker0.type.properties.ImageID** 参数。
5. 使用 CloudFormation 模板创建代表 worker 节点的 AWS 资源堆栈：

重要

您必须在一行内输入命令。

```
-
```

```
$ aws cloudformation create-stack --stack-name <name> 1
    --template-body file://<template>.yaml \ 2
    --parameters file://<parameters>.json 3
```

1 **<name>** 是 CloudFormation 堆栈的名称，如 **cluster-worker-1**。如果您删除集群，则需要此堆栈的名称。

2 **<template>** 是您保存的 CloudFormation 模板 YAML 文件的相对路径和名称。

3 **<parameters>** 是 CloudFormation 参数 JSON 文件的相对路径和名称。

输出示例

```
arn:aws:cloudformation:us-east-1:269333783861:stack/cluster-worker-1/729ee301-1c2a-11eb-348f-sd9888c65b59
```

注意

CloudFormation 模板会创建一个堆栈，它代表一个 worker 节点。

6. 确认模板组件已存在：

```
$ aws cloudformation describe-stacks --stack-name <name>
```

7. 继续创建 worker 堆栈，直到为集群创建了充足的 worker 机器。您可以通过引用同一模板和参数文件并指定不同的堆栈名称来创建额外的 worker 堆栈。

重要

您必须至少创建两台 worker 机器，因此您必须创建至少两个使用此 CloudFormation 模板的堆栈。

6.4.3.11.1. worker 机器的 CloudFormation 模板

您可以使用以下 CloudFormation 模板来部署 OpenShift Container Platform 集群所需的 worker 机器。

例 6.77. worker 机器的 CloudFormation 模板

```
AWSTemplateFormatVersion: 2010-09-09
Description: Template for OpenShift Cluster Node Launch (EC2 worker instance)

Parameters:
  InfrastructureName:
    AllowedPattern: ^([a-zA-Z][a-zA-Z0-9\-]{0,26})$
    MaxLength: 27
    MinLength: 1
    ConstraintDescription: Infrastructure name must be alphanumeric, start with a letter, and have a maximum of 27 characters.
    Description: A short, unique cluster ID used to tag nodes for the kubelet cloud provider.
    Type: String
  RhcosAmi:
    Description: Current Red Hat Enterprise Linux CoreOS AMI to use for bootstrap.
```

```
    Type: AWS::EC2::Image::Id
  Subnet:
    Description: The subnets, recommend private, to launch the worker nodes into.
    Type: AWS::EC2::Subnet::Id
  WorkerSecurityGroupId:
    Description: The worker security group ID to associate with worker nodes.
    Type: AWS::EC2::SecurityGroup::Id
  IgnitionLocation:
    Default: https://api-int.$CLUSTER_NAME.$DOMAIN:22623/config/worker
    Description: Ignition config file location.
    Type: String
  CertificateAuthorities:
    Default: data:text/plain;charset=utf-8;base64,ABC...xYz==
    Description: Base64 encoded certificate authority string to use.
    Type: String
  WorkerInstanceProfileName:
    Description: IAM profile to associate with worker nodes.
    Type: String
  WorkerInstanceType:
    Default: m5.large
    Type: String

Metadata:
  AWS::CloudFormation::Interface:
    ParameterGroups:
    - Label:
        default: "Cluster Information"
      Parameters:
      - InfrastructureName
    - Label:
        default: "Host Information"
      Parameters:
      - WorkerInstanceType
      - RhcosAmi
      - IgnitionLocation
      - CertificateAuthorities
      - WorkerSecurityGroupId
      - WorkerInstanceProfileName
    - Label:
        default: "Network Configuration"
      Parameters:
      - Subnet
    ParameterLabels:
      Subnet:
        default: "Subnet"
      InfrastructureName:
        default: "Infrastructure Name"
      WorkerInstanceType:
        default: "Worker Instance Type"
      WorkerInstanceProfileName:
        default: "Worker Instance Profile Name"
      RhcosAmi:
        default: "Red Hat Enterprise Linux CoreOS AMI ID"
      IgnitionLocation:
        default: "Worker Ignition Source"
      CertificateAuthorities:
```

```
        default: "Ignition CA String"
      WorkerSecurityGroupId:
        default: "Worker Security Group ID"

Resources:
  Worker0:
    Type: AWS::EC2::Instance
    Properties:
      ImageId: !Ref RhcosAmi
      BlockDeviceMappings:
      - DeviceName: /dev/xvda
        Ebs:
          VolumeSize: "120"
          VolumeType: "gp2"
      IamInstanceProfile: !Ref WorkerInstanceProfileName
      InstanceType: !Ref WorkerInstanceType
      NetworkInterfaces:
      - AssociatePublicIpAddress: "false"
        DeviceIndex: "0"
        GroupSet:
        - !Ref "WorkerSecurityGroupId"
        SubnetId: !Ref "Subnet"
      UserData:
        Fn::Base64: !Sub
        - '{"ignition":{"config":{"merge":[{"source":"${SOURCE}"}]},"security":{"tls":
{"certificateAuthorities":[{"source":"${CA_BUNDLE}"}]}},"version":"3.1.0"}}'
        - {
          SOURCE: !Ref IgnitionLocation,
          CA_BUNDLE: !Ref CertificateAuthorities,
        }
      Tags:
      - Key: !Join ["", ["kubernetes.io/cluster/", !Ref InfrastructureName]]
        Value: "shared"

Outputs:
  PrivateIP:
    Description: The compute node private IP address.
    Value: !GetAtt Worker0.PrivateIp
```

其他资源

- 您可以通过导航 AWS CloudFormation 控制台来查看您创建的 CloudFormation 堆栈的详情。

6.4.3.12. 使用用户置备的基础架构在 AWS 上初始化 bootstrap 序列

在 Amazon Web Services（AWS）中创建所有所需的基础架构后，您可以启动初始化 OpenShift Container Platform control plane 的 bootstrap 序列。

先决条件

- 已配置了一个 AWS 帐户。
- 您可以通过运行 **aws configure**，将 AWS 密钥和区域添加到本地 AWS 配置集中。

- 已为集群生成 Ignition 配置文件。
- 您在 AWS 中创建并配置了 VPC 及相关子网。
- 您在 AWS 中创建并配置了 DNS、负载均衡器和监听程序。
- 您在 AWS 中创建了集群所需的安全组和角色。
- 已创建 bootstrap 机器。
- 已创建 control plane 机器。
- 已创建 worker 节点。

流程

1. 更改为包含安装程序的目录，并启动初始化 OpenShift Container Platform control plane 的 bootstrap 过程：

```
$ ./openshift-install wait-for bootstrap-complete --dir <installation_directory> \ 1
    --log-level=info 2
```

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

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

输出示例

```
INFO Waiting up to 20m0s for the Kubernetes API at
https://api.mycluster.example.com:6443...
INFO API v1.29.4 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
INFO Time elapsed: 1s
```

如果命令退出时没有 **FATAL** 警告，则 OpenShift Container Platform control plane 已被初始化。

注意

在 control plane 初始化后，它会设置计算节点，并以 Operator 的形式安装其他服务。

其他资源

- 如需了解在 OpenShift Container Platform 安装过程中监控安装、bootstrap 和 control plane 日志的详细信息，请参阅监控安装进度。
- 如需有关对 bootstrap 过程进行故障排除的信息，请参阅收集 bootstrap 节点诊断数据。
- 您可以使用 AWS EC2 控制台查看正在运行的实例的详情。

6.4.3.13. 使用 CLI 登录集群

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

先决条件

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

流程

1. 导出 **kubeadmin** 凭证：

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

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

2. 验证您可以使用导出的配置成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

6.4.3.14. 批准机器的证书签名请求

当您将机器添加到集群时，会为您添加的每台机器生成两个待处理证书签名请求(CSR)。您必须确认这些 CSR 已获得批准，或根据需要自行批准。必须首先批准客户端请求，然后批准服务器请求。

先决条件

- 您已将机器添加到集群中。

流程

1. 确认集群可以识别这些机器：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0 Ready    master  63m  v1.29.4
master-1 Ready    master  63m  v1.29.4
master-2 Ready    master  64m  v1.29.4
```

输出中列出了您创建的所有机器。

注意

在有些 CSR 被批准前，前面的输出可能不包括计算节点（也称为 worker 节点）。

2. 检查待处理的 CSR，并确保添加到集群中的每台机器都有 **Pending** 或 **Approved** 状态的客户端请求：

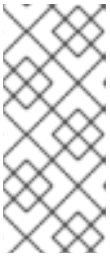
```
$ oc get csr
```

输出示例

```
NAME        AGE    REQUESTOR                                                   CONDITION
csr-8b2br   15m    system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
csr-8vnps   15m    system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
...
```

在本例中，两台机器加入集群。您可能会在列表中看到更多已批准的 CSR。

3. 如果 CSR 没有获得批准，在您添加的机器的所有待处理 CSR 都处于 **Pending 状态** 后，请批准集群机器的 CSR：



注意

由于 CSR 会自动轮转，因此请在将机器添加到集群后一小时内批准您的 CSR。如果没有在一小时内批准它们，证书将会轮转，每个节点会存在多个证书。您必须批准所有这些证书。批准客户端 CSR 后，Kubelet 为服务证书创建一个二级 CSR，这需要手动批准。然后，如果 Kubelet 请求具有相同参数的新证书，则后续提供证书续订请求由 **machine-approver** 自动批准。

注意

对于在未启用机器 API 的平台上运行的集群，如裸机和其他用户置备的基础架构，您必须实施一种方法来自动批准 kubelet 提供证书请求(CSR)。如果没有批准请求，则 **oc exec**、oc**rsh** 和 **oc logs** 命令将无法成功，因为 API 服务器连接到 kubelet 时需要服务证书。与 Kubelet 端点联系的任何操作都需要此证书批准。该方法必须监视新的 CSR，确认 CSR 由 system: **node** 或 **system:admin** 组中的 **node-bootstrapper** 服务帐户提交，并确认节点的身份。

- 要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```

注意

在有些 CSR 被批准前，一些 Operator 可能无法使用。

4. 现在，您的客户端请求已被批准，您必须查看添加到集群中的每台机器的服务器请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE     REQUESTOR                                                   CONDITION
csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal       Pending
csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal       Pending
...
```

5. 如果剩余的 CSR 没有被批准，且处于 **Pending** 状态，请批准集群机器的 CSR：

- 要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

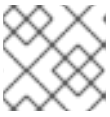
```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs oc adm certificate approve
```

6. 批准所有客户端和服务器 CSR 后，机器将 处于 **Ready 状态**。运行以下命令验证：

```
$ oc get nodes
```

输出示例

```
NAME      STATUS    ROLES   AGE  VERSION
master-0  Ready     master  73m  v1.29.4
master-1  Ready     master  73m  v1.29.4
master-2  Ready     master  74m  v1.29.4
worker-0  Ready     worker  11m  v1.29.4
worker-1  Ready     worker  11m  v1.29.4
```



注意

批准服务器 CSR 后可能需要几分钟时间让机器过渡到 **Ready 状态**。

其他信息

- 如需有关 CSR 的更多信息，请参阅 证书签名请求。

6.4.3.15. 初始 Operator 配置

在 control plane 初始化后，您必须立即配置一些 Operator，以便它们都可用。

先决条件

- 您的 control plane 已初始化。

流程

1. 观察集群组件上线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION   AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.16.0    True        False         False      19m
baremetal                                  4.16.0    True        False         False      37m
cloud-credential                           4.16.0    True        False         False      40m
cluster-autoscaler                         4.16.0    True        False         False      37m
config-operator                            4.16.0    True        False         False      38m
console                                    4.16.0    True        False         False      26m
csi-snapshot-controller                    4.16.0    True        False         False      37m
dns                                        4.16.0    True        False         False      37m
etcd                                       4.16.0    True        False         False      36m
image-registry                             4.16.0    True        False         False      31m
ingress                                    4.16.0    True        False         False      30m
insights                                   4.16.0    True        False         False      31m
kube-apiserver                             4.16.0    True        False         False      26m
kube-controller-manager                    4.16.0    True        False         False      36m
kube-scheduler                             4.16.0    True        False         False      36m
kube-storage-version-migrator              4.16.0    True        False         False      37m
machine-api                                4.16.0    True        False         False      29m
machine-approver                           4.16.0    True        False         False      37m
machine-config                             4.16.0    True        False         False      36m
marketplace                                4.16.0    True        False         False      37m
monitoring                                 4.16.0    True        False         False      29m
network                                    4.16.0    True        False         False      38m
node-tuning                                4.16.0    True        False         False      37m
openshift-apiserver                        4.16.0    True        False         False      32m
openshift-controller-manager               4.16.0    True        False         False      30m
openshift-samples                          4.16.0    True        False         False      32m
operator-lifecycle-manager                 4.16.0    True        False         False      37m
operator-lifecycle-manager-catalog         4.16.0    True        False         False      37m
operator-lifecycle-manager-packageserver   4.16.0    True        False         False      32m
service-ca                                 4.16.0    True        False         False      38m
storage                                    4.16.0    True        False         False      37m
```

2. 配置不可用的 Operator。

6.4.3.15.1. 镜像 registry 存储配置

Amazon Web Services 提供默认存储，这意味着 Image Registry Operator 在安装后可用。但是，如果 Registry Operator 无法创建 S3 存储桶并自动配置存储，您需要手工配置 registry 存储。

示配置生产集群所需的持久性卷的说明。如果适用，显示有关将空目录配置为存储位置的说明，这仅适用于非生产集群。

另外还提供了在升级过程中使用 **Recreate** rollout 策略来允许镜像 registry 使用块存储类型的说明。

您可以在 AWS 中为用户置备的基础架构配置 registry 存储,以将 OpenShift Container Platform 部署到隐藏的区域。请参阅为 AWS 用户置备的基础架构配置 registry。

6.4.3.15.1.1. 为使用用户置备的基础架构的 AWS 配置 registry 存储

在安装过程中，使用您的云凭据就可以创建一个 Amazon S3 存储桶，Registry Operator 将会自动配置存储。

如果 Registry Operator 无法创建 S3 存储桶或自动配置存储，您可以按照以下流程创建 S3 存储桶并配置存储。

先决条件

- 在带有用户置备的基础架构的 AWS 上有一个集群。
- 对于 Amazon S3 存储，secret 应该包含以下两个键：
  - **REGISTRY_STORAGE_S3_ACCESSKEY**
  - **REGISTRY_STORAGE_S3_SECRETKEY**

流程

如果 Registry Operator 无法创建 S3 存储桶并自动配置存储，请进行以下操作。

1. 设置一个 Bucket Lifecycle Policy 用来终止已有一天之久的未完成的分段上传操作。
2. 在**configs.imageregistry.operator.openshift.io/cluster**中中输入存储配置：

```
$ oc edit configs.imageregistry.operator.openshift.io/cluster
```

配置示例

```
storage:
  s3:
    bucket: <bucket-name>
    region: <region-name>
```

警告

为了保护 AWS 中 registry 镜像的安全，阻止对 S3 存储桶的公共访问。

6.4.3.15.1.2. 在非生产集群中配置镜像 registry 存储

您必须为 Image Registry Operator 配置存储。对于非生产集群，您可以将镜像 registry 设置为空目录。如果您这样做，重启 registry 时会丢失所有镜像。

流程

- 将镜像 registry 存储设置为空目录：

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec":{"storage":{"emptyDir":{}}}}'
```

警告

仅为非生产集群配置这个选项。

如果在 Image Registry Operator 初始化其组件前运行这个命令，**oc patch** 命令会失败并显示以下错误：

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster" not found
```

等待几分钟，然后再次运行该命令。

6.4.3.16. 删除 bootstrap 资源：

完成集群的初始 Operator 配置后，从 Amazon Web Services (AWS) 中删除 bootstrap 资源。

先决条件

- 已为集群完成初始的 Operator 配置。

流程

1. 删除 bootstrap 资源。如果您使用了 CloudFormation 模板，请删除其堆栈：
   - 使用 AWS CLI 删除堆栈：

```
$ aws cloudformation delete-stack --stack-name <name> 1
```

1 **<name>** 是 bootstrap 堆栈的名称。

   - 使用 AWS CloudFormation 控制台删除堆栈。

6.4.3.17. 创建 Ingress DNS 记录

如果您删除了 DNS 区配置，请手动创建指向 Ingress 负载均衡器的 DNS 记录。您可以创建一个 wildcard 记录或具体的记录。以下流程使用了 A 记录，但您可以使用其他所需记录类型，如 CNAME 或别名。

先决条件

- 已在 Amazon Web Services (AWS) 上安装了使用您置备的基础架构的 OpenShift Container Platform 集群。
- 已安装 OpenShift CLI（**oc**）。
- 安装了 **jq** 软件包。
- 您下载了 AWS CLI 并安装到您的计算机上。请参阅使用捆绑安装程序（Linux、macOS 或 Unix）安装 AWS CLI的文档。

流程

1. 决定要创建的路由。
   - 要创建一个 wildcard 记录，请使用 ***.apps.<cluster_name>.<domain_name>**，其中 **<cluster_name>** 是集群名称，**<domain_name>** 是 OpenShift Container Platform 集群的 Route 53 基域。
   - 要创建特定的记录，您必须为集群使用的每个路由创建一个记录，如下所示：

     ```
     $ oc get --all-namespaces -o jsonpath='{range .items[*]}{range .status.ingress[*]}{.host}
     {"\n"}{end}{end}' routes
     ```

     输出示例

     ```
     oauth-openshift.apps.<cluster_name>.<domain_name>
     console-openshift-console.apps.<cluster_name>.<domain_name>
     downloads-openshift-console.apps.<cluster_name>.<domain_name>
     alertmanager-main-openshift-monitoring.apps.<cluster_name>.<domain_name>
     prometheus-k8s-openshift-monitoring.apps.<cluster_name>.<domain_name>
     ```

2. 获取 Ingress Operator 负载均衡器状态，并记录其使用的外部 IP 地址值，如 **EXTERNAL-IP** 列所示：

   ```
   $ oc -n openshift-ingress get service router-default
   ```

   输出示例

   ```
   NAME              TYPE           CLUSTER-IP      EXTERNAL-IP                         PORT(S)
   AGE
   router-default    LoadBalancer   172.30.62.215   ab3...28.us-east-2.elb.amazonaws.com
   80:31499/TCP,443:30693/TCP   5m
   ```

3. 为负载均衡器定位托管区 ID：

   ```
   $ aws elb describe-load-balancers | jq -r '.LoadBalancerDescriptions[] | select(.DNSName ==
   "<external_ip>").CanonicalHostedZoneNameID' 1
   ```

   1 对于 **<external_ip>**，请指定您获取的 Ingress Operator 负载均衡器的外部 IP 地址值。

   输出示例

   ```
   Z3AADJGX6KTTL2
   ```

这个命令的输出是负载均衡器托管区 ID。

4. 获取集群域的公共托管区 ID：

```
$ aws route53 list-hosted-zones-by-name \
        --dns-name "<domain_name>" \ 1
        --query 'HostedZones[? Config.PrivateZone != `true` && Name ==
`<domain_name>.`].Id' 2
        --output text
```

1 2 对于 **<domain_name>**，请为 OpenShift Container Platform 集群指定 Route 53 基域。

输出示例

```
/hostedzone/Z3URY6TWQ91KVV
```

命令输出中会显示您的域的公共托管区 ID。在本例中是 **Z3URY6TWQ91KVV**。

5. 在您的私有区中添加别名记录：

```
$ aws route53 change-resource-record-sets --hosted-zone-id "<private_hosted_zone_id>" --
change-batch '{ 1
>   "Changes": [
>     {
>       "Action": "CREATE",
>       "ResourceRecordSet": {
>         "Name": "\\052.apps.<cluster_domain>", 2
>         "Type": "A",
>         "AliasTarget":{
>           "HostedZoneId": "<hosted_zone_id>", 3
>           "DNSName": "<external_ip>.", 4
>           "EvaluateTargetHealth": false
>         }
>       }
>     }
>   ]
> }'
```

1 对于 **<private_hosted_zone_id>**，指定 DNS 和负载均衡的 CloudFormation 模板输出的值。

2 对于 **<cluster_domain>**，请指定用于 OpenShift Container Platform 集群的域或子域。

3 对于 **<hosted_zone_id>**，请为您获得的负载均衡器指定公共托管区 ID。

4 对于 **<external_ip>**，请指定 Ingress Operator 负载均衡器的外部 IP 地址值。请确定在该参数值中包含最后的句点（.）。

6. 在您的公共区中添加记录：

```
$ aws route53 change-resource-record-sets --hosted-zone-id "<public_hosted_zone_id>"" --
```

```
change-batch '{ 1
>   "Changes": [
>     {
>       "Action": "CREATE",
>       "ResourceRecordSet": {
>         "Name": "\\052.apps.<cluster_domain>", 2
>         "Type": "A",
>         "AliasTarget":{
>           "HostedZoneId": "<hosted_zone_id>", 3
>           "DNSName": "<external_ip>.", 4
>           "EvaluateTargetHealth": false
>         }
>       }
>     }
>   ]
> }'
```

1. 对于 **<public_hosted_zone_id>**，请为您的域指定公共托管区。
2. 对于 **<cluster_domain>**，请指定用于 OpenShift Container Platform 集群的域或子域。
3. 对于 **<hosted_zone_id>**，请为您获得的负载均衡器指定公共托管区 ID。
4. 对于 **<external_ip>**，请指定 Ingress Operator 负载均衡器的外部 IP 地址值。请确定在该参数值中包含最后的句点（.）。

6.4.3.18. 在用户置备的基础架构上完成 AWS 安装

在用户置备的基础架构 Amazon Web Service (AWS) 上启动 OpenShift Container Platform 安装后，监视进程并等待安装完成。

先决条件

- 您在用户置备的 AWS 基础架构上为 OpenShift Container Platform 集群删除了 bootstrap 节点。
- 已安装 **oc** CLI。

流程

- 在包含安装程序的目录中完成集群安装：

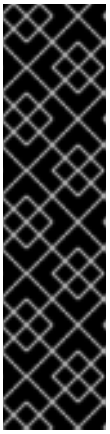
```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

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

输出示例

```
INFO Waiting up to 40m0s for the cluster at https://api.mycluster.example.com:6443 to
initialize...
INFO Waiting up to 10m0s for the openshift-console route to be created...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
```

```
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 1s
```



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

6.4.3.19. 使用 Web 控制台登录到集群

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

先决条件

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

流程

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

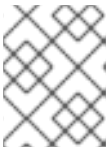
```
$ cat <installation_directory>/auth/kubeadmin-password
```

注意

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

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

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



注意

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

输出示例

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

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

其他资源

- 如需有关 访问和了解 OpenShift Container Platform Web 控制台的更多详情，请参阅 访问 Web 控制台。

6.4.3.20. 其他资源

- 如需有关 AWS CloudFormation 堆栈的更多信息，请参阅 AWS 文档中的使用堆栈。

6.4.3.21. 后续步骤

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

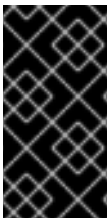
6.4.4. 在带有用户置备的受限网络中的 AWS 上安装集群

在 OpenShift Container Platform 版本 4.16 中，您可以使用您提供的基础架构和安装发行内容的内部镜像在 Amazon Web Services (AWS) 上安装集群。

重要

虽然您可以使用镜像安装发行内容安装 OpenShift Container Platform 集群，但您的集群仍需要访问互联网才能使用 AWS API。

创建此基础架构的一种方法是使用提供的 CloudFormation 模板。您可以修改模板来自定义基础架构，或使用其包含的信息来按照公司策略创建 AWS 对象。

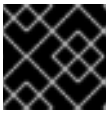


重要

进行用户置备的基础架构安装的步骤仅作为示例。使用您提供的基础架构安装集群需要了解云供应商和 OpenShift Container Platform 安装过程。提供的几个 CloudFormation 模板可帮助完成这些步骤，或者帮助您自行建模。您也可以自由选择通过其他方法创建所需的资源；模板仅作示例之用。

6.4.4.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读选择集群安装方法并为用户准备它的文档。
- 您在镜像主机上创建了一个镜像 registry，并获取您的 OpenShift Container Platform 版本的 **imageContentSources** 数据。



重要

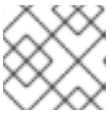
由于安装介质位于堡垒主机上，因此请使用该计算机完成所有安装步骤。

- 已将 AWS 帐户配置为托管集群。

重要

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

- 已准备好用户置备的基础架构。
- 您下载了 AWS CLI 并安装到您的计算机上。请参阅 AWS 文档中的使用捆绑安装程序（Linux、macOS 或 UNIX）安装 AWS CLI。
- 如果您使用防火墙并计划使用 Telemetry 服务，则将防火墙配置为允许集群需要访问的站点。



注意

如果要配置代理，请务必查看此站点列表。

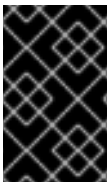
- 如果环境中无法访问云身份和访问管理(IAM)API，或者不想将管理员级别的凭证 secret 存储在 **kube-system** 命名空间中，您可以 手动创建和维护 IAM 凭证。

6.4.4.2. 关于在受限网络中安装

在 OpenShift Container Platform 4.16 中，可以执行不需要有效的互联网连接来获取软件组件的安装。受限网络安装可以使用安装程序置备的基础架构或用户置备的基础架构完成，具体取决于您要安装集群的云平台。

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

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



重要

由于用户置备安装配置的复杂性，在尝试使用用户置备的基础架构受限网络安装前，请考虑完成标准用户置备的基础架构安装。完成此测试安装后，您可以更轻松地隔离和排除在受限网络中安装过程中可能出现的任何问题。

6.4.4.2.1. 其他限制

受限网络中的集群有以下额外限制和限制：

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

6.4.4.3. 创建用于 AWS 的安装文件

要使用用户置备的基础架构在 Amazon Web Services (AWS) 上安装 OpenShift Container Platform，您必须生成并修改安装程序部署集群所需的文件，以便集群只创建要使用的机器。您要生成并自定义 **install-config.yaml** 文件、Kubernetes 清单和 Ignition 配置文件。您还可以选择在安装准备阶段首先设置独立 **var** 分区。

6.4.4.3.1. 可选：创建独立 /var 分区

建议安装程序将 OpenShift Container Platform 的磁盘分区保留给安装程序。然而，在有些情况下您可能需要在文件系统的一部分中创建独立分区。

OpenShift Container Platform 支持添加单个分区来将存储附加到 **/var** 分区或 **/var** 的子目录中。例如：

- **/var/lib/containers** ：保存随着系统中添加更多镜像和容器而增长的容器相关内容。
- **/var/lib/etcd** ：保存您可能希望独立保留的数据，比如 etcd 存储的性能优化。
- **/var** ：保存您可能希望独立保留的数据，以满足审计等目的。

通过单独存储 **/var** 目录的内容，可以更轻松地根据需要为区域扩展存储，并在以后重新安装 OpenShift Container Platform，并保持该数据的完整性。使用这个方法，您不必再次拉取所有容器，在更新系统时也不必复制大量日志文件。

因为 **/var** 在进行一个全新的 Red Hat Enterprise Linux CoreOS（RHCOS）安装前必需存在，所以这个流程会在 OpenShift Container Platform 安装过程的 **openshift-install** 准备阶段插入一个创建的机器配置清单的机器配置来设置独立的 **/var** 分区。

重要

如果按照以下步骤在此流程中创建独立 **/var** 分区，则不需要再次创建 Kubernetes 清单和 Ignition 配置文件，如本节所述。

流程

1. 创建存放 OpenShift Container Platform 安装文件的目录：

```
$ mkdir $HOME/clusterconfig
```

2. 运行 **openshift-install**，以在 **manifest** 和 **openshift** 子目录中创建一组文件。在系统提示时回答系统问题：

```
$ openshift-install create manifests --dir $HOME/clusterconfig
```

输出示例

```
? SSH Public Key ...
INFO Credentials loaded from the "myprofile" profile in file "/home/myuser/.aws/credentials"
INFO Consuming Install Config from target directory
INFO Manifests created in: $HOME/clusterconfig/manifests and
$HOME/clusterconfig/openshift
```

3. 可选：确认安装程序在 **clusterconfig/openshift** 目录中创建了清单：

```
$ ls $HOME/clusterconfig/openshift/
```

输出示例

```
99_kubeadmin-password-secret.yaml
99_openshift-cluster-api_master-machines-0.yaml
99_openshift-cluster-api_master-machines-1.yaml
99_openshift-cluster-api_master-machines-2.yaml
...
```

4. 创建用于配置额外分区的 Butane 配置。例如，将文件命名为 **$HOME/clusterconfig/98-var-partition.bu**，将磁盘设备名称改为 **worker** 系统上存储设备的名称，并根据情况设置存储大小。这个示例将 **/var** 目录放在一个单独的分区中：

```
variant: openshift
version: 4.16.0
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
storage:
  disks:
  - device: /dev/disk/by-id/<device_name> 1
    partitions:
    - label: var
      start_mib: <partition_start_offset> 2
      size_mib: <partition_size> 3
      number: 5
  filesystems:
    - device: /dev/disk/by-partlabel/var
      path: /var
      format: xfs
      mount_options: [defaults, prjquota] 4
      with_mount_unit: true
```

1. 要分区的磁盘的存储设备名称。
2. 在引导磁盘中添加数据分区时，推荐最少使用 25000 MiB(Mebibytes)。root 文件系统会自动调整大小以填充所有可用空间（最多到指定的偏移值）。如果没有指定值，或者指定的值小于推荐的最小值，则生成的 root 文件系统会太小，而在以后进行的 RHCOS 重新安装可能会覆盖数据分区的开始部分。
3. 以兆字节为单位的数据分区大小。
4. 对于用于容器存储的文件系统，必须启用 **prjquota** 挂载选项。

注意

当创建单独的 **/var** 分区时，如果不同的实例类型没有相同的设备名称，则无法为 worker 节点使用不同的实例类型。

5. 从 Butane 配置创建一个清单，并将它保存到 **clusterconfig/openshift** 目录中。例如，运行以下命令：

```
$ butane $HOME/clusterconfig/98-var-partition.bu -o $HOME/clusterconfig/openshift/98-var-partition.yaml
```

6. 再次运行 **openshift-install**，从 **manifest** 和 **openshift** 子目录中的一组文件创建 Ignition 配置：

```
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
auth  bootstrap.ign  master.ign  metadata.json  worker.ign
```

现在，您可以使用 Ignition 配置文件作为安装程序的输入来安装 Red Hat Enterprise Linux CoreOS(RHCOS)系统。

6.4.4.3.2. 创建安装配置文件

生成并自定义安装程序部署集群所需的安装配置文件。

先决条件

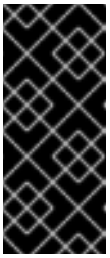
- 已获取 OpenShift Container Platform 安装程序用于用户置备的基础架构和集群的 pull secret。对于受限网络安装，这些文件位于您的堡垒主机上。
- 使用红帽发布的附带 Red Hat Enterprise Linux CoreOS (RHCOS) AMI 检查您是否将集群部署到 AWS 区域。如果要部署到需要自定义 AMI 的 AWS 区域，如 AWS GovCloud 区域，您必须手动创建 **install-config.yaml** 文件。

流程

1. 创建 **install-config.yaml** 文件。
   a. 进入包含安装程序的目录并运行以下命令：

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

1 对于 **<installation_directory>**，请指定要存储安装程序创建的文件的目录名称。



重要

指定一个空目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

   b. 在提示符处，提供云的配置详情：
      i. 可选： 选择用于访问集群机器的 SSH 密钥。

注意

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

ii. 选择 **aws** 作为目标平台。

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

注意

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

iv. 选择要将集群部署到的 AWS Region。

v. 选择您为集群配置的 Route 53 服务的基域。

vi. 为集群输入描述性名称。

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

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

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

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

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

b. 添加 **additionalTrustBundle** 参数和值。该值必须是您用于镜像 registry 的证书文件内容。证书文件可以是现有的可信证书颁发机构，也可以是您为镜像 registry 生成的自签名证书。

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

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

c. 添加镜像内容资源：

```
imageContentSources:
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-v4.0-art-dev
```

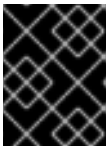
使用命令输出中的 **imageContentSources** 部分来镜像（mirror）仓库，或您从您进入受限网络的介质中的内容时使用的值。

d. 可选：将发布策略设置为 **Internal**：

```
publish: Internal
```

通过设置这个选项，您可以创建一个内部 Ingress Controller 和一个私有负载均衡器。

3. 可选：备份 **install-config.yaml** 文件。



重要

install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用此文件，必须现在备份。

其他资源

- 如需有关 AWS 配置集和凭证配置的更多信息，请参阅 AWS 文档中的配置和凭证文件设置。

6.4.4.3.3. 在安装过程中配置集群范围代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 **Proxy** 对象的 **spec.noProxy** 字段中添加站点来绕过代理。

注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，**Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: ec2.<aws_region>.amazonaws.com,elasticloadbalancing.
<aws_region>.amazonaws.com,s3.<aws_region>.amazonaws.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 **http**。

2 用于创建集群外 HTTPS 连接的代理 URL。

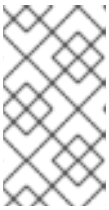
3 要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 **.** 以仅匹配子域。例如，**.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代理。如果您已将 Amazon **EC2**、**Elastic Load Balancing** 和 **S3** VPC 端点添加到 VPC 中，您必须将这些端点添加到 **noProxy** 字段。

4 如果提供，安装程序会在 **openshift-config** 命名空间中生成名为 **user-ca-bundle** 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 **trusted-ca-bundle** 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， **Proxy** 对象的 **trustedCA** 字段中也会引用此配置映射。**additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

5 可选：决定 **Proxy** 对象的配置以引用 **trustedCA** 字段中 **user-ca-bundle** 配置映射的策略。允许的值是 **Proxyonly** 和 **Always**。仅在配置了 **http/https** 代理时，使用 **Proxyonly** 引用 **user-ca-bundle** 配置映射。使用 **Always** 始终引用 **user-ca-bundle** 配置映射。默认值为 **Proxyonly**。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。



注意

如果安装程序超时，重启并使用安装程序的 **wait-for** 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. 保存该文件并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 cluster 的集群范围代理，该代理 **使用** 提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

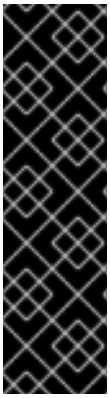
注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

6.4.4.3.4. 创建 Kubernetes 清单和 Ignition 配置文件

由于您必须修改一些集群定义文件并手动启动集群机器，因此您必须生成 Kubernetes 清单和 Ignition 配置文件来配置机器。

安装配置文件转换为 Kubernetes 清单。清单嵌套到 Ignition 配置文件中，稍后用于配置集群机器。



重要

- OpenShift Container Platform 安装程序生成的 Ignition 配置文件包含 24 小时后过期的证书，然后在该时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书中恢复*的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

先决条件

- 已获得 OpenShift Container Platform 安装程序。对于受限网络安装，这些文件位于您的镜像主机上。
- 已创建 **install-config.yaml** 安装配置文件。

流程

1. 进入包含 OpenShift Container Platform 安装程序的目录，并为集群生成 Kubernetes 清单：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定包含您创建的 **install-config.yaml** 文件的安装目录。

2. 删除定义 control plane 机器的 Kubernetes 清单文件：

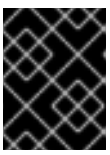
```
$ rm -f <installation_directory>/openshift/99_openshift-cluster-api_master-machines-*.yaml
```

通过删除这些文件，您可以防止集群自动生成 control plane 机器。

3. 删除定义 control plane 机器集的 Kubernetes 清单文件：

```
$ rm -f <installation_directory>/openshift/99_openshift-machine-api_master-control-plane-machine-set.yaml
```

```
$ rm -f <installation_directory>/openshift/99_openshift-cluster-api_worker-machineset-*.yaml
```



重要

如果在用户置备的基础架构上安装集群时禁用了 **MachineAPI** 功能，则必须删除定义 worker 机器的 Kubernetes 清单文件。否则，集群将无法安装。

由于您要自行创建和管理 worker 机器，因此不需要初始化这些机器。

4. 检查 **<installation_directory>/manifests/cluster-scheduler-02-config.yml Kubernetes 清单文件中的 mastersSchedulable** 参数是否已设置为 **false**。此设置可防止在 control plane 机器上调度 pod：

   a. 打开 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** 文件。

b. 找到 **mastersSchedulable** 参数，并确保它被设置为 **false**。

c. 保存并退出 文件。

5. 可选：如果您不希望 Ingress Operator 代表您创建 DNS 记录，请删除 **<installation_directory>/manifests/cluster-dns-02-config.yml** DNS 配置文件中的 **privateZone 和 publicZone** 部分：

```
apiVersion: config.openshift.io/v1
kind: DNS
metadata:
  creationTimestamp: null
  name: cluster
spec:
  baseDomain: example.openshift.com
  privateZone: 1
    id: mycluster-100419-private-zone
  publicZone: 2
    id: example.openshift.com
status: {}
```

1 2 完全删除此部分。

如果这样做，您必须在后续步骤中手动添加入口 DNS 记录。

6. 要创建 Ignition 配置文件，请从包含安装程序的目录运行以下命令：

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定相同的安装目录。

为安装目录中的 bootstrap、control plane 和计算节点创建 Ignition 配置文件。**kubeadmin-password** 和 **kubeconfig** 文件在 **./<installation_directory>/auth** 目录中创建：

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

其他资源

- 手动创建长期凭证

6.4.4.4. 提取基础架构名称

Ignition 配置文件包含一个唯一集群标识符，您可以使用它在 Amazon Web Services (AWS) 中唯一地标识您的集群。基础架构名称还用于在 OpenShift Container Platform 安装过程中定位适当的 AWS 资源。提供的 CloudFormation 模板包含对此基础架构名称的引用，因此您必须提取它。

先决条件

- 已获取 OpenShift Container Platform 安装程序和集群的 pull secret。
- 已为集群生成 Ignition 配置文件。
- 已安装 **jq** 软件包。

流程

- 要从 Ignition 配置文件元数据中提取和查看基础架构名称，请运行以下命令：

```
$ jq -r .infraID <installation_directory>/metadata.json 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

输出示例

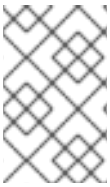
```
openshift-vw9j6 1
```

1 此命令的输出是您的集群名称和随机字符串。

6.4.4.5. 在 AWS 中创建 VPC

您必须在 Amazon Web Services（AWS）中创建 Virtual Private Cloud（VPC），供您的 OpenShift Container Platform 集群使用。您可以自定义 VPC 来满足您的要求，包括 VPN 和路由表。

您可以使用提供的 CloudFormation 模板和自定义参数文件创建代表 VPC 的 AWS 资源堆栈。



注意

如果不使用提供的 CloudFormation 模板来创建 AWS 基础架构，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 已配置了一个 AWS 帐户。
- 您可以通过运行 **aws configure**，将 AWS 密钥和区域添加到本地 AWS 配置集中。
- 已为集群生成 Ignition 配置文件。

流程

1. 创建一个 JSON 文件，其包含模板所需的参数值：

```
[
  {
    "ParameterKey": "VpcCidr", 1
    "ParameterValue": "10.0.0.0/16" 2
  },
```

```
  {
    "ParameterKey": "AvailabilityZoneCount", 3
    "ParameterValue": "1" 4
  },
  {
    "ParameterKey": "SubnetBits", 5
    "ParameterValue": "12" 6
  }
]
```

1. VPC 的 CIDR 块。
2. 以 **x.x.x.x/16-24** 格式指定 CIDR 块。
3. 在其中部署 VPC 的可用区的数量。
4. 指定一个 **1** 到 **3** 之间的整数。
5. 各个可用区中每个子网的大小。
6. 指定 **5** 到 **13** 之间的整数，其中 **5** 为 **/27**，**13** 为 **/19**。

2. 复制本主题的 **VPC 的 CloudFormation 模板**部分中的模板，并将它以 YAML 文件形式保存到计算机上。此模板描述了集群所需的 VPC。
3. 启动 CloudFormation 模板，以创建代表 VPC 的 AWS 资源堆栈：

重要

您必须在一行内输入命令。

```
$ aws cloudformation create-stack --stack-name <name> 1
    --template-body file://<template>.yaml 2
    --parameters file://<parameters>.json 3
```

1. **<name>** 是 CloudFormation 堆栈的名称，如 **cluster-VPC**。如果您删除集群，则需要此堆栈的名称。
2. **<template>** 是您保存的 CloudFormation 模板 YAML 文件的相对路径和名称。
3. **<parameters>** 是 CloudFormation 参数 JSON 文件的相对路径和名称。

输出示例

```
arn:aws:cloudformation:us-east-1:269333783861:stack/cluster-vpc/dbedae40-2fd3-11eb-820e-12a48460849f
```

4. 确认模板组件已存在：

```
$ aws cloudformation describe-stacks --stack-name <name>
```

在 **StackStatus** 显示 **CREATE_COMPLETE** 后，输出会显示以下参数的值。您必须将这些参数值提供给您在创建集群时要运行的其他 CloudFormation 模板：

VpcId	您的 VPC ID。
PublicSubnetIds	新公共子网的 ID。
PrivateSubnetIds	新专用子网的 ID。

6.4.4.5.1. VPC 的 CloudFormation 模板

您可以使用以下 CloudFormation 模板来部署 OpenShift Container Platform 集群所需的 VPC。

例 6.78. VPC 的 CloudFormation 模板

```
AWSTemplateFormatVersion: 2010-09-09
Description: Template for Best Practice VPC with 1-3 AZs

Parameters:
  VpcCidr:
    AllowedPattern: ^(([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])\.){3}([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])(\/(1[6-9]|2[0-4]))$
    ConstraintDescription: CIDR block parameter must be in the form x.x.x.x/16-24.
    Default: 10.0.0.0/16
    Description: CIDR block for VPC.
    Type: String
  AvailabilityZoneCount:
    ConstraintDescription: "The number of availability zones. (Min: 1, Max: 3)"
    MinValue: 1
    MaxValue: 3
    Default: 1
    Description: "How many AZs to create VPC subnets for. (Min: 1, Max: 3)"
    Type: Number
  SubnetBits:
    ConstraintDescription: CIDR block parameter must be in the form x.x.x.x/19-27.
    MinValue: 5
    MaxValue: 13
    Default: 12
    Description: "Size of each subnet to create within the availability zones. (Min: 5 = /27, Max: 13 = /19)"
    Type: Number

Metadata:
  AWS::CloudFormation::Interface:
    ParameterGroups:
    - Label:
        default: "Network Configuration"
      Parameters:
      - VpcCidr
      - SubnetBits
    - Label:
        default: "Availability Zones"
```

```
      Parameters:
      - AvailabilityZoneCount
    ParameterLabels:
      AvailabilityZoneCount:
        default: "Availability Zone Count"
      VpcCidr:
        default: "VPC CIDR"
      SubnetBits:
        default: "Bits Per Subnet"

Conditions:
  DoAz3: !Equals [3, !Ref AvailabilityZoneCount]
  DoAz2: !Or [!Equals [2, !Ref AvailabilityZoneCount], Condition: DoAz3]

Resources:
  VPC:
    Type: "AWS::EC2::VPC"
    Properties:
      EnableDnsSupport: "true"
      EnableDnsHostnames: "true"
      CidrBlock: !Ref VpcCidr
  PublicSubnet:
    Type: "AWS::EC2::Subnet"
    Properties:
      VpcId: !Ref VPC
      CidrBlock: !Select [0, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
      AvailabilityZone: !Select
      - 0
      - Fn::GetAZs: !Ref "AWS::Region"
  PublicSubnet2:
    Type: "AWS::EC2::Subnet"
    Condition: DoAz2
    Properties:
      VpcId: !Ref VPC
      CidrBlock: !Select [1, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
      AvailabilityZone: !Select
      - 1
      - Fn::GetAZs: !Ref "AWS::Region"
  PublicSubnet3:
    Type: "AWS::EC2::Subnet"
    Condition: DoAz3
    Properties:
      VpcId: !Ref VPC
      CidrBlock: !Select [2, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
      AvailabilityZone: !Select
      - 2
      - Fn::GetAZs: !Ref "AWS::Region"
  InternetGateway:
    Type: "AWS::EC2::InternetGateway"
  GatewayToInternet:
    Type: "AWS::EC2::VPCGatewayAttachment"
    Properties:
      VpcId: !Ref VPC
      InternetGatewayId: !Ref InternetGateway
  PublicRouteTable:
    Type: "AWS::EC2::RouteTable"
```

```
    Properties:
      VpcId: !Ref VPC
  PublicRoute:
    Type: "AWS::EC2::Route"
    DependsOn: GatewayToInternet
    Properties:
      RouteTableId: !Ref PublicRouteTable
      DestinationCidrBlock: 0.0.0.0/0
      GatewayId: !Ref InternetGateway
  PublicSubnetRouteTableAssociation:
    Type: "AWS::EC2::SubnetRouteTableAssociation"
    Properties:
      SubnetId: !Ref PublicSubnet
      RouteTableId: !Ref PublicRouteTable
  PublicSubnetRouteTableAssociation2:
    Type: "AWS::EC2::SubnetRouteTableAssociation"
    Condition: DoAz2
    Properties:
      SubnetId: !Ref PublicSubnet2
      RouteTableId: !Ref PublicRouteTable
  PublicSubnetRouteTableAssociation3:
    Condition: DoAz3
    Type: "AWS::EC2::SubnetRouteTableAssociation"
    Properties:
      SubnetId: !Ref PublicSubnet3
      RouteTableId: !Ref PublicRouteTable
  PrivateSubnet:
    Type: "AWS::EC2::Subnet"
    Properties:
      VpcId: !Ref VPC
      CidrBlock: !Select [3, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
      AvailabilityZone: !Select
      - 0
      - Fn::GetAZs: !Ref "AWS::Region"
  PrivateRouteTable:
    Type: "AWS::EC2::RouteTable"
    Properties:
      VpcId: !Ref VPC
  PrivateSubnetRouteTableAssociation:
    Type: "AWS::EC2::SubnetRouteTableAssociation"
    Properties:
      SubnetId: !Ref PrivateSubnet
      RouteTableId: !Ref PrivateRouteTable
  NAT:
    DependsOn:
    - GatewayToInternet
    Type: "AWS::EC2::NatGateway"
    Properties:
      AllocationId:
        "Fn::GetAtt":
        - EIP
        - AllocationId
      SubnetId: !Ref PublicSubnet
  EIP:
    Type: "AWS::EC2::EIP"
    Properties:
```

```
    Domain: vpc
Route:
  Type: "AWS::EC2::Route"
  Properties:
    RouteTableId:
      Ref: PrivateRouteTable
    DestinationCidrBlock: 0.0.0.0/0
    NatGatewayId:
      Ref: NAT
PrivateSubnet2:
  Type: "AWS::EC2::Subnet"
  Condition: DoAz2
  Properties:
    VpcId: !Ref VPC
    CidrBlock: !Select [4, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
    AvailabilityZone: !Select
    - 1
    - Fn::GetAZs: !Ref "AWS::Region"
PrivateRouteTable2:
  Type: "AWS::EC2::RouteTable"
  Condition: DoAz2
  Properties:
    VpcId: !Ref VPC
PrivateSubnetRouteTableAssociation2:
  Type: "AWS::EC2::SubnetRouteTableAssociation"
  Condition: DoAz2
  Properties:
    SubnetId: !Ref PrivateSubnet2
    RouteTableId: !Ref PrivateRouteTable2
NAT2:
  DependsOn:
  - GatewayToInternet
  Type: "AWS::EC2::NatGateway"
  Condition: DoAz2
  Properties:
    AllocationId:
      "Fn::GetAtt":
      - EIP2
      - AllocationId
    SubnetId: !Ref PublicSubnet2
EIP2:
  Type: "AWS::EC2::EIP"
  Condition: DoAz2
  Properties:
    Domain: vpc
Route2:
  Type: "AWS::EC2::Route"
  Condition: DoAz2
  Properties:
    RouteTableId:
      Ref: PrivateRouteTable2
    DestinationCidrBlock: 0.0.0.0/0
    NatGatewayId:
      Ref: NAT2
PrivateSubnet3:
  Type: "AWS::EC2::Subnet"
```

```
  Condition: DoAz3
  Properties:
    VpcId: !Ref VPC
    CidrBlock: !Select [5, !Cidr [!Ref VpcCidr, 6, !Ref SubnetBits]]
    AvailabilityZone: !Select
    - 2
    - Fn::GetAZs: !Ref "AWS::Region"
PrivateRouteTable3:
  Type: "AWS::EC2::RouteTable"
  Condition: DoAz3
  Properties:
    VpcId: !Ref VPC
PrivateSubnetRouteTableAssociation3:
  Type: "AWS::EC2::SubnetRouteTableAssociation"
  Condition: DoAz3
  Properties:
    SubnetId: !Ref PrivateSubnet3
    RouteTableId: !Ref PrivateRouteTable3
NAT3:
  DependsOn:
  - GatewayToInternet
  Type: "AWS::EC2::NatGateway"
  Condition: DoAz3
  Properties:
    AllocationId:
      "Fn::GetAtt":
      - EIP3
      - AllocationId
    SubnetId: !Ref PublicSubnet3
EIP3:
  Type: "AWS::EC2::EIP"
  Condition: DoAz3
  Properties:
    Domain: vpc
Route3:
  Type: "AWS::EC2::Route"
  Condition: DoAz3
  Properties:
    RouteTableId:
      Ref: PrivateRouteTable3
    DestinationCidrBlock: 0.0.0.0/0
    NatGatewayId:
      Ref: NAT3
S3Endpoint:
  Type: AWS::EC2::VPCEndpoint
  Properties:
    PolicyDocument:
      Version: 2012-10-17
      Statement:
      - Effect: Allow
        Principal: '*'
        Action:
        - '*'
        Resource:
        - '*'
    RouteTableIds:
```

```
      - !Ref PublicRouteTable
      - !Ref PrivateRouteTable
      - !If [DoAz2, !Ref PrivateRouteTable2, !Ref "AWS::NoValue"]
      - !If [DoAz3, !Ref PrivateRouteTable3, !Ref "AWS::NoValue"]
      ServiceName: !Join
      - ''
      - - com.amazonaws.
        - !Ref 'AWS::Region'
        - .s3
      VpcId: !Ref VPC

Outputs:
  VpcId:
    Description: ID of the new VPC.
    Value: !Ref VPC
  PublicSubnetIds:
    Description: Subnet IDs of the public subnets.
    Value:
      !Join [
        ",",
        [!Ref PublicSubnet, !If [DoAz2, !Ref PublicSubnet2, !Ref "AWS::NoValue"], !If [DoAz3, !Ref
PublicSubnet3, !Ref "AWS::NoValue"]]
      ]
  PrivateSubnetIds:
    Description: Subnet IDs of the private subnets.
    Value:
      !Join [
        ",",
        [!Ref PrivateSubnet, !If [DoAz2, !Ref PrivateSubnet2, !Ref "AWS::NoValue"], !If [DoAz3, !Ref
PrivateSubnet3, !Ref "AWS::NoValue"]]
      ]
  PublicRouteTableId:
    Description: Public Route table ID
    Value: !Ref PublicRouteTable
  PrivateRouteTableIds:
    Description: Private Route table IDs
    Value:
      !Join [
        ",",
        [
          !Join ["=", [
            !Select [0, "Fn::GetAZs": !Ref "AWS::Region"],
            !Ref PrivateRouteTable
          ]],
          !If [DoAz2,
              !Join ["=", [!Select [1, "Fn::GetAZs": !Ref "AWS::Region"], !Ref PrivateRouteTable2]],
              !Ref "AWS::NoValue"
          ],
          !If [DoAz3,
              !Join ["=", [!Select [2, "Fn::GetAZs": !Ref "AWS::Region"], !Ref PrivateRouteTable3]],
              !Ref "AWS::NoValue"
          ]
        ]
      ]
```

6.4.4.6. 在 AWS 中创建网络和负载均衡组件

您必须在 OpenShift Container Platform 集群可以使用的 Amazon Web Services（AWS）中配置网络、经典或网络负载均衡。

您可以使用提供的 CloudFormation 模板和自定义参数文件来创建 AWS 资源堆栈。堆栈代表 OpenShift Container Platform 集群所需的网络和负载均衡组件。该模板还创建一个托管区和子网标签。

您可以在单一虚拟私有云(VPC)内多次运行该模板。

注意

如果不使用提供的 CloudFormation 模板来创建 AWS 基础架构，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 已配置了一个 AWS 帐户。
- 您可以通过运行 **aws configure**，将 AWS 密钥和区域添加到本地 AWS 配置集中。
- 已为集群生成 Ignition 配置文件。
- 您在 AWS 中创建并配置了 VPC 及相关子网。

流程

1. 获取您在 **install-config.yaml** 文件中为集群指定的 Route 53 基域的托管区 ID。您可以运行以下命令来获取托管区的详细信息：

```
$ aws route53 list-hosted-zones-by-name --dns-name <route53_domain> 1
```

1 对于 **<route53_domain>**，请指定您为集群生成 **install-config.yaml** 文件时所用的 Route53 基域。

输出示例

```
mycluster.example.com. False 100
HOSTEDZONES 65F8F38E-2268-B835-E15C-AB55336FCBFA
/hostedzone/Z21IXYZABCZ2A4 mycluster.example.com. 10
```

在示例输出中，托管区 ID 为 **Z21IXYZABCZ2A4**。

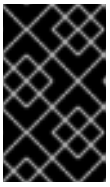
2. 创建一个 JSON 文件，其包含模板所需的参数值：

```
[
  {
    "ParameterKey": "ClusterName", 1
    "ParameterValue": "mycluster" 2
  },
  {
    "ParameterKey": "InfrastructureName", 3
```

```
    "ParameterValue": "mycluster-<random_string>" 4
  },
  {
    "ParameterKey": "HostedZoneId", 5
    "ParameterValue": "<random_string>" 6
  },
  {
    "ParameterKey": "HostedZoneName", 7
    "ParameterValue": "example.com" 8
  },
  {
    "ParameterKey": "PublicSubnets", 9
    "ParameterValue": "subnet-<random_string>" 10
  },
  {
    "ParameterKey": "PrivateSubnets", 11
    "ParameterValue": "subnet-<random_string>" 12
  },
  {
    "ParameterKey": "VpcId", 13
    "ParameterValue": "vpc-<random_string>" 14
  }
]
```

1. 一个简短的、代表集群的名称用于主机名等。
2. 指定您为集群生成 **install-config.yaml** 文件时所用的集群名称。
3. 您的 Ignition 配置文件中为集群编码的集群基础架构名称。
4. 指定从 Ignition 配置文件元数据中提取的基础架构名称，其格式为 **<cluster-name>-<random-string>**。
5. 用来注册目标的 Route 53 公共区 ID。
6. 指定 Route 53 公共区 ID，其格式与 **Z21IXYZABCZ2A4** 类似。您可以从 AWS 控制台获取这个值。
7. 用来注册目标的 Route 53 区。
8. 指定您为集群生成 **install-config.yaml** 文件时所用的 Route 53 基域。请勿包含 AWS 控制台中显示的结尾句点 (.)。
9. 为 VPC 创建的公共子网。
10. 指定 VPC 的 CloudFormation 模板输出的 **PublicSubnetIds** 值。
11. 为 VPC 创建的专用子网。
12. 指定 VPC 的 CloudFormation 模板输出的 **PrivateSubnetIds** 值。
13. 为集群创建的 VPC。
14. 指定 VPC 的 CloudFormation 模板输出的 **VpcId** 值。

3. 复制本主题的**网络和负载均衡器的 CloudFormation 模板**部分中的模板，并将它以 YAML 文件形式保存到计算机上。此模板描述了集群所需的网络和负载均衡对象。



重要

如果要将集群部署到 AWS 政府或 secret 区域，您必须更新 CloudFormation 模板中的 **InternalApiServerRecord**，以使用 **CNAME** 记录。AWS 政府区不支持 **ALIAS** 类型的记录。

4. 启动 CloudFormation 模板，以创建 AWS 资源堆栈，该堆栈提供网络和负载均衡组件：

重要

您必须在一行内输入命令。

```
$ aws cloudformation create-stack --stack-name <name> 1
    --template-body file://<template>.yaml 2
    --parameters file://<parameters>.json 3
    --capabilities CAPABILITY_NAMED_IAM 4
```

1. **<name>** 是 CloudFormation 堆栈的名称，如 **cluster-dns**。如果您删除集群，则需要此堆栈的名称。
2. **<template>** 是您保存的 CloudFormation 模板 YAML 文件的相对路径和名称。
3. **<parameters>** 是 CloudFormation 参数 JSON 文件的相对路径和名称。
4. 您必须明确声明 **CAPABILITY_NAMED_IAM** 功能，因为提供的模板会创建一些 **AWS::IAM::Role** 资源。

输出示例

```
arn:aws:cloudformation:us-east-1:269333783861:stack/cluster-dns/cd3e5de0-2fd4-11eb-5cf0-12be5c33a183
```

5. 确认模板组件已存在：

```
$ aws cloudformation describe-stacks --stack-name <name>
```

在 **StackStatus** 显示 **CREATE_COMPLETE** 后，输出会显示以下参数的值。您必须将这些参数值提供给您在创建集群时要运行的其他 CloudFormation 模板：

PrivateHostedZoneId	专用 DNS 的托管区 ID。
ExternalApiLoadBalancerName	外部 API 负载均衡器的完整名称。

InternalApiLoadBalancerName	内部 API 负载均衡器的完整名称。
ApiServerDnsName	API 服务器的完整主机名。
RegisterNlbIpTargetsLambda	有助于为这些负载均衡器注册/撤销注册 IP 目标的 Lambda ARN。
ExternalApiTargetGroupArn	外部 API 目标组的 ARN。
InternalApiTargetGroupArn	内部 API 目标组的 ARN。
InternalServiceTargetGroupArn	内部服务目标组群的 ARN。

6.4.4.6.1. 网络和负载均衡器的 CloudFormation 模板

您可以使用以下 CloudFormation 模板来部署 OpenShift Container Platform 集群所需的网络对象和负载均衡器。

例 6.79. 网络和负载均衡器的 CloudFormation 模板

```
AWSTemplateFormatVersion: 2010-09-09
Description: Template for OpenShift Cluster Network Elements (Route53 & LBs)

Parameters:
  ClusterName:
    AllowedPattern: ^([a-zA-Z][a-zA-Z0-9\-]{0,26})$
    MaxLength: 27
    MinLength: 1
    ConstraintDescription: Cluster name must be alphanumeric, start with a letter, and have a
maximum of 27 characters.
    Description: A short, representative cluster name to use for host names and other identifying
names.
    Type: String
  InfrastructureName:
    AllowedPattern: ^([a-zA-Z][a-zA-Z0-9\-]{0,26})$
    MaxLength: 27
    MinLength: 1
    ConstraintDescription: Infrastructure name must be alphanumeric, start with a letter, and have a
maximum of 27 characters.
    Description: A short, unique cluster ID used to tag cloud resources and identify items owned or
used by the cluster.
```

```
    Type: String
  HostedZoneId:
    Description: The Route53 public zone ID to register the targets with, such as
Z21IXYZABCZ2A4.
    Type: String
  HostedZoneName:
    Description: The Route53 zone to register the targets with, such as example.com. Omit the
trailing period.
    Type: String
    Default: "example.com"
  PublicSubnets:
    Description: The internet-facing subnets.
    Type: List<AWS::EC2::Subnet::Id>
  PrivateSubnets:
    Description: The internal subnets.
    Type: List<AWS::EC2::Subnet::Id>
  VpcId:
    Description: The VPC-scoped resources will belong to this VPC.
    Type: AWS::EC2::VPC::Id

Metadata:
  AWS::CloudFormation::Interface:
    ParameterGroups:
    - Label:
        default: "Cluster Information"
      Parameters:
      - ClusterName
      - InfrastructureName
    - Label:
        default: "Network Configuration"
      Parameters:
      - VpcId
      - PublicSubnets
      - PrivateSubnets
    - Label:
        default: "DNS"
      Parameters:
      - HostedZoneName
      - HostedZoneId
    ParameterLabels:
      ClusterName:
        default: "Cluster Name"
      InfrastructureName:
        default: "Infrastructure Name"
      VpcId:
        default: "VPC ID"
      PublicSubnets:
        default: "Public Subnets"
      PrivateSubnets:
        default: "Private Subnets"
      HostedZoneName:
        default: "Public Hosted Zone Name"
      HostedZoneId:
        default: "Public Hosted Zone ID"

Resources:
```

```
ExtApiElb:
  Type: AWS::ElasticLoadBalancingV2::LoadBalancer
  Properties:
    Name: !Join ["-", [!Ref InfrastructureName, "ext"]]
    IpAddressType: ipv4
    Subnets: !Ref PublicSubnets
    Type: network

IntApiElb:
  Type: AWS::ElasticLoadBalancingV2::LoadBalancer
  Properties:
    Name: !Join ["-", [!Ref InfrastructureName, "int"]]
    Scheme: internal
    IpAddressType: ipv4
    Subnets: !Ref PrivateSubnets
    Type: network

IntDns:
  Type: "AWS::Route53::HostedZone"
  Properties:
    HostedZoneConfig:
      Comment: "Managed by CloudFormation"
    Name: !Join [".", [!Ref ClusterName, !Ref HostedZoneName]]
    HostedZoneTags:
    - Key: Name
      Value: !Join ["-", [!Ref InfrastructureName, "int"]]
    - Key: !Join ["", ["kubernetes.io/cluster/", !Ref InfrastructureName]]
      Value: "owned"
    VPCs:
    - VPCId: !Ref VpcId
      VPCRegion: !Ref "AWS::Region"

ExternalApiServerRecord:
  Type: AWS::Route53::RecordSetGroup
  Properties:
    Comment: Alias record for the API server
    HostedZoneId: !Ref HostedZoneId
    RecordSets:
    - Name:
        !Join [
          ".",
          ["api", !Ref ClusterName, !Join ["", [!Ref HostedZoneName, "."]]],
        ]
      Type: A
      AliasTarget:
        HostedZoneId: !GetAtt ExtApiElb.CanonicalHostedZoneID
        DNSName: !GetAtt ExtApiElb.DNSName

InternalApiServerRecord:
  Type: AWS::Route53::RecordSetGroup
  Properties:
    Comment: Alias record for the API server
    HostedZoneId: !Ref IntDns
    RecordSets:
    - Name:
        !Join [
```

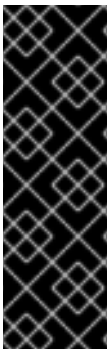
```
          ".",
          ["api", !Ref ClusterName, !Join ["", [!Ref HostedZoneName, "."]]],
        ]
      Type: A
      AliasTarget:
        HostedZoneId: !GetAtt IntApiElb.CanonicalHostedZoneID
        DNSName: !GetAtt IntApiElb.DNSName
    - Name:
        !Join [
          ".",
          ["api-int", !Ref ClusterName, !Join ["", [!Ref HostedZoneName, "."]]],
        ]
      Type: A
      AliasTarget:
        HostedZoneId: !GetAtt IntApiElb.CanonicalHostedZoneID
        DNSName: !GetAtt IntApiElb.DNSName

ExternalApiListener:
  Type: AWS::ElasticLoadBalancingV2::Listener
  Properties:
    DefaultActions:
    - Type: forward
      TargetGroupArn:
        Ref: ExternalApiTargetGroup
    LoadBalancerArn:
      Ref: ExtApiElb
    Port: 6443
    Protocol: TCP

ExternalApiTargetGroup:
  Type: AWS::ElasticLoadBalancingV2::TargetGroup
  Properties:
    HealthCheckIntervalSeconds: 10
    HealthCheckPath: "/readyz"
    HealthCheckPort: 6443
    HealthCheckProtocol: HTTPS
    HealthyThresholdCount: 2
    UnhealthyThresholdCount: 2
    Port: 6443
    Protocol: TCP
    TargetType: ip
    VpcId:
      Ref: VpcId
    TargetGroupAttributes:
    - Key: deregistration_delay.timeout_seconds
      Value: 60

InternalApiListener:
  Type: AWS::ElasticLoadBalancingV2::Listener
  Properties:
    DefaultActions:
    - Type: forward
      TargetGroupArn:
        Ref: InternalApiTargetGroup
    LoadBalancerArn:
      Ref: IntApiElb
```

```
    Port: 6443
    Protocol: TCP

  InternalApiTargetGroup:
    Type: AWS::ElasticLoadBalancingV2::TargetGroup
    Properties:
      HealthCheckIntervalSeconds: 10
      HealthCheckPath: "/readyz"
      HealthCheckPort: 6443
      HealthCheckProtocol: HTTPS
      HealthyThresholdCount: 2
      UnhealthyThresholdCount: 2
      Port: 6443
      Protocol: TCP
      TargetType: ip
      VpcId:
        Ref: VpcId
      TargetGroupAttributes:
      - Key: deregistration_delay.timeout_seconds
        Value: 60

  InternalServiceInternalListener:
    Type: AWS::ElasticLoadBalancingV2::Listener
    Properties:
      DefaultActions:
      - Type: forward
        TargetGroupArn:
          Ref: InternalServiceTargetGroup
      LoadBalancerArn:
        Ref: IntApiElb
      Port: 22623
      Protocol: TCP

  InternalServiceTargetGroup:
    Type: AWS::ElasticLoadBalancingV2::TargetGroup
    Properties:
      HealthCheckIntervalSeconds: 10
      HealthCheckPath: "/healthz"
      HealthCheckPort: 22623
      HealthCheckProtocol: HTTPS
      HealthyThresholdCount: 2
      UnhealthyThresholdCount: 2
      Port: 22623
      Protocol: TCP
      TargetType: ip
      VpcId:
        Ref: VpcId
      TargetGroupAttributes:
      - Key: deregistration_delay.timeout_seconds
        Value: 60

  RegisterTargetLambdaIamRole:
    Type: AWS::IAM::Role
    Properties:
      RoleName: !Join ["-", [!Ref InfrastructureName, "nlb", "lambda", "role"]]
      AssumeRolePolicyDocument:
```

```
        Version: "2012-10-17"
        Statement:
        - Effect: "Allow"
          Principal:
            Service:
            - "lambda.amazonaws.com"
          Action:
          - "sts:AssumeRole"
      Path: "/"
      Policies:
      - PolicyName: !Join ["-", [!Ref InfrastructureName, "master", "policy"]]
        PolicyDocument:
          Version: "2012-10-17"
          Statement:
          - Effect: "Allow"
            Action:
              [
                "elasticloadbalancing:RegisterTargets",
                "elasticloadbalancing:DeregisterTargets",
              ]
            Resource: !Ref InternalApiTargetGroup
          - Effect: "Allow"
            Action:
              [
                "elasticloadbalancing:RegisterTargets",
                "elasticloadbalancing:DeregisterTargets",
              ]
            Resource: !Ref InternalServiceTargetGroup
          - Effect: "Allow"
            Action:
              [
                "elasticloadbalancing:RegisterTargets",
                "elasticloadbalancing:DeregisterTargets",
              ]
            Resource: !Ref ExternalApiTargetGroup

  RegisterNlbIpTargets:
    Type: "AWS::Lambda::Function"
    Properties:
      Handler: "index.handler"
      Role:
        Fn::GetAtt:
        - "RegisterTargetLambdaIamRole"
        - "Arn"
      Code:
        ZipFile: |
          import json
          import boto3
          import cfnresponse
          def handler(event, context):
            elb = boto3.client('elbv2')
            if event['RequestType'] == 'Delete':
              elb.deregister_targets(TargetGroupArn=event['ResourceProperties']
['TargetArn'],Targets=[{'Id': event['ResourceProperties']['TargetIp']}])
            elif event['RequestType'] == 'Create':
              elb.register_targets(TargetGroupArn=event['ResourceProperties']['TargetArn'],Targets=
```

```
[{'Id': event['ResourceProperties']['TargetIp']}])
            responseData = {}
            cfnresponse.send(event, context, cfnresponse.SUCCESS, responseData,
event['ResourceProperties']['TargetArn']+event['ResourceProperties']['TargetIp'])
      Runtime: "python3.8"
      Timeout: 120

  RegisterSubnetTagsLambdaIamRole:
    Type: AWS::IAM::Role
    Properties:
      RoleName: !Join ["-", [!Ref InfrastructureName, "subnet-tags-lambda-role"]]
      AssumeRolePolicyDocument:
        Version: "2012-10-17"
        Statement:
        - Effect: "Allow"
          Principal:
            Service:
            - "lambda.amazonaws.com"
          Action:
          - "sts:AssumeRole"
      Path: "/"
      Policies:
      - PolicyName: !Join ["-", [!Ref InfrastructureName, "subnet-tagging-policy"]]
        PolicyDocument:
          Version: "2012-10-17"
          Statement:
          - Effect: "Allow"
            Action:
              [
                "ec2:DeleteTags",
                "ec2:CreateTags"
              ]
            Resource: "arn:aws:ec2:*:*:subnet/*"
          - Effect: "Allow"
            Action:
              [
                "ec2:DescribeSubnets",
                "ec2:DescribeTags"
              ]
            Resource: "*"

  RegisterSubnetTags:
    Type: "AWS::Lambda::Function"
    Properties:
      Handler: "index.handler"
      Role:
        Fn::GetAtt:
        - "RegisterSubnetTagsLambdaIamRole"
        - "Arn"
      Code:
        ZipFile: |
          import json
          import boto3
          import cfnresponse
          def handler(event, context):
            ec2_client = boto3.client('ec2')
```

```
            if event['RequestType'] == 'Delete':
              for subnet_id in event['ResourceProperties']['Subnets']:
                ec2_client.delete_tags(Resources=[subnet_id], Tags=[{'Key': 'kubernetes.io/cluster/' +
event['ResourceProperties']['InfrastructureName']}]);
            elif event['RequestType'] == 'Create':
              for subnet_id in event['ResourceProperties']['Subnets']:
                ec2_client.create_tags(Resources=[subnet_id], Tags=[{'Key': 'kubernetes.io/cluster/' +
event['ResourceProperties']['InfrastructureName'], 'Value': 'shared'}]);
            responseData = {}
            cfnresponse.send(event, context, cfnresponse.SUCCESS, responseData,
event['ResourceProperties']['InfrastructureName']+event['ResourceProperties']['Subnets'][0])
      Runtime: "python3.8"
      Timeout: 120

  RegisterPublicSubnetTags:
    Type: Custom::SubnetRegister
    Properties:
      ServiceToken: !GetAtt RegisterSubnetTags.Arn
      InfrastructureName: !Ref InfrastructureName
      Subnets: !Ref PublicSubnets

  RegisterPrivateSubnetTags:
    Type: Custom::SubnetRegister
    Properties:
      ServiceToken: !GetAtt RegisterSubnetTags.Arn
      InfrastructureName: !Ref InfrastructureName
      Subnets: !Ref PrivateSubnets

Outputs:
  PrivateHostedZoneId:
    Description: Hosted zone ID for the private DNS, which is required for private records.
    Value: !Ref IntDns
  ExternalApiLoadBalancerName:
    Description: Full name of the external API load balancer.
    Value: !GetAtt ExtApiElb.LoadBalancerFullName
  InternalApiLoadBalancerName:
    Description: Full name of the internal API load balancer.
    Value: !GetAtt IntApiElb.LoadBalancerFullName
  ApiServerDnsName:
    Description: Full hostname of the API server, which is required for the Ignition config files.
    Value: !Join [".", ["api-int", !Ref ClusterName, !Ref HostedZoneName]]
  RegisterNlbIpTargetsLambda:
    Description: Lambda ARN useful to help register or deregister IP targets for these load
balancers.
    Value: !GetAtt RegisterNlbIpTargets.Arn
  ExternalApiTargetGroupArn:
    Description: ARN of the external API target group.
    Value: !Ref ExternalApiTargetGroup
  InternalApiTargetGroupArn:
    Description: ARN of the internal API target group.
    Value: !Ref InternalApiTargetGroup
  InternalServiceTargetGroupArn:
    Description: ARN of the internal service target group.
    Value: !Ref InternalServiceTargetGroup
```



重要

如果要将集群部署到 AWS 政府或 secret 区域，您必须更新 **InternalApiServerRecord** 以使用 **CNAME** 记录。AWS 政府区不支持 **ALIAS** 类型的记录。例如：

```
Type: CNAME
TTL: 10
ResourceRecords:
- !GetAtt IntApiElb.DNSName
```

其他资源

- 有关列出公共托管区的更多信息，请参阅 AWS 文档中的列出公共托管区。

6.4.4.7. 在 AWS 中创建安全组和角色

您必须在 Amazon Web Services (AWS) 中创建安全组和角色，供您的 OpenShift Container Platform 集群使用。

您可以使用提供的 CloudFormation 模板和自定义参数文件来创建 AWS 资源堆栈。堆栈代表 OpenShift Container Platform 集群所需的安全组和角色。

注意

如果不使用提供的 CloudFormation 模板来创建 AWS 基础架构，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 已配置了一个 AWS 帐户。
- 您可以通过运行 **aws configure**，将 AWS 密钥和区域添加到本地 AWS 配置集中。
- 已为集群生成 Ignition 配置文件。
- 您在 AWS 中创建并配置了 VPC 及相关子网。

流程

1. 创建一个 JSON 文件，其包含模板所需的参数值：

```
[
  {
    "ParameterKey": "InfrastructureName", 1
    "ParameterValue": "mycluster-<random_string>" 2
  },
  {
    "ParameterKey": "VpcCidr", 3
    "ParameterValue": "10.0.0.0/16" 4
  },
  {
    "ParameterKey": "PrivateSubnets", 5
    "ParameterValue": "subnet-<random_string>" 6
```

```
  },
  {
    "ParameterKey": "VpcId", 7
    "ParameterValue": "vpc-<random_string>" 8
  }
]
```

1. 您的 Ignition 配置文件中为集群编码的集群基础架构名称。
2. 指定从 Ignition 配置文件元数据中提取的基础架构名称，其格式为 **<cluster-name>-<random-string>**。
3. VPC 的 CIDR 块。
4. 指定以 **x.x.x.x/16-24** 格式定义的用于 VPC 的 CIDR 地址块。
5. 为 VPC 创建的专用子网。
6. 指定 VPC 的 CloudFormation 模板输出的 **PrivateSubnetIds** 值。
7. 为集群创建的 VPC。
8. 指定 VPC 的 CloudFormation 模板输出的 **VpcId** 值。

2. 复制本主题的**安全对象的 CloudFormation 模板**部分中的模板，并将它以 YAML 文件形式保存到计算机上。此模板描述了集群所需的安全组和角色。
3. 启动 CloudFormation 模板，以创建代表安全组和角色的 AWS 资源堆栈：

重要

您必须在一行内输入命令。

```
$ aws cloudformation create-stack --stack-name <name> 1
    --template-body file://<template>.yaml 2
    --parameters file://<parameters>.json 3
    --capabilities CAPABILITY_NAMED_IAM 4
```

1. **<name>** 是 CloudFormation 堆栈的名称，如 **cluster-sec**。如果您删除集群，则需要此堆栈的名称。
2. **<template>** 是您保存的 CloudFormation 模板 YAML 文件的相对路径和名称。
3. **<parameters>** 是 CloudFormation 参数 JSON 文件的相对路径和名称。
4. 您必须明确声明 **CAPABILITY_NAMED_IAM** 功能，因为提供的模板会创建一些 **AWS::IAM::Role** 和 **AWS::IAM::InstanceProfile** 资源。

输出示例

```
arn:aws:cloudformation:us-east-1:269333783861:stack/cluster-sec/03bd4210-2ed7-11eb-6d7a-13fc0b61e9db
```

4. 确认模板组件已存在：

```
$ aws cloudformation describe-stacks --stack-name <name>
```

在 **StackStatus** 显示 **CREATE_COMPLETE** 后，输出会显示以下参数的值。您必须将这些参数值提供给您在创建集群时要运行的其他 CloudFormation 模板：

MasterSecurityGroupId	Master 安全组 ID
WorkerSecurityGroupId	worker 安全组 ID
MasterInstanceProfile	Master IAM 实例配置集
WorkerInstanceProfile	worker IAM 实例配置集

6.4.4.7.1. 安全对象的 CloudFormation 模板

您可以使用以下 CloudFormation 模板来部署 OpenShift Container Platform 集群所需的安全对象。

例 6.80. 安全对象的 CloudFormation 模板

```
AWSTemplateFormatVersion: 2010-09-09
Description: Template for OpenShift Cluster Security Elements (Security Groups & IAM)

Parameters:
  InfrastructureName:
    AllowedPattern: ^([a-zA-Z][a-zA-Z0-9\-]{0,26})$
    MaxLength: 27
    MinLength: 1
    ConstraintDescription: Infrastructure name must be alphanumeric, start with a letter, and have a maximum of 27 characters.
    Description: A short, unique cluster ID used to tag cloud resources and identify items owned or used by the cluster.
    Type: String
  VpcCidr:
    AllowedPattern: ^(([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])\.){3}([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])(\/(1[6-9]|2[0-4]))$
    ConstraintDescription: CIDR block parameter must be in the form x.x.x.x/16-24.
    Default: 10.0.0.0/16
    Description: CIDR block for VPC.
    Type: String
  VpcId:
    Description: The VPC-scoped resources will belong to this VPC.
    Type: AWS::EC2::VPC::Id
  PrivateSubnets:
```

```
    Description: The internal subnets.
    Type: List<AWS::EC2::Subnet::Id>

Metadata:
  AWS::CloudFormation::Interface:
    ParameterGroups:
    - Label:
        default: "Cluster Information"
      Parameters:
      - InfrastructureName
    - Label:
        default: "Network Configuration"
      Parameters:
      - VpcId
      - VpcCidr
      - PrivateSubnets
    ParameterLabels:
      InfrastructureName:
        default: "Infrastructure Name"
      VpcId:
        default: "VPC ID"
      VpcCidr:
        default: "VPC CIDR"
      PrivateSubnets:
        default: "Private Subnets"

Resources:
  MasterSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: Cluster Master Security Group
      SecurityGroupIngress:
      - IpProtocol: icmp
        FromPort: 0
        ToPort: 0
        CidrIp: !Ref VpcCidr
      - IpProtocol: tcp
        FromPort: 22
        ToPort: 22
        CidrIp: !Ref VpcCidr
      - IpProtocol: tcp
        ToPort: 6443
        FromPort: 6443
        CidrIp: !Ref VpcCidr
      - IpProtocol: tcp
        FromPort: 22623
        ToPort: 22623
        CidrIp: !Ref VpcCidr
      VpcId: !Ref VpcId

  WorkerSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: Cluster Worker Security Group
      SecurityGroupIngress:
      - IpProtocol: icmp
```

```
      FromPort: 0
      ToPort: 0
      CidrIp: !Ref VpcCidr
    - IpProtocol: tcp
      FromPort: 22
      ToPort: 22
      CidrIp: !Ref VpcCidr
    VpcId: !Ref VpcId

MasterIngressEtcd:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: etcd
    FromPort: 2379
    ToPort: 2380
    IpProtocol: tcp

MasterIngressVxlan:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: Vxlan packets
    FromPort: 4789
    ToPort: 4789
    IpProtocol: udp

MasterIngressWorkerVxlan:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
    Description: Vxlan packets
    FromPort: 4789
    ToPort: 4789
    IpProtocol: udp

MasterIngressGeneve:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: Geneve packets
    FromPort: 6081
    ToPort: 6081
    IpProtocol: udp

MasterIngressWorkerGeneve:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
    Description: Geneve packets
    FromPort: 6081
```

```
      ToPort: 6081
      IpProtocol: udp

  MasterIngressIpsecIke:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt MasterSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
      Description: IPsec IKE packets
      FromPort: 500
      ToPort: 500
      IpProtocol: udp

  MasterIngressIpsecNat:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt MasterSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
      Description: IPsec NAT-T packets
      FromPort: 4500
      ToPort: 4500
      IpProtocol: udp

  MasterIngressIpsecEsp:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt MasterSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
      Description: IPsec ESP packets
      IpProtocol: 50

  MasterIngressWorkerIpsecIke:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt MasterSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
      Description: IPsec IKE packets
      FromPort: 500
      ToPort: 500
      IpProtocol: udp

  MasterIngressWorkerIpsecNat:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt MasterSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
      Description: IPsec NAT-T packets
      FromPort: 4500
      ToPort: 4500
      IpProtocol: udp

  MasterIngressWorkerIpsecEsp:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt MasterSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
```

```
    Description: IPsec ESP packets
    IpProtocol: 50

MasterIngressInternal:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: Internal cluster communication
    FromPort: 9000
    ToPort: 9999
    IpProtocol: tcp

MasterIngressWorkerInternal:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
    Description: Internal cluster communication
    FromPort: 9000
    ToPort: 9999
    IpProtocol: tcp

MasterIngressInternalUDP:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: Internal cluster communication
    FromPort: 9000
    ToPort: 9999
    IpProtocol: udp

MasterIngressWorkerInternalUDP:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
    Description: Internal cluster communication
    FromPort: 9000
    ToPort: 9999
    IpProtocol: udp

MasterIngressKube:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt MasterSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: Kubernetes kubelet, scheduler and controller manager
    FromPort: 10250
    ToPort: 10259
    IpProtocol: tcp

MasterIngressWorkerKube:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
```

```
      GroupId: !GetAtt MasterSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
      Description: Kubernetes kubelet, scheduler and controller manager
      FromPort: 10250
      ToPort: 10259
      IpProtocol: tcp

  MasterIngressIngressServices:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt MasterSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
      Description: Kubernetes ingress services
      FromPort: 30000
      ToPort: 32767
      IpProtocol: tcp

  MasterIngressWorkerIngressServices:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt MasterSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
      Description: Kubernetes ingress services
      FromPort: 30000
      ToPort: 32767
      IpProtocol: tcp

  MasterIngressIngressServicesUDP:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt MasterSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
      Description: Kubernetes ingress services
      FromPort: 30000
      ToPort: 32767
      IpProtocol: udp

  MasterIngressWorkerIngressServicesUDP:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt MasterSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
      Description: Kubernetes ingress services
      FromPort: 30000
      ToPort: 32767
      IpProtocol: udp

  WorkerIngressVxlan:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt WorkerSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
      Description: Vxlan packets
      FromPort: 4789
      ToPort: 4789
      IpProtocol: udp
```

```
WorkerIngressMasterVxlan:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: Vxlan packets
    FromPort: 4789
    ToPort: 4789
    IpProtocol: udp

WorkerIngressGeneve:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
    Description: Geneve packets
    FromPort: 6081
    ToPort: 6081
    IpProtocol: udp

WorkerIngressMasterGeneve:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: Geneve packets
    FromPort: 6081
    ToPort: 6081
    IpProtocol: udp

WorkerIngressIpsecIke:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
    Description: IPsec IKE packets
    FromPort: 500
    ToPort: 500
    IpProtocol: udp

WorkerIngressIpsecNat:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
    Description: IPsec NAT-T packets
    FromPort: 4500
    ToPort: 4500
    IpProtocol: udp

WorkerIngressIpsecEsp:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
```

```
    Description: IPsec ESP packets
    IpProtocol: 50

WorkerIngressMasterIpsecIke:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: IPsec IKE packets
    FromPort: 500
    ToPort: 500
    IpProtocol: udp

WorkerIngressMasterIpsecNat:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: IPsec NAT-T packets
    FromPort: 4500
    ToPort: 4500
    IpProtocol: udp

WorkerIngressMasterIpsecEsp:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: IPsec ESP packets
    IpProtocol: 50

WorkerIngressInternal:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
    Description: Internal cluster communication
    FromPort: 9000
    ToPort: 9999
    IpProtocol: tcp

WorkerIngressMasterInternal:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
    Description: Internal cluster communication
    FromPort: 9000
    ToPort: 9999
    IpProtocol: tcp

WorkerIngressInternalUDP:
  Type: AWS::EC2::SecurityGroupIngress
  Properties:
    GroupId: !GetAtt WorkerSecurityGroup.GroupId
    SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
```

```
    Description: Internal cluster communication
    FromPort: 9000
    ToPort: 9999
    IpProtocol: udp

  WorkerIngressMasterInternalUDP:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt WorkerSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
      Description: Internal cluster communication
      FromPort: 9000
      ToPort: 9999
      IpProtocol: udp

  WorkerIngressKube:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt WorkerSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
      Description: Kubernetes secure kubelet port
      FromPort: 10250
      ToPort: 10250
      IpProtocol: tcp

  WorkerIngressWorkerKube:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt WorkerSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
      Description: Internal Kubernetes communication
      FromPort: 10250
      ToPort: 10250
      IpProtocol: tcp

  WorkerIngressIngressServices:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt WorkerSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
      Description: Kubernetes ingress services
      FromPort: 30000
      ToPort: 32767
      IpProtocol: tcp

  WorkerIngressMasterIngressServices:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt WorkerSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
      Description: Kubernetes ingress services
      FromPort: 30000
      ToPort: 32767
      IpProtocol: tcp

  WorkerIngressIngressServicesUDP:
```

```
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt WorkerSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt WorkerSecurityGroup.GroupId
      Description: Kubernetes ingress services
      FromPort: 30000
      ToPort: 32767
      IpProtocol: udp

  WorkerIngressMasterIngressServicesUDP:
    Type: AWS::EC2::SecurityGroupIngress
    Properties:
      GroupId: !GetAtt WorkerSecurityGroup.GroupId
      SourceSecurityGroupId: !GetAtt MasterSecurityGroup.GroupId
      Description: Kubernetes ingress services
      FromPort: 30000
      ToPort: 32767
      IpProtocol: udp

  MasterIamRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: "2012-10-17"
        Statement:
        - Effect: "Allow"
          Principal:
            Service:
            - "ec2.amazonaws.com"
          Action:
          - "sts:AssumeRole"
      Policies:
      - PolicyName: !Join ["-", [!Ref InfrastructureName, "master", "policy"]]
        PolicyDocument:
          Version: "2012-10-17"
          Statement:
          - Effect: "Allow"
            Action:
            - "ec2:AttachVolume"
            - "ec2:AuthorizeSecurityGroupIngress"
            - "ec2:CreateSecurityGroup"
            - "ec2:CreateTags"
            - "ec2:CreateVolume"
            - "ec2:DeleteSecurityGroup"
            - "ec2:DeleteVolume"
            - "ec2:Describe*"
            - "ec2:DetachVolume"
            - "ec2:ModifyInstanceAttribute"
            - "ec2:ModifyVolume"
            - "ec2:RevokeSecurityGroupIngress"
            - "elasticloadbalancing:AddTags"
            - "elasticloadbalancing:AttachLoadBalancerToSubnets"
            - "elasticloadbalancing:ApplySecurityGroupsToLoadBalancer"
            - "elasticloadbalancing:CreateListener"
            - "elasticloadbalancing:CreateLoadBalancer"
            - "elasticloadbalancing:CreateLoadBalancerPolicy"
```

```
          - "elasticloadbalancing:CreateLoadBalancerListeners"
          - "elasticloadbalancing:CreateTargetGroup"
          - "elasticloadbalancing:ConfigureHealthCheck"
          - "elasticloadbalancing:DeleteListener"
          - "elasticloadbalancing:DeleteLoadBalancer"
          - "elasticloadbalancing:DeleteLoadBalancerListeners"
          - "elasticloadbalancing:DeleteTargetGroup"
          - "elasticloadbalancing:DeregisterInstancesFromLoadBalancer"
          - "elasticloadbalancing:DeregisterTargets"
          - "elasticloadbalancing:Describe*"
          - "elasticloadbalancing:DetachLoadBalancerFromSubnets"
          - "elasticloadbalancing:ModifyListener"
          - "elasticloadbalancing:ModifyLoadBalancerAttributes"
          - "elasticloadbalancing:ModifyTargetGroup"
          - "elasticloadbalancing:ModifyTargetGroupAttributes"
          - "elasticloadbalancing:RegisterInstancesWithLoadBalancer"
          - "elasticloadbalancing:RegisterTargets"
          - "elasticloadbalancing:SetLoadBalancerPoliciesForBackendServer"
          - "elasticloadbalancing:SetLoadBalancerPoliciesOfListener"
          - "kms:DescribeKey"
          Resource: "*"

  MasterInstanceProfile:
    Type: "AWS::IAM::InstanceProfile"
    Properties:
      Roles:
      - Ref: "MasterIamRole"

  WorkerIamRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: "2012-10-17"
        Statement:
        - Effect: "Allow"
          Principal:
            Service:
            - "ec2.amazonaws.com"
          Action:
          - "sts:AssumeRole"
      Policies:
      - PolicyName: !Join ["-", [!Ref InfrastructureName, "worker", "policy"]]
        PolicyDocument:
          Version: "2012-10-17"
          Statement:
          - Effect: "Allow"
            Action:
            - "ec2:DescribeInstances"
            - "ec2:DescribeRegions"
            Resource: "*"

  WorkerInstanceProfile:
    Type: "AWS::IAM::InstanceProfile"
    Properties:
      Roles:
      - Ref: "WorkerIamRole"
```

```
Outputs:
  MasterSecurityGroupId:
    Description: Master Security Group ID
    Value: !GetAtt MasterSecurityGroup.GroupId

  WorkerSecurityGroupId:
    Description: Worker Security Group ID
    Value: !GetAtt WorkerSecurityGroup.GroupId

  MasterInstanceProfile:
    Description: Master IAM Instance Profile
    Value: !Ref MasterInstanceProfile

  WorkerInstanceProfile:
    Description: Worker IAM Instance Profile
    Value: !Ref WorkerInstanceProfile
```

6.4.4.8. 使用流元数据访问 RHCOS AMI

在 OpenShift Container Platform 中，*流元数据*以 JSON 格式提供与 RHCOS 相关的标准化元数据，并将元数据注入集群中。流元数据是一种稳定的格式，支持多种架构，旨在自我记录以维护自动化。

您可以使用 **openshift-install** 的 **coreos print-stream-json** 子命令访问流元数据格式的引导镜像的信息。此命令提供了一种以可脚本、机器可读格式打印流元数据的方法。

对于用户置备的安装，**openshift-install** 二进制文件包含对经过测试用于 OpenShift Container Platform 的 RHCOS 引导镜像版本的引用，如 AWS AMI。

流程

要解析流元数据，请使用以下方法之一：

- 在 Go 程序中使用位于 https://github.com/coreos/stream-metadata-go 的正式 **stream-metadata-go** 库。您还可以查看库中的示例代码。
- 在 Python 或 Ruby 等其他编程语言中使用您首选编程语言的 JSON 库。
- 在处理 JSON 数据的命令行工具中，如 **jq**：
  - 为 AWS 区域输出当前的 **x86_64** 或 **aarch64** AMI，如 **us-west-1** ：

    对于 x86_64

    ```
    $ openshift-install coreos print-stream-json | jq -r
    '.architectures.x86_64.images.aws.regions["us-west-1"].image'
    ```

    输出示例

    ```
    ami-0d3e625f84626bbda
    ```

    对于 aarch64

```
$ openshift-install coreos print-stream-json | jq -r
'.architectures.aarch64.images.aws.regions["us-west-1"].image'
```

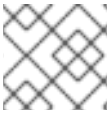
输出示例

```
ami-0af1d3b7fa5be2131
```

这个命令的输出是您指定的架构和 **us-west-1** 区域的 AWS AMI ID。AMI 必须与集群属于同一区域。

6.4.4.9. AWS 基础架构的 RHCOS AMI

红帽提供了对可手动为 OpenShift Container Platform 节点指定的各种 AWS 区域和实例架构有效的 Red Hat Enterprise Linux CoreOS(RHCOS)AMI。



注意

通过导入您自己的 AMI，您还可以安装到没有公布的 RHCOS AMI 的区域。

表 6.25. x86_64 RHCOS AMIs

AWS 区	AWS AMI
af-south-1	**ami-018de6b1be470b7e6**
ap-east-1	**ami-092f09a4c8aacf56a**
ap-northeast-1	**ami-001c9fc85b77505f4**
ap-northeast-2	**ami-0a5d7f1966d88fba3**
ap-northeast-3	**ami-085a2726ceb4f5eeb**
ap-south-1	**ami-03f2c19071fb6eab3**
ap-south-2	**ami-0c82522890979d94b**
ap-southeast-1	**ami-06e5b9d16ead6c55c**
ap-southeast-2	**ami-0ccd429129fbf6bc0**
ap-southeast-3	**ami-096f9b4d41116d95c**
ap-southeast-4	**ami-0b511ab55f9f7d052**
ca-central-1	**ami-0e357287c651be1f2**
ca-west-1	**ami-0b1692e5776740901**

AWS 区	AWS AMI
eu-central-1	ami-08b13fb388ba5610d
eu-central-2	ami-04777298b55045ea9
eu-north-1	ami-05421993a4ee567b9
eu-south-1	ami-00959341869d34746
eu-south-2	ami-0a745b610522a87cc
eu-west-1	ami-0e48f85ec57bd7baa
eu-west-2	ami-0c015b1387acddc5e
eu-west-3	ami-01e466af77a95b93d
il-central-1	ami-0a1b706793b54d087
me-central-1	ami-09d58e8b2099ae5bc
me-south-1	ami-0f9c259bc4346599c
sa-east-1	ami-06277517d966881a3
us-east-1	ami-05483066c3caaccf5
us-east-2	ami-0c704ce516493a479
us-gov-east-1	ami-0695b2cbdd1ec4d48
us-gov-west-1	ami-004404a0eaa427b39
us-west-1	ami-0aaa56637063be772
us-west-2	ami-0870c7a3d5ef4d2b3

表 6.26. aarch64 RHCOS AMI

AWS 区	AWS AMI
af-south-1	ami-0d96d447d3df14f4e
ap-east-1	ami-010236402dfba1717

AWS 区	AWS AMI
ap-northeast-1	ami-08feeb6d9068bb12e
ap-northeast-2	ami-0772082c119147205
ap-northeast-3	ami-05bcbb13493d0516f
ap-south-1	ami-0034a539e8adcb817
ap-south-2	ami-0fc56b7c53c38ff70
ap-southeast-1	ami-0b50a0e92a58f3ad8
ap-southeast-2	ami-0697a9174ae206182
ap-southeast-3	ami-05ae309319a7ad504
ap-southeast-4	ami-00500414843baa657
ca-central-1	ami-05e76bf99d3b0d4c8
ca-west-1	ami-099fc953c221ec590
eu-central-1	ami-0d2e2698884020b71
eu-central-2	ami-0a96ba21ddee01719
eu-north-1	ami-0ab8a1070d0b1d9a5
eu-south-1	ami-0d0465299a8b196c1
eu-south-2	ami-07d5aa3c6d468c738
eu-west-1	ami-08e7d209f26c09c80
eu-west-2	ami-0c8336d4e2c1f13cb
eu-west-3	ami-085d804b98acf0648
il-central-1	ami-02ce030e36aeb7643
me-central-1	ami-0dea3ac466040b77f
me-south-1	ami-02ef2a46887b9786d

AWS 区	AWS AMI
sa-east-1	**ami-0d19817d31b2399f9**
us-east-1	**ami-04859c888566a2c2f**
us-east-2	**ami-02f7e4a5dc66361bb**
us-gov-east-1	**ami-067ef782980ea96ad**
us-gov-west-1	**ami-0ce67540c1bb2319d**
us-west-1	**ami-0a09c5d2f287718a3**
us-west-2	**ami-06b94e64e595f6cee**

6.4.4.10. 在 AWS 中创建 bootstrap 节点

您必须在 Amazon Web Services (AWS) 中创建 bootstrap 节点，以便在 OpenShift Container Platform 集群初始化过程中使用。您可以按照以下方法：

- 为集群提供 **bootstrap.ign** Ignition 配置文件的位置。此文件位于您的安装目录中。提供的 CloudFormation 模板假定集群的 Ignition 配置文件由 S3 存储桶提供。如果选择从其他位置提供文件，您必须修改模板。
- 使用提供的 CloudFormation 模板和自定义参数文件来创建 AWS 资源堆栈。堆栈代表 OpenShift Container Platform 安装所需的 bootstrap 节点。

注意

如果不使用提供的 CloudFormation 模板来创建 bootstrap 节点，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 已配置了一个 AWS 帐户。
- 您可以通过运行 **aws configure**，将 AWS 密钥和区域添加到本地 AWS 配置集中。
- 已为集群生成 Ignition 配置文件。
- 您在 AWS 中创建并配置了 VPC 及相关子网。
- 您在 AWS 中创建并配置了 DNS、负载均衡器和监听程序。
- 您在 AWS 中创建了集群所需的安全组和角色。

流程

1. 运行以下命令来创建存储桶：

```
$ aws s3 mb s3://<cluster-name>-infra 1
```

1 **<cluster-name>-infra** 是存储桶名称。在创建 **install-config.yaml** 文件时，将 **<cluster-name>** 替换为为集群指定的名称。

如果需要，您必须为 S3 存储桶使用预签名 URL，而不是 **s3://** 模式：

- 部署到具有与 AWS SDK 不同端点的区域。
- 部署代理。
- 提供您自己的自定义端点。

2. 运行以下命令，将 **bootstrap.ign** Ignition 配置文件上传到存储桶：

```
$ aws s3 cp <installation_directory>/bootstrap.ign s3://<cluster-name>-infra/bootstrap.ign 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

3. 运行以下命令验证文件是否已上传：

```
$ aws s3 ls s3://<cluster-name>-infra/
```

输出示例

```
2019-04-03 16:15:16    314878 bootstrap.ign
```

注意

bootstrap Ignition 配置文件包含 secret，如 X.509 密钥。以下步骤为 S3 存储桶提供基本安全性。若要提供额外的安全性，您可以启用 S3 存储桶策略，仅允许某些用户（如 OpenShift IAM 用户）访问存储桶中包含的对象。您可以完全避开 S3，并从 bootstrap 可访问的任意地址提供 bootstrap Ignition 配置文件。

4. 创建一个 JSON 文件，其包含模板所需的参数值：

```
[
  {
    "ParameterKey": "InfrastructureName", 1
    "ParameterValue": "mycluster-<random_string>" 2
  },
  {
    "ParameterKey": "RhcosAmi", 3
    "ParameterValue": "ami-<random_string>" 4
  },
  {
    "ParameterKey": "AllowedBootstrapSshCidr", 5
    "ParameterValue": "0.0.0.0/0" 6
  },
  {
    "ParameterKey": "PublicSubnet", 7
```

```
    "ParameterValue": "subnet-<random_string>" 8
  },
  {
    "ParameterKey": "MasterSecurityGroupId", 9
    "ParameterValue": "sg-<random_string>" 10
  },
  {
    "ParameterKey": "VpcId", 11
    "ParameterValue": "vpc-<random_string>" 12
  },
  {
    "ParameterKey": "BootstrapIgnitionLocation", 13
    "ParameterValue": "s3://<bucket_name>/bootstrap.ign" 14
  },
  {
    "ParameterKey": "AutoRegisterELB", 15
    "ParameterValue": "yes" 16
  },
  {
    "ParameterKey": "RegisterNlbIpTargetsLambdaArn", 17
    "ParameterValue": "arn:aws:lambda:<aws_region>:<account_number>:function:
<dns_stack_name>-RegisterNlbIpTargets-<random_string>" 18
  },
  {
    "ParameterKey": "ExternalApiTargetGroupArn", 19
    "ParameterValue": "arn:aws:elasticloadbalancing:<aws_region>:
<account_number>:targetgroup/<dns_stack_name>-Exter-<random_string>" 20
  },
  {
    "ParameterKey": "InternalApiTargetGroupArn", 21
    "ParameterValue": "arn:aws:elasticloadbalancing:<aws_region>:
<account_number>:targetgroup/<dns_stack_name>-Inter-<random_string>" 22
  },
  {
    "ParameterKey": "InternalServiceTargetGroupArn", 23
    "ParameterValue": "arn:aws:elasticloadbalancing:<aws_region>:
<account_number>:targetgroup/<dns_stack_name>-Inter-<random_string>" 24
  }
]
```

1. 您的 Ignition 配置文件中为集群编码的集群基础架构名称。
2. 指定从 Ignition 配置文件元数据中提取的基础架构名称，其格式为 **<cluster-name>-<random-string>**。
3. 根据您选择的架构，当前 Red Hat Enterprise Linux CoreOS (RHCOS) AMI 用于 bootstrap 节点。
4. 指定有效的 **AWS::EC2::Image::Id** 值。
5. 允许通过 SSH 访问 bootstrap 节点的 CIDR 块。
6. 以 **x.x.x.x/16-24** 格式指定 CIDR 块。

7. 与 VPC 关联的公共子网，将 bootstrap 节点启动到其中。
8. 指定 VPC 的 CloudFormation 模板输出的 **PublicSubnetIds** 值。
9. master 安全组 ID（用于注册临时规则）
10. 指定安全组和角色的 CloudFormation 模板输出的 **MasterSecurityGroupId** 值。
11. 创建的资源将从属于的 VPC。
12. 指定 VPC 的 CloudFormation 模板输出的 **VpcId** 值。
13. 从中获取 bootstrap Ignition 配置文件的位置。
14. 指定 S3 存储桶和文件名，格式为 **s3://<bucket_name>/bootstrap.ign**。
15. 是否要注册网络负载均衡器 (NLB) 。
16. 指定 **yes** 或 **no**。如果指定 **yes**，您必须提供一个 Lambda Amazon Resource Name (ARN) 值。
17. NLB IP 目标注册 lambda 组的 ARN。
18. 指定 DNS 和负载均衡的 CloudFormation 模板输出的 **RegisterNlbIpTargetsLambda** 值。如果将集群部署到 AWS GovCloud 区域，请使用 **arn:aws-us-gov**。
19. 外部 API 负载均衡器目标组的 ARN。
20. 指定 DNS 和负载均衡的 CloudFormation 模板输出的 **ExternalApiTargetGroupArn** 值。如果将集群部署到 AWS GovCloud 区域，请使用 **arn:aws-us-gov**。
21. 内部 API 负载均衡器目标组群的 ARN。
22. 指定 DNS 和负载均衡的 CloudFormation 模板输出的 **InternalApiTargetGroupArn** 值。如果将集群部署到 AWS GovCloud 区域，请使用 **arn:aws-us-gov**。
23. 内部服务负载均衡器目标组群的 ARN。
24. 指定 DNS 和负载均衡的 CloudFormation 模板输出的 **InternalServiceTargetGroupArn** 值。如果将集群部署到 AWS GovCloud 区域，请使用 **arn:aws-us-gov**。

5. 复制本主题的 **Bootstrap 机器的 CloudFormation 模板**部分中的模板，并将它以 YAML 文件形式保存到计算机上。此模板描述了集群所需的 bootstrap 机器。
6. 可选：如果要使用代理部署集群，您必须更新模板中的 ignition 以添加 **ignition.config.proxy** 字段。另外，如果您已将 Amazon EC2、Elastic Load Balancing 和 S3 VPC 端点添加到 VPC 中，您必须将这些端点添加到 **noProxy** 字段。
7. 启动 CloudFormation 模板，以创建代表 bootstrap 节点的 AWS 资源堆栈：

重要

您必须在一行内输入命令。

```
$ aws cloudformation create-stack --stack-name <name> 1
```

```
  --template-body file://<template>.yaml 2
  --parameters file://<parameters>.json 3
  --capabilities CAPABILITY_NAMED_IAM 4
```

1. **<name>** 是 CloudFormation 堆栈的名称，如 **cluster-bootstrap**。如果您删除集群，则需要此堆栈的名称。
2. **<template>** 是您保存的 CloudFormation 模板 YAML 文件的相对路径和名称。
3. **<parameters>** 是 CloudFormation 参数 JSON 文件的相对路径和名称。
4. 您必须明确声明 **CAPABILITY_NAMED_IAM** 功能，因为提供的模板会创建一些 **AWS::IAM::Role** 和 **AWS::IAM::InstanceProfile** 资源。

输出示例

```
arn:aws:cloudformation:us-east-1:269333783861:stack/cluster-bootstrap/12944486-2add-11eb-9dee-12dace8e3a83
```

8. 确认模板组件已存在：

```
$ aws cloudformation describe-stacks --stack-name <name>
```

在 **StackStatus** 显示 **CREATE_COMPLETE** 后，输出会显示以下参数的值。您必须将这些参数值提供给您在创建集群时要运行的其他 CloudFormation 模板：

BootstrapInstanceId	bootstrap 实例 ID。
BootstrapPublicIp	bootstrap 节点公共 IP 地址。
BootstrapPrivateIp	bootstrap 节点专用 IP 地址。

6.4.4.10.1. bootstrap 机器的 CloudFormation 模板

您可以使用以下 CloudFormation 模板来部署 OpenShift Container Platform 集群所需的 bootstrap 机器。

例 6.81. bootstrap 机器的 CloudFormation 模板

```
AWSTemplateFormatVersion: 2010-09-09
Description: Template for OpenShift Cluster Bootstrap (EC2 Instance, Security Groups and IAM)

Parameters:
  InfrastructureName:
    AllowedPattern: ^([a-zA-Z][a-zA-Z0-9\-]{0,26})$
    MaxLength: 27
    MinLength: 1
    ConstraintDescription: Infrastructure name must be alphanumeric, start with a letter, and have a
```

```
maximum of 27 characters.
    Description: A short, unique cluster ID used to tag cloud resources and identify items owned or
used by the cluster.
    Type: String
  RhcosAmi:
    Description: Current Red Hat Enterprise Linux CoreOS AMI to use for bootstrap.
    Type: AWS::EC2::Image::Id
  AllowedBootstrapSshCidr:
    AllowedPattern: ^(([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-4][0-9]|25[0-5])\.){3}([0-9]|[1-9][0-9]|1[0-9]{2}|2[0-
4][0-9]|25[0-5])(\/([0-9]|1[0-9]|2[0-9]|3[0-2]))$
    ConstraintDescription: CIDR block parameter must be in the form x.x.x.x/0-32.
    Default: 0.0.0.0/0
    Description: CIDR block to allow SSH access to the bootstrap node.
    Type: String
  PublicSubnet:
    Description: The public subnet to launch the bootstrap node into.
    Type: AWS::EC2::Subnet::Id
  MasterSecurityGroupId:
    Description: The master security group ID for registering temporary rules.
    Type: AWS::EC2::SecurityGroup::Id
  VpcId:
    Description: The VPC-scoped resources will belong to this VPC.
    Type: AWS::EC2::VPC::Id
  BootstrapIgnitionLocation:
    Default: s3://my-s3-bucket/bootstrap.ign
    Description: Ignition config file location.
    Type: String
  AutoRegisterELB:
    Default: "yes"
    AllowedValues:
    - "yes"
    - "no"
    Description: Do you want to invoke NLB registration, which requires a Lambda ARN parameter?
    Type: String
  RegisterNlbIpTargetsLambdaArn:
    Description: ARN for NLB IP target registration lambda.
    Type: String
  ExternalApiTargetGroupArn:
    Description: ARN for external API load balancer target group.
    Type: String
  InternalApiTargetGroupArn:
    Description: ARN for internal API load balancer target group.
    Type: String
  InternalServiceTargetGroupArn:
    Description: ARN for internal service load balancer target group.
    Type: String
  BootstrapInstanceType:
    Description: Instance type for the bootstrap EC2 instance
    Default: "i3.large"
    Type: String

Metadata:
  AWS::CloudFormation::Interface:
    ParameterGroups:
    - Label:
        default: "Cluster Information"
```

```
      Parameters:
      - InfrastructureName
    - Label:
        default: "Host Information"
      Parameters:
      - RhcosAmi
      - BootstrapIgnitionLocation
      - MasterSecurityGroupId
    - Label:
        default: "Network Configuration"
      Parameters:
      - VpcId
      - AllowedBootstrapSshCidr
      - PublicSubnet
    - Label:
        default: "Load Balancer Automation"
      Parameters:
      - AutoRegisterELB
      - RegisterNlbIpTargetsLambdaArn
      - ExternalApiTargetGroupArn
      - InternalApiTargetGroupArn
      - InternalServiceTargetGroupArn
    ParameterLabels:
      InfrastructureName:
        default: "Infrastructure Name"
      VpcId:
        default: "VPC ID"
      AllowedBootstrapSshCidr:
        default: "Allowed SSH Source"
      PublicSubnet:
        default: "Public Subnet"
      RhcosAmi:
        default: "Red Hat Enterprise Linux CoreOS AMI ID"
      BootstrapIgnitionLocation:
        default: "Bootstrap Ignition Source"
      MasterSecurityGroupId:
        default: "Master Security Group ID"
      AutoRegisterELB:
        default: "Use Provided ELB Automation"

Conditions:
  DoRegistration: !Equals ["yes", !Ref AutoRegisterELB]

Resources:
  BootstrapIamRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: "2012-10-17"
        Statement:
        - Effect: "Allow"
          Principal:
            Service:
            - "ec2.amazonaws.com"
          Action:
          - "sts:AssumeRole"
```

```
      Path: "/"
      Policies:
      - PolicyName: !Join ["-", [!Ref InfrastructureName, "bootstrap", "policy"]]
        PolicyDocument:
          Version: "2012-10-17"
          Statement:
          - Effect: "Allow"
            Action: "ec2:Describe*"
            Resource: "*"
          - Effect: "Allow"
            Action: "ec2:AttachVolume"
            Resource: "*"
          - Effect: "Allow"
            Action: "ec2:DetachVolume"
            Resource: "*"
          - Effect: "Allow"
            Action: "s3:GetObject"
            Resource: "*"

    BootstrapInstanceProfile:
      Type: "AWS::IAM::InstanceProfile"
      Properties:
        Path: "/"
        Roles:
        - Ref: "BootstrapIamRole"

    BootstrapSecurityGroup:
      Type: AWS::EC2::SecurityGroup
      Properties:
        GroupDescription: Cluster Bootstrap Security Group
        SecurityGroupIngress:
        - IpProtocol: tcp
          FromPort: 22
          ToPort: 22
          CidrIp: !Ref AllowedBootstrapSshCidr
        - IpProtocol: tcp
          ToPort: 19531
          FromPort: 19531
          CidrIp: 0.0.0.0/0
        VpcId: !Ref VpcId

    BootstrapInstance:
      Type: AWS::EC2::Instance
      Properties:
        ImageId: !Ref RhcosAmi
        IamInstanceProfile: !Ref BootstrapInstanceProfile
        InstanceType: !Ref BootstrapInstanceType
        NetworkInterfaces:
        - AssociatePublicIpAddress: "true"
          DeviceIndex: "0"
          GroupSet:
          - !Ref "BootstrapSecurityGroup"
          - !Ref "MasterSecurityGroupId"
          SubnetId: !Ref "PublicSubnet"
        UserData:
          Fn::Base64: !Sub
```

```
        - '{"ignition":{"config":{"replace":{"source":"${S3Loc}"}},"version":"3.1.0"}}'
        - {
          S3Loc: !Ref BootstrapIgnitionLocation
        }

  RegisterBootstrapApiTarget:
    Condition: DoRegistration
    Type: Custom::NLBRegister
    Properties:
      ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
      TargetArn: !Ref ExternalApiTargetGroupArn
      TargetIp: !GetAtt BootstrapInstance.PrivateIp

  RegisterBootstrapInternalApiTarget:
    Condition: DoRegistration
    Type: Custom::NLBRegister
    Properties:
      ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
      TargetArn: !Ref InternalApiTargetGroupArn
      TargetIp: !GetAtt BootstrapInstance.PrivateIp

  RegisterBootstrapInternalServiceTarget:
    Condition: DoRegistration
    Type: Custom::NLBRegister
    Properties:
      ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
      TargetArn: !Ref InternalServiceTargetGroupArn
      TargetIp: !GetAtt BootstrapInstance.PrivateIp

Outputs:
  BootstrapInstanceId:
    Description: Bootstrap Instance ID.
    Value: !Ref BootstrapInstance

  BootstrapPublicIp:
    Description: The bootstrap node public IP address.
    Value: !GetAtt BootstrapInstance.PublicIp

  BootstrapPrivateIp:
    Description: The bootstrap node private IP address.
    Value: !GetAtt BootstrapInstance.PrivateIp
```

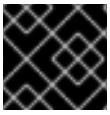
其他资源

- 如需有关 AWS 区的 Red Hat Enterprise Linux CoreOS（RHCOS）AMI 的详细信息，请参阅 AWS 基础架构的 RHCOS AMI。

6.4.4.11. 在 AWS 中创建 control plane 机器

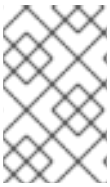
您必须在集群要使用的 Amazon Web Services（AWS）中创建 control plane 机器。

您可以使用提供的 CloudFormation 模板和自定义参数文件，创建代表 control plane 节点的 AWS 资源堆栈。



重要

CloudFormation 模板会创建一个堆栈，它代表三个 control plane 节点。



注意

如果不使用提供的 CloudFormation 模板来创建 control plane 节点，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 已配置了一个 AWS 帐户。
- 您可以通过运行 **aws configure**，将 AWS 密钥和区域添加到本地 AWS 配置集中。
- 已为集群生成 Ignition 配置文件。
- 您在 AWS 中创建并配置了 VPC 及相关子网。
- 您在 AWS 中创建并配置了 DNS、负载均衡器和监听程序。
- 您在 AWS 中创建了集群所需的安全组和角色。
- 已创建 bootstrap 机器。

流程

1. 创建一个 JSON 文件，其包含模板所需的参数值：

```
[
  {
    "ParameterKey": "InfrastructureName", 1
    "ParameterValue": "mycluster-<random_string>" 2
  },
  {
    "ParameterKey": "RhcosAmi", 3
    "ParameterValue": "ami-<random_string>" 4
  },
  {
    "ParameterKey": "AutoRegisterDNS", 5
    "ParameterValue": "yes" 6
  },
  {
    "ParameterKey": "PrivateHostedZoneId", 7
    "ParameterValue": "<random_string>" 8
  },
  {
    "ParameterKey": "PrivateHostedZoneName", 9
    "ParameterValue": "mycluster.example.com" 10
  },
  {
    "ParameterKey": "Master0Subnet", 11
    "ParameterValue": "subnet-<random_string>" 12
```

```
  },
  {
    "ParameterKey": "Master1Subnet", 13
    "ParameterValue": "subnet-<random_string>" 14
  },
  {
    "ParameterKey": "Master2Subnet", 15
    "ParameterValue": "subnet-<random_string>" 16
  },
  {
    "ParameterKey": "MasterSecurityGroupId", 17
    "ParameterValue": "sg-<random_string>" 18
  },
  {
    "ParameterKey": "IgnitionLocation", 19
    "ParameterValue": "https://api-int.<cluster_name>.<domain_name>:22623/config/master"
20
  },
  {
    "ParameterKey": "CertificateAuthorities", 21
    "ParameterValue": "data:text/plain;charset=utf-8;base64,ABC...xYz==" 22
  },
  {
    "ParameterKey": "MasterInstanceProfileName", 23
    "ParameterValue": "<roles_stack>-MasterInstanceProfile-<random_string>" 24
  },
  {
    "ParameterKey": "MasterInstanceType", 25
    "ParameterValue": "" 26
  },
  {
    "ParameterKey": "AutoRegisterELB", 27
    "ParameterValue": "yes" 28
  },
  {
    "ParameterKey": "RegisterNlbIpTargetsLambdaArn", 29
    "ParameterValue": "arn:aws:lambda:<aws_region>:<account_number>:function:
<dns_stack_name>-RegisterNlbIpTargets-<random_string>" 30
  },
  {
    "ParameterKey": "ExternalApiTargetGroupArn", 31
    "ParameterValue": "arn:aws:elasticloadbalancing:<aws_region>:
<account_number>:targetgroup/<dns_stack_name>-Exter-<random_string>" 32
  },
  {
    "ParameterKey": "InternalApiTargetGroupArn", 33
    "ParameterValue": "arn:aws:elasticloadbalancing:<aws_region>:
<account_number>:targetgroup/<dns_stack_name>-Inter-<random_string>" 34
  },
  {
    "ParameterKey": "InternalServiceTargetGroupArn", 35
    "ParameterValue": "arn:aws:elasticloadbalancing:<aws_region>:
```

```
<account_number>:targetgroup/<dns_stack_name>-Inter-<random_string>" 36
  }
]
```

1. 您的 Ignition 配置文件中为集群编码的集群基础架构名称。
2. 指定从 Ignition 配置文件元数据中提取的基础架构名称，其格式为 **<cluster-name>-<random-string>**。
3. 根据您选择的架构，当前用于 control plane 机器的 Red Hat Enterprise Linux CoreOS (RHCOS) AMI。
4. 指定 **AWS::EC2::Image::Id** 值。
5. 是否要执行 DNS etcd 注册。
6. 指定 **yes** 或 **no**。如果指定 **yes**，您必须提供托管区信息。
7. 用来注册 etcd 目标的 Route 53 专用区 ID。
8. 指定 DNS 和负载均衡的 CloudFormation 模板输出的 **PrivateHostedZoneId** 值。
9. 用来注册目标的 Route 53 区。
10. 指定 **<cluster_name>.<domain_name>**，其中 **<domain_name>** 是您为集群生成 **install-config.yaml** 文件时所用的 Route 53 基域。请勿包含 AWS 控制台中显示的结尾句点 (.)。
11. 13. 15. 在其中启动 control plane 机器的子网，最好是专用子网。
12. 14. 16. 从 DNS 和负载均衡的 CloudFormation 模板输出的 **PrivateSubnets** 值指定子网。
17. 与 control plane 节点关联的 master 安全组 ID。
18. 指定安全组和角色的 CloudFormation 模板输出的 **MasterSecurityGroupId** 值。
19. 从中获取 control plane Ignition 配置文件的位置。
20. 指定生成的 Ignition 配置文件的位置，**https://api-int.<cluster_name>.<domain_name>:22623/config/master**。
21. 要使用的 base64 编码证书颁发机构字符串。
22. 指定安装目录中 **master.ign** 文件中的值。这个值是一个长字符串，格式为 **data:text/plain;charset=utf-8;base64,ABC...xYz==**。
23. 与 control plane 节点关联的 IAM 配置集。
24. 指定安全组和角色的 CloudFormation 模板输出的 **MasterInstanceProfile** 参数值。
25. 根据您选择的架构，用于 control plane 机器的 AWS 实例类型。
26. 实例类型值与 control plane 机器的最低资源要求对应。例如，**m6i.xlarge** 是 AMD64 的类型，**m6g.xlarge** 是 ARM64 的类型。
27. 是否要注册网络负载均衡器 (NLB)。
28. 指定 **yes** 或 **no**。如果指定 **yes**，您必须提供一个 Lambda Amazon Resource Name (ARN) 值。

29 NLB IP 目标注册 lambda 组的 ARN。

30 指定 DNS 和负载均衡的 CloudFormation 模板输出的 **RegisterNlbIpTargetsLambda** 值。如果将集群部署到 AWS GovCloud 区域，请使用 **arn:aws-us-gov**。

31 外部 API 负载均衡器目标组的 ARN。

32 指定 DNS 和负载均衡的 CloudFormation 模板输出的 **ExternalApiTargetGroupArn** 值。如果将集群部署到 AWS GovCloud 区域，请使用 **arn:aws-us-gov**。

33 内部 API 负载均衡器目标组群的 ARN。

34 指定 DNS 和负载均衡的 CloudFormation 模板输出的 **InternalApiTargetGroupArn** 值。如果将集群部署到 AWS GovCloud 区域，请使用 **arn:aws-us-gov**。

35 内部服务负载均衡器目标组群的 ARN。

36 指定 DNS 和负载均衡的 CloudFormation 模板输出的 **InternalServiceTargetGroupArn** 值。如果将集群部署到 AWS GovCloud 区域，请使用 **arn:aws-us-gov**。

2. 复制**control plane 机器的 CloudFormation 模板**一节中的模板，并将它以 YAML 文件形式保存到计算机上。此模板描述了集群所需的 control plane 机器。

3. 如果您将 **m5** 实例类型指定为 **MasterInstanceType** 的值，请将该实例类型添加到 CloudFormation 模板中的 **MasterInstanceType.AllowedValues** 参数。

4. 启动 CloudFormation 模板，以创建代表 control plane 节点的 AWS 资源堆栈：

重要

您必须在一行内输入命令。

```
$ aws cloudformation create-stack --stack-name <name> 1
    --template-body file://<template>.yaml 2
    --parameters file://<parameters>.json 3
```

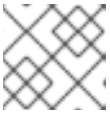
1 **<name>** 是 CloudFormation 堆栈的名称，如 **cluster-control-plane**。如果您删除集群，则需要此堆栈的名称。

2 **<template>** 是您保存的 CloudFormation 模板 YAML 文件的相对路径和名称。

3 **<parameters>** 是 CloudFormation 参数 JSON 文件的相对路径和名称。

输出示例

```
arn:aws:cloudformation:us-east-1:269333783861:stack/cluster-control-plane/21c7e2b0-2ee2-11eb-c6f6-0aa34627df4b
```



注意

CloudFormation 模板会创建一个堆栈，它代表三个 control plane 节点。

5. 确认模板组件已存在：

```
$ aws cloudformation describe-stacks --stack-name <name>
```

6.4.4.11.1. control plane 机器的 CloudFormation 模板

您可以使用以下 CloudFormation 模板来部署 OpenShift Container Platform 集群所需的 control plane 机器。

例 6.82. control plane 机器的 CloudFormation 模板

```
AWSTemplateFormatVersion: 2010-09-09
Description: Template for OpenShift Cluster Node Launch (EC2 master instances)

Parameters:
  InfrastructureName:
    AllowedPattern: ^([a-zA-Z][a-zA-Z0-9\-]{0,26})$
    MaxLength: 27
    MinLength: 1
    ConstraintDescription: Infrastructure name must be alphanumeric, start with a letter, and have a
maximum of 27 characters.
    Description: A short, unique cluster ID used to tag nodes for the kubelet cloud provider.
    Type: String
  RhcosAmi:
    Description: Current Red Hat Enterprise Linux CoreOS AMI to use for bootstrap.
    Type: AWS::EC2::Image::Id
  AutoRegisterDNS:
    Default: ""
    Description: unused
    Type: String
  PrivateHostedZoneId:
    Default: ""
    Description: unused
    Type: String
  PrivateHostedZoneName:
    Default: ""
    Description: unused
    Type: String
  Master0Subnet:
    Description: The subnets, recommend private, to launch the master nodes into.
    Type: AWS::EC2::Subnet::Id
  Master1Subnet:
    Description: The subnets, recommend private, to launch the master nodes into.
    Type: AWS::EC2::Subnet::Id
  Master2Subnet:
    Description: The subnets, recommend private, to launch the master nodes into.
    Type: AWS::EC2::Subnet::Id
  MasterSecurityGroupId:
    Description: The master security group ID to associate with master nodes.
    Type: AWS::EC2::SecurityGroup::Id
  IgnitionLocation:
    Default: https://api-int.$CLUSTER_NAME.$DOMAIN:22623/config/master
    Description: Ignition config file location.
    Type: String
  CertificateAuthorities:
```

```
    Default: data:text/plain;charset=utf-8;base64,ABC...xYz==
    Description: Base64 encoded certificate authority string to use.
    Type: String
  MasterInstanceProfileName:
    Description: IAM profile to associate with master nodes.
    Type: String
  MasterInstanceType:
    Default: m5.xlarge
    Type: String

  AutoRegisterELB:
    Default: "yes"
    AllowedValues:
    - "yes"
    - "no"
    Description: Do you want to invoke NLB registration, which requires a Lambda ARN parameter?
    Type: String
  RegisterNlbIpTargetsLambdaArn:
    Description: ARN for NLB IP target registration lambda. Supply the value from the cluster
infrastructure or select "no" for AutoRegisterELB.
    Type: String
  ExternalApiTargetGroupArn:
    Description: ARN for external API load balancer target group. Supply the value from the cluster
infrastructure or select "no" for AutoRegisterELB.
    Type: String
  InternalApiTargetGroupArn:
    Description: ARN for internal API load balancer target group. Supply the value from the cluster
infrastructure or select "no" for AutoRegisterELB.
    Type: String
  InternalServiceTargetGroupArn:
    Description: ARN for internal service load balancer target group. Supply the value from the
cluster infrastructure or select "no" for AutoRegisterELB.
    Type: String

Metadata:
  AWS::CloudFormation::Interface:
    ParameterGroups:
    - Label:
        default: "Cluster Information"
      Parameters:
      - InfrastructureName
    - Label:
        default: "Host Information"
      Parameters:
      - MasterInstanceType
      - RhcosAmi
      - IgnitionLocation
      - CertificateAuthorities
      - MasterSecurityGroupId
      - MasterInstanceProfileName
    - Label:
        default: "Network Configuration"
      Parameters:
      - VpcId
      - AllowedBootstrapSshCidr
      - Master0Subnet
```

```
      - Master1Subnet
      - Master2Subnet
    - Label:
        default: "Load Balancer Automation"
      Parameters:
      - AutoRegisterELB
      - RegisterNlbIpTargetsLambdaArn
      - ExternalApiTargetGroupArn
      - InternalApiTargetGroupArn
      - InternalServiceTargetGroupArn
    ParameterLabels:
      InfrastructureName:
        default: "Infrastructure Name"
      VpcId:
        default: "VPC ID"
      Master0Subnet:
        default: "Master-0 Subnet"
      Master1Subnet:
        default: "Master-1 Subnet"
      Master2Subnet:
        default: "Master-2 Subnet"
      MasterInstanceType:
        default: "Master Instance Type"
      MasterInstanceProfileName:
        default: "Master Instance Profile Name"
      RhcosAmi:
        default: "Red Hat Enterprise Linux CoreOS AMI ID"
      BootstrapIgnitionLocation:
        default: "Master Ignition Source"
      CertificateAuthorities:
        default: "Ignition CA String"
      MasterSecurityGroupId:
        default: "Master Security Group ID"
      AutoRegisterELB:
        default: "Use Provided ELB Automation"

Conditions:
  DoRegistration: !Equals ["yes", !Ref AutoRegisterELB]

Resources:
  Master0:
    Type: AWS::EC2::Instance
    Properties:
      ImageId: !Ref RhcosAmi
      BlockDeviceMappings:
      - DeviceName: /dev/xvda
        Ebs:
          VolumeSize: "120"
          VolumeType: "gp2"
      IamInstanceProfile: !Ref MasterInstanceProfileName
      InstanceType: !Ref MasterInstanceType
      NetworkInterfaces:
      - AssociatePublicIpAddress: "false"
        DeviceIndex: "0"
        GroupSet:
        - !Ref "MasterSecurityGroupId"
```

```
      SubnetId: !Ref "Master0Subnet"
    UserData:
      Fn::Base64: !Sub
      - '{"ignition":{"config":{"merge":[{"source":"${SOURCE}"}]},"security":{"tls":
{"certificateAuthorities":[{"source":"${CA_BUNDLE}"}]}},"version":"3.1.0"}}'
      - {
        SOURCE: !Ref IgnitionLocation,
        CA_BUNDLE: !Ref CertificateAuthorities,
      }
    Tags:
    - Key: !Join ["", ["kubernetes.io/cluster/", !Ref InfrastructureName]]
      Value: "shared"

RegisterMaster0:
  Condition: DoRegistration
  Type: Custom::NLBRegister
  Properties:
    ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
    TargetArn: !Ref ExternalApiTargetGroupArn
    TargetIp: !GetAtt Master0.PrivateIp

RegisterMaster0InternalApiTarget:
  Condition: DoRegistration
  Type: Custom::NLBRegister
  Properties:
    ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
    TargetArn: !Ref InternalApiTargetGroupArn
    TargetIp: !GetAtt Master0.PrivateIp

RegisterMaster0InternalServiceTarget:
  Condition: DoRegistration
  Type: Custom::NLBRegister
  Properties:
    ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
    TargetArn: !Ref InternalServiceTargetGroupArn
    TargetIp: !GetAtt Master0.PrivateIp

Master1:
  Type: AWS::EC2::Instance
  Properties:
    ImageId: !Ref RhcosAmi
    BlockDeviceMappings:
    - DeviceName: /dev/xvda
      Ebs:
        VolumeSize: "120"
        VolumeType: "gp2"
    IamInstanceProfile: !Ref MasterInstanceProfileName
    InstanceType: !Ref MasterInstanceType
    NetworkInterfaces:
    - AssociatePublicIpAddress: "false"
      DeviceIndex: "0"
      GroupSet:
      - !Ref "MasterSecurityGroupId"
      SubnetId: !Ref "Master1Subnet"
    UserData:
      Fn::Base64: !Sub
```

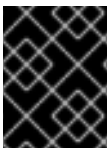
```
      - '{"ignition":{"config":{"merge":[{"source":"${SOURCE}"}]},"security":{"tls":
{"certificateAuthorities":[{"source":"${CA_BUNDLE}"}]}},"version":"3.1.0"}}'
      - {
        SOURCE: !Ref IgnitionLocation,
        CA_BUNDLE: !Ref CertificateAuthorities,
      }
    Tags:
    - Key: !Join ["", ["kubernetes.io/cluster/", !Ref InfrastructureName]]
      Value: "shared"

  RegisterMaster1:
    Condition: DoRegistration
    Type: Custom::NLBRegister
    Properties:
      ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
      TargetArn: !Ref ExternalApiTargetGroupArn
      TargetIp: !GetAtt Master1.PrivateIp

  RegisterMaster1InternalApiTarget:
    Condition: DoRegistration
    Type: Custom::NLBRegister
    Properties:
      ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
      TargetArn: !Ref InternalApiTargetGroupArn
      TargetIp: !GetAtt Master1.PrivateIp

  RegisterMaster1InternalServiceTarget:
    Condition: DoRegistration
    Type: Custom::NLBRegister
    Properties:
      ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
      TargetArn: !Ref InternalServiceTargetGroupArn
      TargetIp: !GetAtt Master1.PrivateIp

  Master2:
    Type: AWS::EC2::Instance
    Properties:
      ImageId: !Ref RhcosAmi
      BlockDeviceMappings:
      - DeviceName: /dev/xvda
        Ebs:
          VolumeSize: "120"
          VolumeType: "gp2"
      IamInstanceProfile: !Ref MasterInstanceProfileName
      InstanceType: !Ref MasterInstanceType
      NetworkInterfaces:
      - AssociatePublicIpAddress: "false"
        DeviceIndex: "0"
        GroupSet:
        - !Ref "MasterSecurityGroupId"
        SubnetId: !Ref "Master2Subnet"
      UserData:
        Fn::Base64: !Sub
        - '{"ignition":{"config":{"merge":[{"source":"${SOURCE}"}]},"security":{"tls":
{"certificateAuthorities":[{"source":"${CA_BUNDLE}"}]}},"version":"3.1.0"}}'
        - {
```

```
          SOURCE: !Ref IgnitionLocation,
          CA_BUNDLE: !Ref CertificateAuthorities,
        }
      Tags:
      - Key: !Join ["", ["kubernetes.io/cluster/", !Ref InfrastructureName]]
        Value: "shared"

  RegisterMaster2:
    Condition: DoRegistration
    Type: Custom::NLBRegister
    Properties:
      ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
      TargetArn: !Ref ExternalApiTargetGroupArn
      TargetIp: !GetAtt Master2.PrivateIp

  RegisterMaster2InternalApiTarget:
    Condition: DoRegistration
    Type: Custom::NLBRegister
    Properties:
      ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
      TargetArn: !Ref InternalApiTargetGroupArn
      TargetIp: !GetAtt Master2.PrivateIp

  RegisterMaster2InternalServiceTarget:
    Condition: DoRegistration
    Type: Custom::NLBRegister
    Properties:
      ServiceToken: !Ref RegisterNlbIpTargetsLambdaArn
      TargetArn: !Ref InternalServiceTargetGroupArn
      TargetIp: !GetAtt Master2.PrivateIp

Outputs:
  PrivateIPs:
    Description: The control-plane node private IP addresses.
    Value:
      !Join [
        ",",
        [!GetAtt Master0.PrivateIp, !GetAtt Master1.PrivateIp, !GetAtt Master2.PrivateIp]
      ]
```

6.4.4.12. 在 AWS 中创建 worker 节点

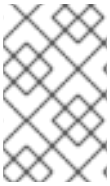
您可以在 Amazon Web Services (AWS) 中创建 worker 节点，供集群使用。

您可以使用提供的 CloudFormation 模板和自定义参数文件创建代表 worker 节点的 AWS 资源堆栈。



重要

CloudFormation 模板会创建一个堆栈，它代表一个 worker 节点。您必须为每个 worker 节点创建一个堆栈。



注意

如果不使用提供的 CloudFormation 模板来创建 worker 节点，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 已配置了一个 AWS 帐户。
- 您可以通过运行 **aws configure**，将 AWS 密钥和区域添加到本地 AWS 配置集中。
- 已为集群生成 Ignition 配置文件。
- 您在 AWS 中创建并配置了 VPC 及相关子网。
- 您在 AWS 中创建并配置了 DNS、负载均衡器和监听程序。
- 您在 AWS 中创建了集群所需的安全组和角色。
- 已创建 bootstrap 机器。
- 已创建 control plane 机器。

流程

1. 创建一个 JSON 文件，其包含 CloudFormation 模板需要的参数值：

```
[
  {
    "ParameterKey": "InfrastructureName", 1
    "ParameterValue": "mycluster-<random_string>" 2
  },
  {
    "ParameterKey": "RhcosAmi", 3
    "ParameterValue": "ami-<random_string>" 4
  },
  {
    "ParameterKey": "Subnet", 5
    "ParameterValue": "subnet-<random_string>" 6
  },
  {
    "ParameterKey": "WorkerSecurityGroupId", 7
    "ParameterValue": "sg-<random_string>" 8
  },
  {
    "ParameterKey": "IgnitionLocation", 9
    "ParameterValue": "https://api-int.<cluster_name>.<domain_name>:22623/config/worker"
10
  },
  {
    "ParameterKey": "CertificateAuthorities", 11
    "ParameterValue": "" 12
  },
```

```
  {
    "ParameterKey": "WorkerInstanceProfileName", 13
    "ParameterValue": "" 14
  },
  {
    "ParameterKey": "WorkerInstanceType", 15
    "ParameterValue": "" 16
  }
]
```

1. 您的 Ignition 配置文件中为集群编码的集群基础架构名称。
2. 指定从 Ignition 配置文件元数据中提取的基础架构名称，其格式为 **<cluster-name>-<random-string>**。
3. 根据您选择的架构，当前 Red Hat Enterprise Linux CoreOS (RHCOS) AMI 用于 worker 节点。
4. 指定 **AWS::EC2::Image::Id** 值。
5. 在其中启动 worker 节点的子网，最好是专用子网。
6. 从 DNS 和负载均衡的 CloudFormation 模板输出的 **PrivateSubnets** 值指定子网。
7. 与 worker 节点关联的 worker 安全组 ID。
8. 指定安全组和角色的 CloudFormation 模板输出的 **WorkerSecurityGroupId** 值。
9. 从中获取 bootstrap Ignition 配置文件的位置。
10. 指定生成的 Ignition 配置的位置，**https://api-int.<cluster_name>.<domain_name>:22623/config/worker**。
11. 要使用的 Base64 编码证书颁发机构字符串。
12. 指定安装目录下 **worker.ign** 文件中的值。这个值是一个长字符串，格式为 **data:text/plain;charset=utf-8;base64,ABC...xYz==**。
13. 与 worker 节点关联的 IAM 配置集。
14. 指定安全组和角色的 CloudFormation 模板输出的 **WorkerInstanceProfile** 参数值。
15. 根据您选择的架构，用于计算机器的 AWS 实例类型。
16. 实例类型值对应于计算机器的最低资源要求。例如，**m6i.large** 是 AMD64 的类型，**m6g.large** 是 ARM64 的类型。

2. 复制 **worker 机器的 CloudFormation 模板**一节中的模板，并将它以 YAML 文件形式保存到计算机上。此模板描述了集群所需的网络对象和负载均衡器。
3. 可选：如果将 **m5** 实例类型指定为 **WorkerInstanceType** 的值，请将该实例类型添加到 CloudFormation 模板中的 **WorkerInstanceType.AllowedValues** 参数。
4. 可选：如果您使用 AWS Marketplace 镜像部署，请使用从订阅获取的 AMI ID 更新 **Worker0.type.properties.ImageID** 参数。

5. 使用 CloudFormation 模板创建代表 worker 节点的 AWS 资源堆栈：

重要

您必须在一行内输入命令。

```
$ aws cloudformation create-stack --stack-name <name> 1
    --template-body file://<template>.yaml \ 2
    --parameters file://<parameters>.json 3
```

1. **<name>** 是 CloudFormation 堆栈的名称，如 **cluster-worker-1**。如果您删除集群，则需要此堆栈的名称。
2. **<template>** 是您保存的 CloudFormation 模板 YAML 文件的相对路径和名称。
3. **<parameters>** 是 CloudFormation 参数 JSON 文件的相对路径和名称。

输出示例

```
arn:aws:cloudformation:us-east-1:269333783861:stack/cluster-worker-1/729ee301-1c2a-11eb-348f-sd9888c65b59
```

注意

CloudFormation 模板会创建一个堆栈，它代表一个 worker 节点。

6. 确认模板组件已存在：

```
$ aws cloudformation describe-stacks --stack-name <name>
```

7. 继续创建 worker 堆栈，直到为集群创建了充足的 worker 机器。您可以通过引用同一模板和参数文件并指定不同的堆栈名称来创建额外的 worker 堆栈。

重要

您必须至少创建两台 worker 机器，因此您必须创建至少两个使用此 CloudFormation 模板的堆栈。

6.4.4.12.1. worker 机器的 CloudFormation 模板

您可以使用以下 CloudFormation 模板来部署 OpenShift Container Platform 集群所需的 worker 机器。

例 6.83. worker 机器的 CloudFormation 模板

```
AWSTemplateFormatVersion: 2010-09-09
Description: Template for OpenShift Cluster Node Launch (EC2 worker instance)

Parameters:
  InfrastructureName:
    AllowedPattern: ^([a-zA-Z][a-zA-Z0-9\-]{0,26})$
    MaxLength: 27
```

```
    MinLength: 1
    ConstraintDescription: Infrastructure name must be alphanumeric, start with a letter, and have a
maximum of 27 characters.
    Description: A short, unique cluster ID used to tag nodes for the kubelet cloud provider.
    Type: String
  RhcosAmi:
    Description: Current Red Hat Enterprise Linux CoreOS AMI to use for bootstrap.
    Type: AWS::EC2::Image::Id
  Subnet:
    Description: The subnets, recommend private, to launch the worker nodes into.
    Type: AWS::EC2::Subnet::Id
  WorkerSecurityGroupId:
    Description: The worker security group ID to associate with worker nodes.
    Type: AWS::EC2::SecurityGroup::Id
  IgnitionLocation:
    Default: https://api-int.$CLUSTER_NAME.$DOMAIN:22623/config/worker
    Description: Ignition config file location.
    Type: String
  CertificateAuthorities:
    Default: data:text/plain;charset=utf-8;base64,ABC...xYz==
    Description: Base64 encoded certificate authority string to use.
    Type: String
  WorkerInstanceProfileName:
    Description: IAM profile to associate with worker nodes.
    Type: String
  WorkerInstanceType:
    Default: m5.large
    Type: String

Metadata:
  AWS::CloudFormation::Interface:
    ParameterGroups:
    - Label:
        default: "Cluster Information"
      Parameters:
      - InfrastructureName
    - Label:
        default: "Host Information"
      Parameters:
      - WorkerInstanceType
      - RhcosAmi
      - IgnitionLocation
      - CertificateAuthorities
      - WorkerSecurityGroupId
      - WorkerInstanceProfileName
    - Label:
        default: "Network Configuration"
      Parameters:
      - Subnet
    ParameterLabels:
      Subnet:
        default: "Subnet"
      InfrastructureName:
        default: "Infrastructure Name"
      WorkerInstanceType:
        default: "Worker Instance Type"
```

```
      WorkerInstanceProfileName:
        default: "Worker Instance Profile Name"
      RhcosAmi:
        default: "Red Hat Enterprise Linux CoreOS AMI ID"
      IgnitionLocation:
        default: "Worker Ignition Source"
      CertificateAuthorities:
        default: "Ignition CA String"
      WorkerSecurityGroupId:
        default: "Worker Security Group ID"

Resources:
  Worker0:
    Type: AWS::EC2::Instance
    Properties:
      ImageId: !Ref RhcosAmi
      BlockDeviceMappings:
      - DeviceName: /dev/xvda
        Ebs:
          VolumeSize: "120"
          VolumeType: "gp2"
      IamInstanceProfile: !Ref WorkerInstanceProfileName
      InstanceType: !Ref WorkerInstanceType
      NetworkInterfaces:
      - AssociatePublicIpAddress: "false"
        DeviceIndex: "0"
        GroupSet:
        - !Ref "WorkerSecurityGroupId"
        SubnetId: !Ref "Subnet"
      UserData:
        Fn::Base64: !Sub
        - '{"ignition":{"config":{"merge":[{"source":"${SOURCE}"}]},"security":{"tls":
{"certificateAuthorities":[{"source":"${CA_BUNDLE}"}]}},"version":"3.1.0"}}'
        - {
          SOURCE: !Ref IgnitionLocation,
          CA_BUNDLE: !Ref CertificateAuthorities,
        }
      Tags:
      - Key: !Join ["", ["kubernetes.io/cluster/", !Ref InfrastructureName]]
        Value: "shared"

Outputs:
  PrivateIP:
    Description: The compute node private IP address.
    Value: !GetAtt Worker0.PrivateIp
```

6.4.4.13. 使用用户置备的基础架构在 AWS 上初始化 bootstrap 序列

在 Amazon Web Services（AWS）中创建所有所需的基础架构后，您可以启动初始化 OpenShift Container Platform control plane 的 bootstrap 序列。

先决条件

- 已配置了一个 AWS 帐户。

- 您可以通过运行 **aws configure**，将 AWS 密钥和区域添加到本地 AWS 配置集中。
- 已为集群生成 Ignition 配置文件。
- 您在 AWS 中创建并配置了 VPC 及相关子网。
- 您在 AWS 中创建并配置了 DNS、负载均衡器和监听程序。
- 您在 AWS 中创建了集群所需的安全组和角色。
- 已创建 bootstrap 机器。
- 已创建 control plane 机器。
- 已创建 worker 节点。

流程

1. 更改为包含安装程序的目录，并启动初始化 OpenShift Container Platform control plane 的 bootstrap 过程：

```
$ ./openshift-install wait-for bootstrap-complete --dir <installation_directory> \ 1
    --log-level=info 2
```

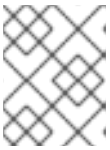
1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不是 **info**。

输出示例

```
INFO Waiting up to 20m0s for the Kubernetes API at
https://api.mycluster.example.com:6443...
INFO API v1.29.4 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
INFO Time elapsed: 1s
```

如果命令退出时没有 **FATAL** 警告，则 OpenShift Container Platform control plane 已被初始化。



注意

在 control plane 初始化后，它会设置计算节点，并以 Operator 的形式安装其他服务。

其他资源

- 如需了解在 OpenShift Container Platform 安装过程中监控安装、bootstrap 和 control plane 日志的详细信息，请参阅监控安装进度。
- 如需有关对 bootstrap 过程进行故障排除的信息，请参阅收集 bootstrap 节点诊断数据。

6.4.4.14. 批准机器的证书签名请求

当您将机器添加到集群时，会为您添加的每台机器生成两个待处理证书签名请求(CSR)。您必须确认这些 CSR 已获得批准，或根据需要自行批准。必须首先批准客户端请求，然后批准服务器请求。

先决条件

- 您已将机器添加到集群中。

流程

1. 确认集群可以识别这些机器：

```
$ oc get nodes
```

输出示例

```
NAME      STATUS    ROLES   AGE  VERSION
master-0  Ready     master  63m  v1.29.4
master-1  Ready     master  63m  v1.29.4
master-2  Ready     master  64m  v1.29.4
```

输出中列出了您创建的所有机器。

注意

在有些 CSR 被批准前，前面的输出可能不包括计算节点（也称为 worker 节点）。

2. 检查待处理的 CSR，并确保添加到集群中的每台机器都有 **Pending** 或 **Approved** 状态的客户端请求：

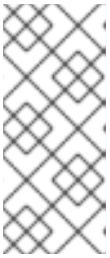
```
$ oc get csr
```

输出示例

```
NAME        AGE    REQUESTOR                                                   CONDITION
csr-8b2br   15m    system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper   Pending
csr-8vnps   15m    system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper   Pending
...
```

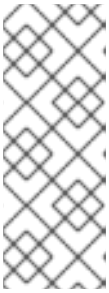
在本例中，两台机器加入集群。您可能会在列表中看到更多已批准的 CSR。

3. 如果 CSR 没有获得批准，在您添加的机器的所有待处理 CSR 都处于 **Pending 状态** 后，请批准集群机器的 CSR：



注意

由于 CSR 会自动轮转，因此请在将机器添加到集群后一小时内批准您的 CSR。如果没有在一小时内批准它们，证书将会轮转，每个节点会存在多个证书。您必须批准所有这些证书。批准客户端 CSR 后，Kubelet 为服务证书创建一个二级 CSR，这需要手动批准。然后，如果 Kubelet 请求具有相同参数的新证书，则后续提供证书续订请求由 **machine-approver** 自动批准。



注意

对于在未启用机器 API 的平台上运行的集群，如裸机和其他用户置备的基础架构，您必须实施一种方法来自动批准 kubelet 提供证书请求(CSR)。如果没有批准请求，则 **oc exec**、oc**rsh** 和 **oc logs** 命令将无法成功，因为 API 服务器连接到 kubelet 时需要服务证书。与 Kubelet 端点联系的任何操作都需要此证书批准。该方法必须监视新的 CSR，确认 CSR 由 system: **node** 或 **system:admin** 组中的 **node-bootstrapper** 服务帐户提交，并确认节点的身份。

- 要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```

注意

在有些 CSR 被批准前，一些 Operator 可能无法使用。

4. 现在，您的客户端请求已被批准，您必须查看添加到集群中的每台机器的服务器请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE     REQUESTOR                                                   CONDITION
csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

5. 如果剩余的 CSR 没有被批准，且处于 **Pending** 状态，请批准集群机器的 CSR：

- 要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

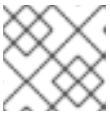
```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs oc adm certificate approve
```

6. 批准所有客户端和服务器 CSR 后，机器将 处于 **Ready** 状态。运行以下命令验证：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0 Ready    master  73m  v1.29.4
master-1 Ready    master  73m  v1.29.4
master-2 Ready    master  74m  v1.29.4
worker-0 Ready    worker  11m  v1.29.4
worker-1 Ready    worker  11m  v1.29.4
```



注意

批准服务器 CSR 后可能需要几分钟时间让机器过渡到 **Ready** 状态。

其他信息

- 如需有关 CSR 的更多信息，请参阅 证书签名请求。

6.4.4.15. 初始 Operator 配置

在 control plane 初始化后，您必须立即配置一些 Operator，以便它们都可用。

先决条件

- 您的 control plane 已初始化。

流程

1. 观察集群组件上线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION   AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.16.0    True        False         False      19m
baremetal                                  4.16.0    True        False         False      37m
cloud-credential                           4.16.0    True        False         False      40m
cluster-autoscaler                         4.16.0    True        False         False      37m
config-operator                            4.16.0    True        False         False      38m
console                                    4.16.0    True        False         False      26m
csi-snapshot-controller                    4.16.0    True        False         False      37m
dns                                        4.16.0    True        False         False      37m
etcd                                       4.16.0    True        False         False      36m
image-registry                             4.16.0    True        False         False      31m
ingress                                    4.16.0    True        False         False      30m
insights                                   4.16.0    True        False         False      31m
kube-apiserver                             4.16.0    True        False         False      26m
kube-controller-manager                    4.16.0    True        False         False      36m
kube-scheduler                             4.16.0    True        False         False      36m
kube-storage-version-migrator              4.16.0    True        False         False      37m
```

```
machine-api                                4.16.0    True        False         False      29m
machine-approver                           4.16.0    True        False         False      37m
machine-config                             4.16.0    True        False         False      36m
marketplace                                4.16.0    True        False         False      37m
monitoring                                 4.16.0    True        False         False      29m
network                                    4.16.0    True        False         False      38m
node-tuning                                4.16.0    True        False         False      37m
openshift-apiserver                        4.16.0    True        False         False      32m
openshift-controller-manager               4.16.0    True        False         False      30m
openshift-samples                          4.16.0    True        False         False      32m
operator-lifecycle-manager                 4.16.0    True        False         False      37m
operator-lifecycle-manager-catalog         4.16.0    True        False         False      37m
operator-lifecycle-manager-packageserver   4.16.0    True        False         False      32m
service-ca                                 4.16.0    True        False         False      38m
storage                                    4.16.0    True        False         False      37m
```

2. 配置不可用的 Operator。

6.4.4.15.1. 禁用默认的 OperatorHub 目录源

在 OpenShift Container Platform 安装过程中，默认为 OperatorHub 配置由红帽和社区项目提供的源内容的 operator 目录。在受限网络环境中，必须以集群管理员身份禁用默认目录。

流程

- 通过在 **OperatorHub** 对象中添加 **disableAllDefaultSources: true 来** 禁用默认目录的源：

```
$ oc patch OperatorHub cluster --type json \
    -p '[{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]'
```

提示

或者，您可以使用 Web 控制台管理目录源。在 **Administration → Cluster Settings → Configuration → OperatorHub** 页面中，点 **Sources** 选项卡，您可以在其中创建、更新、删除、禁用和启用单独的源。

6.4.4.15.2. 镜像 registry 存储配置

Amazon Web Services 提供默认存储，这意味着 Image Registry Operator 在安装后可用。但是，如果 Registry Operator 无法创建 S3 存储桶并自动配置存储，您需要手工配置 registry 存储。

示配置生产集群所需的持久性卷的说明。如果适用，显示有关将空目录配置为存储位置的说明，这仅适用于非生产集群。

另外还提供了在升级过程中使用 **Recreate** rollout 策略来允许镜像 registry 使用块存储类型的说明。

6.4.4.15.2.1. 为使用用户置备的基础架构的 AWS 配置 registry 存储

在安装过程中，使用您的云凭据就可以创建一个 Amazon S3 存储桶，Registry Operator 将会自动配置存储。

如果 Registry Operator 无法创建 S3 存储桶或自动配置存储，您可以按照以下流程创建 S3 存储桶并配置存储。

先决条件

- 在带有用户置备的基础架构的 AWS 上有一个集群。
- 对于 Amazon S3 存储，secret 应该包含以下两个键：
  - **REGISTRY_STORAGE_S3_ACCESSKEY**
  - **REGISTRY_STORAGE_S3_SECRETKEY**

流程

如果 Registry Operator 无法创建 S3 存储桶并自动配置存储，请进行以下操作。

1. 设置一个 Bucket Lifecycle Policy 用来终止已有一天之久的未完成的分段上传操作。
2. 在**configs.imageregistry.operator.openshift.io/cluster**中中输入存储配置：

```
$ oc edit configs.imageregistry.operator.openshift.io/cluster
```

配置示例

```
storage:
  s3:
    bucket: <bucket-name>
    region: <region-name>
```

警告

为了保护 AWS 中 registry 镜像的安全，阻止对 S3 存储桶的公共访问。

6.4.4.15.2.2. 在非生产集群中配置镜像 registry 存储

您必须为 Image Registry Operator 配置存储。对于非生产集群，您可以将镜像 registry 设置为空目录。如果您这样做，重启 registry 时会丢失所有镜像。

流程

- 将镜像 registry 存储设置为空目录：

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec":{"storage":{"emptyDir":{}}}}'
```

警告

仅为非生产集群配置这个选项。

如果在 Image Registry Operator 初始化其组件前运行这个命令，**oc patch** 命令会失败并显示以下错误：

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster" not found
```

等待几分钟，然后再次运行该命令。

6.4.4.16. 删除 bootstrap 资源：

完成集群的初始 Operator 配置后，从 Amazon Web Services (AWS) 中删除 bootstrap 资源。

先决条件

- 已为集群完成初始的 Operator 配置。

流程

1. 删除 bootstrap 资源。如果您使用了 CloudFormation 模板，请删除其堆栈：
   - 使用 AWS CLI 删除堆栈：

     ```
     $ aws cloudformation delete-stack --stack-name <name> 1
     ```

     1 **<name>** 是 bootstrap 堆栈的名称。

   - 使用 AWS CloudFormation 控制台删除堆栈。

6.4.4.17. 创建 Ingress DNS 记录

如果您删除了 DNS 区配置，请手动创建指向 Ingress 负载均衡器的 DNS 记录。您可以创建一个 wildcard 记录或具体的记录。以下流程使用了 A 记录，但您可以使用其他所需记录类型，如 CNAME 或别名。

先决条件

- 已在 Amazon Web Services (AWS) 上安装了使用您置备的基础架构的 OpenShift Container Platform 集群。
- 已安装 OpenShift CLI（**oc**）。
- 安装了 **jq** 软件包。
- 您下载了 AWS CLI 并安装到您的计算机上。请参阅使用捆绑安装程序（Linux、macOS 或 Unix）安装 AWS CLI的文档。

流程

1. 决定要创建的路由。
   - 要创建一个 wildcard 记录，请使用 ***.apps.<cluster_name>.<domain_name>**，其中 **<cluster_name>** 是集群名称，**<domain_name>** 是 OpenShift Container Platform 集群的 Route 53 基域。
   - 要创建特定的记录，您必须为集群使用的每个路由创建一个记录，如下所示：

```
$ oc get --all-namespaces -o jsonpath='{range .items[*]}{range .status.ingress[*]}{.host}
{"\n"}{end}{end}' routes
```

输出示例

```
oauth-openshift.apps.<cluster_name>.<domain_name>
console-openshift-console.apps.<cluster_name>.<domain_name>
downloads-openshift-console.apps.<cluster_name>.<domain_name>
alertmanager-main-openshift-monitoring.apps.<cluster_name>.<domain_name>
prometheus-k8s-openshift-monitoring.apps.<cluster_name>.<domain_name>
```

2. 获取 Ingress Operator 负载均衡器状态，并记录其使用的外部 IP 地址值，如 **EXTERNAL-IP** 列所示：

```
$ oc -n openshift-ingress get service router-default
```

输出示例

```
NAME              TYPE           CLUSTER-IP      EXTERNAL-IP                            PORT(S)
AGE
router-default    LoadBalancer   172.30.62.215   ab3...28.us-east-2.elb.amazonaws.com
80:31499/TCP,443:30693/TCP   5m
```

3. 为负载均衡器定位托管区 ID：

```
$ aws elb describe-load-balancers | jq -r '.LoadBalancerDescriptions[] | select(.DNSName ==
"<external_ip>").CanonicalHostedZoneNameID' 1
```

1 对于 **<external_ip>**，请指定您获取的 Ingress Operator 负载均衡器的外部 IP 地址值。

输出示例

```
Z3AADJGX6KTTL2
```

这个命令的输出是负载均衡器托管区 ID。

4. 获取集群域的公共托管区 ID：

```
$ aws route53 list-hosted-zones-by-name \
        --dns-name "<domain_name>" \ 1
        --query 'HostedZones[? Config.PrivateZone != `true` && Name ==
`<domain_name>.`].Id' 2
        --output text
```

1 2 对于 **<domain_name>**，请为 OpenShift Container Platform 集群指定 Route 53 基域。

输出示例

```
/hostedzone/Z3URY6TWQ91KVV
```

命令输出中会显示您的域的公共托管区 ID。在本例中是 **Z3URY6TWQ91KVV**。

5. 在您的私有区中添加别名记录：

```
$ aws route53 change-resource-record-sets --hosted-zone-id "<private_hosted_zone_id>" --
change-batch '{ (1)
>   "Changes": [
>     {
>       "Action": "CREATE",
>       "ResourceRecordSet": {
>         "Name": "\\052.apps.<cluster_domain>", (2)
>         "Type": "A",
>         "AliasTarget":{
>           "HostedZoneId": "<hosted_zone_id>", (3)
>           "DNSName": "<external_ip>.", (4)
>           "EvaluateTargetHealth": false
>         }
>       }
>     }
>   ]
> }'
```

(1) 对于 **<private_hosted_zone_id>**，指定 DNS 和负载均衡的 CloudFormation 模板输出的值。

(2) 对于 **<cluster_domain>**，请指定用于 OpenShift Container Platform 集群的域或子域。

(3) 对于 **<hosted_zone_id>**，请为您获得的负载均衡器指定公共托管区 ID。

(4) 对于 **<external_ip>**，请指定 Ingress Operator 负载均衡器的外部 IP 地址值。请确定在该参数值中包含最后的句点（.）。

6. 在您的公共区中添加记录：

```
$ aws route53 change-resource-record-sets --hosted-zone-id "<public_hosted_zone_id>"" --
change-batch '{ (1)
>   "Changes": [
>     {
>       "Action": "CREATE",
>       "ResourceRecordSet": {
>         "Name": "\\052.apps.<cluster_domain>", (2)
>         "Type": "A",
>         "AliasTarget":{
>           "HostedZoneId": "<hosted_zone_id>", (3)
>           "DNSName": "<external_ip>.", (4)
>           "EvaluateTargetHealth": false
>         }
>       }
>     }
>   ]
> }'
```

(1) 对于 **<public_hosted_zone_id>**，请为您的域指定公共托管区。

2 对于 **<cluster_domain>**，请指定用于 OpenShift Container Platform 集群的域或子域。

3 对于 **<hosted_zone_id>**，请为您获得的负载均衡器指定公共托管区 ID。

4 对于 **<external_ip>**，请指定 Ingress Operator 负载均衡器的外部 IP 地址值。请确定在该参数值中包含最后的句点（.）。

6.4.4.18. 在用户置备的基础架构上完成 AWS 安装

在用户置备的基础架构 Amazon Web Service (AWS) 上启动 OpenShift Container Platform 安装后，监视进程并等待安装完成。

先决条件

- 您在用户置备的 AWS 基础架构上为 OpenShift Container Platform 集群删除了 bootstrap 节点。
- 已安装 **oc** CLI。

流程

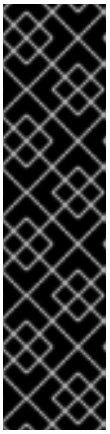
1. 在包含安装程序的目录中完成集群安装：

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

输出示例

```
INFO Waiting up to 40m0s for the cluster at https://api.mycluster.example.com:6443 to
initialize...
INFO Waiting up to 10m0s for the openshift-console route to be created...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 1s
```



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书中恢复的文档*。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

2. 在 Cluster registration 页面注册您的集群。

6.4.4.19. 使用 CLI 登录到集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

6.4.4.20. 使用 Web 控制台登录到集群

kubeadmin 用户默认在 OpenShift Container Platform 安装后存在。您可以使用 OpenShift Container Platform Web 控制台以 **kubeadmin** 用户身份登录集群。

先决条件

- 有访问安装主机的访问权限。
- 您完成了集群安装，所有集群 Operator 都可用。

流程

1. 从安装主机上的 **kubeadmin -password** 文件中获取 kubeadmin 用户的密码：

```
$ cat <installation_directory>/auth/kubeadmin-password
```

注意

另外，您还可以从安装主机上的 **<installation_directory>/.openshift_install.log** 日志文件获取 **kubeadmin** 密码。

2. 列出 OpenShift Container Platform Web 控制台路由：

```
$ oc get routes -n openshift-console | grep 'console-openshift'
```

注意

另外，您还可以从安装主机上的 **<installation_directory>/.openshift_install.log 日志** 文件获取 OpenShift Container Platform 路由。

输出示例

```
console     console-openshift-console.apps.<cluster_name>.<base_domain>           console
https   reencrypt/Redirect   None
```

3. 在 Web 浏览器中导航到上一命令输出中包括的路由，以 **kubeadmin** 用户身份登录。

其他资源

- 如需有关 访问和了解 OpenShift Container Platform Web 控制台的更多详情，请参阅 访问 Web 控制台。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于 远程健康监控

6.4.4.21. 其他资源

- 如需有关 AWS CloudFormation 堆栈的更多信息，请参阅 AWS 文档中的使用堆栈。

6.4.4.22. 后续步骤

- 验证安装。
- 自定义集群。
- 为 Cluster Samples Operator 和 **must-gather** 工具 配置镜像流。
- 了解如何在 受限网络中使用 Operator Lifecycle Manager(OLM)。
- 如果您用来安装集群的镜像 registry 具有可信任的 CA，请通过 配置额外的信任存储将其添加到集群中。
- 如果需要，您可以选择 不使用远程健康报告。
- 如果需要，请参阅 注册断开连接的集群
- 如果需要，您可以删除云供应商凭证。

6.5. 在 AWS 上安装三节点集群

在 OpenShift Container Platform 版本 4.16 中，您可以在 Amazon Web Services (AWS) 上安装三节点集群。三节点集群包含三个 control plane 机器，它们也可以充当计算机器。这种类型的集群提供了一个较小的、效率更高的集群，供集群管理员和开发人员用于测试、开发和生产。

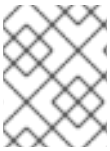
您可以使用安装程序置备或用户置备的基础架构安装三节点集群。

注意

不支持使用 AWS Marketplace 镜像部署三节点集群。

6.5.1. 配置三节点集群

在部署集群前，您可以通过将 **install-config.yaml** 文件中的 worker 节点数量设置为 **0** 来配置三节点集群。将 worker 节点数量设置为 **0** 可确保 control plane 机器可以调度。这允许调度应用程序工作负载从 control plane 节点运行。



注意

因为应用程序工作负载从 control plane 节点运行，所以需要额外的订阅，因为 control plane 节点被视为计算节点。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。

流程

1. 将 **install-config.yaml** 文件中的计算副本数量设置为 **0**，如以下 **compute** 小节中所示：

   三节点集群的 install-config.yaml 文件示例

   ```
   apiVersion: v1
   baseDomain: example.com
   compute:
   - name: worker
     platform: {}
     replicas: 0
   # ...
   ```

2. 如果您使用用户置备的基础架构部署集群：

   - 创建 Kubernetes 清单文件后，请确保在 **cluster-scheduler-02-config.yml** 文件中将 **spec.mastersSchedulable** 参数设置为 **true**。您可以在 **<installation_directory>/manifests** 中找到此文件。如需更多信息，请参阅"使用 CloudFormation 模板在 AWS 中用户置备的基础架构上安装集群"中的"创建 Kubernetes 清单和 Ignition 配置文件"。

   - 不要创建额外的 worker 节点。

三节点集群的 cluster-scheduler-02-config.yml 文件示例

```
apiVersion: config.openshift.io/v1
kind: Scheduler
metadata:
  creationTimestamp: null
  name: cluster
spec:
  mastersSchedulable: true
```

```
  policy:
    name: ""
status: {}
```

6.5.2. 后续步骤

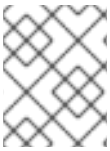
- 使用自定义在 AWS 上安装集群
- 使用 CloudFormation 模板在 AWS 中用户置备的基础架构上安装集群

6.6. 在 AWS 上卸载集群

您可以删除部署到 Amazon Web Services (AWS) 的集群。

6.6.1. 删除使用安装程序置备的基础架构的集群

您可以从云中删除使用安装程序置备的基础架构的集群。



注意

卸载后，检查云供应商是否有未正确删除的资源，特别是在用户置备基础架构(UPI)集群中。可能存在安装程序未创建或安装程序无法访问的资源。

先决条件

- 有用于部署集群的安装程序副本。
- 有创建集群时安装程序生成的文件。

流程

1. 在用来安装集群的计算机中包含安装程序的目录中，运行以下命令：

```
$ ./openshift-install destroy cluster \
--dir <installation_directory> --log-level info 1 2
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2 要查看不同的详情，请指定 **warn**、**debug** 或 **error**，而不是 **info**。

注意

您必须为集群指定包含集群定义文件的目录。安装程序需要此目录中的 **metadata.json** 文件来删除集群。

2. 可选：删除 **<installation_directory>** 目录和 OpenShift Container Platform 安装程序。

6.6.2. 使用 Cloud Credential Operator 实用程序删除 Amazon Web Services 资源

卸载使用在集群外管理的短期凭证的 OpenShift Container Platform 集群后，您可以使用 CCO 实用程序 (**ccoctl**) 删除 **ccoctl** 在安装过程中创建的 Amazon Web Services (AWS) 资源。

先决条件

- 提取并准备 **ccoctl** 二进制文件。
- 卸载使用短期凭证的 AWS 上的 OpenShift Container Platform 集群。

流程

- 运行以下命令，删除 **ccoctl** 创建的 AWS 资源：

```
$ ccoctl aws delete \
  --name=<name> \ 1
  --region=<aws_region> 2
```

1 **<name>** 与最初用于创建和标记云资源的名称匹配。

2 **<aws_region>** 是要删除云资源的 AWS 区域。

输出示例

```
2021/04/08 17:50:41 Identity Provider object .well-known/openid-configuration deleted from the bucket <name>-oidc
2021/04/08 17:50:42 Identity Provider object keys.json deleted from the bucket <name>-oidc
2021/04/08 17:50:43 Identity Provider bucket <name>-oidc deleted
2021/04/08 17:51:05 Policy <name>-openshift-cloud-credential-operator-cloud-credential-o associated with IAM Role <name>-openshift-cloud-credential-operator-cloud-credential-o deleted
2021/04/08 17:51:05 IAM Role <name>-openshift-cloud-credential-operator-cloud-credential-o deleted
2021/04/08 17:51:07 Policy <name>-openshift-cluster-csi-drivers-ebs-cloud-credentials associated with IAM Role <name>-openshift-cluster-csi-drivers-ebs-cloud-credentials deleted
2021/04/08 17:51:07 IAM Role <name>-openshift-cluster-csi-drivers-ebs-cloud-credentials deleted
2021/04/08 17:51:08 Policy <name>-openshift-image-registry-installer-cloud-credentials associated with IAM Role <name>-openshift-image-registry-installer-cloud-credentials deleted
2021/04/08 17:51:08 IAM Role <name>-openshift-image-registry-installer-cloud-credentials deleted
2021/04/08 17:51:09 Policy <name>-openshift-ingress-operator-cloud-credentials associated with IAM Role <name>-openshift-ingress-operator-cloud-credentials deleted
2021/04/08 17:51:10 IAM Role <name>-openshift-ingress-operator-cloud-credentials deleted
2021/04/08 17:51:11 Policy <name>-openshift-machine-api-aws-cloud-credentials associated with IAM Role <name>-openshift-machine-api-aws-cloud-credentials deleted
2021/04/08 17:51:11 IAM Role <name>-openshift-machine-api-aws-cloud-credentials deleted
2021/04/08 17:51:39 Identity Provider with ARN arn:aws:iam::<aws_account_id>:oidc-provider/<name>-oidc.s3.<aws_region>.amazonaws.com deleted
```

验证

- 要验证资源是否已删除，请查询 AWS。如需更多信息，请参阅 AWS 文档。

6.6.3. 使用配置的 AWS Local Zone 基础架构删除集群

在 Amazon Web Services (AWS) 上安装集群后，进入现有的 Virtual Private Cloud (VPC)，并为每个 Local Zone 位置设置子网，您可以删除集群以及与之关联的任何 AWS 资源。

该流程中的示例假设您使用 CloudFormation 模板创建了 VPC 及其子网。

先决条件

- 您知道创建网络期间使用的 CloudFormation 堆栈 **<local_zone_stack_name>** 和 **<vpc_stack_name>** 的名称。您需要堆栈的名称来删除集群。
- 您可以访问包含安装程序创建的安装文件的目录的权限。
- 您的帐户包含一个策略，为您提供了删除 CloudFormation 堆栈的权限。

流程

1. 进入包含存储的安装程序的目录，并使用 **destroy cluster** 命令删除集群：

```
$ ./openshift-install destroy cluster --dir <installation_directory> \ 1
  --log-level=debug 2
```

1 对于 **<installation_directory>**，请指定保存安装程序创建的任何文件的目录。

2 要查看不同的日志详细信息，请指定 **error**、**info** 或 **warn**，而不是 **debug**。

2. 删除 Local Zone 子网的 CloudFormation 堆栈：

```
$ aws cloudformation delete-stack --stack-name <local_zone_stack_name>
```

3. 删除代表 VPC 的资源堆栈：

```
$ aws cloudformation delete-stack --stack-name <vpc_stack_name>
```

验证

- 通过在 AWS CLI 中执行以下命令，检查您是否删除了堆栈资源。AWS CLI 输出没有模板组件。

```
$ aws cloudformation describe-stacks --stack-name <local_zone_stack_name>
```

```
$ aws cloudformation describe-stacks --stack-name <vpc_stack_name>
```

其他资源

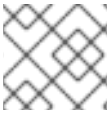
- 如需有关 AWS CloudFormation 堆栈的更多信息，请参阅 AWS 文档中的使用堆栈。
- 选择 AWS 本地区域
- AWS 本地区域可用位置
- AWS Local Zones 功能

6.7. AWS 的安装配置参数

在 AWS 上部署 OpenShift Container Platform 集群前，您可以提供参数来自定义集群和托管它的平台。在创建 **install-config.yaml** 文件时，您可以通过命令行为所需参数提供值。然后，您可以修改 **install-config.yaml** 文件以进一步自定义集群。

6.7.1. AWS 可用的安装配置参数

下表指定您可以在安装过程中设置所需的、可选和 AWS 特定安装配置参数。



注意

安装后，您无法在 **install-config.yaml** 文件中修改这些参数。

6.7.1.1. 所需的配置参数

下表描述了所需的安装配置参数：

表 6.27. 所需的参数

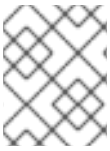
参数	描述	值
apiVersion:	**install-config.yaml** 内容的 API 版本。当前版本为 **v1**。安装程序可能还支持旧的 API 版本。	字符串
baseDomain:	云供应商的基域。基域用于创建到 OpenShift Container Platform 集群组件的路由。集群的完整 DNS 名称是 **baseDomain** 和 **metadata.name** 参数值的组合，其格式为 **<metadata.name>.<baseDomain>**。	完全限定域名或子域名，如 **example.com**。
metadata:	Kubernetes 资源 **ObjectMeta**，其中只消耗 **name** 参数。	对象
metadata: name:	集群的名称。集群的 DNS 记录是 **{{.metadata.name}}.{{.baseDomain}}** 的子域。	小写字母、连字符(**-**)和句点(**.**)字符串，如 **dev**。
platform:	对于特定平台的配置取决于执行安装的环境： **aws**, **baremetal**, **azure**, **gcp**, **ibmcloud**, **nutanix**, **openstack**, **powervs**, **vsphere**, 或 **{}**.有关 **platform.<platform>** 参数的更多信息，请参考下表中您的特定平台。	对象

参数	描述	值
pullSecret:	从 Red Hat OpenShift Cluster Manager 获取 pull secret，验证从 Quay.io 等服务中下载 OpenShift Container Platform 组件的容器镜像。	{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }

6.7.1.2. 网络配置参数

您可以根据现有网络基础架构的要求自定义安装配置。例如，您可以扩展集群网络的 IP 地址块，或者提供不同于默认值的不同 IP 地址块。

仅支持 IPv4 地址。



注意

Red Hat OpenShift Data Foundation 灾难恢复解决方案不支持 Globalnet。对于区域灾难恢复场景，请确保为每个集群中的集群和服务网络使用非重叠的专用 IP 地址。

表 6.28. 网络参数

参数	描述	值
networking:	集群网络的配置。	对象  **注意** 您无法在安装后修改 **网络** 对象指定的参数。
networking: networkType:	要安装的 Red Hat OpenShift Networking 网络插件。	**OVNKubernetes**。**OVNKubernetes** 是 Linux 网络和包含 Linux 和 Windows 服务器的混合网络的 CNI 插件。默认值为 **OVNKubernetes**。

参数	描述	值
networking: clusterNetwork:	pod 的 IP 地址块。 默认值为 **10.128.0.0/14**，主机前缀为 **/23**。 如果您指定了多个 IP 地址块，块不得重叠。	对象数组。例如： networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23
networking: clusterNetwork: cidr:	使用 **networking.clusterNetwork** 时需要此项。IP 地址块。 IPv4 网络。	无类别域间路由(CIDR)表示法中的 IP 地址块。IPv4 块的前缀长度介于 **0 到 32** 之间。
networking: clusterNetwork: hostPrefix:	分配给每个节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从 given **cidr** 中分配 a **/23** 子网。**hostPrefix** 值 **23** 提供 510（2^(32 - 23)- 2）pod IP 地址。	子网前缀。 默认值为 **23**。
networking: serviceNetwork:	服务的 IP 地址块。默认值为 **172.30.0.0/16**。 OVN-Kubernetes 网络插件只支持服务网络的一个 IP 地址块。	CIDR 格式具有 IP 地址块的数组。例如： networking: serviceNetwork: - 172.30.0.0/16
networking: machineNetwork:	机器的 IP 地址块。 如果您指定了多个 IP 地址块，块不得重叠。	对象数组。例如： networking: machineNetwork: - cidr: 10.0.0.0/16
networking: machineNetwork: cidr:	使用 **networking.machineNetwork** 时需要此项。IP 地址块。对于 libvirt 和 IBM Power® Virtual Server 以外的所有平台，默认值为 **10.0.0.0/16**。对于 libvirt，默认值 为 **192.168.126.0/24**。对于 IBM Power® Virtual Server，默认值为 **192.168.0.0/24**。	CIDR 表示法中的 IP 网络块。 例如： **10.0.0.0/16**。 **注意** 将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

6.7.1.3. 可选的配置参数

下表描述了可选的安装配置参数：

表 6.29. 可选参数

参数	描述	值
additionalTrustBundle:	添加到节点可信证书存储中的 PEM 编码 X.509 证书捆绑包。配置了代理时，也可以使用此信任捆绑包。	字符串
capabilities:	控制可选核心组件的安装。您可以通过禁用可选组件来减少 OpenShift Container Platform 集群的空间。如需更多信息，请参阅*安装*中的"集群功能"页面。	字符串数组
capabilities: baselineCapabilitySet:	选择要启用的一组初始可选功能。有效值为 **None**、**v4.11**、**v4.12** 和 **vCurrent**。默认值为 **vCurrent**。	字符串
capabilities: additionalEnabledCapabilities:	将可选功能集合扩展到您在 **baselineCapabilitySet** 中指定的范围。您可以在此参数中指定多个功能。	字符串数组
cpuPartitioningMode:	启用工作负载分区，它会隔离 OpenShift Container Platform 服务、集群管理工作负载和基础架构 pod，以便在保留的一组 CPU 上运行。工作负载分区只能在安装过程中启用，且在安装后无法禁用。虽然此字段启用工作负载分区，但它不会将工作负载配置为使用特定的 CPU。如需更多信息，请参阅 *Scalability and Performance* 部分中的 *Workload partitioning* 页面。	**None** 或 **AllNodes**.**None** 是默认值。
compute:	组成计算节点的机器的配置。	**MachinePool** 对象的数组。
compute: architecture:	决定池中机器的指令集合架构。目前，不支持具有不同架构的集群。所有池都必须指定相同的架构。有效值为 **amd64** 和 **arm64**。并非所有安装选项都支持 64 位 ARM 架构。要验证您的平台上是否支持您的安装选项，请参阅 *选择集群安装方法并为用户准备它*中的*不同平台支持的安装方法*。	字符串

参数	描述	值
compute: hyperthreading:	是否在计算机器上启用或禁用并发多**线程或超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。	**enabled** 或 **Disabled**
compute: name:	使用 **compute** 时需要此项。机器池的名称。	**worker**
compute: platform:	使用 **compute** 时需要此项。使用此参数指定托管 worker 机器的云供应商。此参数值必须与 **controlPlane.platform** 参数值匹配。	**aws**, **azure**, **gcp**, **ibmcloud**, **nutanix**, **openstack**, **powervs**, **vsphere**, 或 **{}**
compute: replicas:	要置备的计算机器数量，也称为 worker 机器。	大于或等于 **2** 的正整数。默认值为 **3**。
featureSet:	为功能集启用集群。功能集是 OpenShift Container Platform 功能的集合，默认情况下不启用。有关在安装过程中启用功能集的更多信息，请参阅"使用功能门启用功能"。	字符串.要启用的功能集的名称，如 **TechPreviewNoUpgrade**。
controlPlane:	组成 control plane 的机器的配置。	**MachinePool** 对象的数组。
controlPlane: architecture:	决定池中机器的指令集合架构。目前，不支持具有不同架构的集群。所有池都必须指定相同的架构。有效值为 **amd64** 和 **arm64**。并非所有安装选项都支持 64 位 ARM 架构。要验证您的平台上是否支持您的安装选项，请参阅 *选择集群安装方法并为用户准备它*中的*不同平台支持的安装方法*。	字符串

参数	描述	值
controlPlane: hyperthreading:	是否在 control plane 机器上启用或禁用并发多**线程或超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。	**enabled** 或 **Disabled**
controlPlane: name:	使用 **controlPlane** 时需要此项。机器池的名称。	**master**
controlPlane: platform:	使用 **controlPlane** 时需要此项。使用此参数指定托管 control plane 机器的云供应商。此参数值必须与 **compute.platform** 参数值匹配。	**aws**, **azure**, **gcp**, **ibmcloud**, **nutanix**, **openstack**, **powervs**, **vsphere**, 或 **{}**
controlPlane: replicas:	要置备的 control plane 机器数量。	部署单节点 OpenShift 时支持的值为 **3** 或 **1**。
credentialsMode:	Cloud Credential Operator(CCO)模式。如果没有指定模式，CCO 会动态尝试决定提供的凭证的功能，在支持多个模式的平台上首选 mint 模式。	**Mint**、**Passthrough**、**Manual** 或空字符串(**""**)。[1]

参数	描述	值
fips:	启用或禁用 FIPS 模式。默认值为 **false**（禁用）。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。  **重要** 要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 Red Hat Enterprise Linux (RHEL) 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅在 FIPS 模式中安装该系统。 当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS) 时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。  **注意** 如果使用 Azure File 存储，则无法启用 FIPS 模式。	**false** 或 **true**
imageContentSources:	release-image 内容的源和存储库。	对象数组。包括一个 **source** 以及可选的 **mirrors**，如本表的以下行所述。
imageContentSources: source:	使用 **imageContentSources** 时需要此项。指定用户在镜像拉取规格中引用的存储库。	字符串

参数	描述	值
imageContentSources: mirrors:	指定可能还包含同一镜像的一个或多个仓库。	字符串数组
platform: aws: lbType:	在 AWS 中设置 NLB 负载均衡器类型需要此项。有效值为 **Classic** 或 **NLB**。如果没有指定值，安装程序将默认为 **Classic**。安装程序设置 ingress 集群配置对象中提供的值。如果您没有为其他 Ingress Controller 指定负载均衡器类型，它们将使用此参数中设置的类型。	**Classic** 或 **NLB**.默认值为 **Classic**。
publish:	如何发布或公开集群的面向用户的端点，如 Kubernetes API、OpenShift 路由。	**内部或外部 .**要部署无法从互联网访问的私有集群，请将 **publish** 设置为 **Internal**。默认值为 **External**。
sshKey:	用于验证对集群机器的访问的 SSH 密钥。 **注意** 对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。	例如，**sshKey: ssh-ed25519 AAAA..**。

1. 不是所有 CCO 模式都支持所有云供应商。有关 CCO 模式的更多信息，请参阅*身份验证和授权*内容中的"管理云供应商凭证"条目。

注意

如果您的 AWS 帐户启用了服务控制策略 (SCP)，必须将 **credentialsMode** 参数配置为 **Mint**、**Passthrough** 或 **Manual**。

重要

将此参数设置为 **Manual** 可启用在 **kube-system** 项目中存储管理员级别的 secret 的替代方案，这需要额外的配置步骤。如需更多信息，请参阅"在 kube-system 项目中存储管理员级别的 secret"。

6.7.1.4. 可选的 AWS 配置参数

下表描述了可选的 AWS 配置参数：

表 6.30. 可选的 AWS 参数

参数	描述	值
compute: platform: aws: amiID:	用于为集群引导计算机器的 AWS AMI。对于需要自定义 RHCOS AMI 的区域来说，这是必需的。	属于集合 AWS 区域的任何已发布或自定义 RHCOS AMI。如需可用的 AMI ID，请参阅 *RHCOS AMIs for AWS infrastructure*。
compute: platform: aws: iamRole:	一个已存在的 AWS IAM 角色应用到计算机器池实例配置集。您可以使用这些字段与命名方案匹配，并为您的 IAM 角色包含预定义的权限界限。如果未定义，安装程序会创建一个新的 IAM 角色。	有效 AWS IAM 角色的名称。
compute: platform: aws: rootVolume: iops:	为根卷保留的每秒输入/输出操作 (IOPS) 数。	整数，如 **4000**。
compute: platform: aws: rootVolume: size:	以 GiB 为单位的根卷大小。	整数，如 **500**。
compute: platform: aws: rootVolume: type:	根卷的类型。	有效的 AWS EBS 卷类型，如 **io1**。
compute: platform: aws: rootVolume: kmsKeyARN:	KMS 密钥的 Amazon 资源名称（密钥 ARN）。这需要使用特定的 KMS 密钥加密 worker 节点的操作系统卷。	有效的 密钥 ID 或密钥 ARN。

参数	描述	值

`compute:`
`  platform:`
`  aws:`
`  type:`
	计算机器的 EC2 实例类型。	有效的 AWS 实例类型，如 **m4.2xlarge**。请参阅以下 **支持的 AWS 机器类型** 表。

`compute:`
`  platform:`
`  aws:`
`  zones:`
	安装程序在其中为计算机机器池创建机器的可用区。如果您提供自己的 VPC，则必须在那个可用域中提供一个子网。	有效 AWS 可用区的列表，如 **us-east-1c**，以 YAML 序列表示。

`compute:`
`  aws:`
`  region:`
	安装程序在其中创建计算资源的 AWS 区域。	任何有效的 AWS 区域，如 **us-east-1**。您可以使用 AWS CLI 访问您选择的实例类型可用的区域。例如：
		`aws ec2 describe-instance-type-offerings --filters Name=instance-type,Values=c7g.xlarge`
		 
		重要
		在基于 ARM 的 AWS 实例上运行时，请确保进入 AWS Graviton 处理器可用的区域。请参阅 AWS 文档中的全局可用性 映射。目前，只有一些区域才提供 AWS Graviton3 处理器。

`controlPlane:`
`  platform:`
`  aws:`
`  amiID:`
	用于为集群引导 control plane 机器的 AWS AMI。对于需要自定义 RHCOS AMI 的区域来说，这是必需的。	属于集合 AWS 区域的任何已发布或自定义 RHCOS AMI。如需可用的 AMI ID，请参阅 *RHCOS AMIs for AWS infrastructure*。

`controlPlane:`
`  platform:`
`  aws:`
`  iamRole:`
	应用到 control plane 机器池实例配置集的已存在的 AWS IAM 角色。您可以使用这些字段与命名方案匹配，并为您的 IAM 角色包含预定义的权限界限。如果未定义，安装程序会创建一个新的 IAM 角色。	有效 AWS IAM 角色的名称。

参数	描述	值
controlPlane: platform: aws: rootVolume: iops:	为 control plane 机器上的根卷保留的每秒输入/输出操作 (IOPS)。	整数，如 **4000**。
controlPlane: platform: aws: rootVolume: size:	control plane 机器的根卷大小（以 GiB 为单位）。	整数，如 **500**。
controlPlane: platform: aws: rootVolume: type:	control plane 机器的根卷的类型。	有效的 AWS EBS 卷类型，如 **io1**。
controlPlane: platform: aws: rootVolume: kmsKeyARN:	KMS 密钥的 Amazon 资源名称（密钥 ARN）。这需要使用特定的 KMS 密钥加密 control plane 节点的操作系统卷。	有效的密钥 ID 和密钥 ARN。
controlPlane: platform: aws: type:	control plane 机器的 EC2 实例类型。	有效的 AWS 实例类型，如 **m6i.xlarge**。请参阅以下 **支持的 AWS 机器类型** 表。
controlPlane: platform: aws: zones:	安装程序在其中为 control plane 机器池创建机器的可用区。	有效 AWS 可用区的列表，如 **us-east-1c**，以 YAML 序列表示。

参数	描述	值
controlPlane: aws: region:	安装程序在其中创建 control plane 资源的 AWS 区域。	有效的 AWS 区域，如 **us-east-1**。
platform: aws: amiID:	用于为集群引导所有机器的 AWS AMI。如果设置，AMI 必须属于与集群相同的区域。对于需要自定义 RHCOS AMI 的区域来说，这是必需的。	属于集合 AWS 区域的任何已发布或自定义 RHCOS AMI。如需可用的 AMI ID，请参阅 *RHCOS AMIs for AWS infrastructure*。
platform: aws: hostedZone:	集群的现有 Route 53 私有托管区。您只能在提供自己的 VPC 时使用已存在的托管区。安装前，托管区必须已经与用户提供的 VPC 关联。另外，托管区的域必须是集群域或集群域的父域。如果未定义，安装程序会创建一个新的托管区。	字符串，如 **Z3URY6TWQ91KVV**。
platform: aws: hostedZoneRole:	在包含指定托管区的帐户中现有 IAM 角色的 Amazon 资源名称(ARN)。在托管区上执行操作时，安装程序和集群操作器会假定此角色。只有在您将集群安装到共享 VPC 中时，才应使用此参数。	字符串，如 **arn:aws:iam::1234567890:role/shared-vpc-role**。
platform: aws: serviceEndpoints: - name: url:	AWS 服务端点名称和 URL。只有在必须使用替代 AWS 端点（如 FIPS）时，才需要自定义端点。可以为 EC2、S3、IAM、Elastic Load Balancing、Tagging、Route 53 和 STS AWS 服务指定自定义 API 端点。	有效的 AWS 服务端点 名称和有效的 AWS 服务端点 URL。
platform: aws: userTags:	键与值的映射，安装程序将其作为标签添加到它所创建的所有资源。	任何有效的 YAML 映射，如 **<key>: <value>** 格式的键值对。如需有关 AWS 标签的更多信息，请参阅 AWS 文档中的标记您的 Amazon EC2 资源。 **注意** 您可以在安装过程中最多添加 25 个用户定义的标签。剩余的 25 个标签是为 OpenShift Container Platform 保留的。

参数	描述	值
platform: aws: propagateUserTags:	指示集群 Operator 中的标记，在 Operator 创建的 AWS 资源标签中包含指定的用户标签。	布尔值，如 **true** 或 **false**。
platform: aws: subnets:	如果您提供 VPC，而不是让安装程序为您创建 VPC，请指定要使用的集群子网。子网必须是您指定的同一 **machineNetwork[].cidr** 范围的一部分。 对于标准集群，为每个可用区指定一个公共和私有子网。 对于私有集群，为每个可用区指定一个私有子网。 对于使用 AWS Local Zones 的集群，您必须将 AWS Local Zone 子网添加到此列表中，以确保创建边缘机器池。	有效的子网 ID。
platform: aws: publicIpv4Pool:	当 **publish** 被设置为 **External** 时，用于分配 Elastic IP (EIPs) 的公共 IPv4 池 ID。您必须在集群的同一 AWS 帐户和区域中置备并公告池。您必须确保池中有 2n + 1 个 IPv4 可用，其中 *n* 是用于部署 API、NAT 网关和 bootstrap 节点的 Network Load Balancer (NLB) 的 AWS 区域总数。有关在 AWS 中包含您自己的 IP 地址 (BYOIP) 的更多信息，请参阅 BYOIP。	有效的 公共 IPv4 池 id **注意** BYOIP 只能为没有网络限制的自定义安装启用。
platform: aws: preserveBootstrapIgnition:	防止在完成 bootstrap 后删除 S3 存储桶。	**true** 或 **false**。默认值为 **false**，这会导致 S3 存储桶被删除。

第 7 章 在 AZURE 上安装

7.1. 准备在 AZURE 上安装

7.1.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读选择集群安装方法并为用户准备它的文档。

7.1.2. 在 Azure 上安装 OpenShift Container Platform 的要求

在 Microsoft Azure 上安装 OpenShift Container Platform 前，您必须配置 Azure 帐户。如需了解有关帐户配置、帐户限值、公共 DNS 区配置、所需角色、创建服务主体和支持的 Azure 区域的详细信息，请参阅配置 Azure 帐户。

如果环境中无法访问云身份和访问管理 (IAM) API，或者不想将管理员级别的凭证 secret 存储在 **kube-system** 命名空间中，请参阅在 kube-system 项目中存储管理员级别的 secret 以了解其他选项。

7.1.3. 选择在 Azure 上安装 OpenShift Container Platform 的方法

您可以在安装程序置备的基础架构或用户置备的基础架构上安装 OpenShift Container Platform。默认安装类型使用安装程序置备的基础架构，安装程序会在其中为集群置备底层基础架构。您还可以在您置备的基础架构上安装 OpenShift Container Platform。如果不使用安装程序置备的基础架构，您必须自己管理和维护集群资源。

如需有关安装程序置备和用户置备的安装过程的更多信息，请参阅 安装过程。

7.1.3.1. 在安装程序置备的基础架构上安装集群

您可以使用以下方法之一在 OpenShift Container Platform 安装程序置备的 Azure 基础架构上安装集群：

- **在 Azure 上快速安装集群**：您可以在由 OpenShift Container Platform 安装程序置备的 Azure 基础架构上安装 OpenShift Container Platform。您可以使用默认配置选项快速安装集群。
- **在 Azure 上安装自定义集群**：您可以在安装程序置备的 Azure 基础架构上安装自定义集群。安装程序允许在安装阶段应用一些自定义。其它自定义选项可在安装后使用。
- **使用自定义网络在 Azure 上安装集群**：您可以在安装过程中自定义 OpenShift Container Platform 网络配置，以便集群可以与现有的 IP 地址分配共存,并遵循您的网络要求。
- **在 Azure 上将集群安装到现有的 VNet 中**：您可以在 Azure 上的现有 Azure Virtual Network（VNet）上安装 OpenShift Container Platform。如果您按照公司的说明设置了限制，可以使用这个安装方法，例如在创建新帐户或基础架构时的限制。
- **在 Azure 上安装私有集群**：您可以在 Azure 上将私有集群安装到现有 Azure Virtual Network（VNet）中。您可以使用此方法将 OpenShift Container Platform 部署到互联网中不可见的内部网络中。
- **在 Azure 上将集群安装到一个政府区域**：OpenShift Container Platform 可以部署到 Microsoft Azure Government（MAG）区域，这些区域是为需要运行敏感工作负载的美国政府机构、州和本地级别的政府机构、企业和其他需要运行敏感工作负载的美国客户而设计的。

7.1.3.2. 在用户置备的基础架构上安装集群

您可以使用以下方法在您置备的 Azure 基础架构上安装集群：

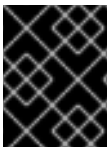
- **使用 ARM 模板在 Azure 上安装集群**：您可以使用您提供的基础架构在 Azure 上安装 OpenShift Container Platform。您可以使用提供的 Azure Resource Manager（ARM）模板来协助安装。

7.1.4. 后续步骤

- 配置 Azure 帐户

7.2. 配置 AZURE 帐户

在安装 OpenShift Container Platform 之前，您必须配置 Microsoft Azure 帐户来满足安装要求。



重要

所有通过公共端点提供的 Azure 资源均存在资源名称的限制，您无法创建使用某些名称的资源。如需 Azure 限制词语列表，请参阅 Azure 文档中的解决保留资源名称错误。

7.2.1. Azure 帐户限值

OpenShift Container Platform 集群使用诸多 Microsoft Azure 组件，默认的 Azure 订阅和服务限值、配额和约束会影响您安装 OpenShift Container Platform 集群的能力。

重要

默认的限制因服务类别的不同（如 Free Trial 或 Pay-As-You-Go）以及系列的不同（如 Dv2、F 或 G）而有所不同。例如，对于 Enterprise Agreement 订阅的默认限制是 350 个内核。

在 Azure 上安装默认集群前，请检查您的订阅类型的限制，如有必要，请提高帐户的配额限制。

下表总结了 Azure 组件，它们的限值会影响您安装和运行 OpenShift Container Platform 集群的能力。

组件	默认所需的组件数	默认 Azure 限值	描述

组件	默认所需的组件数	默认 Azure 限值	描述
vCPU	44	每个区域 20 个	默认集群需要 44 个 vCPU，因此您必须提高帐户限值。 默认情况下，每个集群创建以下实例： ● 一台 Bootstrap 机器，在安装后删除 ● 三个 control plane 机器 ● 三个计算（compute）机器 因为 bootstrap 和 control plane 机器使用 **Standard_D8s_v3** 虚拟机 (使用 8 个 vCPU)，因此计算机器使用 **Standard_D4s_v3** 虚拟机（使用 4 个 vCPU），所以默认集群需要 44 个 vCPU。bootstrap 节点 VM（使用 8 个 vCPU）只在安装过程中使用。 若要部署更多 worker 节点、启用自动扩展、部署大型工作负载或使用不同的实例类型，您必须进一步提高帐户的 vCPU 限值，以确保集群可以部署您需要的机器。
OS Disk	7		每个集群机器必须至少有 100 GB 存储和 300 IOPS。虽然这些值是最低支持的值，但对于具有密集型工作负载的生产环境集群和集群，建议使用更快的存储。有关优化性能存储的更多信息，请参阅"扩展和性能"部分中的页面标题为"优化存储"。
VNet	1	每个区域 1000 个	每个默认集群都需要一个虚拟网络 (VNet)，此网络包括两个子网。
网络接口	7	每个区域 65,536 个	每个默认集群都需要 7 个网络接口。如果您要创建更多机器或者您部署的工作负载要创建负载均衡器，则集群会使用更多的网络接口。
网络安全组	2	5000	每个集群为 VNet 中的每个子网创建网络安全组。默认集群为 control plane 和计算节点子网创建网络安全组： **controlplane**：允许从任何位置通过端口 6443 访问 control plane 机器 **node**：允许从互联网通过端口 80 和 443 访问 worker 节点

组件	默认所需的组件数	默认 Azure 限值	描述
网络负载均衡器	3	每个区域 1000 个	每个集群都会创建以下负载均衡器： **default** 用于在 worker 机器之间对端口 80 和 443 的请求进行负载均衡的公共 IP 地址 **internal** 用于在 control plane 机器之间对端口 6443 和 22623 的请求进行负载均衡的专用 IP 地址 **external** 用于在 control plane 机器之间对端口 6443 的请求进行负载均衡的公共 IP 地址 如果您的应用程序创建了更多的 Kubernetes **LoadBalancer** 服务对象，您的集群会使用更多的负载均衡器。
公共 IP 地址	3		两个公共负载均衡器各自使用一个公共 IP 地址。bootstrap 机器也使用一个公共 IP 地址，以便您可以在安装期间通过 SSH 连接到该机器来进行故障排除。bootstrap 节点的 IP 地址仅在安装过程中使用。
专用 IP 地址	7		内部负载均衡器、三台 control plane 机器中的每一台以及三台 worker 机器中的每一台各自使用一个专用 IP 地址。
Spot VM vCPU（可选）	0 如果配置 spot 虚拟机，您的集群必须为每个计算节点有两个 spot VM vCPU。	每个区域 20 个	这是可选组件。要使用 spot 虚拟机，您必须将 Azure 默认限值增加到集群中至少有两倍的计算节点数量。 **注意** 不建议将 spot 虚拟机用于 control plane 节点。

其他资源

- 优化存储.

7.2.2. 在 Azure 中配置公共 DNS 区

要安装 OpenShift Container Platform，您使用的 Microsoft Azure 帐户必须在帐户中具有一个专用的公共托管 DNS 区。此区域必须对域具有权威。此服务为集群外部连接提供集群 DNS 解析和名称查询。

流程

1. 标识您的域或子域，以及注册商（registrar）。您可以转移现有的域和注册商，或通过 Azure 或其他来源获取新的域和注册商。

注意

如需通过 Azure 购买域的更多信息，请参阅 Azure 文档中的购买 Azure 应用服务的自定义域名。

2. 如果您使用现有的域和注册商，请将其 DNS 迁移到 Azure。请参阅 Azure 文档中的将活动 DNS 名称迁移到 Azure 应用服务。

3. 为您的域配置 DNS。按照 Azure 文档中教程：在 Azure DNS 中托管域部分里的步骤，为您的域或子域创建一个公共托管区，提取新的权威名称服务器，并更新您的域使用的名称服务器的注册商记录。
使用合适的根域（如 **openshiftcorp.com**）或子域（如 **clusters.openshiftcorp.com**）。

4. 如果您使用子域，请按照您公司的流程将其委派记录添加到父域。

7.2.3. 提高 Azure 帐户限值

要提高帐户限值，请在 Azure 门户上提交支持请求。

注意

每一支持请求只能提高一种类型的配额。

流程

1. 从 Azure 门户，点击左下角的 **Help + suport**。

2. 点击 **New support request**，然后选择所需的值：

   a. 从 **Issue type** 列表中，选择 **Service and subscription limits (quotas)**。

   b. 从 **Subscription** 列表中，选择要修改的订阅。

   c. 从 **Quota type** 列表中，选择要提高的配额。例如，选择 **Compute-VM (cores-vCPUs) subscription limit increases** 以增加 vCPU 的数量，这是安装集群所必须的。

   d. 点击 **Next: Solutions**。

3. 在 **Problem Details** 页面中，提供您要提高配额所需的信息：

   a. 点击 **Provide details**，然后在 **Quota details** 窗口中提供所需的详情。

   b. 在 SUPPORT METHOD 和 CONTACT INFO 部分中，提供问题严重性和您的联系详情。

4. 点击 **Next: Review + create**，然后点击 **Create**。

7.2.4. 记录下订阅和租户 ID

安装程序需要与 Azure 帐户关联的订阅和租户 ID。您可以使用 Azure CLI 收集此信息。

先决条件

- 已安装或更新 Azure CLI。

流程

1. 运行以下命令登录到 Azure CLI：

```
$ az login
```

2. 确保您使用正确的订阅：

   a. 运行以下命令，查看可用订阅列表：

```
$ az account list --refresh
```

   输出示例

```
[
  {
    "cloudName": "AzureCloud",
    "id": "8xxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
    "isDefault": true,
    "name": "Subscription Name 1",
    "state": "Enabled",
    "tenantId": "6xxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
    "user": {
      "name": "you@example.com",
      "type": "user"
    }
  },
  {
    "cloudName": "AzureCloud",
    "id": "9xxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
    "isDefault": false,
    "name": "Subscription Name 2",
    "state": "Enabled",
    "tenantId": "7xxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
    "user": {
      "name": "you2@example.com",
      "type": "user"
    }
  }
]
```

   b. 运行以下命令，查看活跃帐户的详情，并确认这是您要使用的订阅：

```
$ az account show
```

   输出示例

```
{
  "environmentName": "AzureCloud",
  "id": "8xxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
  "isDefault": true,
  "name": "Subscription Name 1",
  "state": "Enabled",
  "tenantId": "6xxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
  "user": {
    "name": "you@example.com",
```

```
    "type": "user"
  }
}
```

3. 如果您没有使用正确的订阅：

   a. 运行以下命令来更改活跃订阅：

   ```
   $ az account set -s <subscription_id>
   ```

   b. 运行以下命令验证您是否正在使用所需的订阅：

   ```
   $ az account show
   ```

   输出示例

   ```
   {
     "environmentName": "AzureCloud",
     "id": "9xxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
     "isDefault": true,
     "name": "Subscription Name 2",
     "state": "Enabled",
     "tenantId": "7xxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
     "user": {
       "name": "you2@example.com",
       "type": "user"
     }
   }
   ```

4. 记录输出中的 **id** 和 **tenantId** 参数值。您需要这些值来安装 OpenShift Container Platform 集群。

7.2.5. 支持访问 Azure 资源的身份

OpenShift Container Platform 集群需要一个 Azure 身份来创建和管理 Azure 资源。因此，您需要以下身份之一来完成安装：

- 服务主体
- 系统分配的管理身份
- 用户分配的受管身份

7.2.5.1. 所需的 Azure 角色

OpenShift Container Platform 集群需要一个 Azure 身份来创建和管理 Azure 资源。在创建身份前，请验证您的环境是否满足以下要求：

- 用于创建身份的 Azure 帐户被分配 **User Access Administrator** 和 **Contributor** 角色。在以下情况下需要这些角色：
  - 创建服务主体或用户分配的受管身份。
  - 在虚拟机上启用系统分配的受管身份。

- 如果您要使用服务主体完成安装，请验证您用来创建身份的 Azure 帐户是否已被分配了 Microsoft Entra ID 中的 **microsoft.directory/servicePrincipals/createAsOwner** 权限。

要在 Azure 门户上设置角色，请参阅 Azure 文档中的使用 RBAC 和 Azure 门户管理对 Azure 资源的访问。

7.2.5.2. 安装程序置备的基础架构所需的 Azure 权限

安装程序需要访问具有所需权限的 Azure 服务主体或受管身份，以部署集群并维护其每日操作。这些权限必须授予与身份关联的 Azure 订阅。

以下选项可供您使用：

- 您可以为身份分配 **Contributor** 和 **User Access Administrator** 角色。分配这些角色是授予所有所需权限的最快速方法。
  有关分配角色的更多信息，请参阅 Azure 文档，使用 Azure 门户管理 Azure 资源的访问。
- 如果机构的安全策略需要更严格的权限集，您可以创建具有所需权限的自定义角色。

在 Microsoft Azure 上创建 OpenShift Container Platform 集群需要以下权限。

例 7.1. 创建授权资源所需的权限

- **Microsoft.Authorization/policies/audit/action**
- **Microsoft.Authorization/policies/auditIfNotExists/action**
- **Microsoft.Authorization/roleAssignments/read**
- **Microsoft.Authorization/roleAssignments/write**

例 7.2. 创建计算资源所需的权限

- **Microsoft.Compute/availabilitySets/read**
- **Microsoft.Compute/availabilitySets/write**
- **Microsoft.Compute/disks/beginGetAccess/action**
- **Microsoft.Compute/disks/delete**
- **Microsoft.Compute/disks/read**
- **Microsoft.Compute/disks/write**
- **Microsoft.Compute/galleries/images/read**
- **Microsoft.Compute/galleries/images/versions/read**
- **Microsoft.Compute/galleries/images/versions/write**
- **Microsoft.Compute/galleries/images/write**
- **Microsoft.Compute/galleries/read**
- **Microsoft.Compute/galleries/write**

- **Microsoft.Compute/snapshots/read**
- **Microsoft.Compute/snapshots/write**
- **Microsoft.Compute/snapshots/delete**
- **Microsoft.Compute/virtualMachines/delete**
- **Microsoft.Compute/virtualMachines/powerOff/action**
- **Microsoft.Compute/virtualMachines/read**
- **Microsoft.Compute/virtualMachines/write**

例 7.3. 创建身份管理资源所需的权限

- **Microsoft.ManagedIdentity/userAssignedIdentities/assign/action**
- **Microsoft.ManagedIdentity/userAssignedIdentities/read**
- **Microsoft.ManagedIdentity/userAssignedIdentities/write**

例 7.4. 创建网络资源所需的权限

- **Microsoft.Network/dnsZones/A/write**
- **Microsoft.Network/dnsZones/CNAME/write**
- **Microsoft.Network/dnszones/CNAME/read**
- **Microsoft.Network/dnszones/read**
- **Microsoft.Network/loadBalancers/backendAddressPools/join/action**
- **Microsoft.Network/loadBalancers/backendAddressPools/read**
- **Microsoft.Network/loadBalancers/backendAddressPools/write**
- **Microsoft.Network/loadBalancers/read**
- **Microsoft.Network/loadBalancers/write**
- **Microsoft.Network/networkInterfaces/delete**
- **Microsoft.Network/networkInterfaces/join/action**
- **Microsoft.Network/networkInterfaces/read**
- **Microsoft.Network/networkInterfaces/write**
- **Microsoft.Network/networkSecurityGroups/join/action**
- **Microsoft.Network/networkSecurityGroups/read**
- **Microsoft.Network/networkSecurityGroups/securityRules/delete**

- **Microsoft.Network/networkSecurityGroups/securityRules/read**
- **Microsoft.Network/networkSecurityGroups/securityRules/write**
- **Microsoft.Network/networkSecurityGroups/write**
- **Microsoft.Network/privateDnsZones/A/read**
- **Microsoft.Network/privateDnsZones/A/write**
- **Microsoft.Network/privateDnsZones/A/delete**
- **Microsoft.Network/privateDnsZones/SOA/read**
- **Microsoft.Network/privateDnsZones/read**
- **Microsoft.Network/privateDnsZones/virtualNetworkLinks/read**
- **Microsoft.Network/privateDnsZones/virtualNetworkLinks/write**
- **Microsoft.Network/privateDnsZones/write**
- **Microsoft.Network/publicIPAddresses/delete**
- **Microsoft.Network/publicIPAddresses/join/action**
- **Microsoft.Network/publicIPAddresses/read**
- **Microsoft.Network/publicIPAddresses/write**
- **Microsoft.Network/virtualNetworks/join/action**
- **Microsoft.Network/virtualNetworks/read**
- **Microsoft.Network/virtualNetworks/subnets/join/action**
- **Microsoft.Network/virtualNetworks/subnets/read**
- **Microsoft.Network/virtualNetworks/subnets/write**
- **Microsoft.Network/virtualNetworks/write**

注意

在 Azure 上创建私有 OpenShift Container Platform 集群不需要以下权限。

- **Microsoft.Network/dnsZones/A/write**
- **Microsoft.Network/dnsZones/CNAME/write**
- **Microsoft.Network/dnszones/CNAME/read**
- **Microsoft.Network/dnszones/read**

例 7.5. 检查资源健康状况所需的权限

- **Microsoft.Resourcehealth/healthevent/Activated/action**
- **Microsoft.Resourcehealth/healthevent/InProgress/action**
- **Microsoft.Resourcehealth/healthevent/Pending/action**
- **Microsoft.Resourcehealth/healthevent/Resolved/action**
- **Microsoft.Resourcehealth/healthevent/Updated/action**

例 7.6. 创建资源组所需的权限

- **Microsoft.Resources/subscriptions/resourceGroups/read**
- **Microsoft.Resources/subscriptions/resourcegroups/write**

例 7.7. 创建资源标签所需的权限

- **Microsoft.Resources/tags/write**

例 7.8. 创建存储资源所需的权限

- **Microsoft.Storage/storageAccounts/blobServices/read**
- **Microsoft.Storage/storageAccounts/blobServices/containers/write**
- **Microsoft.Storage/storageAccounts/fileServices/read**
- **Microsoft.Storage/storageAccounts/fileServices/shares/read**
- **Microsoft.Storage/storageAccounts/fileServices/shares/write**
- **Microsoft.Storage/storageAccounts/fileServices/shares/delete**
- **Microsoft.Storage/storageAccounts/listKeys/action**
- **Microsoft.Storage/storageAccounts/read**
- **Microsoft.Storage/storageAccounts/write**

例 7.9. 为镜像 registry 创建私有存储端点的可选权限

- **Microsoft.Network/privateEndpoints/write**
- **Microsoft.Network/privateEndpoints/read**
- **Microsoft.Network/privateEndpoints/privateDnsZoneGroups/write**
- **Microsoft.Network/privateEndpoints/privateDnsZoneGroups/read**
- **Microsoft.Network/privateDnsZones/join/action**

- **Microsoft.Storage/storageAccounts/PrivateEndpointConnectionsApproval/action**

例 7.10. 创建 marketplace 虚拟机资源的可选权限

- **Microsoft.MarketplaceOrdering/offertypes/publishers/offers/plans/agreements/read**
- **Microsoft.MarketplaceOrdering/offertypes/publishers/offers/plans/agreements/write**

例 7.11. 创建计算资源的可选权限

- **Microsoft.Compute/availabilitySets/delete**
- **Microsoft.Compute/images/read**
- **Microsoft.Compute/images/write**
- **Microsoft.Compute/images/delete**

例 7.12. 启用用户管理加密的可选权限

- **Microsoft.Compute/diskEncryptionSets/read**
- **Microsoft.Compute/diskEncryptionSets/write**
- **Microsoft.Compute/diskEncryptionSets/delete**
- **Microsoft.KeyVault/vaults/read**
- **Microsoft.KeyVault/vaults/write**
- **Microsoft.KeyVault/vaults/delete**
- **Microsoft.KeyVault/vaults/deploy/action**
- **Microsoft.KeyVault/vaults/keys/read**
- **Microsoft.KeyVault/vaults/keys/write**
- **Microsoft.Features/providers/features/register/action**

例 7.13. 使用 NatGateway 出站类型安装集群的可选权限

- **Microsoft.Network/natGateways/read**
- **Microsoft.Network/natGateways/write**

例 7.14. 使用 Azure 网络地址转换 (NAT) 安装私有集群的可选权限

- **Microsoft.Network/natGateways/join/action**

- **Microsoft.Network/natGateways/read**
- **Microsoft.Network/natGateways/write**

例 7.15. 使用 Azure 防火墙安装私有集群的可选权限

- **Microsoft.Network/azureFirewalls/applicationRuleCollections/write**
- **Microsoft.Network/azureFirewalls/read**
- **Microsoft.Network/azureFirewalls/write**
- **Microsoft.Network/routeTables/join/action**
- **Microsoft.Network/routeTables/read**
- **Microsoft.Network/routeTables/routes/read**
- **Microsoft.Network/routeTables/routes/write**
- **Microsoft.Network/routeTables/write**
- **Microsoft.Network/virtualNetworks/peer/action**
- **Microsoft.Network/virtualNetworks/virtualNetworkPeerings/read**
- **Microsoft.Network/virtualNetworks/virtualNetworkPeerings/write**

例 7.16. 运行收集 bootstrap 的可选权限

- **Microsoft.Compute/virtualMachines/instanceView/read**

删除 Microsoft Azure 上的 OpenShift Container Platform 集群需要以下权限。您可以使用相同的权限删除 Azure 上的私有 OpenShift Container Platform 集群。

例 7.17. 删除授权资源所需的权限

- **Microsoft.Authorization/roleAssignments/delete**

例 7.18. 删除计算资源所需的权限

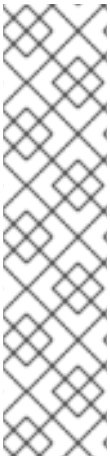
- **Microsoft.Compute/disks/delete**
- **Microsoft.Compute/galleries/delete**
- **Microsoft.Compute/galleries/images/delete**
- **Microsoft.Compute/galleries/images/versions/delete**
- **Microsoft.Compute/virtualMachines/delete**

例 7.19. 删除身份管理资源所需的权限

- **Microsoft.ManagedIdentity/userAssignedIdentities/delete**

例 7.20. 删除网络资源所需的权限

- **Microsoft.Network/dnszones/read**
- **Microsoft.Network/dnsZones/A/read**
- **Microsoft.Network/dnsZones/A/delete**
- **Microsoft.Network/dnsZones/CNAME/read**
- **Microsoft.Network/dnsZones/CNAME/delete**
- **Microsoft.Network/loadBalancers/delete**
- **Microsoft.Network/networkInterfaces/delete**
- **Microsoft.Network/networkSecurityGroups/delete**
- **Microsoft.Network/privateDnsZones/read**
- **Microsoft.Network/privateDnsZones/A/read**
- **Microsoft.Network/privateDnsZones/delete**
- **Microsoft.Network/privateDnsZones/virtualNetworkLinks/delete**
- **Microsoft.Network/publicIPAddresses/delete**
- **Microsoft.Network/virtualNetworks/delete**



注意

在 Azure 上删除私有 OpenShift Container Platform 集群不需要以下权限。

- **Microsoft.Network/dnszones/read**
- **Microsoft.Network/dnsZones/A/read**
- **Microsoft.Network/dnsZones/A/delete**
- **Microsoft.Network/dnsZones/CNAME/read**
- **Microsoft.Network/dnsZones/CNAME/delete**

例 7.21. 检查资源健康状况所需的权限

- **Microsoft.Resourcehealth/healthevent/Activated/action**
- **Microsoft.Resourcehealth/healthevent/Resolved/action**

- **Microsoft.Resourcehealth/healthevent/Updated/action**

例 7.22. 删除资源组所需的权限

- **Microsoft.Resources/subscriptions/resourcegroups/delete**

例 7.23. 删除存储资源所需的权限

- **Microsoft.Storage/storageAccounts/delete**
- **Microsoft.Storage/storageAccounts/listKeys/action**

注意

要在 Azure 上安装 OpenShift Container Platform，您必须将权限范围到您的订阅。之后，您可以将这些权限重新限定到安装程序创建的资源组。如果其他资源组中存在公共 DNS 区域，则必须始终将网络 DNS 区域相关权限应用到您的订阅。默认情况下，OpenShift Container Platform 安装程序分配 **Contributor** 角色的 Azure 身份。

在删除 OpenShift Container Platform 集群时，您可以将订阅的所有权限限定到您的订阅。

7.2.5.3. 使用 Azure 管理的身份

安装程序需要一个 Azure 身份来完成安装。您可以使用系统分配或用户分配的受管身份。

如果无法使用受管身份，您可以使用服务主体。

流程

1. 如果您使用系统分配的受管身份，请在您要从其运行安装程序的虚拟机上启用它。
2. 如果您使用用户分配的受管身份：
   a. 将它分配给您要从中运行安装程序的虚拟机。
   b. 记录其客户端 ID。安装集群时需要这个值。
   有关查看用户分配受管身份的详情的更多信息，请参阅 Microsoft Azure 文档来列出用户分配的管理身份。
3. 验证是否为受管身份分配了所需的权限。

7.2.5.4. 创建服务主体

安装程序需要一个 Azure 身份来完成安装。您可以使用服务主体。

如果无法使用服务主体，您可以使用受管身份。

先决条件

- 已安装或更新 Azure CLI。

- 您有一个 Azure 订阅 ID。
- 如果您没有将 **Contributor** 和 **User Administrator Access** 角色分配给服务主体，您已创建了具有所需 Azure 权限的自定义角色。

流程

1. 运行以下命令，为您的帐户创建服务主体：

```
$ az ad sp create-for-rbac --role <role_name> \ 1
    --name <service_principal> \ 2
    --scopes /subscriptions/<subscription_id> 3
```

1. 定义角色名称。您可以使用 **Contributor** 角色，或者指定包含所需权限的自定义角色。
2. 定义服务主体名称。
3. 指定订阅 ID。

输出示例

```
Creating 'Contributor' role assignment under scope '/subscriptions/<subscription_id>'
The output includes credentials that you must protect. Be sure that you do not
include these credentials in your code or check the credentials into your source
control. For more information, see https://aka.ms/azadsp-cli
{
  "appId": "axxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
  "displayName": <service_principal>",
  "password": "00000000-0000-0000-0000-000000000000",
  "tenantId": "8xxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx"
}
```

2. 记录输出中的 **appId** 和 **password** 参数的值。安装集群时需要这些值。
3. 如果您将 **Contributor** 角色应用到服务主体，请运行以下命令来分配 **User Administrator Access** 角色：

```
$ az role assignment create --role "User Access Administrator" \
  --assignee-object-id $(az ad sp show --id <appId> --query id -o tsv) 1
  --scope /subscriptions/<subscription_id> 2
```

1. 为您的服务主体指定 **appId** 参数值。
2. 指定订阅 ID。

其他资源

- 关于 Cloud Credential Operator

7.2.6. 支持的 Azure Marketplace 区域

使用 Azure Marketplace 镜像安装集群可供购买在北美和 EMEA 中提供的客户提供服务。

虽然优惠必须在北美或 EMEA 处购买，但您可以将集群部署到 OpenShift Container Platform 支持的 Azure 公共分区中。

注意

Azure Government 区域不支持使用 Azure Marketplace 镜像部署集群。

7.2.7. 支持的 Azure 区域

安装程序会根据您的订阅动态地生成可用的 Microsoft Azure 区域列表。

支持的 Azure 公共区域

- **australiacentral** (Australia Central)
- **australiaeast** (Australia East)
- **australiasoutheast** (Australia South East)
- **brazilsouth** (Brazil South)
- **canadacentral** (Canada Central)
- **canadaeast** (Canada East)
- **centralindia** (Central India)
- **centralus** (Central US)
- **eastasia** (East Asia)
- **eastus** (East US)
- **eastus2** (East US 2)
- **francecentral** (France Central)
- **germanywestcentral** (Germany West Central)
- **israelcentral** (Israel Central)
- **italynorth** (Italy North)
- **japaneast** (Japan East)
- **japanwest** (Japan West)
- **koreacentral** (Korea Central)
- **koreasouth** (Korea South)
- **mexicocentral** (Mexico Central)
- **northcentralus** (North Central US)
- **northeurope** (North Europe)

- **norwayeast** (Norway East)
- **polandcentral** (Poland Central)
- **qatarcentral** (Qatar Central)
- **southafricanorth** (South Africa North)
- **southcentralus** (South Central US)
- **southeastasia** (Southeast Asia)
- **southindia** (South India)
- **swedencentral** (Sweden Central)
- **switzerlandnorth** (Switzerland North)
- **uaenorth** (UAE North)
- **uksouth** (UK South)
- **ukwest** (UK West)
- **westcentralus** (West Central US)
- **westeurope** (West Europe)
- **westindia** (West India)
- **westus** (West US)
- **westus2** (West US 2)
- **westus3** (West US 3)

支持的 Azure 政府区域
OpenShift Container Platform 4.6 添加了对以下 Microsoft Azure Government（MAG）区域的支持：

- **usgovtexas** (US Gov Texas)
- **usgovvirginia** (US Gov Virginia)

您可以参阅 Azure 文档来了解与所有可用 MAG 区域的信息。其他 MAG 区域应该可以与 OpenShift Container Platform 一起工作，但并没有经过测试。

7.2.8. 后续步骤

- 在 Azure 上安装 OpenShift Container Platform 集群。您可以 安装自定义集群，或使用默认选项快速安装集群。

7.3. 为 AZURE 启用用户管理的加密

在 OpenShift Container Platform 版本 4.16 中，您可以在 Azure 中使用用户管理的加密密钥安装集群。要启用此功能，您可以在安装前准备 Azure DiskEncryptionSet，修改 **install-config.yaml** 文件，然后完成安装。

7.3.1. 准备 Azure 磁盘加密集

OpenShift Container Platform 安装程序可以使用带有用户管理的密钥的现有磁盘加密集。要启用此功能，您可以在 Azure 中创建磁盘加密集，并为安装程序提供密钥。

流程

1. 运行以下命令，为 Azure 资源组设置以下环境变量：

```
$ export RESOURCEGROUP="<resource_group>" \ 1
  LOCATION="<location>" 2
```

1. 指定您要创建磁盘加密集和加密密钥的 Azure 资源组名称。为了避免在销毁集群后丢失对密钥的访问，您应该在与安装集群的资源组不同的资源组中创建 Disk Encryption Set。
2. 指定您要创建资源组的 Azure 位置。

2. 运行以下命令，为 Azure Key Vault 和 Disk Encryption Set 设置以下环境变量：

```
$ export KEYVAULT_NAME="<keyvault_name>" \ 1
  KEYVAULT_KEY_NAME="<keyvault_key_name>" \ 2
  DISK_ENCRYPTION_SET_NAME="<disk_encryption_set_name>" 3
```

1. 指定您要创建的 Azure Key Vault 的名称。
2. 指定您要创建的加密密钥名称。
3. 指定您要创建的磁盘加密集的名称。

3. 运行以下命令，为您的 Azure Service Principal 设置环境变量：

```
$ export CLUSTER_SP_ID="<service_principal_id>" 1
```

1. 指定用于此安装的服务主体的 ID。

4. 运行以下命令，在 Azure 中启用主机级别加密：

```
$ az feature register --namespace "Microsoft.Compute" --name "EncryptionAtHost"
```

```
$ az feature show --namespace Microsoft.Compute --name EncryptionAtHost
```

```
$ az provider register -n Microsoft.Compute
```

5. 运行以下命令，创建一个 Azure 资源组来保存磁盘加密集和相关资源：

```
$ az group create --name $RESOURCEGROUP --location $LOCATION
```

6. 运行以下命令来创建 Azure 密钥库：

```
$ az keyvault create -n $KEYVAULT_NAME -g $RESOURCEGROUP -l $LOCATION \
  --enable-purge-protection true
```

7. 运行以下命令，在密钥 vault 中创建加密密钥：

```
$ az keyvault key create --vault-name $KEYVAULT_NAME -n $KEYVAULT_KEY_NAME \
  --protection software
```

8. 运行以下命令捕获密钥 vault 的 ID：

```
$ KEYVAULT_ID=$(az keyvault show --name $KEYVAULT_NAME --query "[id]" -o tsv)
```

9. 运行以下命令，捕获密钥 vault 中的密钥 URL：

```
$ KEYVAULT_KEY_URL=$(az keyvault key show --vault-name $KEYVAULT_NAME --name
\
  $KEYVAULT_KEY_NAME --query "[key.kid]" -o tsv)
```

10. 运行以下命令来创建磁盘加密集：

```
$ az disk-encryption-set create -n $DISK_ENCRYPTION_SET_NAME -l $LOCATION -g \
  $RESOURCEGROUP --source-vault $KEYVAULT_ID --key-url $KEYVAULT_KEY_URL
```

11. 运行以下命令，为 DiskEncryptionSet 资源授予对密钥 vault 的访问权限：

```
$ DES_IDENTITY=$(az disk-encryption-set show -n $DISK_ENCRYPTION_SET_NAME -g \
  $RESOURCEGROUP --query "[identity.principalId]" -o tsv)
```

```
$ az keyvault set-policy -n $KEYVAULT_NAME -g $RESOURCEGROUP --object-id \
  $DES_IDENTITY --key-permissions wrapkey unwrapkey get
```

12. 运行以下命令，授予 Azure Service Principal 权限来读取 DiskEncryptionSet：

```
$ DES_RESOURCE_ID=$(az disk-encryption-set show -n
$DISK_ENCRYPTION_SET_NAME -g \
  $RESOURCEGROUP --query "[id]" -o tsv)
```

```
$ az role assignment create --assignee $CLUSTER_SP_ID --role "<reader_role>" \ 1
  --scope $DES_RESOURCE_ID -o jsonc
```

1 指定对磁盘加密集的读取权限的 Azure 角色。您可以使用 **Owner** 角色或具有所需权限的自定义角色。

7.3.2. 后续步骤

- 安装 OpenShift Container Platform 集群：
  - 使用自定义在安装程序置备的基础架构中安装集群
  - 在安装程序置备的基础架构中使用网络自定义安装集群

- 在安装程序置备的基础架构上将集群安装到现有 VNet 中
- 在安装程序置备的基础架构上安装私有集群
- 在安装程序置备的基础架构上将集群安装到一个政府区域

7.4. 在 AZURE 上快速安装集群

在 OpenShift Container Platform 版本 4.16 中，您可以使用默认配置选项在 Microsoft Azure 上安装集群。

7.4.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读选择集群安装方法并为用户准备它的文档。
- 您已将 Azure 帐户配置为托管集群，并决定要将集群部署到的已测试和验证的区域。
- 如果使用防火墙，将其配置为允许集群需要访问的站点。

7.4.2. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类型的基础架构上执行受限网络安装。在此过程中，您可以下载所需的内容，并使用它为镜像 registry 填充安装软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群前，您要更新镜像 registry 的内容。

7.4.3. 为集群节点 SSH 访问生成密钥对

在 OpenShift Container Platform 安装过程中，您可以为安装程序提供 SSH 公钥。密钥通过它们的 Ignition 配置文件传递给 Red Hat Enterprise Linux CoreOS(RHCOS)节点，用于验证对节点的 SSH 访问。密钥添加到每个节点上 **core** 用户的 **~/.ssh/authorized_keys** 列表中，这将启用免密码身份验证。

将密钥传递给节点后，您可以使用密钥对作为用户 **核心** 通过 SSH 连接到 RHCOS 节点。若要通过 SSH 访问节点，必须由 SSH 为您的本地用户管理私钥身份。

如果要通过 SSH 连接到集群节点来执行安装调试或灾难恢复，则必须在安装过程中提供 SSH 公钥。**./openshift-install gather** 命令还需要在集群节点上设置 SSH 公钥。

重要

不要在生产环境中跳过这个过程，在生产环境中需要灾难恢复和调试。

注意

您必须使用本地密钥，而不是使用特定平台方法配置 的密钥，如 AWS 密钥对。

流程

1. 如果您在本地计算机上没有可用于在集群节点上进行身份验证的现有 SSH 密钥对，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 ~/**.ssh/id_ed25519**。如果您已有密钥对，请确保您的公钥位于 ~/**.ssh** 目录中。

注意

如果您计划在 **x86_64**、**ppc64le** 和 **s390x** 架构上安装使用 RHEL 加密库（这些加密库已提交给 NIST 用于 FIPS 140-2/140-3 验证）的 OpenShift Container Platform 集群，则不要创建使用 **ed25519** 算法的密钥。相反，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 查看公共 SSH 密钥：

```
$ cat <path>/<file_name>.pub
```

例如，运行以下命令来查看 ~/**.ssh/id_ed25519.pub** 公钥：

```
$ cat ~/.ssh/id_ed25519.pub
```

3. 将 SSH 私钥身份添加到本地用户的 SSH 代理（如果尚未添加）。在集群节点上，或者要使用 **./openshift-install gather** 命令，需要对该密钥进行 SSH 代理管理，才能在集群节点上进行免密码 SSH 身份验证。

注意

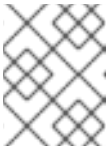
在某些发行版中，自动管理默认 SSH 私钥身份，如 ~/**.ssh/id_rsa** 和 ~/**.ssh/id_dsa**。

a. 如果 **ssh-agent** 进程尚未为您的本地用户运行，请将其作为后台任务启动：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```



注意

如果集群处于 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

4. 将 SSH 私钥添加到 **ssh-agent** ：

```
$ ssh-add <path>/<file_name> 1
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_ed25519.pub**

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

后续步骤

- 安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

7.4.4. 获取安装程序

在安装 OpenShift Container Platform 前，将安装文件下载到您用于安装的主机上。

先决条件

- 您有一台运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB。

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐户，请使用您的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入到安装类型的页面，下载与您的主机操作系统和架构对应的安装程序，并将该文件放在您要存储安装配置文件的目录中。

重要

安装程序会在用来安装集群的计算机上创建几个文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。

重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，请为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。此 pull secret 允许您与所含授权机构提供的服务进行身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

7.4.5. 部署集群

您可以在兼容云平台上安装 OpenShift Container Platform。

重要

在初始安装过程中，您只能运行安装程序的 **create cluster** 命令一次。

先决条件

- 您已使用托管集群的云平台配置了帐户。
- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。
- 您有一个 Azure 订阅 ID 和租户 ID。
- 您有服务主体的应用程序 ID 和密码。

流程

1. 可选： 如果您在此计算机上运行安装程序，并希望使用替代服务主体，请转至 **~/.azure/** 目录并删除 **osServicePrincipal.json** 配置文件。
   删除此文件可防止安装程序自动重复使用之前安装中的订阅和验证值。

2. 进入包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定要存储安装程序创建的文件的目录名称。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不是 **info**。

在指定目录时：

- 验证该目录是否具有**执行**权限。在安装目录中运行 Terraform 二进制文件需要这个权限。
- 使用空目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

3. 在提示符处提供值：

   a. 可选： 选择用于访问集群机器的 SSH 密钥。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

b. 选择 **azure** 作为目标平台。
如果安装程序无法找到之前安装中的 **osServicePrincipal.json** 配置文件，会提示您输入 Azure 订阅和验证值。

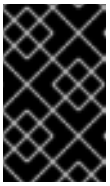
c. 为您的订阅和服务主体指定以下 Azure 参数值：

- **Azure subscription id**：输入用于集群的订阅 ID。
- **Azure 租户 id**：输入租户 ID。
- **Azure 服务主体客户端 id**：输入其应用程序 ID。
- **Azure 服务主体客户端 secret**：输入其密码。

d. 选择要将集群部署到的区域。

e. 选择集群要部署到的基域。基域与您为集群创建的 Azure DNS 区对应。

f. 为集群输入一个描述性名称。



重要

所有通过公共端点提供的 Azure 资源均存在资源名称的限制，您无法创建使用某些名称的资源。如需 Azure 限制词语列表，请参阅 Azure 文档中的解决保留资源名称错误。

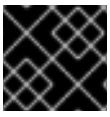
g. 粘贴 Red Hat OpenShift Cluster Manager 中的 pull secret 。

在以前的版本中，安装程序会创建一个 **osServicePrincipal.json** 配置文件，并将此文件存储在计算机上的 **~/.azure/** 目录中。这样可确保安装程序在目标平台上创建 OpenShift Container Platform 集群时可以加载配置集。

验证

当集群部署成功完成时：

- 终端会显示用于访问集群的说明，包括指向 Web 控制台和 **kubeadmin** 用户的凭证的链接。
- 凭证信息还会输出到 **<installation_directory>/.openshift_install.log**.

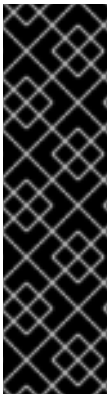


重要

不要删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```

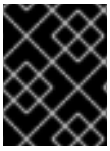


重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

7.4.6. 安装 OpenShift CLI

您可以安装 OpenShift CLI(**oc**)来使用命令行界面与 OpenShift Container Platform 进行交互。您可以在 Linux、Windows 或 macOS 上安装 **oc**。



重要

如果安装了旧版本的 **oc**，则可能无法使用 OpenShift Container Platform 4.16 中的所有命令。下载并安装新版本的 **oc**。

在 Linux 上安装 OpenShift CLI
您可以按照以下流程在 Linux 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **产品变体** 下拉列表中选择架构。
3. 从 **版本** 下拉列表中选择适当的版本。
4. 点 **OpenShift v4.16 Linux Client** 条目旁的 **Download Now** 来保存文件。
5. 解包存档：

```
$ tar xvf <file>
```

6. 将 **oc** 二进制文件放到 **PATH 中的目录中**。
   要查看您的 **PATH**，请执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

在 Windows 上安装 OpenShift CLI
您可以按照以下流程在 Windows 上安装 OpenShift CLI(**oc**)二进制文件。

流程

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 Windows Client** 条目旁的 **Download Now** 来保存文件。
4. 使用 ZIP 程序解压存档。
5. 将 **oc** 二进制文件移到 **PATH 中的目录中**。
   要查看您的 **PATH**，请打开命令提示并执行以下命令：

```
C:\> path
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

在 macOS 上安装 OpenShift CLI
您可以按照以下流程在 macOS 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 macOS Client** 条目旁的 **Download Now** 来保存文件。

注意

对于 macOS arm64，请选择 **OpenShift v4.16 macOS arm64 Client** 条目。

4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
   要查看您的 **PATH**，请打开终端并执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

7.4.7. 使用 CLI 登录集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

其他资源

- 如需有关 访问和了解 OpenShift Container Platform Web 控制台的更多详情，请参阅 访问 Web 控制台。

7.4.8. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于 远程健康监控

7.4.9. 后续步骤

- 自定义集群。
- 如果需要，您可以选择不使用远程健康报告。

7.5. 使用自定义在 AZURE 上安装集群

在 OpenShift Container Platform 版本 4.16 中，您可以在安装程序在 Microsoft Azure 上置备的基础架构上安装自定义的集群。要自定义安装，请在安装集群前修改 **install-config.yaml** 文件中的参数。

7.5.1. 先决条件

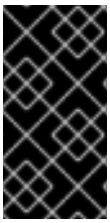
- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读选择集群安装方法并为用户准备它的文档。
- 您已将 Azure 帐户配置为托管集群，并决定要将集群部署到的已测试和验证的区域。
- 如果使用防火墙，将其配置为允许集群需要访问的站点。
- 如果您使用客户管理的加密密钥，准备了用于加密的 Azure 环境。

7.5.2. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。



重要

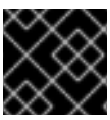
如果您的集群无法直接访问互联网，则可以在置备的某些类型的基础架构上执行受限网络安装。在此过程中，您可以下载所需的内容，并使用它为镜像 registry 填充安装软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群前，您要更新镜像 registry 的内容。

7.5.3. 为集群节点 SSH 访问生成密钥对

在 OpenShift Container Platform 安装过程中，您可以为安装程序提供 SSH 公钥。密钥通过它们的 Ignition 配置文件传递给 Red Hat Enterprise Linux CoreOS(RHCOS)节点，用于验证对节点的 SSH 访问。密钥添加到每个节点上 **core** 用户的 **~/.ssh/authorized_keys** 列表中，这将启用免密码身份验证。

将密钥传递给节点后，您可以使用密钥对作为用户 **核心** 通过 SSH 连接到 RHCOS 节点。若要通过 SSH 访问节点，必须由 SSH 为您的本地用户管理私钥身份。

如果要通过 SSH 连接到集群节点来执行安装调试或灾难恢复，则必须在安装过程中提供 SSH 公钥。**./openshift-install gather** 命令还需要在集群节点上设置 SSH 公钥。



重要

不要在生产环境中跳过这个过程，在生产环境中需要灾难恢复和调试。

注意

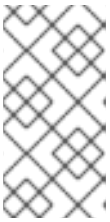
您必须使用本地密钥，而不是使用特定平台方法配置 的密钥，如 AWS 密钥对。

流程

1. 如果您在本地计算机上没有可用于在集群节点上进行身份验证的现有 SSH 密钥对，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 ~/**.ssh/id_ed25519**。如果您已有密钥对，请确保您的公钥位于 ~/**.ssh** 目录中。



注意

如果您计划在 **x86_64**、**ppc64le** 和 **s390x** 架构上安装使用 RHEL 加密库（这些加密库已提交给 NIST 用于 FIPS 140-2/140-3 验证）的 OpenShift Container Platform 集群，则不要创建使用 **ed25519** 算法的密钥。相反，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

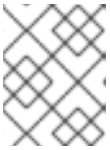
2. 查看公共 SSH 密钥：

```
$ cat <path>/<file_name>.pub
```

例如，运行以下命令来查看 ~/**.ssh/id_ed25519.pub** 公钥：

```
$ cat ~/.ssh/id_ed25519.pub
```

3. 将 SSH 私钥身份添加到本地用户的 SSH 代理（如果尚未添加）。在集群节点上，或者要使用 **./openshift-install gather** 命令，需要对该密钥进行 SSH 代理管理，才能在集群节点上进行免密码 SSH 身份验证。



注意

在某些发行版中，自动管理默认 SSH 私钥身份，如 ~/**.ssh/id_rsa** 和 ~/**.ssh/id_dsa**。

a. 如果 **ssh-agent** 进程尚未为您的本地用户运行，请将其作为后台任务启动：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果集群处于 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

4. 将 SSH 私钥添加到 **ssh-agent** ：

```
$ ssh-add <path>/<file_name> 1
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_ed25519.pub**

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

后续步骤

- 安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

7.5.4. 使用 Azure Marketplace 产品

使用 Azure Marketplace 产品可让您部署 OpenShift Container Platform 集群，该集群按照使用付费（按小时、每个内核）进行计费，同时仍由红帽直接支持。

要使用 Azure Marketplace 产品部署 OpenShift Container Platform 集群，您必须首先获取 Azure Marketplace 镜像。安装程序使用这个镜像来部署 worker 或 control plane 节点。在获取您的镜像时，请考虑以下事项：

- 虽然镜像相同，但 Azure Marketplace publisher 根据您的区域。如果您位于北美，请将 **redhat** 指定为发布者。如果您位于 EMEA，请将 **redhat-limited** 指定为发布者。
- 此项优惠包括 **rh-ocp-worker** SKU 和 **rh-ocp-worker-gen1** SKU。**rh-ocp-worker** SKU 代表 Hyper-V 生成版本 2 虚拟机镜像。OpenShift Container Platform 中使用的默认实例类型与版本 2 兼容。如果您计划使用与版本 1 兼容的实例类型，请使用与 **rh-ocp-worker-gen1** SKU 关联的镜像。**rh-ocp-worker-gen1** SKU 代表 Hyper-V 版本 1 虚拟机镜像。

重要

在使用 64 位 ARM 实例的集群上不支持使用 Azure marketplace 安装镜像。

先决条件

- 已安装 Azure CLI 客户端 **(az)**。
- 您的 Azure 帐户为产品授权，您使用 Azure CLI 客户端登录到此帐户。

流程

1. 运行以下命令之一，显示所有可用的 OpenShift Container Platform 镜像：
   - 北美：

     ```
     $ az vm image list --all --offer rh-ocp-worker --publisher redhat -o table
     ```

     输出示例

     ```
     Offer          Publisher      Sku              Urn                                                          Version
     -------------  -------------  ---------------  -----------------------------------------------------------  -----------------
     rh-ocp-worker  RedHat         rh-ocp-worker    RedHat:rh-ocp-worker:rh-ocp-
     ```

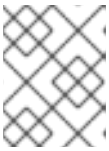
```
worker:413.92.2023101700           413.92.2023101700
rh-ocp-worker  RedHat          rh-ocp-worker-gen1  RedHat:rh-ocp-worker:rh-ocp-worker-
gen1:413.92.2023101700       413.92.2023101700
```

- 欧洲、中东和非洲地区：

```
$  az vm image list --all --offer rh-ocp-worker --publisher redhat-limited -o table
```

输出示例

```
Offer          Publisher       Sku                 Urn
Version
-------------  --------------  ------------------  ------------------------------------------------------------
----           -----------------
rh-ocp-worker  redhat-limited  rh-ocp-worker       redhat-limited:rh-ocp-worker:rh-ocp-
worker:413.92.2023101700           413.92.2023101700
rh-ocp-worker  redhat-limited  rh-ocp-worker-gen1  redhat-limited:rh-ocp-worker:rh-ocp-
worker-gen1:413.92.2023101700       413.92.2023101700
```



注意

使用可用于 compute 和 control plane 节点的最新镜像。如果需要，您的虚拟机会在安装过程中自动升级。

2. 运行以下命令之一检查您的所提供的镜像：

- 北美：

```
$ az vm image show --urn redhat:rh-ocp-worker:rh-ocp-worker:<version>
```

- 欧洲、中东和非洲地区：

```
$ az vm image show --urn redhat-limited:rh-ocp-worker:rh-ocp-worker:<version>
```

3. 运行以下命令之一查看提供的术语：

- 北美：

```
$ az vm image terms show --urn redhat:rh-ocp-worker:rh-ocp-worker:<version>
```

- 欧洲、中东和非洲地区：

```
$ az vm image terms show --urn redhat-limited:rh-ocp-worker:rh-ocp-worker:<version>
```

4. 运行以下命令之一接受产品条款：

- 北美：

```
$ az vm image terms accept --urn redhat:rh-ocp-worker:rh-ocp-worker:<version>
```

- 欧洲、中东和非洲地区：

```
$ az vm image terms accept --urn redhat-limited:rh-ocp-worker:rh-ocp-worker:<version>
```

5. 记录您的所提供的镜像详情。在部署集群前，您必须使用 **publisher**, **offer**, **sku**, 和 **version** 的值来更新 **install-config.yaml** 文件中的 **compute** 部分。您还可以更新 **controlPlane** 部分，以使用指定镜像详情或 **defaultMachinePlatform** 部分部署 control plane 机器，以使用指定镜像详情部署 control plane 和计算机器。将最新的可用镜像用于 control plane 和计算节点。

使用 Azure Marketplace 计算节点的 install-config.yaml 文件示例

```
apiVersion: v1
baseDomain: example.com
compute:
- hyperthreading: Enabled
  name: worker
  platform:
    azure:
      type: Standard_D4s_v5
      osImage:
        publisher: redhat
        offer: rh-ocp-worker
        sku: rh-ocp-worker
        version: 413.92.2023101700
  replicas: 3
```

7.5.5. 获取安装程序

在安装 OpenShift Container Platform 前，将安装文件下载到您用于安装的主机上。

先决条件

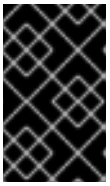
- 您有一台运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB。

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐户，请使用您的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入到安装类型的页面，下载与您的主机操作系统和架构对应的安装程序，并将该文件放在您要存储安装配置文件的目录中。

重要

安装程序会在用来安装集群的计算机上创建几个文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。



重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，请为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。此 pull secret 允许您与所含授权机构提供的服务进行身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

7.5.6. 创建安装配置文件

您可以自定义在 Microsoft Azure 上安装的 OpenShift Container Platform 集群。

先决条件

- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。
- 您有一个 Azure 订阅 ID 和租户 ID。
- 如果要使用服务主体安装集群，则有其应用程序 ID 和密码。
- 如果您要使用系统分配的受管身份安装集群，需要在您要从其中运行安装程序的虚拟机上启用它。
- 如果您要使用用户分配的受管身份安装集群，需要满足以下先决条件：
  - 您有它的客户端 ID。
  - 您已将其分配给您要从其运行安装程序的虚拟机。

流程

1. 可选：如果您之前在这个计算机上运行安装程序，并希望使用替代的服务主体或受管身份，请进入 **~/.azure/** 目录并删除 **osServicePrincipal.json** 配置文件。
   删除此文件可防止安装程序自动重复使用之前安装中的订阅和验证值。
2. 创建 **install-config.yaml** 文件。
   a. 进入包含安装程序的目录并运行以下命令：

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

   1 对于 **<installation_directory>**，请指定要存储安装程序创建的文件的目录名称。

   在指定目录时：

   - 验证该目录是否具有**执行**权限。在安装目录中运行 Terraform 二进制文件需要这个权限。
   - 使用空目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

   b. 在提示符处，提供云的配置详情：

i. 可选： 选择用于访问集群机器的 SSH 密钥。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

ii. 选择 **azure** 作为目标平台。
如果安装程序无法找到之前安装中的 **osServicePrincipal.json** 配置文件，会提示您输入 Azure 订阅和验证值。

iii. 为您的订阅输入以下 Azure 参数值：

- **Azure subscription id**：输入用于集群的订阅 ID。
- **Azure 租户 id**：输入租户 ID。

iv. 根据您用来部署集群的 Azure 身份，在提示输入 **azure 服务主体客户端 id**时执行以下操作之一：

- 如果您使用服务主体，请输入其应用程序 ID。
- 如果您使用系统分配的受管身份，请将此值设为空白。
- 如果您使用用户分配的受管身份，请指定其客户端 ID。

v. 根据您用来部署集群的 Azure 身份，在提示输入 **azure 服务主体客户端 secret**时执行以下操作之一：

- 如果您使用服务主体，请输入其密码。
- 如果您使用系统分配的受管身份，请将此值设为空白。
- 如果您使用用户分配的受管身份，请将此值设为空白。

vi. 选择要将集群部署到的区域。

vii. 选择集群要部署到的基域。基域与您为集群创建的 Azure DNS 区对应。

viii. 为集群输入一个描述性名称。

重要

所有通过公共端点提供的 Azure 资源均存在资源名称的限制，您无法创建使用某些名称的资源。如需 Azure 限制词语的列表，请参阅 Azure 文档中的解决预留资源名称错误。

3. 修改 **install-config.yaml** 文件。您可以在"安装配置参数"部分找到有关可用参数的更多信息。

注意

如果要安装三节点集群，请确保将 **compute.replicas** 参数设置为 **0**。这样可确保集群的 control plane 可以调度。如需更多信息，请参阅"在 Azure 上安装三节点集群"。

4. 备份 **install-config.yaml** 文件，以便您可以使用它安装多个集群。

重要

install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用该文件，您必须立即备份该文件。

在以前的版本中，安装程序会创建一个 **osServicePrincipal.json** 配置文件，并将此文件存储在计算机上的 **~/.azure/** 目录中。这样可确保安装程序在目标平台上创建 OpenShift Container Platform 集群时可以加载配置集。

其他资源

- Azure 的安装配置参数

7.5.6.1. 集群安装的最低资源要求

每台集群机器都必须满足以下最低要求：

表 7.1. 最低资源要求

机器	操作系统	vCPU [1]	虚拟内存	Storage	每秒输入/输出 (IOPS) [2]
bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane（控制平面）	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS、RHEL 8.6 及更新版本 [3]	2	8 GB	100 GB	300

1. 当未启用并发多线程(SMT)或超线程时，一个 vCPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比例：（每个内核数的线程）× sockets = vCPU。

2. OpenShift Container Platform 和 Kubernetes 对磁盘性能非常敏感，建议使用更快的存储速度，特别是 control plane 节点上需要 10 ms p99 fsync 持续时间的 etcd。请注意，在许多云平台上，存储大小和 IOPS 可一起扩展，因此您可能需要过度分配存储卷来获取足够的性能。

3. 与所有用户置备的安装一样，如果您选择在集群中使用 RHEL 计算机器，则负责所有操作系统生命周期管理和维护，包括执行系统更新、应用补丁和完成所有其他必要的任务。RHEL 7 计算机器的使用已弃用，并已在 OpenShift Container Platform 4.10 及更新的版本中删除。

注意

从 OpenShift Container Platform 版本 4.13 开始，RHCOS 基于 RHEL 版本 9.2，它更新了微架构要求。以下列表包含每个架构需要的最小指令集架构 (ISA)：

- x86-64 体系结构需要 x86-64-v2 ISA
- ARM64 架构需要 ARMv8.0-A ISA
- IBM Power 架构需要 Power 9 ISA
- s390x 架构需要 z14 ISA

如需更多信息，请参阅 RHEL 架构。

重要

您需要使用将 **PremiumIO** 参数设置为 **true** 的 Azure 虚拟机。

如果平台的实例类型满足集群机器的最低要求，则 OpenShift Container Platform 支持使用它。

其他资源

- 优化存储

7.5.6.2. 为 Azure 测试的实例类型

以下 Microsoft Azure 实例类型已经 OpenShift Container Platform 测试。

例 7.24. 基于 64 位 x86 架构的机器类型

- **c4.***
- **c5.***
- **c5a.***
- **i3.***
- **m4.***
- **m5.***
- **m5a.***
- **m6a.***
- **m6i.***
- **r4.***
- **r5.***
- **r5a.***
- **r6i.***

- **t3.***
- **t3a.***

7.5.6.3. 在 64 位 ARM 基础架构上为 Azure 测试的实例类型

以下 Microsoft Azure ARM64 实例类型已使用 OpenShift Container Platform 测试。

例 7.25. 基于 64 位 ARM 架构的机器类型

- **c6g.***
- **m6g.***

7.5.6.4. 为 Azure 虚拟机启用可信启动

在 Azure 上安装集群时，您可以启用两个可信启动功能： 安全引导 和虚拟化可信平台模块。

请参阅 Azure 文档中有关 虚拟机大小 的信息，以了解虚拟机支持这些功能的大小。

重要

可信启动只是一个技术预览功能。技术预览功能不受红帽产品服务等级协议（SLA）支持，且功能可能并不完整。红帽不推荐在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。

有关红帽技术预览功能支持范围的更多信息，请参阅技术预览功能支持范围。

先决条件

- 您已创建了 **install-config.yaml** 文件。

流程

- 在部署集群前，使用文本编辑器编辑 **install-config.yaml** 文件并添加以下小节：

```
controlPlane: 1
  platform:
    azure:
      settings:
        securityType: TrustedLaunch 2
        trustedLaunch:
          uefiSettings:
            secureBoot: Enabled 3
            virtualizedTrustedPlatformModule: Enabled 4
```

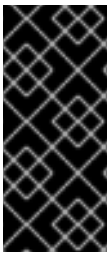
1. 指定 **controlPlane.platform.azure** 或 **compute.platform.azure**，以分别在 control plane 或计算节点上启用可信启动。指定 **platform.azure.defaultMachinePlatform**，以便在所有节点上启用可信启动。
2. 启用可信启动功能。

3 启用安全引导。如需更多信息，请参阅 Azure 文档中有关安全引导的内容。

4 启用虚拟化可信平台模块。如需更多信息，请参阅 Azure 文档中有关 虚拟化可信平台模块的文档。

7.5.6.5. 启用机密虚拟机

您可在安装集群前启用机密虚拟机。您可以为计算节点、control plane 节点或所有节点启用机密虚拟机。



重要

使用机密虚拟机只是一个技术预览功能。技术预览功能不受红帽产品服务等级协议（SLA）支持，且功能可能并不完整。红帽不推荐在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。

有关红帽技术预览功能支持范围的更多信息，请参阅技术预览功能支持范围。

您可以使用带有以下虚拟机大小的机密虚拟机：

- DCasv5-series
- DCadsv5-series
- ECasv5-series
- ECadsv5-series

重要

64 位 ARM 架构目前不支持机密虚拟机。

先决条件

- 您已创建了 **install-config.yaml** 文件。

流程

- 在部署集群前，使用文本编辑器编辑 **install-config.yaml** 文件并添加以下小节：

```
controlPlane: 1
  platform:
    azure:
      settings:
        securityType: ConfidentialVM 2
        confidentialVM:
          uefiSettings:
            secureBoot: Enabled 3
            virtualizedTrustedPlatformModule: Enabled 4
      osDisk:
        securityProfile:
          securityEncryptionType: VMGuestStateOnly 5
```

1 指定 **controlPlane.platform.azure** 或 **compute.platform.azure**，以分别在 control plane

或计算节点上部署机密虚拟机。指定 **platform.azure.defaultMachinePlatform** 以在所有节点上部署机密虚拟机。

2 启用机密虚拟机。

3 启用安全引导。如需更多信息，请参阅 Azure 文档中有关安全引导的内容。

4 启用虚拟化可信平台模块。如需更多信息，请参阅 Azure 文档中有关 虚拟化可信平台模块的文档。

5 指定 **VMGuestStateOnly** 来加密虚拟机客户端状态。

7.5.6.6. Azure 的自定义 install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多详情，或修改所需参数的值。

重要

此示例 YAML 文件仅供参考。您必须使用安装程序来获取 **install-config.yaml** 文件，并进行修改。

```
apiVersion: v1
baseDomain: example.com 1
controlPlane: 2
  hyperthreading: Enabled 3 4
  name: master
  platform:
    azure:
      encryptionAtHost: true
      ultraSSDCapability: Enabled
      osDisk:
        diskSizeGB: 1024 5
        diskType: Premium_LRS
        diskEncryptionSet:
          resourceGroup: disk_encryption_set_resource_group
          name: disk_encryption_set_name
          subscriptionId: secondary_subscription_id
      osImage:
        publisher: example_publisher_name
        offer: example_image_offer
        sku: example_offer_sku
        version: example_image_version
      type: Standard_D8s_v3
  replicas: 3
compute: 6
- hyperthreading: Enabled 7
  name: worker
  platform:
    azure:
      ultraSSDCapability: Enabled
      type: Standard_D2s_v3
      encryptionAtHost: true
      osDisk:
```

```
        diskSizeGB: 512 8
        diskType: Standard_LRS
        diskEncryptionSet:
          resourceGroup: disk_encryption_set_resource_group
          name: disk_encryption_set_name
          subscriptionId: secondary_subscription_id
      osImage:
        publisher: example_publisher_name
        offer: example_image_offer
        sku: example_offer_sku
        version: example_image_version
      zones: 9
      - "1"
      - "2"
      - "3"
  replicas: 5
metadata:
  name: test-cluster 10
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 11
  serviceNetwork:
  - 172.30.0.0/16
platform:
  azure:
    defaultMachinePlatform:
      osImage: 12
        publisher: example_publisher_name
        offer: example_image_offer
        sku: example_offer_sku
        version: example_image_version
      ultraSSDCapability: Enabled
    baseDomainResourceGroupName: resource_group 13
    region: centralus 14
    resourceGroupName: existing_resource_group 15
    outboundType: Loadbalancer
    cloudName: AzurePublicCloud
pullSecret: '{"auths": ...}' 16
fips: false 17
sshKey: ssh-ed25519 AAAA... 18
```

1 10 14 16 必需。安装程序会提示您输入这个值。

2 6 如果没有提供这些参数和值，安装程序会提供默认值。

3 7 **controlPlane** 部分是一个单个映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，controlPlane **部分** 的第一行则不以连字符开头。仅使用一个 control plane 池。

4 是否要启用或禁用并发多线程或 **超线程**。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将 参数值设置为 **Disabled** 来禁用它。如果在某些集群机器中禁用并发多线程，则必须在所

有集群机器中禁用它。

重要

如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。如果您禁用并发多线程，请为您的机器使用较大的虚拟机类型，如 **Standard_D8s_v3**。

5 8 您可以指定要使用的磁盘大小（以 GB 为单位）。control plane 节点的最低推荐值为 1024 GB。

9 指定要将机器部署到的区域列表。如需高可用性，请至少指定两个区域。

11 要安装的集群网络插件。默认值 **OVNKubernetes** 是唯一支持的值。

12 可选：应该用来引导 control plane 和计算机器的自定义 Red Hat Enterprise Linux CoreOS (RHCOS) 镜像。**platform.azure.defaultMachinePlatform.osImage** 下的 **publisher**, **offer**, **sku**, 和 **version** 参数应用到 control plane 和计算机器。如果设置了 **controlPlane.platform.azure.osImage** 或 **compute.platform.azure.osImage** 下的参数，它们会覆盖 **platform.azure.defaultMachinePlatform.osImage** 参数。

13 指定包含基域的 DNS 区的资源组的名称。

15 指定要安装集群的现有资源组的名称。如果未定义，则会为集群创建新的资源组。

17 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 Red Hat Enterprise Linux (RHEL) 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅在 FIPS 模式中安装该系统。

当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS)时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。

18 您可以选择提供您用来访问集群中机器的 **sshKey** 值。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

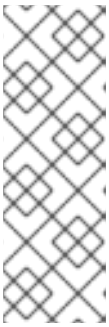
7.5.6.7. 在安装过程中配置集群范围的代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。

- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 **Proxy** 对象的 **spec.noProxy** 字段中添加站点来绕过代理。



注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，**Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 **.** 以仅匹配子域。例如，**.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代理。
4. 如果提供，安装程序会在 **openshift-config** 命名空间中生成名为 **user-ca-bundle** 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 **trusted-ca-bundle** 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， **Proxy** 对象的 **trustedCA** 字段中也会引用此配置映射。**additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。
5. 可选：决定 **Proxy** 对象的配置以引用 **trustedCA** 字段中 **user-ca-bundle** 配置映射的策略。允许的值是 **Proxyonly** 和 **Always**。仅在配置了 **http/https** 代理时，使用 **Proxyonly** 引用 **user-ca-bundle** 配置映射。使用 **Always** 始终引用 **user-ca-bundle** 配置映射。默认值为 **Proxyonly**。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。

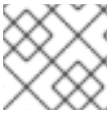
注意

如果安装程序超时，重启并使用安装程序的 **wait-for** 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. 保存该文件并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 cluster 的集群范围代理，该代理 **使用** 提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。



注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

其他资源

- 有关加速网络的详情，请参阅 Microsoft Azure 虚拟机的加速网络。

7.5.7. 为 Azure 配置用户定义的标签

在 OpenShift Container Platform 中，您可以使用标签对资源进行分组，并管理资源访问和成本。您只能在 OpenShift Container Platform 集群创建过程中在 **install-config.yaml** 文件中定义 Azure 资源的标签。您无法在集群创建后修改用户定义的标签。

对用户定义的标签的支持仅适用于 Azure Public Cloud 中创建的资源。升级到 OpenShift Container Platform 4.16 的 OpenShift Container Platform 集群不支持用户定义的标签。

用户定义的和 OpenShift Container Platform 特定的标签只适用于 OpenShift Container Platform 安装程序及其核心 Operator 等资源，如 Machine api provider azure Operator、Cluster Ingress Operator、Cluster Image Registry Operator。

默认情况下，OpenShift Container Platform 安装程序会将 OpenShift Container Platform 标签附加到 Azure 资源。这些 OpenShift Container Platform 标签无法被用户访问。

您可以使用 **install-config.yaml** 文件中的 **.platform.azure.userTags** 字段来定义用户定义的标签列表，如以下 **install-config.yaml** 文件所示。

install-config.yaml 文件示例

```
additionalTrustBundlePolicy: Proxyonly (1)
apiVersion: v1
baseDomain: catchall.azure.devcluster.openshift.com (2)
compute: (3)
- architecture: amd64
  hyperthreading: Enabled (4)
  name: worker
  platform: {}
  replicas: 3
controlPlane: (5)
  architecture: amd64
  hyperthreading: Enabled (6)
  name: master
```

```
  platform: {}
  replicas: 3
metadata:
  creationTimestamp: null
  name: user 7
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 8
  serviceNetwork:
  - 172.30.0.0/16
platform:
  azure:
    baseDomainResourceGroupName: os4-common 9
    cloudName: AzurePublicCloud 10
    outboundType: Loadbalancer
    region: southindia 11
    userTags: 12
      createdBy: user
      environment: dev
```

1. 定义信任捆绑包策略。
2. 必需。**baseDomain** 参数指定您的云供应商的基域。安装程序会提示您输入这个值。
3. 组成计算的机器的配置。**compute** 部分包括了一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头。如果没有提供这些参数和值，安装程序会提供默认值。
4. 启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将 参数值设置为 **Disabled** 来禁用它。如果在某些集群机器中禁用并发多线程，则必须在所有集群机器中禁用它。
5. 组成 control plane 的机器的配置。**controlPlane** 部分是一个单一的映射。**controlPlane** 部分的第一行不能以连字符 **-** 开头。您只能使用一个 control plane 池。如果没有提供这些参数和值，安装程序会提供默认值。
6. 启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将 参数值设置为 **Disabled** 来禁用它。如果在某些集群机器中禁用并发多线程，则必须在所有集群机器中禁用它。
7. 安装程序会提示您输入这个值。
8. 要安装的集群网络插件。默认值 **OVNKubernetes** 是唯一支持的值。
9. 指定 Azure DNS 区基域的资源组。
10. 指定 Azure 云环境的名称。您可以使用 **cloudName** 字段配置带有 Azure API 端点的 Azure SDK。如果没有提供值，则默认值为 Azure Public Cloud。
11. 必需。指定托管集群的 Azure 区域的名称。安装程序会提示您输入这个值。
12. 定义安装程序作为标签添加到它创建的所有 Azure 资源的额外键和值。

用户定义的标签有以下限制：

- 标签键最多可以有 128 个字符。
- 标签键必须以字母、数字或下划线结尾，并且只能包含字母、数字、下划线、句点和连字符。
- 标签键不区分大小写。
- 标签键不能是 **name**。它不能有前缀，如 **kubernetes.io**、**openshift.io**、**microsoft**、**azure** 和 **windows**。
- 标签值最多可有 256 个字符。
- 您可以为资源组和资源配置最多 10 个标签。

如需有关 Azure 标签的更多信息，请参阅 Azure 用户定义的标签

7.5.8. 为 Azure 查询用户定义的标签

创建 OpenShift Container Platform 集群后，您可以访问 Azure 资源定义的标签列表。OpenShift Container Platform 标签的格式为 **kubernetes.io_cluster.<cluster_id>:owned**。**cluster_id** 参数是 **config.openshift.io/Infrastructure** 中存在 **.status.infrastructureName** 的值。

- 运行以下命令，查询为 Azure 资源定义的标签：

```
$ oc get infrastructures.config.openshift.io cluster -o=jsonpath-as-
json='{.status.platformStatus.azure.resourceTags}'
```

输出示例

```
[
  [
    {
      "key": "createdBy",
      "value": "user"
    },
    {
      "key": "environment",
      "value": "dev"
    }
  ]
]
```

7.5.9. 安装 OpenShift CLI

您可以安装 OpenShift CLI(**oc**)来使用命令行界面与 OpenShift Container Platform 进行交互。您可以在 Linux、Windows 或 macOS 上安装 **oc**。

重要

如果安装了旧版本的 **oc**，则可能无法使用 OpenShift Container Platform 4.16 中的所有命令。下载并安装新版本的 **oc**。

在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **产品变体** 下拉列表中选择架构。
3. 从 **版本** 下拉列表中选择适当的版本。
4. 点 **OpenShift v4.16 Linux Client** 条目旁的 **Download Now** 来保存文件。
5. 解包存档：

```
$ tar xvf <file>
```

6. 将 **oc** 二进制文件放到 **PATH 中的目录中**。
   要查看您的 **PATH**，请执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

在 Windows 上安装 OpenShift CLI
您可以按照以下流程在 Windows 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 Windows Client** 条目旁的 **Download Now** 来保存文件。
4. 使用 ZIP 程序解压存档。
5. 将 **oc** 二进制文件移到 **PATH 中的目录中**。
   要查看您的 **PATH**，请打开命令提示并执行以下命令：

```
C:\> path
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

在 macOS 上安装 OpenShift CLI
您可以按照以下流程在 macOS 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 macOS Client** 条目旁的 **Download Now** 来保存文件。

注意

对于 macOS arm64，请选择 **OpenShift v4.16 macOS arm64 Client** 条目。

4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
要查看您的 **PATH**，请打开终端并执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

7.5.10. 在 kube-system 项目中存储管理员级别的 secret 的替代方案

默认情况下，管理员 secret 存储在 **kube-system** 项目中。如果您在 **install-config.yaml** 文件中将 **credentialsMode** 参数配置为 **Manual**，则必须使用以下替代方案之一：

- 要手动管理长期云凭证，请按照手动创建长期凭证中的步骤操作。
- 要实现在集群外为各个组件管理的短期凭证，请按照配置 Azure 集群以使用短期凭证中的步骤操作。

7.5.10.1. 手动创建长期凭证

在无法访问云身份和访问管理(IAM)API 的环境中，或者管理员更不希望将管理员级别的凭证 secret 存储在集群 **kube-system** 命名空间中时，可以在安装前将 Cloud Credential Operator(CCO)放入手动模式。

流程

1. 如果您没有将 **install-config.yaml** 配置文件中的 **credentialsMode** 参数设置为 **Manual**，请修改值，如下所示：

配置文件片段示例

```
apiVersion: v1
baseDomain: example.com
credentialsMode: Manual
# ...
```

2. 如果您之前还没有创建安装清单文件，请运行以下命令：

```
$ openshift-install create manifests --dir <installation_directory>
```

其中 **<installation_directory>** 是安装程序在其中创建文件的目录。

3. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

4. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 自定义资源 (CR) 列表：

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
  --included \ 1
  --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
  --to=<path_to_directory_for_credentials_requests> 3
```

1. **--included** 参数仅包含特定集群配置所需的清单。
2. 指定 **install-config.yaml** 文件的位置。
3. 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

此命令为每个 **CredentialsRequest** 对象创建一个 YAML 文件。

CredentialsRequest 对象示例

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component_credentials_request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: AzureProviderSpec
    roleBindings:
    - role: Contributor
  ...
```

5. 在之前生成的 **openshift-install** 清单目录中为 secret 创建 YAML 文件。secret 必须使用在 **spec.secretRef** 中为每个 **CredentialsRequest** 定义的命名空间和 secret 名称存储。

带有 secret 的 CredentialsRequest 对象示例

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component_credentials_request>
  namespace: openshift-cloud-credential-operator
```

```
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: AzureProviderSpec
    roleBindings:
    - role: Contributor
      ...
  secretRef:
    name: <component_secret>
    namespace: <component_namespace>
  ...
```

Secret 对象示例

```
apiVersion: v1
kind: Secret
metadata:
  name: <component_secret>
  namespace: <component_namespace>
data:
  azure_subscription_id: <base64_encoded_azure_subscription_id>
  azure_client_id: <base64_encoded_azure_client_id>
  azure_client_secret: <base64_encoded_azure_client_secret>
  azure_tenant_id: <base64_encoded_azure_tenant_id>
  azure_resource_prefix: <base64_encoded_azure_resource_prefix>
  azure_resourcegroup: <base64_encoded_azure_resourcegroup>
  azure_region: <base64_encoded_azure_region>
```

重要

在升级使用手动维护凭证的集群前，您必须确保 CCO 处于可升级状态。

7.5.10.2. 配置 Azure 集群以使用短期凭证

要安装使用 Microsoft Entra Workload ID 的集群，您必须配置 Cloud Credential Operator 工具，并为集群创建所需的 Azure 资源。

7.5.10.2.1. 配置 Cloud Credential Operator 工具

当 Cloud Credential Operator(CCO)以手动模式运行时，要从集群外部创建和管理云凭证，提取并准备 CCO 实用程序(**ccoctl**)二进制文件。

注意

ccoctl 工具是在 Linux 环境中运行的 Linux 二进制文件。

先决条件

- 您可以访问具有集群管理员权限的 OpenShift Container Platform 帐户。
- 已安装 OpenShift CLI(**oc**)。
- 您已为 **ccoctl** 工具创建了全局 Microsoft Azure 帐户，用于以下权限：

例 7.26. 所需的 Azure 权限

- Microsoft.Resources/subscriptions/resourceGroups/read
- Microsoft.Resources/subscriptions/resourceGroups/write
- Microsoft.Resources/subscriptions/resourceGroups/delete
- Microsoft.Authorization/roleAssignments/read
- Microsoft.Authorization/roleAssignments/delete
- Microsoft.Authorization/roleAssignments/write
- Microsoft.Authorization/roleDefinitions/read
- Microsoft.Authorization/roleDefinitions/write
- Microsoft.Authorization/roleDefinitions/delete
- Microsoft.Storage/storageAccounts/listkeys/action
- Microsoft.Storage/storageAccounts/delete
- Microsoft.Storage/storageAccounts/read
- Microsoft.Storage/storageAccounts/write
- Microsoft.Storage/storageAccounts/blobServices/containers/write
- Microsoft.Storage/storageAccounts/blobServices/containers/delete
- Microsoft.Storage/storageAccounts/blobServices/containers/read
- Microsoft.ManagedIdentity/userAssignedIdentities/delete
- Microsoft.ManagedIdentity/userAssignedIdentities/read
- Microsoft.ManagedIdentity/userAssignedIdentities/write
- Microsoft.ManagedIdentity/userAssignedIdentities/federatedIdentityCredentials/read
- Microsoft.ManagedIdentity/userAssignedIdentities/federatedIdentityCredentials/write
- Microsoft.ManagedIdentity/userAssignedIdentities/federatedIdentityCredentials/delete
- Microsoft.Storage/register/action
- Microsoft.ManagedIdentity/register/action

流程

1. 运行以下命令，为 OpenShift Container Platform 发行镜像设置变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. 运行以下命令，从 OpenShift Container Platform 发行镜像获取 CCO 容器镜像：

```
$ CCO_IMAGE=$(oc adm release info --image-for='cloud-credential-operator'
$RELEASE_IMAGE -a ~/.pull-secret)
```

注意

确保 **$RELEASE_IMAGE** 的架构与将使用 **ccoctl** 工具的环境架构相匹配。

3. 运行以下命令，将 CCO 容器镜像中的 **ccoctl** 二进制文件提取到 OpenShift Container Platform 发行镜像中：

```
$ oc image extract $CCO_IMAGE \
  --file="/usr/bin/ccoctl.<rhel_version>" \ 1
  -a ~/.pull-secret
```

1 对于 **<rhel_version>**，请指定与主机使用的 Red Hat Enterprise Linux (RHEL) 版本对应的值。如果没有指定值，则默认使用 **ccoctl.rhel8**。以下值有效：

- **rhel8**: 为使用 RHEL 8 的主机指定这个值。
- **rhel9** ：为使用 RHEL 9 的主机指定这个值。

4. 运行以下命令更改权限以使 **ccoctl** 可执行：

```
$ chmod 775 ccoctl.<rhel_version>
```

验证

- 要验证 **ccoctl** 是否可使用，请运行以下命令来显示帮助文件：

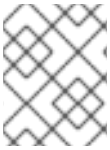
```
$ ccoctl --help
```

ccoctl --help 的输出

```
OpenShift credentials provisioning tool

Usage:
  ccoctl [command]

Available Commands:
  aws         Manage credentials objects for AWS cloud
  azure       Manage credentials objects for Azure
  gcp         Manage credentials objects for Google cloud
  help        Help about any command
  ibmcloud    Manage credentials objects for IBM Cloud
  nutanix     Manage credentials objects for Nutanix

Flags:
  -h, --help   help for ccoctl

Use "ccoctl [command] --help" for more information about a command.
```

7.5.10.2.2. 使用 Cloud Credential Operator 实用程序创建 Azure 资源

您可以使用 **ccoctl azure create-all** 命令自动创建 Azure 资源。



注意

默认情况下，**ccoctl** 在运行命令的目录中创建对象。要在其他目录中创建对象，请使用 **--output-dir** 标志。此流程使用 **<path_to_ccoctl_output_dir>** 来引用这个目录。

先决条件

您必须：

- 提取并准备好 **ccoctl** 二进制文件。
- 使用 Azure CLI 访问 Microsoft Azure 帐户。

流程

1. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 对象列表：

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
  --included \ 1
  --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
  --to=<path_to_directory_for_credentials_requests> 3
```

1. **--included** 参数仅包含特定集群配置所需的清单。
2. 指定 **install-config.yaml** 文件的位置。
3. 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

注意

此命令可能需要一些时间才能运行。

3. 要启用 **ccoctl** 工具自动检测 Azure 凭证，请运行以下命令登录到 Azure CLI：

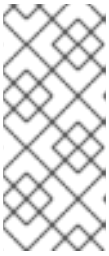
```
$ az login
```

4. 运行以下命令，使用 **ccoctl** 工具处理所有 **CredentialsRequest** 对象：

```
$ ccoctl azure create-all \
  --name=<azure_infra_name> \ 1
```

```
--output-dir=<ccoctl_output_dir> \ 2
--region=<azure_region> \ 3
--subscription-id=<azure_subscription_id> \ 4
--credentials-requests-dir=<path_to_credentials_requests_directory> \ 5
--dnszone-resource-group-name=<azure_dns_zone_resource_group_name> \ 6
--tenant-id=<azure_tenant_id> 7
```

1. 为用于跟踪的所有创建 Azure 资源指定用户定义的名称。
2. 可选：指定您希望 **ccoctl** 实用程序在其中创建对象的目录。默认情况下，实用程序在运行命令的目录中创建对象。
3. 指定在其中创建云资源的 Azure 区域。
4. 指定要使用的 Azure 订阅 ID。
5. 指定包含组件 **CredentialsRequest** 对象文件的目录。
6. 指定包含集群基域 Azure DNS 区的资源组名称。
7. 指定要使用的 Azure 租户 ID。



注意

如果您的集群使用 **TechPreviewNoUpgrade** 功能集启用的技术预览功能，则必须包含 **--enable-tech-preview** 参数。

要查看其他可选参数以及如何使用它们的说明，请运行 **azure create-all --help** 命令。

验证

- 要验证 OpenShift Container Platform secret 是否已创建，列出 **<path_to_ccoctl_output_dir>/manifests** 目录中的文件：

```
$ ls <path_to_ccoctl_output_dir>/manifests
```

输出示例

```
azure-ad-pod-identity-webhook-config.yaml
cluster-authentication-02-config.yaml
openshift-cloud-controller-manager-azure-cloud-credentials-credentials.yaml
openshift-cloud-network-config-controller-cloud-credentials-credentials.yaml
openshift-cluster-api-capz-manager-bootstrap-credentials-credentials.yaml
openshift-cluster-csi-drivers-azure-disk-credentials-credentials.yaml
openshift-cluster-csi-drivers-azure-file-credentials-credentials.yaml
openshift-image-registry-installer-cloud-credentials-credentials.yaml
openshift-ingress-operator-cloud-credentials-credentials.yaml
openshift-machine-api-azure-cloud-credentials-credentials.yaml
```

您可以验证 Microsoft Entra ID 服务账户通过查询 Azure 而创建。如需更多信息，请参阅 Azure 文档中有关列出 Entra ID 服务帐户的内容。

7.5.10.2.3. 整合 Cloud Credential Operator 实用程序清单

要为单个组件在集群外实现短期安全凭证，您必须将创建 Cloud Credential Operator 实用程序 (**ccoctl**) 的清单文件移到安装程序的正确目录中。

先决条件

- 您已使用托管集群的云平台配置了帐户。
- 您已配置了 Cloud Credential Operator 实用程序 (**ccoctl**)。
- 已使用 **ccoctl** 工具创建了集群所需的云供应商资源。

流程

1. 如果您没有将 **install-config.yaml** 配置文件中的 **credentialsMode** 参数设置为 **Manual**，请修改值，如下所示：

   配置文件片段示例

   ```
   apiVersion: v1
   baseDomain: example.com
   credentialsMode: Manual
   # ...
   ```

2. 如果您使用 **ccoctl** 实用程序创建新的 Azure 资源组，而不是使用现有资源组，请修改 **install-config.yaml** 中的 **resourceGroupName** 参数，如下所示：

   配置文件片段示例

   ```
   apiVersion: v1
   baseDomain: example.com
   # ...
   platform:
     azure:
       resourceGroupName: <azure_infra_name> 1
   # ...
   ```

   1 这个值必须与 **ccoctl azure create-all** 命令的 **--name** 参数指定的 Azure 资源用户定义的名称匹配。

3. 如果您之前还没有创建安装清单文件，请运行以下命令：

   ```
   $ openshift-install create manifests --dir <installation_directory>
   ```

   其中 **<installation_directory>** 是安装程序在其中创建文件的目录。

4. 运行以下命令，将 **ccoctl** 工具生成的清单复制到安装程序创建的 **manifests** 目录中：

   ```
   $ cp /<path_to_ccoctl_output_dir>/manifests/* ./manifests/
   ```

5. 运行以下命令，将 **ccoctl** 工具在 **tls** 目录中生成的私钥复制到安装目录中：

```
$ cp -a /<path_to_ccoctl_output_dir>/tls .
```

7.5.11. 部署集群

您可以在兼容云平台上安装 OpenShift Container Platform。

重要

在初始安装过程中，您只能运行安装程序的 **create cluster** 命令一次。

先决条件

- 您已使用托管集群的云平台配置了帐户。
- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。
- 您有一个 Azure 订阅 ID 和租户 ID。

流程

- 进入包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定自定义 **./install-config.yaml** 文件的位置。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不是 **info**。

验证

当集群部署成功完成时：

- 终端会显示用于访问集群的说明，包括指向 Web 控制台和 **kubeadmin** 用户的凭证的链接。
- 凭证信息还会输出到 **<installation_directory>/.openshift_install.log**.

重要

不要删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

7.5.12. 使用 CLI 登录集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

其他资源

- 如需有关 访问和了解 OpenShift Container Platform Web 控制台的更多详情，请参阅 访问 Web 控制台。

7.5.13. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于 远程健康监控

7.5.14. 后续步骤

- 自定义集群。
- 如果需要，您可以选择不使用远程健康报告。

7.6. 使用网络自定义在 AZURE 上安装集群

在 OpenShift Container Platform 版本 4.16 中，您可以使用自定义的网络配置在安装程序在 Microsoft Azure 上置备的基础架构上安装集群。通过自定义网络配置，您的集群可以与环境中现有的 IP 地址分配共存，并与现有的 MTU 和 VXLAN 配置集成。

大部分网络配置参数必须在安装过程中设置，只有 **kubeProxy** 配置参数可以在运行的集群中修改。

7.6.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读选择集群安装方法并为用户准备它的文档。
- 您已将 Azure 帐户配置为托管集群，并决定要将集群部署到的已测试和验证的区域。
- 如果使用防火墙，将其配置为允许集群需要访问的站点。
- 如果您使用客户管理的加密密钥，准备了用于加密的 Azure 环境。

7.6.2. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类型的基础架构上执行受限网络安装。在此过程中，您可以下载所需的内容，并使用它为镜像 registry 填充安装软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群前，您要更新镜像 registry 的内容。

7.6.3. 为集群节点 SSH 访问生成密钥对

在 OpenShift Container Platform 安装过程中，您可以为安装程序提供 SSH 公钥。密钥通过它们的 Ignition 配置文件传递给 Red Hat Enterprise Linux CoreOS(RHCOS)节点，用于验证对节点的 SSH 访问。密钥添加到每个节点上 **core** 用户的 **~/.ssh/authorized_keys** 列表中，这将启用免密码身份验证。

将密钥传递给节点后，您可以使用密钥对作为用户 **核心** 通过 SSH 连接到 RHCOS 节点。若要通过 SSH 访问节点，必须由 SSH 为您的本地用户管理私钥身份。

如果要通过 SSH 连接到集群节点来执行安装调试或灾难恢复，则必须在安装过程中提供 SSH 公钥。**./openshift-install gather** 命令还需要在集群节点上设置 SSH 公钥。

重要

不要在生产环境中跳过这个过程，在生产环境中需要灾难恢复和调试。

注意

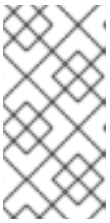
您必须使用本地密钥，而不是使用特定平台方法配置 的密钥，如 AWS 密钥对。

流程

1. 如果您在本地计算机上没有可用于在集群节点上进行身份验证的现有 SSH 密钥对，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_ed25519**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。



注意

如果您计划在 **x86_64**、**ppc64le** 和 **s390x** 架构上安装使用 RHEL 加密库（这些加密库已提交给 NIST 用于 FIPS 140-2/140-3 验证）的 OpenShift Container Platform 集群，则不要创建使用 **ed25519** 算法的密钥。相反，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

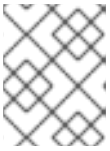
2. 查看公共 SSH 密钥：

```
$ cat <path>/<file_name>.pub
```

例如，运行以下命令来查看 **~/.ssh/id_ed25519.pub** 公钥：

```
$ cat ~/.ssh/id_ed25519.pub
```

3. 将 SSH 私钥身份添加到本地用户的 SSH 代理（如果尚未添加）。在集群节点上，或者要使用 **./openshift-install gather** 命令，需要对该密钥进行 SSH 代理管理，才能在集群节点上进行免密码 SSH 身份验证。



注意

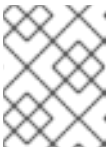
在某些发行版中，自动管理默认 SSH 私钥身份，如 ~/**.ssh/id_rsa** 和 ~/**.ssh/id_dsa**。

a. 如果 **ssh-agent** 进程尚未为您的本地用户运行，请将其作为后台任务启动：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```



注意

如果集群处于 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

4. 将 SSH 私钥添加到 **ssh-agent** ：

```
$ ssh-add <path>/<file_name> 1
```

1 指定 SSH 私钥的路径和文件名，如 ~/**.ssh/id_ed25519.pub**

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

后续步骤

- 安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

7.6.4. 获取安装程序

在安装 OpenShift Container Platform 前，将安装文件下载到您用于安装的主机上。

先决条件

- 您有一台运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB。

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐户，请使用您的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入到安装类型的页面，下载与您的主机操作系统和架构对应的安装程序，并将该文件放在您要存储安装配置文件的目录中。

重要

安装程序会在用来安装集群的计算机上创建几个文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。

重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，请为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。此 pull secret 允许您与所含授权机构提供的服务进行身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

7.6.5. 创建安装配置文件

您可以自定义在 Microsoft Azure 上安装的 OpenShift Container Platform 集群。

先决条件

- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。
- 您有一个 Azure 订阅 ID 和租户 ID。
- 如果要使用服务主体安装集群，则有其应用程序 ID 和密码。
- 如果您要使用系统分配的受管身份安装集群，需要在您要从其中运行安装程序的虚拟机上启用它。
- 如果您要使用用户分配的受管身份安装集群，需要满足以下先决条件：
  - 您有它的客户端 ID。
  - 您已将其分配给您要从其运行安装程序的虚拟机。

流程

1. 可选：如果您之前在这个计算机上运行安装程序，并希望使用替代的服务主体或受管身份，请进入 **~/.azure/** 目录并删除 **osServicePrincipal.json** 配置文件。
   删除此文件可防止安装程序自动重复使用之前安装中的订阅和验证值。
2. 创建 **install-config.yaml** 文件。
   a. 进入包含安装程序的目录并运行以下命令：

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定要存储安装程序创建的文件的目录名称。

在指定目录时：

- 验证该目录是否具有**执行**权限。在安装目录中运行 Terraform 二进制文件需要这个权限。
- 使用空目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

b. 在提示符处，提供云的配置详情：

i. 可选： 选择用于访问集群机器的 SSH 密钥。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

ii. 选择 **azure** 作为目标平台。
如果安装程序无法找到之前安装中的 **osServicePrincipal.json** 配置文件，会提示您输入 Azure 订阅和验证值。

iii. 为您的订阅输入以下 Azure 参数值：

- **Azure subscription id**：输入用于集群的订阅 ID。
- **Azure 租户 id**：输入租户 ID。

iv. 根据您用来部署集群的 Azure 身份，在提示输入 **azure 服务主体客户端 id**时执行以下操作之一：

- 如果您使用服务主体，请输入其应用程序 ID。
- 如果您使用系统分配的受管身份，请将此值设为空白。
- 如果您使用用户分配的受管身份，请指定其客户端 ID。

v. 根据您用来部署集群的 Azure 身份，在提示输入 **azure 服务主体客户端 secret**时执行以下操作之一：

- 如果您使用服务主体，请输入其密码。
- 如果您使用系统分配的受管身份，请将此值设为空白。
- 如果您使用用户分配的受管身份，请将此值设为空白。

vi. 选择要将集群部署到的区域。

vii. 选择集群要部署到的基域。基域与您为集群创建的 Azure DNS 区对应。

viii. 为集群输入一个描述性名称。

重要

所有通过公共端点提供的 Azure 资源均存在资源名称的限制，您无法创建使用某些名称的资源。如需 Azure 限制词语的列表，请参阅 Azure 文档中的解决预留资源名称错误。

3. 修改 **install-config.yaml** 文件。您可以在"安装配置参数"部分找到有关可用参数的更多信息。
4. 备份 **install-config.yaml** 文件，以便您可以使用它安装多个集群。

重要

install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用该文件，您必须立即备份该文件。

在以前的版本中，安装程序会创建一个 **osServicePrincipal.json** 配置文件，并将此文件存储在计算机上的 **~/.azure/** 目录中。这样可确保安装程序在目标平台上创建 OpenShift Container Platform 集群时可以加载配置集。

其他资源

- Azure 的安装配置参数

7.6.5.1. 集群安装的最低资源要求

每台集群机器都必须满足以下最低要求：

表 7.2. 最低资源要求

机器	操作系统	vCPU [1]	虚拟内存	Storage	每秒输入/输出 (IOPS) [2]
bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane（控制平面）	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS、RHEL 8.6 及更新版本 [3]	2	8 GB	100 GB	300

1. 当未启用并发多线程(SMT)或超线程时，一个 vCPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比例：（每个内核数的线程）× sockets = vCPU。
2. OpenShift Container Platform 和 Kubernetes 对磁盘性能非常敏感，建议使用更快的存储速度，特别是 control plane 节点上需要 10 ms p99 fsync 持续时间的 etcd。请注意，在许多云平台上，存储大小和 IOPS 可一起扩展，因此您可能需要过度分配存储卷来获取足够的性能。
3. 与所有用户置备的安装一样，如果您选择在集群中使用 RHEL 计算机器，则负责所有操作系统生命周期管理和维护，包括执行系统更新、应用补丁和完成所有其他必要的任务。RHEL 7 计算机器的使用已弃用，并已在 OpenShift Container Platform 4.10 及更新的版本中删除。

注意

从 OpenShift Container Platform 版本 4.13 开始，RHCOS 基于 RHEL 版本 9.2，它更新了微架构要求。以下列表包含每个架构需要的最小指令集架构 (ISA)：

- x86-64 体系结构需要 x86-64-v2 ISA
- ARM64 架构需要 ARMv8.0-A ISA
- IBM Power 架构需要 Power 9 ISA
- s390x 架构需要 z14 ISA

如需更多信息，请参阅 RHEL 架构。

重要

您需要使用将 **PremiumIO** 参数设置为 **true** 的 Azure 虚拟机。

如果平台的实例类型满足集群机器的最低要求，则 OpenShift Container Platform 支持使用它。

其他资源

- 优化存储

7.6.5.2. 为 Azure 测试的实例类型

以下 Microsoft Azure 实例类型已经 OpenShift Container Platform 测试。

例 7.27. 基于 64 位 x86 架构的机器类型

- **c4.***
- **c5.***
- **c5a.***
- **i3.***
- **m4.***
- **m5.***
- **m5a.***
- **m6a.***
- **m6i.***
- **r4.***
- **r5.***
- **r5a.***
- **r6i.***

- **t3.***
- **t3a.***

7.6.5.3. 在 64 位 ARM 基础架构上为 Azure 测试的实例类型

以下 Microsoft Azure ARM64 实例类型已使用 OpenShift Container Platform 测试。

例 7.28. 基于 64 位 ARM 架构的机器类型

- **c6g.***
- **m6g.***

7.6.5.4. 为 Azure 虚拟机启用可信启动

在 Azure 上安装集群时，您可以启用两个可信启动功能： 安全引导 和虚拟化可信平台模块。

请参阅 Azure 文档中有关 虚拟机大小 的信息，以了解虚拟机支持这些功能的大小。

重要

可信启动只是一个技术预览功能。技术预览功能不受红帽产品服务等级协议（SLA）支持，且功能可能并不完整。红帽不推荐在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。

有关红帽技术预览功能支持范围的更多信息，请参阅技术预览功能支持范围。

先决条件

- 您已创建了 **install-config.yaml** 文件。

流程

- 在部署集群前，使用文本编辑器编辑 **install-config.yaml** 文件并添加以下小节：

```
controlPlane: 1
  platform:
    azure:
      settings:
        securityType: TrustedLaunch 2
        trustedLaunch:
          uefiSettings:
            secureBoot: Enabled 3
            virtualizedTrustedPlatformModule: Enabled 4
```

1 指定 **controlPlane.platform.azure** 或 **compute.platform.azure**，以分别在 control plane 或计算节点上启用可信启动。指定 **platform.azure.defaultMachinePlatform**，以便在所有节点上启用可信启动。

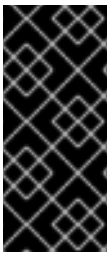
2 启用可信启动功能。

3 启用安全引导。如需更多信息，请参阅 Azure 文档中有关安全引导的内容。

4 启用虚拟化可信平台模块。如需更多信息，请参阅 Azure 文档中有关 虚拟化可信平台模块的文档。

7.6.5.5. 启用机密虚拟机

您可在安装集群前启用机密虚拟机。您可以为计算节点、control plane 节点或所有节点启用机密虚拟机。



重要

使用机密虚拟机只是一个技术预览功能。技术预览功能不受红帽产品服务等级协议（SLA）支持，且功能可能并不完整。红帽不推荐在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。

有关红帽技术预览功能支持范围的更多信息，请参阅技术预览功能支持范围。

您可以使用带有以下虚拟机大小的机密虚拟机：

- DCasv5-series
- DCadsv5-series
- ECasv5-series
- ECadsv5-series

重要

64 位 ARM 架构目前不支持机密虚拟机。

先决条件

- 您已创建了 **install-config.yaml** 文件。

流程

- 在部署集群前，使用文本编辑器编辑 **install-config.yaml** 文件并添加以下小节：

```
controlPlane: 1
  platform:
    azure:
      settings:
        securityType: ConfidentialVM 2
        confidentialVM:
          uefiSettings:
            secureBoot: Enabled 3
            virtualizedTrustedPlatformModule: Enabled 4
      osDisk:
        securityProfile:
          securityEncryptionType: VMGuestStateOnly 5
```

1 指定 **controlPlane.platform.azure** 或 **compute.platform.azure**，以分别在 control plane

或计算节点上部署机密虚拟机。指定 **platform.azure.defaultMachinePlatform** 以在所有节点上部署机密虚拟机。

2 启用机密虚拟机。

3 启用安全引导。如需更多信息，请参阅 Azure 文档中有关安全引导的内容。

4 启用虚拟化可信平台模块。如需更多信息，请参阅 Azure 文档中有关 虚拟化可信平台模块的文档。

5 指定 **VMGuestStateOnly** 来加密虚拟机客户端状态。

7.6.5.6. Azure 的自定义 install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多详情，或修改所需参数的值。

重要

此示例 YAML 文件仅供参考。您必须使用安装程序来获取 **install-config.yaml** 文件，并进行修改。

```
apiVersion: v1
baseDomain: example.com 1
controlPlane: 2
  hyperthreading: Enabled 3 4
  name: master
  platform:
    azure:
      encryptionAtHost: true
      ultraSSDCapability: Enabled
      osDisk:
        diskSizeGB: 1024 5
        diskType: Premium_LRS
        diskEncryptionSet:
          resourceGroup: disk_encryption_set_resource_group
          name: disk_encryption_set_name
          subscriptionId: secondary_subscription_id
      osImage:
        publisher: example_publisher_name
        offer: example_image_offer
        sku: example_offer_sku
        version: example_image_version
      type: Standard_D8s_v3
  replicas: 3
compute: 6
- hyperthreading: Enabled 7
  name: worker
  platform:
    azure:
      ultraSSDCapability: Enabled
      type: Standard_D2s_v3
      encryptionAtHost: true
      osDisk:
```

```
        diskSizeGB: 512 8
        diskType: Standard_LRS
        diskEncryptionSet:
          resourceGroup: disk_encryption_set_resource_group
          name: disk_encryption_set_name
          subscriptionId: secondary_subscription_id
      osImage:
        publisher: example_publisher_name
        offer: example_image_offer
        sku: example_offer_sku
        version: example_image_version
      zones: 9
      - "1"
      - "2"
      - "3"
  replicas: 5
metadata:
  name: test-cluster 10
networking: 11
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 12
  serviceNetwork:
  - 172.30.0.0/16
platform:
  azure:
    defaultMachinePlatform:
      osImage: 13
        publisher: example_publisher_name
        offer: example_image_offer
        sku: example_offer_sku
        version: example_image_version
      ultraSSDCapability: Enabled
    baseDomainResourceGroupName: resource_group 14
    region: centralus 15
    resourceGroupName: existing_resource_group 16
    outboundType: Loadbalancer
    cloudName: AzurePublicCloud
pullSecret: '{"auths": ...}' 17
fips: false 18
sshKey: ssh-ed25519 AAAA... 19
```

1 10 15 17 必需。安装程序会提示您输入这个值。

2 6 11 如果没有提供这些参数和值，安装程序会提供默认值。

3 7 **controlPlane** 部分是一个单个映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，controlPlane **部分** 的第一行则不以连字符开头。仅使用一个 control plane 池。

4

是否要启用或禁用并发多线程或 **超线程**。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将 参数值设置为 **Disabled** 来禁用它。如果在某些集群机器中禁用并发多线程，则必须在所

重要

如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。如果您禁用并发多线程，请为您的机器使用较大的虚拟机类型，如 **Standard_D8s_v3**。

5 8 您可以指定要使用的磁盘大小（以 GB 为单位）。control plane 节点的最低推荐值为 1024 GB。

9 指定要将机器部署到的区域列表。如需高可用性，请至少指定两个区域。

12 要安装的集群网络插件。默认值 **OVNKubernetes** 是唯一支持的值。

13 可选：应该用来引导 control plane 和计算机器的自定义 Red Hat Enterprise Linux CoreOS (RHCOS) 镜像。**platform.azure.defaultMachinePlatform.osImage** 下的 **publisher**, **offer**, **sku**, 和 **version** 参数应用到 control plane 和计算机器。如果设置了 **controlPlane.platform.azure.osImage** 或 **compute.platform.azure.osImage** 下的参数，它们会覆盖 **platform.azure.defaultMachinePlatform.osImage** 参数。

14 指定包含基域的 DNS 区的资源组的名称。

16 指定要安装集群的现有资源组的名称。如果未定义，则会为集群创建新的资源组。

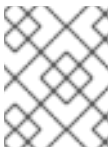
18 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 Red Hat Enterprise Linux (RHEL) 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅在 FIPS 模式中安装该系统。

当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS)时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。

19 您可以选择提供您用来访问集群中机器的 **sshKey** 值。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

7.6.5.7. 在安装过程中配置集群范围的代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。

- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 **Proxy** 对象的 **spec.noProxy** 字段中添加站点来绕过代理。

注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

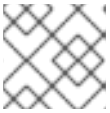
对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，**Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 **.** 以仅匹配子域。例如，**.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代理。
4. 如果提供，安装程序会在 **openshift-config** 命名空间中生成名为 **user-ca-bundle** 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 **trusted-ca-bundle** 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， **Proxy** 对象的 **trustedCA** 字段中也会引用此配置映射。**additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。
5. 可选：决定 **Proxy** 对象的配置以引用 **trustedCA** 字段中 **user-ca-bundle** 配置映射的策略。允许的值是 **Proxyonly** 和 **Always**。仅在配置了 **http/https** 代理时，使用 **Proxyonly** 引用 **user-ca-bundle** 配置映射。使用 **Always** 始终引用 **user-ca-bundle** 配置映射。默认值为 **Proxyonly**。



注意

安装程序不支持代理的 **readinessEndpoints** 字段。

注意

如果安装程序超时，重启并使用安装程序的 **wait-for** 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. 保存该文件并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 cluster 的集群范围代理，该代理 **使用** 提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

7.6.6. 网络配置阶段

OpenShift Container Platform 安装前有两个阶段，您可以在其中自定义网络配置。

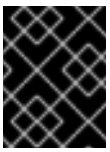
第 1 阶段

在创建清单文件前，您可以自定义 **install-config.yaml** 文件中的以下与网络相关的字段：

- **networking.networkType**
- **networking.clusterNetwork**
- **networking.serviceNetwork**
- **networking.machineNetwork**
  有关这些字段的更多信息，请参阅 *安装配置参数*。

注意

将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。



重要

CIDR 范围 **172.17.0.0/16** 由 libVirt 保留。对于集群中的任何网络，您无法使用此范围或与这个范围重叠的范围。

第 2 阶段

运行 **openshift-install create 清单创建** 清单文件后，您可以只使用您要修改的字段定义自定义 Cluster Network Operator 清单。您可以使用 清单指定高级网络配置。

您不能覆盖在 stage 2 阶段 1 中在 **install-config.yaml** 文件中指定的值。但是，您可以在第 2 阶段进一步自定义网络插件。

7.6.7. 指定高级网络配置

您可以使用网络插件的高级网络配置将集群集成到现有网络环境中。您只能在安装集群前指定高级网络配置。

重要

不支持通过修改安装程序创建的 OpenShift Container Platform 清单文件来自定义网络配置。支持应用您创建的清单文件，如以下流程中所示。

先决条件

- 您已创建 **install-config.yaml** 文件并完成对其所做的任何修改。

流程

1. 进入包含安装程序的目录并创建清单：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 **<installation_directory>** 指定包含集群的 **install-config.yaml** 文件的目录名称。

2. 在 **<installation_directory>/manifests/** 目录中 为**高级网络配置创建一个名为 cluster-network-03-config.yml** 的 stub 清单文件：

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
```

3. 在 **cluster-network-03-config.yml** 文件中指定集群的高级网络配置，如下例所示：

   为 OVN-Kubernetes 网络供应商启用 IPsec

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  defaultNetwork:
    ovnKubernetesConfig:
      ipsecConfig:
        mode: Full
```

4. 可选：备份 **manifests/cluster-network-03-config.yml** 文件。创建 Ignition 配置文件时，安装程序会使用 **manifests/** 目录。

7.6.8. Cluster Network Operator 配置

集群网络的配置作为 Cluster Network Operator(CNO)配置的一部分指定，并存储在名为 **cluster** 的自定义资源(CR)对象中。CR 指定 **operator.openshift.io** API 组中的 **Network** API 的字段。

CNO 配置在集群安装过程中从 **Network.config.openshift.io** API 组中的 **Network** API 继承以下字段：

clusterNetwork

从中分配 Pod IP 地址的 IP 地址池。

serviceNetwork

服务的 IP 地址池.

defaultNetwork.type

集群网络插件。**OVNKubernetes** 是安装期间唯一支持的插件。

您可以通过在名为 **cluster** 的 CNO 对象中设置 **defaultNetwork** 对象的字段来为集群指定集群网络插件配置。

7.6.8.1. Cluster Network Operator 配置对象

下表中描述了 Cluster Network Operator(CNO)的字段：

表 7.3. Cluster Network Operator 配置对象

字段	类型	描述
metadata.name	**字符串**	CNO 对象的名称。这个名称始终是 **集群**。

spec.clusterNetwork	**array**	用于指定从哪些 IP 地址块分配 Pod IP 地址以及集群中每个节点的子网前缀长度的列表。例如：
		`spec:`
		`  clusterNetwork:`
		`  - cidr: 10.128.0.0/19`
		`  hostPrefix: 23`
		`  - cidr: 10.128.32.0/19`
		`  hostPrefix: 23`

spec.serviceNetwork	**array**	服务的 IP 地址块。OpenShift SDN 和 OVN-Kubernetes 网络插件只支持服务网络的一个 IP 地址块。例如：
		`spec:`
		`  serviceNetwork:`
		`  - 172.30.0.0/14`
		您只能在创建清单前在 **install-config.yaml** 文件中自定义此字段。该值在清单文件中是只读的。

spec.defaultNetwork	**object**	为集群网络配置网络插件。
spec.kubeProxyConfig	**object**	此对象的字段指定 kube-proxy 配置。如果使用 OVN-Kubernetes 集群网络供应商，则 kube-proxy 配置不会起作用。

defaultNetwork 对象配置
下表列出了 **defaultNetwork** 对象的值：

表 7.4. defaultNetwork 对象

字段	类型	描述
type	**字符串**	**OVNKubernetes**。Red Hat OpenShift Networking 网络插件在安装过程中被选择。此值在集群安装后无法更改。 **注意** OpenShift Container Platform 默认使用 OVN-Kubernetes 网络插件。OpenShift SDN 不再作为新集群的安装选择提供。
ovnKubernetesConfig	**object**	此对象仅对 OVN-Kubernetes 网络插件有效。

配置 OVN-Kubernetes 网络插件
下表描述了 OVN-Kubernetes 网络插件的配置字段：

表 7.5. **ovnKubernetesConfig** object

字段	类型	描述
mtu	**integer**	Geneve（通用网络虚拟化封装）覆盖网络的最大传输单元 (MTU)。这根据主网络接口的 MTU 自动探测。您通常不需要覆盖检测到的 MTU。 如果自动探测的值不是您期望的值，请确认节点上主网络接口上的 MTU 是否正确。您不能使用这个选项更改节点上主网络接口的 MTU 值。 如果集群中不同节点需要不同的 MTU 值，则必须将此值设置为**比** 集群中的最低 MTU 值小 100。例如，如果集群中的某些节点的 MTU 为 **9001**，而某些节点的 MTU 为 **1500**，则必须将此值设置为 **1400**。
genevePort	**integer**	用于所有 Geneve 数据包的端口。默认值为 **6081**。此值在集群安装后无法更改。
ipsecConfig	**object**	指定用于自定义 IPsec 配置的配置对象。
ipv4	**object**	为 IPv4 设置指定配置对象。
ipv6	**object**	为 IPv6 设置指定配置对象。
policyAuditConfig	**object**	指定用于自定义网络策略审计日志的配置对象。如果未设置，则使用默认的审计日志设置。

字段	类型	描述
gatewayConfig	**object**	可选：指定一个配置对象来自定义如何将出口流量发送到节点网关。  **注意** 在迁移出口流量时，工作负载和服务流量会受到一定影响，直到 Cluster Network Operator (CNO) 成功推出更改。

表 7.6. **ovnKubernetesConfig.ipv4** object

字段	类型	描述
internalTransitSwitchSubnet	字符串	如果您的现有网络基础架构与 **100.88.0.0/16** IPv4 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。启用东西流量的分布式传输交换机的子网。此子网不能与 OVN-Kubernetes 或主机本身使用的任何其他子网重叠。必须足够大，以适应集群中的每个节点一个 IP 地址。 默认值为 **100.88.0.0/16**。
internalJoinSubnet	字符串	如果您的现有网络基础架构与 **100.64.0.0/16** IPv4 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。您必须确保 IP 地址范围没有与 OpenShift Container Platform 安装使用的任何其他子网重叠。IP 地址范围必须大于可添加到集群的最大节点数。例如，如果 **clusterNetwork.cidr** 值为 **10.128.0.0/14**，并且 **clusterNetwork.hostPrefix** 值为 **/23**，则最大节点数量为 **2^(23-14)=512**。 默认值为 **100.64.0.0/16**。

表 7.7. **ovnKubernetesConfig.ipv6** object

字段	类型	描述
internalTransitSwitchSubnet	字符串	如果您的现有网络基础架构与 **fd97::/64** IPv6 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。启用东西流量的分布式传输交换机的子网。此子网不能与 OVN-Kubernetes 或主机本身使用的任何其他子网重叠。必须足够大，以适应集群中的每个节点一个 IP 地址。 默认值为 **fd97::/64**。

字段	类型	描述
internalJoinSubnet	字符串	如果您的现有网络基础架构与 **fd98::/64** IPv6 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。您必须确保 IP 地址范围没有与 OpenShift Container Platform 安装使用的任何其他子网重叠。IP 地址范围必须大于可添加到集群的最大节点数。 默认值为 **fd98::/64**。

表 7.8. **policyAuditConfig** object

字段	类型	描述
rateLimit	整数	每个节点每秒生成一次的消息数量上限。默认值为每秒 **20** 条消息。
maxFileSize	整数	审计日志的最大大小，以字节为单位。默认值为 **50000000** 或 50 MB。
maxLogFiles	整数	保留的日志文件的最大数量。
目的地	字符串	以下附加审计日志目标之一： **libc** 主机上的 journald 进程的 libc **syslog（）** 函数。 **UDP:<host>:<port>** 一个 syslog 服务器。将 **<host>:<port>** 替换为 **syslog 服务器的主机** 和端口。 **Unix:<file>** 由 **<file>** 指定的 Unix 域套接字文件。 **null** 不要将审计日志发送到任何其他目标。
syslogFacility	字符串	syslog 工具，如 as **kern**，如 RFC5424 定义。默认值为 **local0**。

表 7.9. **gatewayConfig** object

字段	类型	描述

字段	类型	描述
routingViaHost	**布尔值**	将此字段设置为 **true**，将来自 pod 的出口流量发送到主机网络堆栈。对于依赖于在内核路由表中手动配置路由的高级别安装和应用程序，您可能需要将出口流量路由到主机网络堆栈。默认情况下，出口流量在 OVN 中进行处理以退出集群，不受内核路由表中的特殊路由的影响。默认值为 **false**。 此字段与 Open vSwitch 硬件卸载功能有交互。如果将此字段设置为 **true**，则不会获得卸载的性能优势，因为主机网络堆栈会处理出口流量。
ipForwarding	**object**	您可以使用 **Network** 资源中的 **ipForwarding** 规格来控制 OVN-Kubernetes 管理接口上所有流量的 IP 转发。指定 **Restricted** 只允许 Kubernetes 相关流量的 IP 转发。指定 **Global** 以允许转发所有 IP 流量。对于新安装，默认值为 **Restricted**。对于到 OpenShift Container Platform 4.14 或更高版本的更新，默认值为 **Global**。
ipv4	**object**	可选：指定一个对象来为主机配置内部 OVN-Kubernetes 伪装地址，以服务 IPv4 地址的流量。
ipv6	**object**	可选：指定一个对象来为主机配置内部 OVN-Kubernetes 伪装地址，以服务 IPv6 地址的流量。

表 7.10. gatewayConfig.ipv4 对象

字段	类型	描述
internalMasqueradeSubnet	**字符串**	内部使用的伪装 IPv4 地址，以启用主机服务流量。主机配置了这些 IP 地址和共享网关网桥接口。默认值为 **169.254.169.0/29**。

表 7.11. gatewayConfig.ipv6 对象

字段	类型	描述
internalMasqueradeSubnet	**字符串**	内部使用的伪装 IPv6 地址，以启用主机服务流量。主机配置了这些 IP 地址和共享网关网桥接口。默认值为 **fd69::/125**。

表 7.12. ipsecConfig 对象

字段	类型	描述

字段	类型	描述
模式	**字符串**	指定 IPsec 实现的行为。必须是以下值之一： • **Disabled**: 在集群节点上不启用 IPsec。 • **External** ：对于带有外部主机的网络流量，启用 IPsec。 • **Full**: IPsec 为带有外部主机的 pod 流量和网络流量启用 IPsec。

启用 IPSec 的 OVN-Kubernetes 配置示例

```
defaultNetwork:
  type: OVNKubernetes
  ovnKubernetesConfig:
    mtu: 1400
    genevePort: 6081
      ipsecConfig:
        mode: Full
```

重要

使用 OVNKubernetes 可能会导致 IBM Power® 上的堆栈耗尽问题。

kubeProxyConfig 对象配置（仅限 OpenShiftSDN 容器网络接口）
kubeProxyConfig 对象的值在下表中定义：

表 7.13. kubeProxyConfig object

字段	类型	描述
iptablesSyncPeriod	**字符串**	**iptables** 规则的刷新周期。默认值为 **30s**。有效的后缀包括 **s**、**m** 和 **h**，具体参见 Go 时间 包 文档。  **注意** 由于 OpenShift Container Platform 4.3 及更高版本中引进了性能改进，不再需要调整 **iptablesSyncPeriod** 参数。

字段	类型	描述
proxyArguments.iptables-min-sync-period	**array**	刷新 **iptables** 规则前的最短持续时间。此字段确保刷新的频率不会过于频繁。有效的后缀包括 **s**、**m** 和 **h**，具体参见 Go **time** 软件包。默认值为： kubeProxyConfig: proxyArguments: iptables-min-sync-period: - 0s

7.6.9. 使用 OVN-Kubernetes 配置混合网络

您可以将集群配置为使用 OVN-Kubernetes 网络插件的混合网络。这允许支持不同节点网络配置的混合集群。

先决条件

- 您在 **install-config.yaml** 文件中为 **networking.networkType** 参数定义了 **OVNKubernetes**。如需更多信息，请参阅有关在所选云供应商上配置 OpenShift Container Platform 网络自定义的安装文档。

流程

1. 进入包含安装程序的目录并创建清单：

```
$ ./openshift-install create manifests --dir <installation_directory>
```

其中：

<installation_directory>
指定包含集群的 **install-config.yaml** 文件的目录名称。

2. 在 **<installation_directory>/manifests/** 目录中为高级网络配置创建一个名为 **cluster-network-03-config.yml** 的 stub 清单文件：

```
$ cat <<EOF > <installation_directory>/manifests/cluster-network-03-config.yml
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
EOF
```

其中：

<installation_directory>
指定包含集群的 **manifests/** 目录的目录名称。

3. 在编辑器中打开 **cluster-network-03-config.yml** 文件，并使用混合网络配置 OVN-Kubernetes，如下例所示：

指定混合网络配置

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  defaultNetwork:
    ovnKubernetesConfig:
      hybridOverlayConfig:
        hybridClusterNetwork: 1
        - cidr: 10.132.0.0/14
          hostPrefix: 23
```

1 指定用于额外覆盖网络上节点的 CIDR 配置。**hybridClusterNetwork** CIDR 无法与 **clusterNetwork** CIDR 重叠。

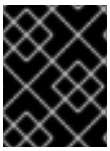
4. 保存 **cluster-network-03-config.yml** 文件，再退出文本编辑器。
5. 可选：备份 **manifests/cluster-network-03-config.yml** 文件。创建集群时，安装程序会删除 **manifests/** 目录。

其他资源

- 有关加速网络的详情，请参阅 Microsoft Azure 虚拟机的加速网络。

7.6.10. 安装 OpenShift CLI

您可以安装 OpenShift CLI(**oc**)来使用命令行界面与 OpenShift Container Platform 进行交互。您可以在 Linux、Windows 或 macOS 上安装 **oc**。



重要

如果安装了旧版本的 **oc**，则可能无法使用 OpenShift Container Platform 4.16 中的所有命令。下载并安装新版本的 **oc**。

在 Linux 上安装 OpenShift CLI
您可以按照以下流程在 Linux 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **产品变体** 下拉列表中选择架构。
3. 从 **版本** 下拉列表中选择适当的版本。
4. 点 **OpenShift v4.16 Linux Client** 条目旁的 **Download Now** 来保存文件。
5. 解包存档：

```
$ tar xvf <file>
```

6. 将 **oc** 二进制文件放到 **PATH 中的目录中**。
要查看您的 **PATH**，请执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

在 Windows 上安装 OpenShift CLI
您可以按照以下流程在 Windows 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 Windows Client** 条目旁的 **Download Now** 来保存文件。
4. 使用 ZIP 程序解压存档。
5. 将 **oc** 二进制文件移到 **PATH 中的目录中**。
要查看您的 **PATH**，请打开命令提示并执行以下命令：

```
C:\> path
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

在 macOS 上安装 OpenShift CLI
您可以按照以下流程在 macOS 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 macOS Client** 条目旁的 **Download Now** 来保存文件。

注意

对于 macOS arm64，请选择 **OpenShift v4.16 macOS arm64 Client** 条目。

4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。

要查看您的 **PATH**，请打开终端并执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

7.6.11. 在 kube-system 项目中存储管理员级别的 secret 的替代方案

默认情况下，管理员 secret 存储在 **kube-system** 项目中。如果您在 **install-config.yaml** 文件中将 **credentialsMode** 参数配置为 **Manual**，则必须使用以下替代方案之一：

- 要手动管理长期云凭证，请按照手动创建长期凭证中的步骤操作。
- 要实现在集群外为各个组件管理的短期凭证，请按照配置 Azure 集群以使用短期凭证中的步骤操作。

7.6.11.1. 手动创建长期凭证

在无法访问云身份和访问管理(IAM)API 的环境中，或者管理员更不希望将管理员级别的凭证 secret 存储在集群 **kube-system** 命名空间中时，可以在安装前将 Cloud Credential Operator(CCO)放入手动模式。

流程

1. 如果您没有将 **install-config.yaml** 配置文件中的 **credentialsMode** 参数设置为 **Manual**，请修改值，如下所示：

   配置文件片段示例

   ```
   apiVersion: v1
   baseDomain: example.com
   credentialsMode: Manual
   # ...
   ```

2. 如果您之前还没有创建安装清单文件，请运行以下命令：

   ```
   $ openshift-install create manifests --dir <installation_directory>
   ```

   其中 **<installation_directory>** 是安装程序在其中创建文件的目录。

3. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

   ```
   $ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
   ```

4. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 自定义资源 (CR) 列表：

   ```
   $ oc adm release extract \
     --from=$RELEASE_IMAGE \
     --credentials-requests \
   ```

```
--included \ 1
--install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
--to=<path_to_directory_for_credentials_requests> 3
```

1 **--included** 参数仅包含特定集群配置所需的清单。

2 指定 **install-config.yaml** 文件的位置。

3 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

此命令为每个 **CredentialsRequest** 对象创建一个 YAML 文件。

CredentialsRequest 对象示例

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component_credentials_request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: AzureProviderSpec
    roleBindings:
    - role: Contributor
  ...
```

5. 在之前生成的 **openshift-install** 清单目录中为 secret 创建 YAML 文件。secret 必须使用在 **spec.secretRef** 中为每个 **CredentialsRequest** 定义的命名空间和 secret 名称存储。

带有 secret 的 CredentialsRequest 对象示例

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component_credentials_request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: AzureProviderSpec
    roleBindings:
    - role: Contributor
      ...
  secretRef:
    name: <component_secret>
    namespace: <component_namespace>
  ...
```

Secret 对象示例

-

```
apiVersion: v1
kind: Secret
metadata:
  name: <component_secret>
  namespace: <component_namespace>
data:
  azure_subscription_id: <base64_encoded_azure_subscription_id>
  azure_client_id: <base64_encoded_azure_client_id>
  azure_client_secret: <base64_encoded_azure_client_secret>
  azure_tenant_id: <base64_encoded_azure_tenant_id>
  azure_resource_prefix: <base64_encoded_azure_resource_prefix>
  azure_resourcegroup: <base64_encoded_azure_resourcegroup>
  azure_region: <base64_encoded_azure_region>
```

重要

在升级使用手动维护凭证的集群前，您必须确保 CCO 处于可升级状态。

7.6.11.2. 配置 Azure 集群以使用短期凭证

要安装使用 Microsoft Entra Workload ID 的集群，您必须配置 Cloud Credential Operator 工具，并为集群创建所需的 Azure 资源。

7.6.11.2.1. 配置 Cloud Credential Operator 工具

当 Cloud Credential Operator(CCO)以手动模式运行时，要从集群外部创建和管理云凭证，提取并准备 CCO 实用程序(**ccoctl**)二进制文件。

注意

ccoctl 工具是在 Linux 环境中运行的 Linux 二进制文件。

先决条件

- 您可以访问具有集群管理员权限的 OpenShift Container Platform 帐户。
- 已安装 OpenShift CLI(**oc**)。
- 您已为 **ccoctl** 工具创建了全局 Microsoft Azure 帐户，用于以下权限：

  例 7.29. 所需的 Azure 权限

  - Microsoft.Resources/subscriptions/resourceGroups/read
  - Microsoft.Resources/subscriptions/resourceGroups/write
  - Microsoft.Resources/subscriptions/resourceGroups/delete
  - Microsoft.Authorization/roleAssignments/read
  - Microsoft.Authorization/roleAssignments/delete
  - Microsoft.Authorization/roleAssignments/write
  - Microsoft.Authorization/roleDefinitions/read

- Microsoft.Authorization/roleDefinitions/write
- Microsoft.Authorization/roleDefinitions/delete
- Microsoft.Storage/storageAccounts/listkeys/action
- Microsoft.Storage/storageAccounts/delete
- Microsoft.Storage/storageAccounts/read
- Microsoft.Storage/storageAccounts/write
- Microsoft.Storage/storageAccounts/blobServices/containers/write
- Microsoft.Storage/storageAccounts/blobServices/containers/delete
- Microsoft.Storage/storageAccounts/blobServices/containers/read
- Microsoft.ManagedIdentity/userAssignedIdentities/delete
- Microsoft.ManagedIdentity/userAssignedIdentities/read
- Microsoft.ManagedIdentity/userAssignedIdentities/write
- Microsoft.ManagedIdentity/userAssignedIdentities/federatedIdentityCredentials/read
- Microsoft.ManagedIdentity/userAssignedIdentities/federatedIdentityCredentials/write
- Microsoft.ManagedIdentity/userAssignedIdentities/federatedIdentityCredentials/delete
- Microsoft.Storage/register/action
- Microsoft.ManagedIdentity/register/action

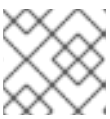
流程

1. 运行以下命令，为 OpenShift Container Platform 发行镜像设置变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. 运行以下命令，从 OpenShift Container Platform 发行镜像获取 CCO 容器镜像：

```
$ CCO_IMAGE=$(oc adm release info --image-for='cloud-credential-operator'
$RELEASE_IMAGE -a ~/.pull-secret)
```



注意

确保 **$RELEASE_IMAGE** 的架构与将使用 **ccoctl** 工具的环境架构相匹配。

3. 运行以下命令，将 CCO 容器镜像中的 **ccoctl** 二进制文件提取到 OpenShift Container Platform 发行镜像中：

```
$ oc image extract $CCO_IMAGE \
  --file="/usr/bin/ccoctl.<rhel_version>" \ 1
  -a ~/.pull-secret
```

1 对于 **<rhel_version>**，请指定与主机使用的 Red Hat Enterprise Linux (RHEL) 版本对应的值。如果没有指定值，则默认使用 **ccoctl.rhel8**。以下值有效：

- **rhel8**: 为使用 RHEL 8 的主机指定这个值。
- **rhel9** ：为使用 RHEL 9 的主机指定这个值。

4. 运行以下命令更改权限以使 **ccoctl** 可执行：

```
$ chmod 775 ccoctl.<rhel_version>
```

验证

- 要验证 **ccoctl** 是否可使用，请运行以下命令来显示帮助文件：

```
$ ccoctl --help
```

ccoctl --help 的输出

```
OpenShift credentials provisioning tool

Usage:
  ccoctl [command]

Available Commands:
  aws         Manage credentials objects for AWS cloud
  azure       Manage credentials objects for Azure
  gcp         Manage credentials objects for Google cloud
  help        Help about any command
  ibmcloud    Manage credentials objects for IBM Cloud
  nutanix     Manage credentials objects for Nutanix

Flags:
  -h, --help   help for ccoctl

Use "ccoctl [command] --help" for more information about a command.
```

7.6.11.2.2. 使用 Cloud Credential Operator 实用程序创建 Azure 资源

您可以使用 **ccoctl azure create-all** 命令自动创建 Azure 资源。

注意

默认情况下，**ccoctl** 在运行命令的目录中创建对象。要在其他目录中创建对象，请使用 **--output-dir** 标志。此流程使用 **<path_to_ccoctl_output_dir>** 来引用这个目录。

先决条件

您必须：

- 提取并准备好 **ccoctl** 二进制文件。
- 使用 Azure CLI 访问 Microsoft Azure 帐户。

流程

1. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 对象列表：

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
  --included \ 1
  --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
  --to=<path_to_directory_for_credentials_requests> 3
```

1. **--included** 参数仅包含特定集群配置所需的清单。
2. 指定 **install-config.yaml** 文件的位置。
3. 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

注意

此命令可能需要一些时间才能运行。

3. 要启用 **ccoctl** 工具自动检测 Azure 凭证，请运行以下命令登录到 Azure CLI：

```
$ az login
```

4. 运行以下命令，使用 **ccoctl** 工具处理所有 **CredentialsRequest** 对象：

```
$ ccoctl azure create-all \
  --name=<azure_infra_name> \ 1
  --output-dir=<ccoctl_output_dir> \ 2
  --region=<azure_region> \ 3
  --subscription-id=<azure_subscription_id> \ 4
  --credentials-requests-dir=<path_to_credentials_requests_directory> \ 5
  --dnszone-resource-group-name=<azure_dns_zone_resource_group_name> \ 6
  --tenant-id=<azure_tenant_id> 7
```

1. 为用于跟踪的所有创建 Azure 资源指定用户定义的名称。
2. 可选：指定您希望 **ccoctl** 实用程序在其中创建对象的目录。默认情况下，实用程序在运行命令的目录中创建对象。

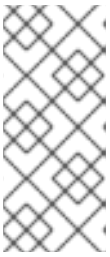
3 指定在其中创建云资源的 Azure 区域。

4 指定要使用的 Azure 订阅 ID。

5 指定包含组件 **CredentialsRequest** 对象文件的目录。

6 指定包含集群基域 Azure DNS 区的资源组名称。

7 指定要使用的 Azure 租户 ID。



注意

如果您的集群使用 **TechPreviewNoUpgrade** 功能集启用的技术预览功能，则必须包含 **--enable-tech-preview** 参数。

要查看其他可选参数以及如何使用它们的说明，请运行 **azure create-all --help** 命令。

验证

- 要验证 OpenShift Container Platform secret 是否已创建，列出 **<path_to_ccoctl_output_dir>/manifests** 目录中的文件：

```
$ ls <path_to_ccoctl_output_dir>/manifests
```

输出示例

```
azure-ad-pod-identity-webhook-config.yaml
cluster-authentication-02-config.yaml
openshift-cloud-controller-manager-azure-cloud-credentials-credentials.yaml
openshift-cloud-network-config-controller-cloud-credentials-credentials.yaml
openshift-cluster-api-capz-manager-bootstrap-credentials-credentials.yaml
openshift-cluster-csi-drivers-azure-disk-credentials-credentials.yaml
openshift-cluster-csi-drivers-azure-file-credentials-credentials.yaml
openshift-image-registry-installer-cloud-credentials-credentials.yaml
openshift-ingress-operator-cloud-credentials-credentials.yaml
openshift-machine-api-azure-cloud-credentials-credentials.yaml
```

您可以验证 Microsoft Entra ID 服务账户通过查询 Azure 而创建。如需更多信息，请参阅 Azure 文档中有关列出 Entra ID 服务帐户的内容。

7.6.11.2.3. 整合 Cloud Credential Operator 实用程序清单

要为单个组件在集群外实现短期安全凭证，您必须将创建 Cloud Credential Operator 实用程序 (**ccoctl**) 的清单文件移到安装程序的正确目录中。

先决条件

- 您已使用托管集群的云平台配置了帐户。
- 您已配置了 Cloud Credential Operator 实用程序 (**ccoctl**)。
- 已使用 **ccoctl** 工具创建了集群所需的云供应商资源。

流程

1. 如果您没有将 **install-config.yaml** 配置文件中的 **credentialsMode** 参数设置为 **Manual**，请修改值，如下所示：

   配置文件片段示例

   ```
   apiVersion: v1
   baseDomain: example.com
   credentialsMode: Manual
   # ...
   ```

2. 如果您使用 **ccoctl** 实用程序创建新的 Azure 资源组，而不是使用现有资源组，请修改 **install-config.yaml** 中的 **resourceGroupName** 参数，如下所示：

   配置文件片段示例

   ```
   apiVersion: v1
   baseDomain: example.com
   # ...
   platform:
     azure:
       resourceGroupName: <azure_infra_name> 1
   # ...
   ```

   1 这个值必须与 **ccoctl azure create-all** 命令的 **--name** 参数指定的 Azure 资源用户定义的名称匹配。

3. 如果您之前还没有创建安装清单文件，请运行以下命令：

   ```
   $ openshift-install create manifests --dir <installation_directory>
   ```

   其中 **<installation_directory>** 是安装程序在其中创建文件的目录。

4. 运行以下命令，将 **ccoctl** 工具生成的清单复制到安装程序创建的 **manifests** 目录中：

   ```
   $ cp /<path_to_ccoctl_output_dir>/manifests/* ./manifests/
   ```

5. 运行以下命令，将 **ccoctl** 工具在 **tls** 目录中生成的私钥复制到安装目录中：

   ```
   $ cp -a /<path_to_ccoctl_output_dir>/tls .
   ```

7.6.12. 部署集群

您可以在兼容云平台上安装 OpenShift Container Platform。

重要

在初始安装过程中，您只能运行安装程序的 **create cluster** 命令一次。

先决条件

- 您已使用托管集群的云平台配置了帐户。
- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。
- 您有一个 Azure 订阅 ID 和租户 ID。

流程

- 进入包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定自定义 **./install-config.yaml** 文件的位置。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不是 **info**。

验证

当集群部署成功完成时：

- 终端会显示用于访问集群的说明，包括指向 Web 控制台和 **kubeadmin** 用户的凭证的链接。
- 凭证信息还会输出到 **<installation_directory>/.openshift_install.log**.

重要

不要删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

7.6.13. 使用 CLI 登录集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

其他资源

- 如需有关 访问和了解 OpenShift Container Platform Web 控制台的更多详情，请参阅 访问 Web 控制台。

7.6.14. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于 远程健康监控

7.6.15. 后续步骤

- 自定义集群。
- 如果需要，您可以选择不使用远程健康报告。

7.7. 将 AZURE 上的集群安装到现有的 VNET

在 OpenShift Container Platform 版本 4.16 中，您可以在 Microsoft Azure 上将集群安装到现有 Azure Virtual Network（VNet）中。安装程序会置备所需基础架构的其余部分，您可以进一步自定义这些基础架构。要自定义安装，请在安装集群前修改 **install-config.yaml** 文件中的参数。

7.7.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读选择集群安装方法并为用户准备它的文档。
- 您已将 Azure 帐户配置为托管集群，并决定要将集群部署到的已测试和验证的区域。
- 如果使用防火墙，将其配置为允许集群需要访问的站点。
- 如果您使用客户管理的加密密钥，准备了用于加密的 Azure 环境。

7.7.2. 关于为 OpenShift Container Platform 集群重复使用 VNet

在 OpenShift Container Platform 4.16 中，您可以在 Microsoft Azure 中将集群部署到现有的 Azure Virtual Network(VNet)中。如果您这样做，还必须在 VNet 和路由规则中使用现有子网。

通过将 OpenShift Container Platform 部署到现有的 Azure VNet 中，您可以避免新帐户中的服务限制，或者更容易地利用公司所设置的操作限制。如果您无法获得创建 VNet 所需的基础架构创建权限，则可以使用这个选项。

7.7.2.1. 使用 VNet 的要求

当使用现有 VNet 部署集群时，必须在安装集群前执行额外的网络配置。在安装程序置备的基础架构集群中，安装程序通常会创建以下组件，但在安装到现有 VNet 时不会创建它们：

- 子网
- 路由表
- VNets
- 网络安全组

注意

安装程序要求您使用由云提供的 DNS 服务器。不支持使用自定义 DNS 服务器，并导致安装失败。

如果使用自定义 VNet，您必须正确配置它及其子网，供安装程序和集群使用。安装程序不能为集群分配要使用的网络范围，为子网设置路由表，或者设置类似 DHCP 的 VNet 选项，因此您必须在安装集群前这样做。

集群必须能够访问包含现有 VNet 和子网的资源组。虽然集群创建的所有资源都放在它创建的单独资源组中，但有些网络资源则从单独的组中使用。有些集群 Operator 必须能够访问这两个资源组中的资源。例如，Machine API 控制器会为它创建的虚拟机附加 NICS，以便从网络资源组中进行子网。

您的 VNet 必须满足以下特征：

- VNet 的 CIDR 块必须包含 **Networking.MachineCIDR** 范围，它是集群机器的 IP 地址池。

- VNet 及其子网必须属于同一资源组，子网必须配置为使用 Azure 分配的 DHCP IP 地址，而不是静态 IP 地址。

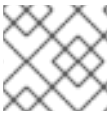
您必须在 VNet 中提供两个子网，一个用于 control plane 机器，一个用于计算机器。因为 Azure 在您指定的区域内的不同可用区中分发机器，所以集群将默认具有高可用性。

注意

默认情况下，如果您在 **install-config.yaml** 文件中指定可用区，安装程序会在一个区（region）内的这些可用区间分发 control plane 机器和计算机器。要确保集群的高可用性，请选择至少含有三个可用区的区域。如果您的区域包含的可用区少于三个，安装程序将在可用区中放置多台 control plane 机器。

为确保您提供的子网适合，安装程序会确认以下数据：

- 所有指定的子网都存在。
- 有两个专用子网，一个用于 control plane 机器，一个用于计算机器。
- 子网 CIDR 属于您指定的机器 CIDR。机器不会在您不为其提供私有子网的可用区中置备。如果需要，安装程序会创建管理 control plane 和 worker 节点的公共负载均衡器，Azure 会为其分配一个公共 IP 地址。



注意

如果您销毁了使用现有 VNet 的集群，则不会删除 VNet。

7.7.2.1.1. 网络安全组要求

托管 compute 和 control plane 机器的子网的网络安全组需要特定的访问权限，以确保集群通信正确。您必须创建规则以允许访问所需的集群通信端口。

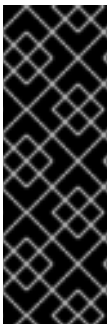
重要

在安装集群前，必须先设置网络安全组规则。如果您试图在没有所需访问权限的情况下安装集群，安装程序无法访问 Azure API，安装会失败。

表 7.14. 所需端口

port	描述	Control plane（控制平面）	Compute
80	允许 HTTP 流量		x
443	允许 HTTPS 流量		x
6443	允许与 control plane 机器通信	x	
22623	允许内部与机器配置服务器通信以用于置备机器	x	

1. 如果使用 Azure Firewall 来限制互联网访问，您可以将 Azure Firewall 配置为允许 Azure API。个需要网络安全组规则。



重要

目前，不支持阻止或限制机器配置服务器端点。机器配置服务器必须公开给网络，以便新置备的机器没有现有配置或状态，才能获取其配置。在这个模型中，信任的根是证书签名请求 (CSR) 端点，即 kubelet 发送其证书签名请求以批准加入集群。因此，机器配置不应用于分发敏感信息，如 secret 和证书。

为确保机器配置服务器端点，端口 22623 和 22624 在裸机场景中是安全的，客户必须配置正确的网络策略。

由于集群组件不会修改 Kubernetes 控制器更新的用户提供的网络安全组，因此为 Kubernetes 控制器在不影响其余环境的情况下创建一个伪网络安全组。

其他资源

- 关于 OpenShift SDN 网络插件
- 配置防火墙

7.7.2.2. 权限划分

从 OpenShift Container Platform 4.3 开始，您不需要安装程序置备的基础架构集群部署所需的所有权限。这与您所在机构可能拥有的权限划分类似：一些个人可以在您的云中创建不同的资源。例如，您可以创建特定于应用程序的对象，如实例、存储和负载均衡器，但不能创建与网络相关的组件，如 VNets、子网或入站规则。

您在创建集群时使用的 Azure 凭证不需要 VNets 和核心网络组件（如子网、路由表、互联网网关、NAT 和 VPN）所需的网络权限。您仍然需要获取集群中的机器需要的应用程序资源的权限，如负载均衡器、安全组、存储帐户和节点。

7.7.2.3. 集群间隔离

因为集群无法修改现有子网中的网络安全组，所以无法在 VNet 中相互隔离集群。

7.7.3. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类型的基础架构上执行受限网络安装。在此过程中，您可以下载所需的内容，并使用它为镜像 registry 填充安装软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群前，您要更新镜像 registry 的内容。

7.7.4. 为集群节点 SSH 访问生成密钥对

在 OpenShift Container Platform 安装过程中，您可以为安装程序提供 SSH 公钥。密钥通过它们的 Ignition 配置文件传递给 Red Hat Enterprise Linux CoreOS(RHCOS)节点，用于验证对节点的 SSH 访问。密钥添加到每个节点上 **core** 用户的 **~/.ssh/authorized_keys** 列表中，这将启用免密码身份验证。

将密钥传递给节点后，您可以使用密钥对作为用户 **核心** 通过 SSH 连接到 RHCOS 节点。若要通过 SSH 访问节点，必须由 SSH 为您的本地用户管理私钥身份。

如果要通过 SSH 连接到集群节点来执行安装调试或灾难恢复，则必须在安装过程中提供 SSH 公钥。**./openshift-install gather** 命令还需要在集群节点上设置 SSH 公钥。

重要

不要在生产环境中跳过这个过程，在生产环境中需要灾难恢复和调试。

注意

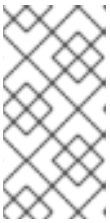
您必须使用本地密钥，而不是使用特定平台方法配置 的密钥，如 AWS 密钥对。

流程

1. 如果您在本地计算机上没有可用于在集群节点上进行身份验证的现有 SSH 密钥对，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_ed25519**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。



注意

如果您计划在 **x86_64**、**ppc64le** 和 **s390x** 架构上安装使用 RHEL 加密库（这些加密库已提交给 NIST 用于 FIPS 140-2/140-3 验证）的 OpenShift Container Platform 集群，则不要创建使用 **ed25519** 算法的密钥。相反，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 查看公共 SSH 密钥：

```
$ cat <path>/<file_name>.pub
```

例如，运行以下命令来查看 **~/.ssh/id_ed25519.pub** 公钥：

```
$ cat ~/.ssh/id_ed25519.pub
```

3. 将 SSH 私钥身份添加到本地用户的 SSH 代理（如果尚未添加）。在集群节点上，或者要使用 **./openshift-install gather** 命令，需要对该密钥进行 SSH 代理管理，才能在集群节点上进行免密码 SSH 身份验证。

注意

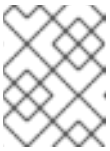
在某些发行版中，自动管理默认 SSH 私钥身份，如 **~/.ssh/id_rsa** 和 **~/.ssh/id_dsa**。

a. 如果 **ssh-agent** 进程尚未为您的本地用户运行，请将其作为后台任务启动：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```



注意

如果集群处于 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

4. 将 SSH 私钥添加到 **ssh-agent** ：

```
$ ssh-add <path>/<file_name> 1
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_ed25519.pub**

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

后续步骤

- 安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

7.7.5. 获取安装程序

在安装 OpenShift Container Platform 前，将安装文件下载到您用于安装的主机上。

先决条件

- 您有一台运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB。

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐户，请使用您的凭证登录。如果没有，请创建一个帐户。

2. 选择您的基础架构供应商。

3. 进入到安装类型的页面，下载与您的主机操作系统和架构对应的安装程序，并将该文件放在您要存储安装配置文件的目录中。

重要

安装程序会在用来安装集群的计算机上创建几个文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。

重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，请为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。此 pull secret 允许您与所含授权机构提供的服务进行身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

7.7.6. 创建安装配置文件

您可以自定义在 Microsoft Azure 上安装的 OpenShift Container Platform 集群。

先决条件

- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。
- 您有一个 Azure 订阅 ID 和租户 ID。
- 如果要使用服务主体安装集群，则有其应用程序 ID 和密码。
- 如果您要使用系统分配的受管身份安装集群，需要在您要从其中运行安装程序的虚拟机上启用它。
- 如果您要使用用户分配的受管身份安装集群，需要满足以下先决条件：
  - 您有它的客户端 ID。
  - 您已将其分配给您要从其运行安装程序的虚拟机。

流程

1. 可选：如果您之前在这个计算机上运行安装程序，并希望使用替代的服务主体或受管身份，请进入 ~/**.azure**/ 目录并删除 **osServicePrincipal.json** 配置文件。
删除此文件可防止安装程序自动重复使用之前安装中的订阅和验证值。

2. 创建 **install-config.yaml** 文件。

   a. 进入包含安装程序的目录并运行以下命令：

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定要存储安装程序创建的文件的目录名称。

在指定目录时：

- 验证该目录是否具有**执行**权限。在安装目录中运行 Terraform 二进制文件需要这个权限。
- 使用空目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

b. 在提示符处，提供云的配置详情：

i. 可选：选择用于访问集群机器的 SSH 密钥。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

ii. 选择 **azure** 作为目标平台。
如果安装程序无法找到之前安装中的 **osServicePrincipal.json** 配置文件，会提示您输入 Azure 订阅和验证值。

iii. 为您的订阅输入以下 Azure 参数值：

- **Azure subscription id**：输入用于集群的订阅 ID。
- **Azure 租户 id**：输入租户 ID。

iv. 根据您用来部署集群的 Azure 身份，在提示输入 **azure 服务主体客户端 id**时执行以下操作之一：

- 如果您使用服务主体，请输入其应用程序 ID。
- 如果您使用系统分配的受管身份，请将此值设为空白。
- 如果您使用用户分配的受管身份，请指定其客户端 ID。

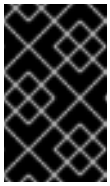
v. 根据您用来部署集群的 Azure 身份，在提示输入 **azure 服务主体客户端 secret**时执行以下操作之一：

- 如果您使用服务主体，请输入其密码。
- 如果您使用系统分配的受管身份，请将此值设为空白。
- 如果您使用用户分配的受管身份，请将此值设为空白。

vi. 选择要将集群部署到的区域。

vii. 选择集群要部署到的基域。基域与您为集群创建的 Azure DNS 区对应。

viii. 为集群输入一个描述性名称。



重要

所有通过公共端点提供的 Azure 资源均存在资源名称的限制，您无法创建使用某些名称的资源。如需 Azure 限制词语的列表，请参阅 Azure 文档中的解决预留资源名称错误。

3. 修改 **install-config.yaml** 文件。您可以在"安装配置参数"部分找到有关可用参数的更多信息。
4. 备份 **install-config.yaml** 文件，以便您可以使用它安装多个集群。

重要

install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用该文件，您必须立即备份该文件。

在以前的版本中，安装程序会创建一个 **osServicePrincipal.json** 配置文件，并将此文件存储在计算机上的 **~/.azure/** 目录中。这样可确保安装程序在目标平台上创建 OpenShift Container Platform 集群时可以加载配置集。

其他资源

- Azure 的安装配置参数

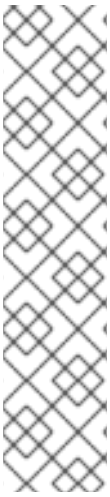
7.7.6.1. 集群安装的最低资源要求

每台集群机器都必须满足以下最低要求：

表 7.15. 最低资源要求

机器	操作系统	vCPU [1]	虚拟内存	Storage	每秒输入/输出 (IOPS) [2]
bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane（控制平面）	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS、RHEL 8.6 及更新版本 [3]	2	8 GB	100 GB	300

1. 当未启用并发多线程(SMT)或超线程时，一个 vCPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比例：（每个内核数的线程）× sockets = vCPU。
2. OpenShift Container Platform 和 Kubernetes 对磁盘性能非常敏感，建议使用更快的存储速度，特别是 control plane 节点上需要 10 ms p99 fsync 持续时间的 etcd。请注意，在许多云平台上，存储大小和 IOPS 可一起扩展，因此您可能需要过度分配存储卷来获取足够的性能。
3. 与所有用户置备的安装一样，如果您选择在集群中使用 RHEL 计算机器，则负责所有操作系统生命周期管理和维护，包括执行系统更新、应用补丁和完成所有其他必要的任务。RHEL 7 计算机器的使用已弃用，并已在 OpenShift Container Platform 4.10 及更新的版本中删除。



注意

从 OpenShift Container Platform 版本 4.13 开始，RHCOS 基于 RHEL 版本 9.2，它更新了微架构要求。以下列表包含每个架构需要的最小指令集架构 (ISA)：

- x86-64 体系结构需要 x86-64-v2 ISA
- ARM64 架构需要 ARMv8.0-A ISA
- IBM Power 架构需要 Power 9 ISA
- s390x 架构需要 z14 ISA

如需更多信息，请参阅 RHEL 架构。

重要

您需要使用将 **PremiumIO** 参数设置为 **true** 的 Azure 虚拟机。

如果平台的实例类型满足集群机器的最低要求，则 OpenShift Container Platform 支持使用它。

其他资源

- 优化存储

7.7.6.2. 为 Azure 测试的实例类型

以下 Microsoft Azure 实例类型已经 OpenShift Container Platform 测试。

例 7.30. 基于 64 位 x86 架构的机器类型

- **c4.***
- **c5.***
- **c5a.***
- **i3.***
- **m4.***
- **m5.***
- **m5a.***
- **m6a.***
- **m6i.***
- **r4.***
- **r5.***
- **r5a.***
- **r6i.***

- **t3.***
- **t3a.***

7.7.6.3. 在 64 位 ARM 基础架构上为 Azure 测试的实例类型

以下 Microsoft Azure ARM64 实例类型已使用 OpenShift Container Platform 测试。

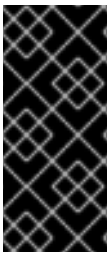
例 7.31. 基于 64 位 ARM 架构的机器类型

- **c6g.***
- **m6g.***

7.7.6.4. 为 Azure 虚拟机启用可信启动

在 Azure 上安装集群时，您可以启用两个可信启动功能： 安全引导 和虚拟化可信平台模块。

请参阅 Azure 文档中有关 虚拟机大小 的信息，以了解虚拟机支持这些功能的大小。



重要

可信启动只是一个技术预览功能。技术预览功能不受红帽产品服务等级协议（SLA）支持，且功能可能并不完整。红帽不推荐在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。

有关红帽技术预览功能支持范围的更多信息，请参阅技术预览功能支持范围。

先决条件

- 您已创建了 **install-config.yaml** 文件。

流程

- 在部署集群前，使用文本编辑器编辑 **install-config.yaml** 文件并添加以下小节：

```
controlPlane: 1
  platform:
    azure:
      settings:
        securityType: TrustedLaunch 2
        trustedLaunch:
          uefiSettings:
            secureBoot: Enabled 3
            virtualizedTrustedPlatformModule: Enabled 4
```

1 指定 **controlPlane.platform.azure** 或 **compute.platform.azure**，以分别在 control plane 或计算节点上启用可信启动。指定 **platform.azure.defaultMachinePlatform**，以便在所有节点上启用可信启动。

2 启用可信启动功能。

3 启用安全引导。如需更多信息，请参阅 Azure 文档中有关安全引导的内容。

4 启用虚拟化可信平台模块。如需更多信息，请参阅 Azure 文档中有关 虚拟化可信平台模块的文档。

7.7.6.5. 启用机密虚拟机

您可在安装集群前启用机密虚拟机。您可以为计算节点、control plane 节点或所有节点启用机密虚拟机。

重要

使用机密虚拟机只是一个技术预览功能。技术预览功能不受红帽产品服务等级协议（SLA）支持，且功能可能并不完整。红帽不推荐在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。

有关红帽技术预览功能支持范围的更多信息，请参阅技术预览功能支持范围。

您可以使用带有以下虚拟机大小的机密虚拟机：

- DCasv5-series
- DCadsv5-series
- ECasv5-series
- ECadsv5-series

重要

64 位 ARM 架构目前不支持机密虚拟机。

先决条件

- 您已创建了 **install-config.yaml** 文件。

流程

- 在部署集群前，使用文本编辑器编辑 **install-config.yaml** 文件并添加以下小节：

```
controlPlane: 1
  platform:
    azure:
      settings:
        securityType: ConfidentialVM 2
        confidentialVM:
          uefiSettings:
            secureBoot: Enabled 3
            virtualizedTrustedPlatformModule: Enabled 4
      osDisk:
        securityProfile:
          securityEncryptionType: VMGuestStateOnly 5
```

1 指定 **controlPlane.platform.azure** 或 **compute.platform.azure**，以分别在 control plane

或计算节点上部署机密虚拟机。指定 **platform.azure.defaultMachinePlatform** 以在所有节点上部署机密虚拟机。

2 启用机密虚拟机。

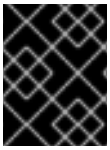
3 启用安全引导。如需更多信息，请参阅 Azure 文档中有关安全引导的内容。

4 启用虚拟化可信平台模块。如需更多信息，请参阅 Azure 文档中有关 虚拟化可信平台模块的文档。

5 指定 **VMGuestStateOnly** 来加密虚拟机客户端状态。

7.7.6.6. Azure 的自定义 install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多详情，或修改所需参数的值。



重要

此示例 YAML 文件仅供参考。您必须使用安装程序来获取 **install-config.yaml** 文件，并进行修改。

```
apiVersion: v1
baseDomain: example.com 1
controlPlane: 2
  hyperthreading: Enabled 3 4
  name: master
  platform:
    azure:
      encryptionAtHost: true
      ultraSSDCapability: Enabled
      osDisk:
        diskSizeGB: 1024 5
        diskType: Premium_LRS
        diskEncryptionSet:
          resourceGroup: disk_encryption_set_resource_group
          name: disk_encryption_set_name
          subscriptionId: secondary_subscription_id
      osImage:
        publisher: example_publisher_name
        offer: example_image_offer
        sku: example_offer_sku
        version: example_image_version
      type: Standard_D8s_v3
  replicas: 3
compute: 6
- hyperthreading: Enabled 7
  name: worker
  platform:
    azure:
      ultraSSDCapability: Enabled
      type: Standard_D2s_v3
      encryptionAtHost: true
      osDisk:
```

```
        diskSizeGB: 512 8
        diskType: Standard_LRS
        diskEncryptionSet:
          resourceGroup: disk_encryption_set_resource_group
          name: disk_encryption_set_name
          subscriptionId: secondary_subscription_id
      osImage:
        publisher: example_publisher_name
        offer: example_image_offer
        sku: example_offer_sku
        version: example_image_version
      zones: 9
      - "1"
      - "2"
      - "3"
  replicas: 5
metadata:
  name: test-cluster 10
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 11
  serviceNetwork:
  - 172.30.0.0/16
platform:
  azure:
    defaultMachinePlatform:
      osImage: 12
        publisher: example_publisher_name
        offer: example_image_offer
        sku: example_offer_sku
        version: example_image_version
      ultraSSDCapability: Enabled
    baseDomainResourceGroupName: resource_group 13
    region: centralus 14
    resourceGroupName: existing_resource_group 15
    networkResourceGroupName: vnet_resource_group 16
    virtualNetwork: vnet 17
    controlPlaneSubnet: control_plane_subnet 18
    computeSubnet: compute_subnet 19
    outboundType: Loadbalancer
    cloudName: AzurePublicCloud
pullSecret: '{"auths": ...}' 20
fips: false 21
sshKey: ssh-ed25519 AAAA... 22
```

1 10 14 20 必需。安装程序会提示您输入这个值。

2 6 如果没有提供这些参数和值，安装程序会提供默认值。

3 7

controlPlane 部分是一个单个映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，controlPlane **部分** 的第一行则不以连字符开头。仅使用一个 control plane 池。

4 是否要启用或禁用并发多线程或 **超线程**。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将 参数值设置为 **Disabled** 来禁用它。如果在某些集群机器中禁用并发多线程，则必须在所有集群机器中禁用它。

重要

如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。如果您禁用并发多线程，请为您的机器使用较大的虚拟机类型，如 **Standard_D8s_v3**。

5 8 您可以指定要使用的磁盘大小（以 GB 为单位）。control plane 节点的最低推荐值为 1024 GB。

9 指定要将机器部署到的区域列表。如需高可用性，请至少指定两个区域。

11 要安装的集群网络插件。默认值 **OVNKubernetes** 是唯一支持的值。

12 可选：应该用来引导 control plane 和计算机器的自定义 Red Hat Enterprise Linux CoreOS (RHCOS) 镜像。**platform.azure.defaultMachinePlatform.osImage** 下的 **publisher**, **offer**, **sku**, 和 **version** 参数应用到 control plane 和计算机器。如果设置了 **controlPlane.platform.azure.osImage** 或 **compute.platform.azure.osImage** 下的参数，它们会覆盖 **platform.azure.defaultMachinePlatform.osImage** 参数。

13 指定包含基域的 DNS 区的资源组的名称。

15 指定要安装集群的现有资源组的名称。如果未定义，则会为集群创建新的资源组。

16 如果使用现有的 VNet，请指定包含它的资源组的名称。

17 如果使用现有的 VNet，请指定其名称。

18 如果使用现有的 VNet，请指定托管 control plane 机器的子网名称。

19 如果使用现有的 VNet，请指定托管计算机器的子网名称。

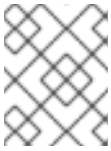
21 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 Red Hat Enterprise Linux (RHEL) 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅在 FIPS 模式中安装该系统。

当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS)时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。

22 您可以选择提供您用来访问集群中机器的 **sshKey** 值。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

7.7.6.7. 在安装过程中配置集群范围的代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 **Proxy** 对象的 **spec.noProxy** 字段中添加站点来绕过代理。

注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，**Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

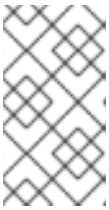
```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 **.** 以仅匹配子域。例如，**.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代理。
4. 如果提供，安装程序会在 **openshift-config** 命名空间中生成名为 **user-ca-bundle** 的配置映

5 可选：决定 **Proxy** 对象的配置以引用 **trustedCA** 字段中 **user-ca-bundle** 配置映射的策略。允许的值是 **Proxyonly** 和 **Always**。仅在配置了 **http/https** 代理时，使用 **Proxyonly**

注意

安装程序不支持代理的 **readinessEndpoints** 字段。



注意

如果安装程序超时，重启并使用安装程序的 **wait-for** 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. 保存该文件并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 cluster 的集群范围代理，该代理 **使用** 提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

注意

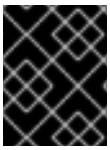
只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

其他资源

- 有关加速网络的详情，请参阅 Microsoft Azure 虚拟机的加速网络。

7.7.7. 安装 OpenShift CLI

您可以安装 OpenShift CLI(**oc**)来使用命令行界面与 OpenShift Container Platform 进行交互。您可以在 Linux、Windows 或 macOS 上安装 **oc**。



重要

如果安装了旧版本的 **oc**，则可能无法使用 OpenShift Container Platform 4.16 中的所有命令。下载并安装新版本的 **oc**。

在 Linux 上安装 OpenShift CLI
您可以按照以下流程在 Linux 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **产品变体** 下拉列表中选择架构。
3. 从 **版本** 下拉列表中选择适当的版本。
4. 点 **OpenShift v4.16 Linux Client** 条目旁的 **Download Now** 来保存文件。
5. 解包存档：

```
$ tar xvf <file>
```

6. 将 **oc** 二进制文件放到 **PATH 中的目录中**。
   要查看您的 **PATH**，请执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

在 Windows 上安装 OpenShift CLI
您可以按照以下流程在 Windows 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 Windows Client** 条目旁的 **Download Now** 来保存文件。
4. 使用 ZIP 程序解压存档。
5. 将 **oc** 二进制文件移到 **PATH 中的目录中**。
   要查看您的 **PATH**，请打开命令提示并执行以下命令：

```
C:\> path
```

验证

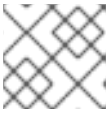
- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

在 macOS 上安装 OpenShift CLI
您可以按照以下流程在 macOS 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 macOS Client** 条目旁的 **Download Now** 来保存文件。



注意

对于 macOS arm64，请选择 **OpenShift v4.16 macOS arm64 Client** 条目。

4. 解包和解压存档。

5. 将 **oc** 二进制文件移到 PATH 的目录中。
要查看您的 **PATH**，请打开终端并执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

7.7.8. 在 kube-system 项目中存储管理员级别的 secret 的替代方案

默认情况下，管理员 secret 存储在 **kube-system** 项目中。如果您在 **install-config.yaml** 文件中将 **credentialsMode** 参数配置为 **Manual**，则必须使用以下替代方案之一：

- 要手动管理长期云凭证，请按照手动创建长期凭证中的步骤操作。
- 要实现在集群外为各个组件管理的短期凭证，请按照 配置 Azure 集群以使用短期凭证中的步骤操作。

7.7.8.1. 手动创建长期凭证

在无法访问云身份和访问管理(IAM)API 的环境中，或者管理员更不希望将管理员级别的凭证 secret 存储在集群 **kube-system** 命名空间中时，可以在安装前将 Cloud Credential Operator(CCO)放入手动模式。

流程

1. 如果您没有将 **install-config.yaml** 配置文件中的 **credentialsMode** 参数设置为 **Manual**，请修改值，如下所示：

   配置文件片段示例

```
apiVersion: v1
baseDomain: example.com
credentialsMode: Manual
# ...
```

2. 如果您之前还没有创建安装清单文件，请运行以下命令：

```
$ openshift-install create manifests --dir <installation_directory>
```

   其中 **<installation_directory>** 是安装程序在其中创建文件的目录。

3. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

4. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 自定义资源 (CR) 列表：

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
```

```
--credentials-requests \
--included \ 1
--install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
--to=<path_to_directory_for_credentials_requests> 3
```

1. **--included** 参数仅包含特定集群配置所需的清单。
2. 指定 **install-config.yaml** 文件的位置。
3. 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

此命令为每个 **CredentialsRequest** 对象创建一个 YAML 文件。

CredentialsRequest 对象示例

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component_credentials_request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: AzureProviderSpec
    roleBindings:
    - role: Contributor
  ...
```

5. 在之前生成的 **openshift-install** 清单目录中为 secret 创建 YAML 文件。secret 必须使用在 **spec.secretRef** 中为每个 **CredentialsRequest** 定义的命名空间和 secret 名称存储。

带有 secret 的 CredentialsRequest 对象示例

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component_credentials_request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: AzureProviderSpec
    roleBindings:
    - role: Contributor
      ...
  secretRef:
    name: <component_secret>
    namespace: <component_namespace>
  ...
```

Secret 对象示例

```
apiVersion: v1
kind: Secret
metadata:
  name: <component_secret>
  namespace: <component_namespace>
data:
  azure_subscription_id: <base64_encoded_azure_subscription_id>
  azure_client_id: <base64_encoded_azure_client_id>
  azure_client_secret: <base64_encoded_azure_client_secret>
  azure_tenant_id: <base64_encoded_azure_tenant_id>
  azure_resource_prefix: <base64_encoded_azure_resource_prefix>
  azure_resourcegroup: <base64_encoded_azure_resourcegroup>
  azure_region: <base64_encoded_azure_region>
```

重要

在升级使用手动维护凭证的集群前，您必须确保 CCO 处于可升级状态。

7.7.8.2. 配置 Azure 集群以使用短期凭证

要安装使用 Microsoft Entra Workload ID 的集群，您必须配置 Cloud Credential Operator 工具，并为集群创建所需的 Azure 资源。

7.7.8.2.1. 配置 Cloud Credential Operator 工具

当 Cloud Credential Operator(CCO)以手动模式运行时，要从集群外部创建和管理云凭证，提取并准备 CCO 实用程序(**ccoctl**)二进制文件。

注意

ccoctl 工具是在 Linux 环境中运行的 Linux 二进制文件。

先决条件

- 您可以访问具有集群管理员权限的 OpenShift Container Platform 帐户。
- 已安装 OpenShift CLI(**oc**)。
- 您已为 **ccoctl** 工具创建了全局 Microsoft Azure 帐户，用于以下权限：

  例 7.32. 所需的 Azure 权限

  - Microsoft.Resources/subscriptions/resourceGroups/read
  - Microsoft.Resources/subscriptions/resourceGroups/write
  - Microsoft.Resources/subscriptions/resourceGroups/delete
  - Microsoft.Authorization/roleAssignments/read
  - Microsoft.Authorization/roleAssignments/delete
  - Microsoft.Authorization/roleAssignments/write
  - Microsoft.Authorization/roleDefinitions/read

- Microsoft.Authorization/roleDefinitions/write
- Microsoft.Authorization/roleDefinitions/delete
- Microsoft.Storage/storageAccounts/listkeys/action
- Microsoft.Storage/storageAccounts/delete
- Microsoft.Storage/storageAccounts/read
- Microsoft.Storage/storageAccounts/write
- Microsoft.Storage/storageAccounts/blobServices/containers/write
- Microsoft.Storage/storageAccounts/blobServices/containers/delete
- Microsoft.Storage/storageAccounts/blobServices/containers/read
- Microsoft.ManagedIdentity/userAssignedIdentities/delete
- Microsoft.ManagedIdentity/userAssignedIdentities/read
- Microsoft.ManagedIdentity/userAssignedIdentities/write
- Microsoft.ManagedIdentity/userAssignedIdentities/federatedIdentityCredentials/read
- Microsoft.ManagedIdentity/userAssignedIdentities/federatedIdentityCredentials/write
- Microsoft.ManagedIdentity/userAssignedIdentities/federatedIdentityCredentials/delete
- Microsoft.Storage/register/action
- Microsoft.ManagedIdentity/register/action

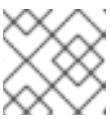
流程

1. 运行以下命令，为 OpenShift Container Platform 发行镜像设置变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. 运行以下命令，从 OpenShift Container Platform 发行镜像获取 CCO 容器镜像：

```
$ CCO_IMAGE=$(oc adm release info --image-for='cloud-credential-operator'
$RELEASE_IMAGE -a ~/.pull-secret)
```



注意

确保 **$RELEASE_IMAGE** 的架构与将使用 **ccoctl** 工具的环境架构相匹配。

3. 运行以下命令，将 CCO 容器镜像中的 **ccoctl** 二进制文件提取到 OpenShift Container Platform 发行镜像中：

```
$ oc image extract $CCO_IMAGE \
  --file="/usr/bin/ccoctl.<rhel_version>" \ 1
  -a ~/.pull-secret
```

1 对于 **<rhel_version>**，请指定与主机使用的 Red Hat Enterprise Linux (RHEL) 版本对应的值。如果没有指定值，则默认使用 **ccoctl.rhel8**。以下值有效：

- **rhel8**: 为使用 RHEL 8 的主机指定这个值。
- **rhel9** ：为使用 RHEL 9 的主机指定这个值。

4. 运行以下命令更改权限以使 **ccoctl** 可执行：

```
$ chmod 775 ccoctl.<rhel_version>
```

验证

- 要验证 **ccoctl** 是否可使用，请运行以下命令来显示帮助文件：

```
$ ccoctl --help
```

ccoctl --help 的输出

```
OpenShift credentials provisioning tool

Usage:
  ccoctl [command]

Available Commands:
  aws         Manage credentials objects for AWS cloud
  azure       Manage credentials objects for Azure
  gcp         Manage credentials objects for Google cloud
  help        Help about any command
  ibmcloud    Manage credentials objects for IBM Cloud
  nutanix     Manage credentials objects for Nutanix

Flags:
  -h, --help   help for ccoctl

Use "ccoctl [command] --help" for more information about a command.
```

7.7.8.2.2. 使用 Cloud Credential Operator 实用程序创建 Azure 资源

您可以使用 **ccoctl azure create-all** 命令自动创建 Azure 资源。

注意

默认情况下，**ccoctl** 在运行命令的目录中创建对象。要在其他目录中创建对象，请使用 **--output-dir** 标志。此流程使用 **<path_to_ccoctl_output_dir>** 来引用这个目录。

先决条件

您必须：

- 提取并准备好 **ccoctl** 二进制文件。
- 使用 Azure CLI 访问 Microsoft Azure 帐户。

流程

1. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 对象列表：

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
  --included \ 1
  --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
  --to=<path_to_directory_for_credentials_requests> 3
```

1. **--included** 参数仅包含特定集群配置所需的清单。
2. 指定 **install-config.yaml** 文件的位置。
3. 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

注意

此命令可能需要一些时间才能运行。

3. 要启用 **ccoctl** 工具自动检测 Azure 凭证，请运行以下命令登录到 Azure CLI：

```
$ az login
```

4. 运行以下命令，使用 **ccoctl** 工具处理所有 **CredentialsRequest** 对象：

```
$ ccoctl azure create-all \
  --name=<azure_infra_name> \ 1
  --output-dir=<ccoctl_output_dir> \ 2
  --region=<azure_region> \ 3
  --subscription-id=<azure_subscription_id> \ 4
  --credentials-requests-dir=<path_to_credentials_requests_directory> \ 5
  --dnszone-resource-group-name=<azure_dns_zone_resource_group_name> \ 6
  --tenant-id=<azure_tenant_id> 7
```

1. 为用于跟踪的所有创建 Azure 资源指定用户定义的名称。
2. 可选：指定您希望 **ccoctl** 实用程序在其中创建对象的目录。默认情况下，实用程序在运行命令的目录中创建对象。

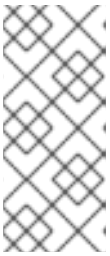
3 指定在其中创建云资源的 Azure 区域。

4 指定要使用的 Azure 订阅 ID。

5 指定包含组件 **CredentialsRequest** 对象文件的目录。

6 指定包含集群基域 Azure DNS 区的资源组名称。

7 指定要使用的 Azure 租户 ID。



注意

如果您的集群使用 **TechPreviewNoUpgrade** 功能集启用的技术预览功能，则必须包含 **--enable-tech-preview** 参数。

要查看其他可选参数以及如何使用它们的说明，请运行 **azure create-all --help** 命令。

验证

- 要验证 OpenShift Container Platform secret 是否已创建，列出 **<path_to_ccoctl_output_dir>/manifests** 目录中的文件：

```
$ ls <path_to_ccoctl_output_dir>/manifests
```

输出示例

```
azure-ad-pod-identity-webhook-config.yaml
cluster-authentication-02-config.yaml
openshift-cloud-controller-manager-azure-cloud-credentials-credentials.yaml
openshift-cloud-network-config-controller-cloud-credentials-credentials.yaml
openshift-cluster-api-capz-manager-bootstrap-credentials-credentials.yaml
openshift-cluster-csi-drivers-azure-disk-credentials-credentials.yaml
openshift-cluster-csi-drivers-azure-file-credentials-credentials.yaml
openshift-image-registry-installer-cloud-credentials-credentials.yaml
openshift-ingress-operator-cloud-credentials-credentials.yaml
openshift-machine-api-azure-cloud-credentials-credentials.yaml
```

您可以验证 Microsoft Entra ID 服务账户通过查询 Azure 而创建。如需更多信息，请参阅 Azure 文档中有关列出 Entra ID 服务帐户的内容。

7.7.8.2.3. 整合 Cloud Credential Operator 实用程序清单

要为单个组件在集群外实现短期安全凭证，您必须将创建 Cloud Credential Operator 实用程序 (**ccoctl**) 的清单文件移到安装程序的正确目录中。

先决条件

- 您已使用托管集群的云平台配置了帐户。
- 您已配置了 Cloud Credential Operator 实用程序 (**ccoctl**)。
- 已使用 **ccoctl** 工具创建了集群所需的云供应商资源。

流程

1. 如果您没有将 **install-config.yaml** 配置文件中的 **credentialsMode** 参数设置为 **Manual**，请修改值，如下所示：

   配置文件片段示例

   ```
   apiVersion: v1
   baseDomain: example.com
   credentialsMode: Manual
   # ...
   ```

2. 如果您使用 **ccoctl** 实用程序创建新的 Azure 资源组，而不是使用现有资源组，请修改 **install-config.yaml** 中的 **resourceGroupName** 参数，如下所示：

   配置文件片段示例

   ```
   apiVersion: v1
   baseDomain: example.com
   # ...
   platform:
     azure:
       resourceGroupName: <azure_infra_name> 1
   # ...
   ```

   1 这个值必须与 **ccoctl azure create-all** 命令的 **--name** 参数指定的 Azure 资源用户定义的名称匹配。

3. 如果您之前还没有创建安装清单文件，请运行以下命令：

   ```
   $ openshift-install create manifests --dir <installation_directory>
   ```

   其中 **<installation_directory>** 是安装程序在其中创建文件的目录。

4. 运行以下命令，将 **ccoctl** 工具生成的清单复制到安装程序创建的 **manifests** 目录中：

   ```
   $ cp /<path_to_ccoctl_output_dir>/manifests/* ./manifests/
   ```

5. 运行以下命令，将 **ccoctl** 工具在 **tls** 目录中生成的私钥复制到安装目录中：

   ```
   $ cp -a /<path_to_ccoctl_output_dir>/tls .
   ```

7.7.9. 部署集群

您可以在兼容云平台上安装 OpenShift Container Platform。

重要

在初始安装过程中，您只能运行安装程序的 **create cluster** 命令一次。

先决条件

- 您已使用托管集群的云平台配置了帐户。
- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。
- 您有一个 Azure 订阅 ID 和租户 ID。

流程

- 进入包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定自定义 **./install-config.yaml** 文件的位置。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不是 **info**。

验证

当集群部署成功完成时：

- 终端会显示用于访问集群的说明，包括指向 Web 控制台和 **kubeadmin** 用户的凭证的链接。
- 凭证信息还会输出到 **<installation_directory>/.openshift_install.log**.

重要

不要删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

其他资源

- 如需有关 访问和了解 OpenShift Container Platform Web 控制台的更多详情，请参阅 访问 Web 控制台。

7.7.10. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于 远程健康监控

7.7.11. 后续步骤

- 自定义集群。
- 如果需要，您可以选择不使用远程健康报告。

7.8. 在 AZURE 上安装私有集群

在 OpenShift Container Platform 版本 4.16 中，您可以在 Microsoft Azure 上将私有集群安装到现有 Azure Virtual Network (VNet) 中。安装程序会置备所需基础架构的其余部分，您可以进一步自定义这些基础架构。要自定义安装，请在安装集群前修改 **install-config.yaml** 文件中的参数。

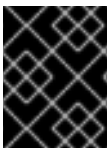
7.8.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读选择集群安装方法并为用户准备它的文档。
- 您已将 Azure 帐户配置为托管集群，并决定要将集群部署到的已测试和验证的区域。
- 如果使用防火墙，将其配置为允许集群需要访问的站点。
- 如果您使用客户管理的加密密钥，准备了用于加密的 Azure 环境。

7.8.2. 私有集群

您可以部署不公开外部端点的私有 OpenShift Container Platform 集群。私有集群只能从内部网络访问，且无法在互联网中看到。

默认情况下，OpenShift Container Platform 被置备为使用可公开访问的 DNS 和端点。在部署集群时，私有集群会将 DNS、Ingress Controller 和 API 服务器设置为私有。这意味着集群资源只能从您的内部网络访问，且不能在互联网中看到。



重要

如果集群有任何公共子网，管理员创建的负载均衡器服务可能会公开访问。为确保集群安全性，请验证这些服务是否已明确标注为私有。

要部署私有集群，您必须：

- 使用满足您的要求的现有网络。集群资源可能会在网络上的其他集群间共享。
- 从有权访问的机器中部署：
  - 您置备的云的 API 服务。
  - 您调配的网络上的主机。
  - 用于获取安装介质的互联网。

您可以使用符合这些访问要求的机器，并按照您的公司规定进行操作。例如，此机器可以是云网络上的堡垒主机，也可以是可通过 VPN 访问网络的机器。

7.8.2.1. Azure 中的私有集群

要在 Microsoft Azure 上创建私有集群，您必须提供一个现有的私有 VNet 和子网来托管集群。安装程序还必须能够解析集群所需的 DNS 记录。安装程序只为内部流量配置 Ingress Operator 和 API 服务器。

根据您的网络如何连接到私有 VNET，您可能需要使用 DNS 转发器来解析集群的私有 DNS 记录。集群的机器在内部使用 **168.63.129.16** 进行 DNS 解析。如需更多信息，请参阅 Azure 文档中的 What is Azure Private DNS? 和 What is IP address 168.63.129.16?。

集群仍然需要访问互联网来访问 Azure API。

安装私有集群时不需要或创建以下项目：

- **BaseDomainResourceGroup**，因为集群不会创建公共记录
- 公共 IP 地址
- 公共 DNS 记录
- 公共端点

> The cluster is configured so that the Operators do not create public records for the cluster and all cluster machines are placed in the private subnets that you specify.

7.8.2.1.1. 限制

Azure 上的私有集群只受到与使用现有 VNet 相关的限制。

7.8.2.2. 用户定义的出站路由

在 OpenShift Container Platform 中，您可以选择自己的出站路由来连接到互联网。这可让您跳过创建公共 IP 地址和公共负载均衡器的步骤。

您可在安装集群前修改 **install-config.yaml** 文件中的参数来配置用户定义的路由。安装集群时，需要一个已存在的 VNet 来使用出站路由，安装程序不负责配置它。

当将集群配置为使用用户定义的路由时，安装程序不会创建以下资源：

- 用于访问互联网的出站规则。
- 公共负载均衡器的公共 IP。

- Kubernetes Service 对象，为出站请求将集群机器添加到公共负载均衡器中。

在设置用户定义的路由前，您必须确保以下项目可用：

- 出口到互联网可以拉取容器镜像，除非使用 OpenShift image registry 镜像。
- 集群可以访问 Azure API。
- 配置了各种允许列表端点。您可以在 *配置防火墙* 部分引用这些端点。

支持一些已存在的网络设置，使用用户定义的路由访问互联网。

带有网络地址转换的私有集群
您可以使用 Azure VNET 网络地址转换(NAT) 为集群中的子网提供出站互联网访问。请参阅 Azure 文档中的 使用 Azure CLI 创建 NAT 网关。

使用 Azure NAT 和用户定义的路由的 VNet 设置时，您可以创建没有公共端点的私有集群。

使用 Azure 防火墙的私有集群
您可以使用 Azure Firewall 为用来安装集群的 VNet 提供出站路由。请参阅 Azure 文档中的Azure Firewall 提供用户定义的路由 的更多信息。

使用 Azure Firewall 和用户定义的路由的 VNet 设置时，您可以创建没有公共端点的私有集群。

带有代理配置的私有集群
您可以使用带有用户定义的路由的代理来允许到互联网的出口。您必须确保集群 Operator 不使用代理访问 Azure API。Operator 必须有权访问代理外的 Azure API。

当使用子网的默认路由表时，Azure **会自动填充0.0.0.0/0**，所有 Azure API 请求都会通过 Azure 的内部网络路由，即使 IP 地址是公共的。只要网络安全组规则允许出口到 Azure API 端点，配置了用户定义的路由的代理就可以在没有公共端点的情况下创建私有集群。

没有互联网访问的私有集群
您可以安装专用网络，以限制对互联网的访问，但 Azure API 除外。这可以通过在本地镜像 registry 来完成。您的集群必须有权访问以下内容：

- 允许拉取容器镜像的 OpenShift image registry 镜像
- 访问 Azure API

在满足这些要求时，您可以使用用户定义的路由来创建没有公共端点的私有集群。

7.8.3. 关于为 OpenShift Container Platform 集群重复使用 VNet

在 OpenShift Container Platform 4.16 中，您可以在 Microsoft Azure 中将集群部署到现有的 Azure Virtual Network(VNet)中。如果您这样做，还必须在 VNet 和路由规则中使用现有子网。

通过将 OpenShift Container Platform 部署到现有的 Azure VNet 中，您可以避免新帐户中的服务限制，或者更容易地利用公司所设置的操作限制。如果您无法获得创建 VNet 所需的基础架构创建权限，则可以使用这个选项。

7.8.3.1. 使用 VNet 的要求

当使用现有 VNet 部署集群时，必须在安装集群前执行额外的网络配置。在安装程序置备的基础架构集群中，安装程序通常会创建以下组件，但在安装到现有 VNet 时不会创建它们：

- 子网

- 路由表
- VNets
- 网络安全组

注意

安装程序要求您使用由云提供的 DNS 服务器。不支持使用自定义 DNS 服务器，并导致安装失败。

如果使用自定义 VNet，您必须正确配置它及其子网，供安装程序和集群使用。安装程序不能为集群分配要使用的网络范围，为子网设置路由表，或者设置类似 DHCP 的 VNet 选项，因此您必须在安装集群前这样做。

集群必须能够访问包含现有 VNet 和子网的资源组。虽然集群创建的所有资源都放在它创建的单独资源组中，但有些网络资源则从单独的组中使用。有些集群 Operator 必须能够访问这两个资源组中的资源。例如，Machine API 控制器会为它创建的虚拟机附加 NICS，以便从网络资源组中进行子网。

您的 VNet 必须满足以下特征：

- VNet 的 CIDR 块必须包含 **Networking.MachineCIDR** 范围，它是集群机器的 IP 地址池。
- VNet 及其子网必须属于同一资源组，子网必须配置为使用 Azure 分配的 DHCP IP 地址，而不是静态 IP 地址。

您必须在 VNet 中提供两个子网，一个用于 control plane 机器，一个用于计算机器。因为 Azure 在您指定的区域内的不同可用区中分发机器，所以集群将默认具有高可用性。

注意

默认情况下，如果您在 **install-config.yaml** 文件中指定可用区，安装程序会在一个区（region）内的这些可用区间分发 control plane 机器和计算机器。要确保集群的高可用性，请选择至少含有三个可用区的区域。如果您的区域包含的可用区少于三个，安装程序将在可用区中放置多台 control plane 机器。

为确保您提供的子网适合，安装程序会确认以下数据：

- 所有指定的子网都存在。
- 有两个专用子网，一个用于 control plane 机器，一个用于计算机器。
- 子网 CIDR 属于您指定的机器 CIDR。机器不会在没有为其提供私有子网的可用区中置备。

注意

如果您销毁了使用现有 VNet 的集群，则不会删除 VNet。

7.8.3.1.1. 网络安全组要求

托管 compute 和 control plane 机器的子网的网络安全组需要特定的访问权限，以确保集群通信正确。您必须创建规则以允许访问所需的集群通信端口。

重要

在安装集群前，必须先设置网络安全组规则。如果您试图在没有所需访问权限的情况下安装集群，安装程序无法访问 Azure API，安装会失败。

表 7.16. 所需端口

port	描述	Control plane（控制平面）	Compute
80	允许 HTTP 流量		x
443	允许 HTTPS 流量		x
6443	允许与 control plane 机器通信	x	
22623	允许内部与机器配置服务器通信以用于置备机器	x	

1. 如果使用 Azure Firewall 来限制互联网访问，您可以将 Azure Firewall 配置为允许 Azure API。不需要网络安全组规则。

重要

目前，不支持阻止或限制机器配置服务器端点。机器配置服务器必须公开给网络，以便新置备的机器没有现有配置或状态，才能获取其配置。在这个模型中，信任的根是证书签名请求 (CSR) 端点，即 kubelet 发送其证书签名请求以批准加入集群。因此，机器配置不应用于分发敏感信息，如 secret 和证书。

为确保机器配置服务器端点，端口 22623 和 22624 在裸机场景中是安全的，客户必须配置正确的网络策略。

由于集群组件不会修改 Kubernetes 控制器更新的用户提供的网络安全组，因此为 Kubernetes 控制器在不影响其余环境的情况下创建一个伪网络安全组。

其他资源

- 关于 OpenShift SDN 网络插件
- 配置防火墙

7.8.3.2. 权限划分

从 OpenShift Container Platform 4.3 开始，您不需要安装程序置备的基础架构集群部署所需的所有权限。这与您所在机构可能拥有的权限划分类似：一些个人可以在您的云中创建不同的资源。例如，您可以创建特定于应用程序的对象，如实例、存储和负载均衡器，但不能创建与网络相关的组件，如 VNets、子网或入站规则。

您在创建集群时使用的 Azure 凭证不需要 VNets 和核心网络组件（如子网、路由表、互联网网关、NAT 和 VPN）所需的网络权限。您仍然需要获取集群中的机器需要的应用程序资源的权限，如负载均衡器、安全组、存储帐户和节点。

7.8.3.3. 集群间隔离

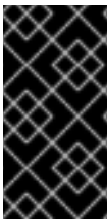
因为集群无法修改现有子网中的网络安全组，所以无法在 VNet 中相互隔离集群。

7.8.4. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。



重要

如果您的集群无法直接访问互联网，则可以在置备的某些类型的基础架构上执行受限网络安装。在此过程中，您可以下载所需的内容，并使用它为镜像 registry 填充安装软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群前，您要更新镜像 registry 的内容。

7.8.5. 为集群节点 SSH 访问生成密钥对

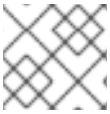
在 OpenShift Container Platform 安装过程中，您可以为安装程序提供 SSH 公钥。密钥通过它们的 Ignition 配置文件传递给 Red Hat Enterprise Linux CoreOS(RHCOS)节点，用于验证对节点的 SSH 访问。密钥添加到每个节点上 **core** 用户的 **~/.ssh/authorized_keys** 列表中，这将启用免密码身份验证。

将密钥传递给节点后，您可以使用密钥对作为用户 **核心** 通过 SSH 连接到 RHCOS 节点。若要通过 SSH 访问节点，必须由 SSH 为您的本地用户管理私钥身份。

如果要通过 SSH 连接到集群节点来执行安装调试或灾难恢复，则必须在安装过程中提供 SSH 公钥。**./openshift-install gather** 命令还需要在集群节点上设置 SSH 公钥。

重要

不要在生产环境中跳过这个过程，在生产环境中需要灾难恢复和调试。



注意

您必须使用本地密钥，而不是使用特定平台方法配置 的密钥，如 AWS 密钥对。

流程

1. 如果您在本地计算机上没有可用于在集群节点上进行身份验证的现有 SSH 密钥对，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_ed25519**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

注意

如果您计划在 **x86_64**、**ppc64le** 和 **s390x** 架构上安装使用 RHEL 加密库（这些加密库已提交给 NIST 用于 FIPS 140-2/140-3 验证）的 OpenShift Container Platform 集群，则不要创建使用 **ed25519** 算法的密钥。相反，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 查看公共 SSH 密钥：

```
$ cat <path>/<file_name>.pub
```

例如，运行以下命令来查看 **~/.ssh/id_ed25519.pub** 公钥：

```
$ cat ~/.ssh/id_ed25519.pub
```

3. 将 SSH 私钥身份添加到本地用户的 SSH 代理（如果尚未添加）。在集群节点上，或者要使用 **./openshift-install gather** 命令，需要对该密钥进行 SSH 代理管理，才能在集群节点上进行免密码 SSH 身份验证。

注意

在某些发行版中，自动管理默认 SSH 私钥身份，如 **~/.ssh/id_rsa** 和 **~/.ssh/id_dsa**。

a. 如果 **ssh-agent** 进程尚未为您的本地用户运行，请将其作为后台任务启动：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果集群处于 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

4. 将 SSH 私钥添加到 **ssh-agent** ：

```
$ ssh-add <path>/<file_name> (1)
```

(1) 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_ed25519.pub**

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

后续步骤

- 安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

7.8.6. 获取安装程序

在安装 OpenShift Container Platform 前，将安装文件下载到您用于安装的主机上。

先决条件

- 您有一台运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB。

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐户，请使用您的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入到安装类型的页面，下载与您的主机操作系统和架构对应的安装程序，并将该文件放在您要存储安装配置文件的目录中。

重要

安装程序会在用来安装集群的计算机上创建几个文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。

重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，请为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。此 pull secret 允许您与所含授权机构提供的服务进行身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

7.8.7. 手动创建安装配置文件

安装集群要求您手动创建安装配置文件。

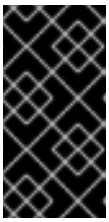
先决条件

- 您在本地机器上有一个 SSH 公钥来提供给安装程序。该密钥将用于在集群节点上进行 SSH 身份验证，以进行调试和灾难恢复。
- 已获取 OpenShift Container Platform 安装程序和集群的 pull secret。

流程

1. 创建一个安装目录来存储所需的安装资产：

```
$ mkdir <installation_directory>
```



重要

您必须创建一个目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

2. 自定义提供的 **install-config.yaml** 文件模板示例，并将其保存在 **<installation_directory>** 中。

注意

此配置文件必须命名为 **install-config.yaml**。

3. 备份 **install-config.yaml** 文件，以便您可以使用它安装多个集群。

重要

install-config.yaml 文件会在安装过程的下一步中使用。现在必须备份它。

其他资源

- Azure 的安装配置参数

7.8.7.1. 集群安装的最低资源要求

每台集群机器都必须满足以下最低要求：

表 7.17. 最低资源要求

机器	操作系统	vCPU [1]	虚拟内存	Storage	每秒输入/输出 (IOPS) [2]
bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane（控制平面）	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS、RHEL 8.6 及更新版本 [3]	2	8 GB	100 GB	300

1. 当未启用并发多线程(SMT)或超线程时，一个 vCPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比例：（每个内核数的线程）× sockets = vCPU。

2. OpenShift Container Platform 和 Kubernetes 对磁盘性能非常敏感，建议使用更快的存储速度，特别是 control plane 节点上需要 10 ms p99 fsync 持续时间的 etcd。请注意，在许多云平台上，存储大小和 IOPS 可一起扩展，因此您可能需要过度分配存储卷来获取足够的性能。

3. 与所有用户置备的安装一样，如果您选择在集群中使用 RHEL 计算机器，则负责所有操作系统生命周期管理和维护，包括执行系统更新、应用补丁和完成所有其他必要的任务。RHEL 7 计算机器的使用已弃用，并已在 OpenShift Container Platform 4.10 及更新的版本中删除。

注意

从 OpenShift Container Platform 版本 4.13 开始，RHCOS 基于 RHEL 版本 9.2，它更新了微架构要求。以下列表包含每个架构需要的最小指令集架构 (ISA)：

- x86-64 体系结构需要 x86-64-v2 ISA
- ARM64 架构需要 ARMv8.0-A ISA
- IBM Power 架构需要 Power 9 ISA
- s390x 架构需要 z14 ISA

如需更多信息，请参阅 RHEL 架构。

重要

您需要使用将 **PremiumIO** 参数设置为 **true** 的 Azure 虚拟机。

如果平台的实例类型满足集群机器的最低要求，则 OpenShift Container Platform 支持使用它。

其他资源

- 优化存储

7.8.7.2. 为 Azure 测试的实例类型

以下 Microsoft Azure 实例类型已经 OpenShift Container Platform 测试。

例 7.33. 基于 64 位 x86 架构的机器类型

- **c4.***
- **c5.***
- **c5a.***
- **i3.***
- **m4.***
- **m5.***
- **m5a.***
- **m6a.***
- **m6i.***
- **r4.***
- **r5.***
- **r5a.***
- **r6i.***

- **t3.***
- **t3a.***

7.8.7.3. 在 64 位 ARM 基础架构上为 Azure 测试的实例类型

以下 Microsoft Azure ARM64 实例类型已使用 OpenShift Container Platform 测试。

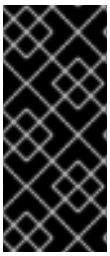
例 7.34. 基于 64 位 ARM 架构的机器类型

- **c6g.***
- **m6g.***

7.8.7.4. 为 Azure 虚拟机启用可信启动

在 Azure 上安装集群时，您可以启用两个可信启动功能： 安全引导 和虚拟化可信平台模块。

请参阅 Azure 文档中有关 虚拟机大小 的信息，以了解虚拟机支持这些功能的大小。



重要

可信启动只是一个技术预览功能。技术预览功能不受红帽产品服务等级协议（SLA）支持，且功能可能并不完整。红帽不推荐在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。

有关红帽技术预览功能支持范围的更多信息，请参阅技术预览功能支持范围。

先决条件

- 您已创建了 **install-config.yaml** 文件。

流程

- 在部署集群前，使用文本编辑器编辑 **install-config.yaml** 文件并添加以下小节：

```
controlPlane: 1
  platform:
    azure:
      settings:
        securityType: TrustedLaunch 2
        trustedLaunch:
          uefiSettings:
            secureBoot: Enabled 3
            virtualizedTrustedPlatformModule: Enabled 4
```

1. 指定 **controlPlane.platform.azure** 或 **compute.platform.azure**，以分别在 control plane 或计算节点上启用可信启动。指定 **platform.azure.defaultMachinePlatform**，以便在所有节点上启用可信启动。
2. 启用可信启动功能。

3 启用安全引导。如需更多信息，请参阅 Azure 文档中有关安全引导的内容。

4 启用虚拟化可信平台模块。如需更多信息，请参阅 Azure 文档中有关 虚拟化可信平台模块的文档。

7.8.7.5. 启用机密虚拟机

您可在安装集群前启用机密虚拟机。您可以为计算节点、control plane 节点或所有节点启用机密虚拟机。

重要

使用机密虚拟机只是一个技术预览功能。技术预览功能不受红帽产品服务等级协议（SLA）支持，且功能可能并不完整。红帽不推荐在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。

有关红帽技术预览功能支持范围的更多信息，请参阅技术预览功能支持范围。

您可以使用带有以下虚拟机大小的机密虚拟机：

- DCasv5-series
- DCadsv5-series
- ECasv5-series
- ECadsv5-series

重要

64 位 ARM 架构目前不支持机密虚拟机。

先决条件

- 您已创建了 **install-config.yaml** 文件。

流程

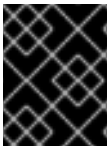
- 在部署集群前，使用文本编辑器编辑 **install-config.yaml** 文件并添加以下小节：

```
controlPlane: 1
  platform:
    azure:
      settings:
        securityType: ConfidentialVM 2
        confidentialVM:
          uefiSettings:
            secureBoot: Enabled 3
            virtualizedTrustedPlatformModule: Enabled 4
      osDisk:
        securityProfile:
          securityEncryptionType: VMGuestStateOnly 5
```

1. 指定 **controlPlane.platform.azure** 或 **compute.platform.azure**，以分别在 control plane 或计算节点上部署机密虚拟机。指定 **platform.azure.defaultMachinePlatform** 以在所有节点上部署机密虚拟机。
2. 启用机密虚拟机。
3. 启用安全引导。如需更多信息，请参阅 Azure 文档中有关安全引导的内容。
4. 启用虚拟化可信平台模块。如需更多信息，请参阅 Azure 文档中有关 虚拟化可信平台模块的文档。
5. 指定 **VMGuestStateOnly** 来加密虚拟机客户端状态。

7.8.7.6. Azure 的自定义 install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多详情，或修改所需参数的值。



重要

此示例 YAML 文件仅供参考。您必须使用安装程序来获取 **install-config.yaml** 文件，并进行修改。

```
apiVersion: v1
baseDomain: example.com (1)
controlPlane: (2)
  hyperthreading: Enabled (3) (4)
  name: master
  platform:
    azure:
      encryptionAtHost: true
      ultraSSDCapability: Enabled
      osDisk:
        diskSizeGB: 1024 (5)
        diskType: Premium_LRS
        diskEncryptionSet:
          resourceGroup: disk_encryption_set_resource_group
          name: disk_encryption_set_name
          subscriptionId: secondary_subscription_id
      osImage:
        publisher: example_publisher_name
        offer: example_image_offer
        sku: example_offer_sku
        version: example_image_version
      type: Standard_D8s_v3
  replicas: 3
compute: (6)
- hyperthreading: Enabled (7)
  name: worker
  platform:
    azure:
      ultraSSDCapability: Enabled
      type: Standard_D2s_v3
```

```
      encryptionAtHost: true
      osDisk:
        diskSizeGB: 512 8
        diskType: Standard_LRS
        diskEncryptionSet:
          resourceGroup: disk_encryption_set_resource_group
          name: disk_encryption_set_name
          subscriptionId: secondary_subscription_id
      osImage:
        publisher: example_publisher_name
        offer: example_image_offer
        sku: example_offer_sku
        version: example_image_version
      zones: 9
      - "1"
      - "2"
      - "3"
  replicas: 5
metadata:
  name: test-cluster 10
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 11
  serviceNetwork:
  - 172.30.0.0/16
platform:
  azure:
    defaultMachinePlatform:
      osImage: 12
        publisher: example_publisher_name
        offer: example_image_offer
        sku: example_offer_sku
        version: example_image_version
      ultraSSDCapability: Enabled
    baseDomainResourceGroupName: resource_group 13
    region: centralus 14
    resourceGroupName: existing_resource_group 15
    networkResourceGroupName: vnet_resource_group 16
    virtualNetwork: vnet 17
    controlPlaneSubnet: control_plane_subnet 18
    computeSubnet: compute_subnet 19
    outboundType: UserDefinedRouting 20
    cloudName: AzurePublicCloud
pullSecret: '{"auths": ...}' 21
fips: false 22
sshKey: ssh-ed25519 AAAA... 23
publish: Internal 24
```

1 10 14 21 必需。安装程序会提示您输入这个值。

2 6 如果没有提供这些参数和值，安装程序会提供默认值。

3 7 **controlPlane** 部分是一个单个映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，controlPlane **部分** 的第一行则不以连字符开头。仅使用一个 control plane 池。

4 是否要启用或禁用并发多线程或 **超线程**。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将 参数值设置为 **Disabled** 来禁用它。如果在某些集群机器中禁用并发多线程，则必须在所有集群机器中禁用它。

重要

如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。如果您禁用并发多线程，请为您的机器使用较大的虚拟机类型，如 **Standard_D8s_v3**。

5 8 您可以指定要使用的磁盘大小（以 GB 为单位）。control plane 节点的最低推荐值为 1024 GB。

9 指定要将机器部署到的区域列表。如需高可用性，请至少指定两个区域。

11 要安装的集群网络插件。默认值 **OVNKubernetes** 是唯一支持的值。

12 可选：应该用来引导 control plane 和计算机器的自定义 Red Hat Enterprise Linux CoreOS (RHCOS) 镜像。**platform.azure.defaultMachinePlatform.osImage** 下的 **publisher**, **offer**, **sku**, 和 **version** 参数应用到 control plane 和计算机器。如果设置了 **controlPlane.platform.azure.osImage** 或 **compute.platform.azure.osImage** 下的参数，它们会覆盖 **platform.azure.defaultMachinePlatform.osImage** 参数。

13 指定包含基域的 DNS 区的资源组的名称。

15 指定要安装集群的现有资源组的名称。如果未定义，则会为集群创建新的资源组。

16 如果使用现有的 VNet，请指定包含它的资源组的名称。

17 如果使用现有的 VNet，请指定其名称。

18 如果使用现有的 VNet，请指定托管 control plane 机器的子网名称。

19 如果使用现有的 VNet，请指定托管计算机器的子网名称。

20 您可以自定义自己的出站路由。配置用户定义的路由可防止在集群中公开外部端点。出口的用户定义路由需要将集群部署到现有的 VNet。

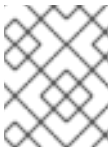
22 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 Red Hat Enterprise Linux (RHEL) 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅在 FIPS 模式中安装该系统。

当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS)时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。

23 您可以选择提供您用来访问集群中机器的 **sshKey** 值。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

24 如何发布集群的面向用户的端点。将 **publish** 设置为 **Internal** 以部署一个私有集群，它不能被互联网访问。默认值为 **External**。

7.8.7.7. 在安装过程中配置集群范围的代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 **Proxy** 对象的 **spec.noProxy** 字段中添加站点来绕过代理。

注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，**Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 **http**。

2 用于创建集群外 HTTPS 连接的代理 URL。

3 要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 **.** 以仅匹配子域。例如，**.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代

4 如果提供，安装程序会在 **openshift-config** 命名空间中生成名为 **user-ca-bundle** 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 **trusted-ca-bundle** 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， **Proxy** 对象的 **trustedCA** 字段中也会引用此配置映射。**additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

5 可选：决定 **Proxy** 对象的配置以引用 **trustedCA** 字段中 **user-ca-bundle** 配置映射的策略。允许的值是 **Proxyonly** 和 **Always**。仅在配置了 **http/https** 代理时，使用 **Proxyonly** 引用 **user-ca-bundle** 配置映射。使用 **Always** 始终引用 **user-ca-bundle** 配置映射。默认值为 **Proxyonly**。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。

注意

如果安装程序超时，重启并使用安装程序的 **wait-for** 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. 保存该文件并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 cluster 的集群范围代理，该代理 **使用** 提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

注意

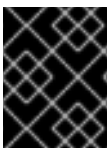
只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

其他资源

- 有关加速网络的详情，请参阅 Microsoft Azure 虚拟机的加速网络。

7.8.8. 安装 OpenShift CLI

您可以安装 OpenShift CLI(**oc**)来使用命令行界面与 OpenShift Container Platform 进行交互。您可以在 Linux、Windows 或 macOS 上安装 **oc**。



重要

如果安装了旧版本的 **oc**，则可能无法使用 OpenShift Container Platform 4.16 中的所有命令。下载并安装新版本的 **oc**。

在 Linux 上安装 OpenShift CLI
您可以按照以下流程在 Linux 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **产品变体** 下拉列表中选择架构。
3. 从 **版本** 下拉列表中选择适当的版本。
4. 点 **OpenShift v4.16 Linux Client** 条目旁的 **Download Now** 来保存文件。
5. 解包存档：

```
$ tar xvf <file>
```

6. 将 **oc** 二进制文件放到 **PATH 中的目录中**。
   要查看您的 **PATH**，请执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

在 Windows 上安装 OpenShift CLI
您可以按照以下流程在 Windows 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 Windows Client** 条目旁的 **Download Now** 来保存文件。
4. 使用 ZIP 程序解压存档。
5. 将 **oc** 二进制文件移到 **PATH 中的目录中**。
   要查看您的 **PATH**，请打开命令提示并执行以下命令：

```
C:\> path
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

在 macOS 上安装 OpenShift CLI
您可以按照以下流程在 macOS 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 macOS Client** 条目旁的 **Download Now** 来保存文件。

注意

对于 macOS arm64，请选择 **OpenShift v4.16 macOS arm64 Client** 条目。

4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
要查看您的 **PATH**，请打开终端并执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

7.8.9. 在 kube-system 项目中存储管理员级别的 secret 的替代方案

默认情况下，管理员 secret 存储在 **kube-system** 项目中。如果您在 **install-config.yaml** 文件中将 **credentialsMode** 参数配置为 **Manual**，则必须使用以下替代方案之一：

- 要手动管理长期云凭证，请按照手动创建长期凭证中的步骤操作。
- 要实现在集群外为各个组件管理的短期凭证，请按照配置 Azure 集群以使用短期凭证中的步骤操作。

7.8.9.1. 手动创建长期凭证

在无法访问云身份和访问管理(IAM)API 的环境中，或者管理员更不希望将管理员级别的凭证 secret 存储在集群 **kube-system** 命名空间中时，可以在安装前将 Cloud Credential Operator(CCO)放入手动模式。

流程

1. 如果您没有将 **install-config.yaml** 配置文件中的 **credentialsMode** 参数设置为 **Manual**，请修改值，如下所示：

配置文件片段示例

```
apiVersion: v1
baseDomain: example.com
credentialsMode: Manual
# ...
```

2. 如果您之前还没有创建安装清单文件，请运行以下命令：

```
$ openshift-install create manifests --dir <installation_directory>
```

其中 **<installation_directory>** 是安装程序在其中创建文件的目录。

3. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

4. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 自定义资源 (CR) 列表：

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
  --included \ 1
  --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
  --to=<path_to_directory_for_credentials_requests> 3
```

1. **--included** 参数仅包含特定集群配置所需的清单。
2. 指定 **install-config.yaml** 文件的位置。
3. 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

此命令为每个 **CredentialsRequest** 对象创建一个 YAML 文件。

CredentialsRequest 对象示例

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component_credentials_request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: AzureProviderSpec
    roleBindings:
    - role: Contributor
  ...
```

5. 在之前生成的 **openshift-install** 清单目录中为 secret 创建 YAML 文件。secret 必须使用在 **spec.secretRef** 中为每个 **CredentialsRequest** 定义的命名空间和 secret 名称存储。

带有 secret 的 CredentialsRequest 对象示例

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component_credentials_request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
```

```
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: AzureProviderSpec
    roleBindings:
    - role: Contributor
      ...
  secretRef:
    name: <component_secret>
    namespace: <component_namespace>
  ...
```

Secret 对象示例

```
apiVersion: v1
kind: Secret
metadata:
  name: <component_secret>
  namespace: <component_namespace>
data:
  azure_subscription_id: <base64_encoded_azure_subscription_id>
  azure_client_id: <base64_encoded_azure_client_id>
  azure_client_secret: <base64_encoded_azure_client_secret>
  azure_tenant_id: <base64_encoded_azure_tenant_id>
  azure_resource_prefix: <base64_encoded_azure_resource_prefix>
  azure_resourcegroup: <base64_encoded_azure_resourcegroup>
  azure_region: <base64_encoded_azure_region>
```

重要

在升级使用手动维护凭证的集群前，您必须确保 CCO 处于可升级状态。

7.8.9.2. 配置 Azure 集群以使用短期凭证

要安装使用 Microsoft Entra Workload ID 的集群，您必须配置 Cloud Credential Operator 工具，并为集群创建所需的 Azure 资源。

7.8.9.2.1. 配置 Cloud Credential Operator 工具

当 Cloud Credential Operator(CCO)以手动模式运行时，要从集群外部创建和管理云凭证，提取并准备 CCO 实用程序(**ccoctl**)二进制文件。

注意

ccoctl 工具是在 Linux 环境中运行的 Linux 二进制文件。

先决条件

- 您可以访问具有集群管理员权限的 OpenShift Container Platform 帐户。
- 已安装 OpenShift CLI(**oc**)。
- 您已为 **ccoctl** 工具创建了全局 Microsoft Azure 帐户，用于以下权限：

  例 7.35. 所需的 Azure 权限

- Microsoft.Resources/subscriptions/resourceGroups/read
- Microsoft.Resources/subscriptions/resourceGroups/write
- Microsoft.Resources/subscriptions/resourceGroups/delete
- Microsoft.Authorization/roleAssignments/read
- Microsoft.Authorization/roleAssignments/delete
- Microsoft.Authorization/roleAssignments/write
- Microsoft.Authorization/roleDefinitions/read
- Microsoft.Authorization/roleDefinitions/write
- Microsoft.Authorization/roleDefinitions/delete
- Microsoft.Storage/storageAccounts/listkeys/action
- Microsoft.Storage/storageAccounts/delete
- Microsoft.Storage/storageAccounts/read
- Microsoft.Storage/storageAccounts/write
- Microsoft.Storage/storageAccounts/blobServices/containers/write
- Microsoft.Storage/storageAccounts/blobServices/containers/delete
- Microsoft.Storage/storageAccounts/blobServices/containers/read
- Microsoft.ManagedIdentity/userAssignedIdentities/delete
- Microsoft.ManagedIdentity/userAssignedIdentities/read
- Microsoft.ManagedIdentity/userAssignedIdentities/write
- Microsoft.ManagedIdentity/userAssignedIdentities/federatedIdentityCredentials/read
- Microsoft.ManagedIdentity/userAssignedIdentities/federatedIdentityCredentials/write
- Microsoft.ManagedIdentity/userAssignedIdentities/federatedIdentityCredentials/delete
- Microsoft.Storage/register/action
- Microsoft.ManagedIdentity/register/action

流程

1. 运行以下命令，为 OpenShift Container Platform 发行镜像设置变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. 运行以下命令，从 OpenShift Container Platform 发行镜像获取 CCO 容器镜像：

```
$ CCO_IMAGE=$(oc adm release info --image-for='cloud-credential-operator'
$RELEASE_IMAGE -a ~/.pull-secret)
```

注意

确保 **$RELEASE_IMAGE** 的架构与将使用 **ccoctl** 工具的环境架构相匹配。

3. 运行以下命令，将 CCO 容器镜像中的 **ccoctl** 二进制文件提取到 OpenShift Container Platform 发行镜像中：

```
$ oc image extract $CCO_IMAGE \
  --file="/usr/bin/ccoctl.<rhel_version>" \ 1
  -a ~/.pull-secret
```

1 对于 **<rhel_version>**，请指定与主机使用的 Red Hat Enterprise Linux (RHEL) 版本对应的值。如果没有指定值，则默认使用 **ccoctl.rhel8**。以下值有效：

- **rhel8**: 为使用 RHEL 8 的主机指定这个值。
- **rhel9** ：为使用 RHEL 9 的主机指定这个值。

4. 运行以下命令更改权限以使 **ccoctl** 可执行：

```
$ chmod 775 ccoctl.<rhel_version>
```

验证

- 要验证 **ccoctl** 是否可使用，请运行以下命令来显示帮助文件：

```
$ ccoctl --help
```

ccoctl --help 的输出

```
OpenShift credentials provisioning tool

Usage:
  ccoctl [command]

Available Commands:
  aws         Manage credentials objects for AWS cloud
  azure       Manage credentials objects for Azure
  gcp         Manage credentials objects for Google cloud
  help        Help about any command
  ibmcloud    Manage credentials objects for IBM Cloud
  nutanix     Manage credentials objects for Nutanix

Flags:
  -h, --help   help for ccoctl

Use "ccoctl [command] --help" for more information about a command.
```

7.8.9.2.2. 使用 Cloud Credential Operator 实用程序创建 Azure 资源

您可以使用 **ccoctl azure create-all** 命令自动创建 Azure 资源。

注意

默认情况下，**ccoctl** 在运行命令的目录中创建对象。要在其他目录中创建对象，请使用 **--output-dir** 标志。此流程使用 **<path_to_ccoctl_output_dir>** 来引用这个目录。

先决条件

您必须：

- 提取并准备好 **ccoctl** 二进制文件。
- 使用 Azure CLI 访问 Microsoft Azure 帐户。

流程

1. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 对象列表：

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
  --included \ 1
  --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
  --to=<path_to_directory_for_credentials_requests> 3
```

1. **--included** 参数仅包含特定集群配置所需的清单。
2. 指定 **install-config.yaml** 文件的位置。
3. 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

注意

此命令可能需要一些时间才能运行。

3. 要启用 **ccoctl** 工具自动检测 Azure 凭证，请运行以下命令登录到 Azure CLI：

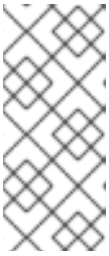
```
$ az login
```

4. 运行以下命令，使用 **ccoctl** 工具处理所有 **CredentialsRequest** 对象：

```
$ ccoctl azure create-all \
  --name=<azure_infra_name> \ 1
```

```
--output-dir=<ccoctl_output_dir> \ 2
--region=<azure_region> \ 3
--subscription-id=<azure_subscription_id> \ 4
--credentials-requests-dir=<path_to_credentials_requests_directory> \ 5
--dnszone-resource-group-name=<azure_dns_zone_resource_group_name> \ 6
--tenant-id=<azure_tenant_id> 7
```

1. 为用于跟踪的所有创建 Azure 资源指定用户定义的名称。
2. 可选：指定您希望 **ccoctl** 实用程序在其中创建对象的目录。默认情况下，实用程序在运行命令的目录中创建对象。
3. 指定在其中创建云资源的 Azure 区域。
4. 指定要使用的 Azure 订阅 ID。
5. 指定包含组件 **CredentialsRequest** 对象文件的目录。
6. 指定包含集群基域 Azure DNS 区的资源组名称。
7. 指定要使用的 Azure 租户 ID。



注意

如果您的集群使用 **TechPreviewNoUpgrade** 功能集启用的技术预览功能，则必须包含 **--enable-tech-preview** 参数。

要查看其他可选参数以及如何使用它们的说明，请运行 **azure create-all --help** 命令。

验证

- 要验证 OpenShift Container Platform secret 是否已创建，列出 **<path_to_ccoctl_output_dir>/manifests** 目录中的文件：

```
$ ls <path_to_ccoctl_output_dir>/manifests
```

输出示例

```
azure-ad-pod-identity-webhook-config.yaml
cluster-authentication-02-config.yaml
openshift-cloud-controller-manager-azure-cloud-credentials-credentials.yaml
openshift-cloud-network-config-controller-cloud-credentials-credentials.yaml
openshift-cluster-api-capz-manager-bootstrap-credentials-credentials.yaml
openshift-cluster-csi-drivers-azure-disk-credentials-credentials.yaml
openshift-cluster-csi-drivers-azure-file-credentials-credentials.yaml
openshift-image-registry-installer-cloud-credentials-credentials.yaml
openshift-ingress-operator-cloud-credentials-credentials.yaml
openshift-machine-api-azure-cloud-credentials-credentials.yaml
```

您可以验证 Microsoft Entra ID 服务账户通过查询 Azure 而创建。如需更多信息，请参阅 Azure 文档中有关列出 Entra ID 服务帐户的内容。

7.8.9.2.3. 整合 Cloud Credential Operator 实用程序清单

要为单个组件在集群外实现短期安全凭证，您必须将创建 Cloud Credential Operator 实用程序 (**ccoctl**) 的清单文件移到安装程序的正确目录中。

先决条件

- 您已使用托管集群的云平台配置了帐户。
- 您已配置了 Cloud Credential Operator 实用程序 (**ccoctl**)。
- 已使用 **ccoctl** 工具创建了集群所需的云供应商资源。

流程

1. 如果您没有将 **install-config.yaml** 配置文件中的 **credentialsMode** 参数设置为 **Manual**，请修改值，如下所示：

   配置文件片段示例

   ```
   apiVersion: v1
   baseDomain: example.com
   credentialsMode: Manual
   # ...
   ```

2. 如果您使用 **ccoctl** 实用程序创建新的 Azure 资源组，而不是使用现有资源组，请修改 **install-config.yaml** 中的 **resourceGroupName** 参数，如下所示：

   配置文件片段示例

   ```
   apiVersion: v1
   baseDomain: example.com
   # ...
   platform:
     azure:
       resourceGroupName: <azure_infra_name> 1
   # ...
   ```

   1 这个值必须与 **ccoctl azure create-all** 命令的 **--name** 参数指定的 Azure 资源用户定义的名称匹配。

3. 如果您之前还没有创建安装清单文件，请运行以下命令：

   ```
   $ openshift-install create manifests --dir <installation_directory>
   ```

   其中 **<installation_directory>** 是安装程序在其中创建文件的目录。

4. 运行以下命令，将 **ccoctl** 工具生成的清单复制到安装程序创建的 **manifests** 目录中：

   ```
   $ cp /<path_to_ccoctl_output_dir>/manifests/* ./manifests/
   ```

5. 运行以下命令，将 **ccoctl** 工具在 **tls** 目录中生成的私钥复制到安装目录中：

```
$ cp -a /<path_to_ccoctl_output_dir>/tls .
```

7.8.10. 可选：为私有镜像 registry 准备私有 Microsoft Azure 集群

通过在私有 Microsoft Azure 集群上安装私有镜像 registry，您可以创建私有存储端点。私有存储端点在 registry 的存储帐户中禁用面向公共的端点，为 OpenShift Container Platform 部署添加额外的安全层。使用以下指南准备私有 Microsoft Azure 集群，以使用私有镜像 registry 安装。

先决条件

- 您可以访问具有集群管理员权限的 OpenShift Container Platform 帐户。
- 已安装 OpenShift CLI（oc）。
- 您已准备了一个 **install-config.yaml**，其中包含以下信息：
  - **publish** 字段设置为 **Internal**
- 您已设置了创建私有存储端点的权限。如需更多信息，请参阅"安装程序置备的基础架构的Azure 权限"。

流程

1. 如果您之前还没有创建安装清单文件，请运行以下命令：

   ```
   $ ./openshift-install create manifests --dir <installation_directory>
   ```

   这个命令显示以下信息：

   输出示例

   ```
   INFO Consuming Install Config from target directory
   INFO Manifests created in: <installation_directory>/manifests and
   <installation_directory>/openshift
   ```

2. 创建镜像 registry 配置对象，并传递 Microsoft Azure 提供的 **networkResourceGroupName**、**subnetName** 和 **vnetName**。例如：

   ```
   $ touch imageregistry-config.yaml
   ```

   ```
   apiVersion: imageregistry.operator.openshift.io/v1
   kind: Config
   metadata:
     name: cluster
   spec:
     managementState: "Managed"
     replicas: 2
     rolloutStrategy: RollingUpdate
     storage:
       azure:
         networkAccess:
           internal:
             networkResourceGroupName: <vnet_resource_group> 1
   ```

```
    subnetName: <subnet_name> 2
    vnetName: <vnet_name> 3
  type: Internal
```

1. 可选。如果您有一个现有的 VNet 和子网设置，将 **<vnet_resource_group>** 替换为包含现有虚拟网络 (VNet) 的资源组名称。
2. 可选。如果您有一个现有的 VNet 和子网设置，将 **<subnet_name>** 替换为指定资源组中现有计算子网的名称。
3. 可选。如果您有一个现有的 VNet 和子网设置，请将 &lt **;vnet_name** > 替换为指定资源组中现有虚拟网络(VNet)的名称。

注意

imageregistry-config.yaml 文件会在安装过程中消耗。如果需要，您必须在安装前备份。

3. 运行以下命令，将 **imageregistry-config.yaml** 文件移到 **<installation_directory/manifests>** 文件夹中：

```
$ mv imageregistry-config.yaml <installation_directory/manifests/>
```

后续步骤

- 将 **imageregistry-config.yaml** 文件移到 **<installation_directory/manifests>** 文件夹并设置所需权限后，继续执行 "Deploying the cluster"。

其他资源

- 有关创建私有存储端点所需的权限列表，请参阅安装程序置备的基础架构所需的 Azure 权限。

7.8.11. 部署集群

您可以在兼容云平台上安装 OpenShift Container Platform。

重要

在初始安装过程中，您只能运行安装程序的 **create cluster** 命令一次。

先决条件

- 您已使用托管集群的云平台配置了帐户。
- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。
- 您有一个 Azure 订阅 ID 和租户 ID。
- 如果要使用服务主体安装集群，则有其应用程序 ID 和密码。
- 如果您要使用系统分配的受管身份安装集群，需要在您要从其中运行安装程序的虚拟机上启用它。

- 如果您要使用用户分配的受管身份安装集群，需要满足以下先决条件：
  - 您有它的客户端 ID。
  - 您已将其分配给您要从其运行安装程序的虚拟机。

流程

1. 可选：如果您之前在这个计算机上运行安装程序，并希望使用替代的服务主体或受管身份，请进入 **~/.azure/** 目录并删除 **osServicePrincipal.json** 配置文件。
   删除此文件可防止安装程序自动重复使用之前安装中的订阅和验证值。
2. 进入包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定自定义 **./install-config.yaml** 文件的位置。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不是 **info**。

   如果安装程序无法找到之前安装中的 **osServicePrincipal.json** 配置文件，会提示您输入 Azure 订阅和验证值。

3. 为您的订阅输入以下 Azure 参数值：
   - **Azure subscription id**：输入用于集群的订阅 ID。
   - **Azure 租户 id**：输入租户 ID。
4. 根据您用来部署集群的 Azure 身份，在提示输入 **azure 服务主体客户端 id**时执行以下操作之一：
   - 如果您使用服务主体，请输入其应用程序 ID。
   - 如果您使用系统分配的受管身份，请将此值设为空白。
   - 如果您使用用户分配的受管身份，请指定其客户端 ID。
5. 根据您用来部署集群的 Azure 身份，在提示输入 **azure 服务主体客户端 secret**时执行以下操作之一：
   - 如果您使用服务主体，请输入其密码。
   - 如果您使用系统分配的受管身份，请将此值设为空白。
   - 如果您使用用户分配的受管身份，请将此值设为空白。

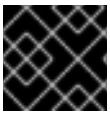
在以前的版本中，安装程序会创建一个 **osServicePrincipal.json** 配置文件，并将此文件存储在计算机上的 **~/.azure/** 目录中。这样可确保安装程序在目标平台上创建 OpenShift Container Platform 集群时可以加载配置集。

验证

当集群部署成功完成时：

- 终端会显示用于访问集群的说明，包括指向 Web 控制台和 **kubeadmin** 用户的凭证的链接。

- 凭证信息还会输出到 **<installation_directory>/.openshift_install.log**.

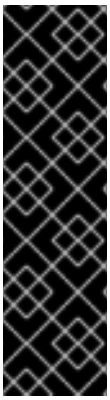


重要

不要删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

7.8.12. 使用 CLI 登录集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

其他资源

- 如需有关 访问和了解 OpenShift Container Platform Web 控制台的更多详情，请参阅 访问 Web 控制台。

7.8.13. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于 远程健康监控

7.8.14. 后续步骤

- 自定义集群。
- 如果需要，您可以选择 不使用远程健康报告。

7.9. 在 AZURE 上将集群安装到一个政府区域

在 OpenShift Container Platform 版本 4.16 中，您可以在 Microsoft Azure 上将集群安装到一个政府区域。要配置政府区域，请在安装集群前修改 **install-config.yaml** 文件中的参数。

7.9.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读有关 选择集群安装方法的文档，并为用户准备它。
- 您已 将 Azure 帐户配置为 托管集群，并确定要将集群部署到的经过测试和验证的政府区域。
- 如果使用防火墙，将其配置为允许集群需要访问的站点。
- 如果环境中无法访问云身份和访问管理(IAM)API，或者不想将管理员级别的凭证 secret 存储在 **kube-system** 命名空间中，您可以 手动创建和维护 IAM 凭证。
- 如果您使用客户管理的加密密钥，准备了用于加密的 Azure 环境。

7.9.2. Azure 政府区域

OpenShift Container Platform 支持将集群部署到 Microsoft Azure Government(MAG) 区域。MAG 是专门为需要运行敏感负载的美国政府机构、企业、企业和其他美国客户特别设计的。MAG 由仅政府数据中心区域组成，它们都赋予了 影响级别 5 授权。

安装到 MAG 区域需要在 **install-config.yaml** 文件中手动配置 Azure Government 专用云实例和区域。您还必须更新服务主体以引用适当的政府环境。

注意

Azure 政府区域不能使用安装程序的引导终端提示来选择。您必须在 **install-config.yaml** 文件中手动定义区域。记得还要根据指定的区域设置专用云实例，如 **AzureUSrianCloud**。

7.9.3. 私有集群

您可以部署不公开外部端点的私有 OpenShift Container Platform 集群。私有集群只能从内部网络访问，且无法在互联网中看到。

默认情况下，OpenShift Container Platform 被置备为使用可公开访问的 DNS 和端点。在部署集群时，私有集群会将 DNS、Ingress Controller 和 API 服务器设置为私有。这意味着集群资源只能从您的内部网络访问，且不能在互联网中看到。

重要

如果集群有任何公共子网，管理员创建的负载均衡器服务可能会公开访问。为确保集群安全性，请验证这些服务是否已明确标注为私有。

要部署私有集群，您必须：

- 使用满足您的要求的现有网络。集群资源可能会在网络上的其他集群间共享。
- 从有权访问的机器中部署：
  - 您置备的云的 API 服务。
  - 您调配的网络上的主机。
  - 用于获取安装介质的互联网。

您可以使用符合这些访问要求的机器，并按照您的公司规定进行操作。例如，此机器可以是云网络上的堡垒主机，也可以是可通过 VPN 访问网络的机器。

7.9.3.1. Azure 中的私有集群

要在 Microsoft Azure 上创建私有集群，您必须提供一个现有的私有 VNet 和子网来托管集群。安装程序还必须能够解析集群所需的 DNS 记录。安装程序只为内部流量配置 Ingress Operator 和 API 服务器。

根据您的网络如何连接到私有 VNET，您可能需要使用 DNS 转发器来解析集群的私有 DNS 记录。集群的机器在内部使用 **168.63.129.16** 进行 DNS 解析。如需更多信息，请参阅 Azure 文档中的 What is Azure Private DNS? 和 What is IP address 168.63.129.16?。

集群仍然需要访问互联网来访问 Azure API。

安装私有集群时不需要或创建以下项目：

- **BaseDomainResourceGroup**，因为集群不会创建公共记录
- 公共 IP 地址
- 公共 DNS 记录
- 公共端点

> The cluster is configured so that the Operators do not create public records for the cluster and all cluster machines are placed in the private subnets that you specify.

7.9.3.1.1. 限制

Azure 上的私有集群只受到与使用现有 VNet 相关的限制。

7.9.3.2. 用户定义的出站路由

在 OpenShift Container Platform 中，您可以选择自己的出站路由来连接到互联网。这可让您跳过创建公共 IP 地址和公共负载均衡器的步骤。

您可在安装集群前修改 **install-config.yaml** 文件中的参数来配置用户定义的路由。安装集群时，需要一个已存在的 VNet 来使用出站路由，安装程序不负责配置它。

当将集群配置为使用用户定义的路由时，安装程序不会创建以下资源：

- 用于访问互联网的出站规则。
- 公共负载均衡器的公共 IP。
- Kubernetes Service 对象，为出站请求将集群机器添加到公共负载均衡器中。

在设置用户定义的路由前，您必须确保以下项目可用：

- 出口到互联网可以拉取容器镜像，除非使用 OpenShift image registry 镜像。
- 集群可以访问 Azure API。
- 配置了各种允许列表端点。您可以在 *配置防火墙* 部分引用这些端点。

支持一些已存在的网络设置，使用用户定义的路由访问互联网。

7.9.4. 关于为 OpenShift Container Platform 集群重复使用 VNet

在 OpenShift Container Platform 4.16 中，您可以在 Microsoft Azure 中将集群部署到现有的 Azure Virtual Network(VNet)中。如果您这样做，还必须在 VNet 和路由规则中使用现有子网。

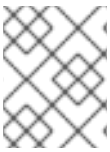
通过将 OpenShift Container Platform 部署到现有的 Azure VNet 中，您可以避免新帐户中的服务限制，或者更容易地利用公司所设置的操作限制。如果您无法获得创建 VNet 所需的基础架构创建权限，则可以使用这个选项。

7.9.4.1. 使用 VNet 的要求

当使用现有 VNet 部署集群时，必须在安装集群前执行额外的网络配置。在安装程序置备的基础架构集群中，安装程序通常会创建以下组件，但在安装到现有 VNet 时不会创建它们：

- 子网

- 路由表
- VNets
- 网络安全组



注意

安装程序要求您使用由云提供的 DNS 服务器。不支持使用自定义 DNS 服务器，并导致安装失败。

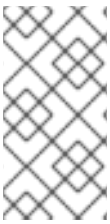
如果使用自定义 VNet，您必须正确配置它及其子网，供安装程序和集群使用。安装程序不能为集群分配要使用的网络范围，为子网设置路由表，或者设置类似 DHCP 的 VNet 选项，因此您必须在安装集群前这样做。

集群必须能够访问包含现有 VNet 和子网的资源组。虽然集群创建的所有资源都放在它创建的单独资源组中，但有些网络资源则从单独的组中使用。有些集群 Operator 必须能够访问这两个资源组中的资源。例如，Machine API 控制器会为它创建的虚拟机附加 NICS，以便从网络资源组中进行子网。

您的 VNet 必须满足以下特征：

- VNet 的 CIDR 块必须包含 **Networking.MachineCIDR** 范围，它是集群机器的 IP 地址池。
- VNet 及其子网必须属于同一资源组，子网必须配置为使用 Azure 分配的 DHCP IP 地址，而不是静态 IP 地址。

您必须在 VNet 中提供两个子网，一个用于 control plane 机器，一个用于计算机器。因为 Azure 在您指定的区域内的不同可用区中分发机器，所以集群将默认具有高可用性。



注意

默认情况下，如果您在 **install-config.yaml** 文件中指定可用区，安装程序会在一个区（region）内的这些可用区间分发 control plane 机器和计算机器。要确保集群的高可用性，请选择至少含有三个可用区的区域。如果您的区域包含的可用区少于三个，安装程序将在可用区中放置多台 control plane 机器。

为确保您提供的子网适合，安装程序会确认以下数据：

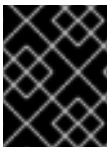
- 所有指定的子网都存在。
- 有两个专用子网，一个用于 control plane 机器，一个用于计算机器。
- 子网 CIDR 属于您指定的机器 CIDR。机器不会在您不为其提供私有子网的可用区中置备。如果需要，安装程序会创建管理 control plane 和 worker 节点的公共负载均衡器，Azure 会为其分配一个公共 IP 地址。

注意

如果您销毁了使用现有 VNet 的集群，则不会删除 VNet。

7.9.4.1.1. 网络安全组要求

托管 compute 和 control plane 机器的子网的网络安全组需要特定的访问权限，以确保集群通信正确。您必须创建规则以允许访问所需的集群通信端口。



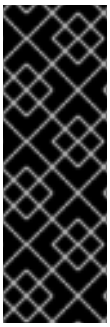
重要

在安装集群前，必须先设置网络安全组规则。如果您试图在没有所需访问权限的情况下安装集群，安装程序无法访问 Azure API，安装会失败。

表 7.18. 所需端口

port	描述	Control plane（控制平面）	Compute
80	允许 HTTP 流量		x
443	允许 HTTPS 流量		x
6443	允许与 control plane 机器通信	x	
22623	允许内部与机器配置服务器通信以用于置备机器	x	

1. 如果使用 Azure Firewall 来限制互联网访问，您可以将 Azure Firewall 配置为允许 Azure API。不需要网络安全组规则。



重要

目前，不支持阻止或限制机器配置服务器端点。机器配置服务器必须公开给网络，以便新置备的机器没有现有配置或状态，才能获取其配置。在这个模型中，信任的根是证书签名请求 (CSR) 端点，即 kubelet 发送其证书签名请求以批准加入集群。因此，机器配置不应用于分发敏感信息，如 secret 和证书。

为确保机器配置服务器端点，端口 22623 和 22624 在裸机场景中是安全的，客户必须配置正确的网络策略。

由于集群组件不会修改 Kubernetes 控制器更新的用户提供的网络安全组，因此为 Kubernetes 控制器在不影响其余环境的情况下创建一个伪网络安全组。

其他资源

- 关于 OpenShift SDN 网络插件
- 配置防火墙

7.9.4.2. 权限划分

从 OpenShift Container Platform 4.3 开始，您不需要安装程序置备的基础架构集群部署所需的所有权限。这与您所在机构可能拥有的权限划分类似：一些个人可以在您的云中创建不同的资源。例如，您可以创建特定于应用程序的对象，如实例、存储和负载均衡器，但不能创建与网络相关的组件，如 VNets、子网或入站规则。

您在创建集群时使用的 Azure 凭证不需要 VNets 和核心网络组件（如子网、路由表、互联网网关、NAT 和 VPN）所需的网络权限。您仍然需要获取集群中的机器需要的应用程序资源的权限，如负载均衡器、安全组、存储帐户和节点。

7.9.4.3. 集群间隔离

因为集群无法修改现有子网中的网络安全组，所以无法在 VNet 中相互隔离集群。

7.9.5. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类型的基础架构上执行受限网络安装。在此过程中，您可以下载所需的内容，并使用它为镜像 registry 填充安装软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群前，您要更新镜像 registry 的内容。

7.9.6. 为集群节点 SSH 访问生成密钥对

在 OpenShift Container Platform 安装过程中，您可以为安装程序提供 SSH 公钥。密钥通过它们的 Ignition 配置文件传递给 Red Hat Enterprise Linux CoreOS(RHCOS)节点，用于验证对节点的 SSH 访问。密钥添加到每个节点上 **core** 用户的 **~/.ssh/authorized_keys** 列表中，这将启用免密码身份验证。

将密钥传递给节点后，您可以使用密钥对作为用户 **核心** 通过 SSH 连接到 RHCOS 节点。若要通过 SSH 访问节点，必须由 SSH 为您的本地用户管理私钥身份。

如果要通过 SSH 连接到集群节点来执行安装调试或灾难恢复，则必须在安装过程中提供 SSH 公钥。**./openshift-install gather** 命令还需要在集群节点上设置 SSH 公钥。

重要

不要在生产环境中跳过这个过程，在生产环境中需要灾难恢复和调试。

注意

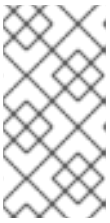
您必须使用本地密钥，而不是使用特定平台方法配置 的密钥，如 AWS 密钥对。

流程

1. 如果您在本地计算机上没有可用于在集群节点上进行身份验证的现有 SSH 密钥对，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_ed25519**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。



注意

如果您计划在 **x86_64**、**ppc64le** 和 **s390x** 架构上安装使用 RHEL 加密库（这些加密库已提交给 NIST 用于 FIPS 140-2/140-3 验证）的 OpenShift Container Platform 集群，则不要创建使用 **ed25519** 算法的密钥。相反，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 查看公共 SSH 密钥：

```
$ cat <path>/<file_name>.pub
```

例如，运行以下命令来查看 **~/.ssh/id_ed25519.pub** 公钥：

```
$ cat ~/.ssh/id_ed25519.pub
```

3. 将 SSH 私钥身份添加到本地用户的 SSH 代理（如果尚未添加）。在集群节点上，或者要使用 **./openshift-install gather** 命令，需要对该密钥进行 SSH 代理管理，才能在集群节点上进行免密码 SSH 身份验证。

注意

在某些发行版中，自动管理默认 SSH 私钥身份，如 **~/.ssh/id_rsa** 和 **~/.ssh/id_dsa**。

a. 如果 **ssh-agent** 进程尚未为您的本地用户运行，请将其作为后台任务启动：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果集群处于 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

4. 将 SSH 私钥添加到 **ssh-agent** ：

```
$ ssh-add <path>/<file_name> (1)
```

(1) 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_ed25519.pub**

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

后续步骤

- 安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

7.9.7. 获取安装程序

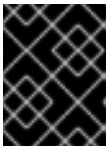
在安装 OpenShift Container Platform 前，将安装文件下载到您用于安装的主机上。

先决条件

- 您有一台运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB。

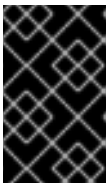
流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐户，请使用您的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入到安装类型的页面，下载与您的主机操作系统和架构对应的安装程序，并将该文件放在您要存储安装配置文件的目录中。



重要

安装程序会在用来安装集群的计算机上创建几个文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。



重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，请为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。此 pull secret 允许您与所含授权机构提供的服务进行身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

7.9.8. 手动创建安装配置文件

安装集群要求您手动创建安装配置文件。

先决条件

- 您在本地机器上有一个 SSH 公钥来提供给安装程序。该密钥将用于在集群节点上进行 SSH 身份验证，以进行调试和灾难恢复。
- 已获取 OpenShift Container Platform 安装程序和集群的 pull secret。

流程

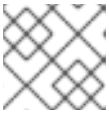
1. 创建一个安装目录来存储所需的安装资产：

```
$ mkdir <installation_directory>
```

重要

您必须创建一个目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

2. 自定义提供的 **install-config.yaml** 文件模板示例，并将其保存在 **<installation_directory>** 中。



注意

此配置文件必须命名为 **install-config.yaml**。

3. 备份 **install-config.yaml** 文件，以便您可以使用它安装多个集群。

重要

install-config.yaml 文件会在安装过程的下一步中使用。现在必须备份它。

其他资源

- Azure 的安装配置参数

7.9.8.1. 集群安装的最低资源要求

每台集群机器都必须满足以下最低要求：

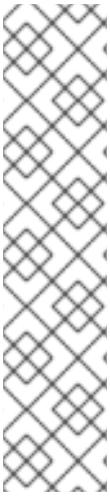
表 7.19. 最低资源要求

机器	操作系统	vCPU [1]	虚拟内存	Storage	每秒输入/输出 (IOPS) [2]
bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane（控制平面）	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS、RHEL 8.6 及更新版本 [3]	2	8 GB	100 GB	300

1. 当未启用并发多线程(SMT)或超线程时，一个 vCPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比例：（每个内核数的线程）× sockets = vCPU。

2. OpenShift Container Platform 和 Kubernetes 对磁盘性能非常敏感，建议使用更快的存储速度，特别是 control plane 节点上需要 10 ms p99 fsync 持续时间的 etcd。请注意，在许多云平台上，存储大小和 IOPS 可一起扩展，因此您可能需要过度分配存储卷来获取足够的性能。

3. 与所有用户置备的安装一样，如果您选择在集群中使用 RHEL 计算机器，则负责所有操作系统生命周期管理和维护，包括执行系统更新、应用补丁和完成所有其他必要的任务。RHEL 7 计算机器的使用已弃用，并已在 OpenShift Container Platform 4.10 及更新的版本中删除。



注意

从 OpenShift Container Platform 版本 4.13 开始，RHCOS 基于 RHEL 版本 9.2，它更新了微架构要求。以下列表包含每个架构需要的最小指令集架构 (ISA)：

- x86-64 体系结构需要 x86-64-v2 ISA
- ARM64 架构需要 ARMv8.0-A ISA
- IBM Power 架构需要 Power 9 ISA
- s390x 架构需要 z14 ISA

如需更多信息，请参阅 RHEL 架构。

重要

您需要使用将 **PremiumIO** 参数设置为 **true** 的 Azure 虚拟机。

如果平台的实例类型满足集群机器的最低要求，则 OpenShift Container Platform 支持使用它。

其他资源

- 优化存储

7.9.8.2. 为 Azure 测试的实例类型

以下 Microsoft Azure 实例类型已经 OpenShift Container Platform 测试。

例 7.36. 基于 64 位 x86 架构的机器类型

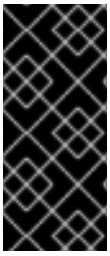
- **c4.***
- **c5.***
- **c5a.***
- **i3.***
- **m4.***
- **m5.***
- **m5a.***
- **m6a.***
- **m6i.***
- **r4.***
- **r5.***

- **r5a.***
- **r6i.***
- **t3.***
- **t3a.***

7.9.8.3. 为 Azure 虚拟机启用可信启动

在 Azure 上安装集群时，您可以启用两个可信启动功能： 安全引导 和虚拟化可信平台模块。

请参阅 Azure 文档中有关 虚拟机大小 的信息，以了解虚拟机支持这些功能的大小。



重要

可信启动只是一个技术预览功能。技术预览功能不受红帽产品服务等级协议（SLA）支持，且功能可能并不完整。红帽不推荐在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。

有关红帽技术预览功能支持范围的更多信息，请参阅技术预览功能支持范围。

先决条件

- 您已创建了 **install-config.yaml** 文件。

流程

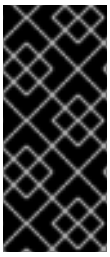
- 在部署集群前，使用文本编辑器编辑 **install-config.yaml** 文件并添加以下小节：

```
controlPlane: 1
  platform:
    azure:
      settings:
        securityType: TrustedLaunch 2
        trustedLaunch:
          uefiSettings:
            secureBoot: Enabled 3
            virtualizedTrustedPlatformModule: Enabled 4
```

1. 指定 **controlPlane.platform.azure** 或 **compute.platform.azure**，以分别在 control plane 或计算节点上启用可信启动。指定 **platform.azure.defaultMachinePlatform**，以便在所有节点上启用可信启动。
2. 启用可信启动功能。
3. 启用安全引导。如需更多信息，请参阅 Azure 文档中有关安全引导的内容。
4. 启用虚拟化可信平台模块。如需更多信息，请参阅 Azure 文档中有关 虚拟化可信平台模块的文档。

7.9.8.4. 启用机密虚拟机

您可在安装集群前启用机密虚拟机。您可以为计算节点、control plane 节点或所有节点启用机密虚拟机。



重要

使用机密虚拟机只是一个技术预览功能。技术预览功能不受红帽产品服务等级协议（SLA）支持，且功能可能并不完整。红帽不推荐在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。

有关红帽技术预览功能支持范围的更多信息，请参阅技术预览功能支持范围。

您可以使用带有以下虚拟机大小的机密虚拟机：

- DCasv5-series
- DCadsv5-series
- ECasv5-series
- ECadsv5-series

重要

64 位 ARM 架构目前不支持机密虚拟机。

先决条件

- 您已创建了 **install-config.yaml** 文件。

流程

- 在部署集群前，使用文本编辑器编辑 **install-config.yaml** 文件并添加以下小节：

```
controlPlane: 1
  platform:
    azure:
      settings:
        securityType: ConfidentialVM 2
        confidentialVM:
          uefiSettings:
            secureBoot: Enabled 3
            virtualizedTrustedPlatformModule: Enabled 4
      osDisk:
        securityProfile:
          securityEncryptionType: VMGuestStateOnly 5
```

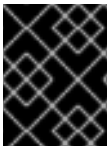
1. 指定 **controlPlane.platform.azure** 或 **compute.platform.azure**，以分别在 control plane 或计算节点上部署机密虚拟机。指定 **platform.azure.defaultMachinePlatform** 以在所有节点上部署机密虚拟机。
2. 启用机密虚拟机。
3. 启用安全引导。如需更多信息，请参阅 Azure 文档中有关安全引导的内容。
4.

启用虚拟化可信平台模块。如需更多信息，请参阅 Azure 文档中有关 虚拟化可信平台模块的文档。

5 指定 **VMGuestStateOnly** 来加密虚拟机客户端状态。

7.9.8.5. Azure 的自定义 install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多详情，或修改所需参数的值。



重要

此示例 YAML 文件仅供参考。您必须使用安装程序来获取 **install-config.yaml** 文件，并进行修改。

```
apiVersion: v1
baseDomain: example.com 1
controlPlane: 2
  hyperthreading: Enabled 3 4
  name: master
  platform:
    azure:
      encryptionAtHost: true
      ultraSSDCapability: Enabled
      osDisk:
        diskSizeGB: 1024 5
        diskType: Premium_LRS
        diskEncryptionSet:
          resourceGroup: disk_encryption_set_resource_group
          name: disk_encryption_set_name
          subscriptionId: secondary_subscription_id
      osImage:
        publisher: example_publisher_name
        offer: example_image_offer
        sku: example_offer_sku
        version: example_image_version
      type: Standard_D8s_v3
  replicas: 3
compute: 6
- hyperthreading: Enabled 7
  name: worker
  platform:
    azure:
      ultraSSDCapability: Enabled
      type: Standard_D2s_v3
      encryptionAtHost: true
      osDisk:
        diskSizeGB: 512 8
        diskType: Standard_LRS
        diskEncryptionSet:
          resourceGroup: disk_encryption_set_resource_group
          name: disk_encryption_set_name
          subscriptionId: secondary_subscription_id
```

```
      osImage:
        publisher: example_publisher_name
        offer: example_image_offer
        sku: example_offer_sku
        version: example_image_version
      zones: 9
      - "1"
      - "2"
      - "3"
  replicas: 5
metadata:
  name: test-cluster 10
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 11
  serviceNetwork:
  - 172.30.0.0/16
platform:
  azure:
    defaultMachinePlatform:
      osImage: 12
        publisher: example_publisher_name
        offer: example_image_offer
        sku: example_offer_sku
        version: example_image_version
      ultraSSDCapability: Enabled
    baseDomainResourceGroupName: resource_group 13
    region: usgovvirginia
    resourceGroupName: existing_resource_group 14
    networkResourceGroupName: vnet_resource_group 15
    virtualNetwork: vnet 16
    controlPlaneSubnet: control_plane_subnet 17
    computeSubnet: compute_subnet 18
    outboundType: UserDefinedRouting 19
    cloudName: AzureUSGovernmentCloud 20
pullSecret: '{"auths": ...}' 21
fips: false 22
sshKey: ssh-ed25519 AAAA... 23
publish: Internal 24
```

1 10 21 必需。

2 6 如果没有提供这些参数和值，安装程序会提供默认值。

3 7 **controlPlane** 部分是一个单个映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，controlPlane **部分** 的第一行则不以连字符开头。仅使用一个 control plane 池。

4

是否要启用或禁用并发多线程或 **超线程**。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将 参数值设置为 **Disabled** 来禁用它。如果在某些集群机器中禁用并发多线程，则必须在所

重要

如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。如果您禁用并发多线程，请为您的机器使用较大的虚拟机类型，如 **Standard_D8s_v3**。

5 8 您可以指定要使用的磁盘大小（以 GB 为单位）。control plane 节点的最低推荐值为 1024 GB。

9 指定要将机器部署到的区域列表。如需高可用性，请至少指定两个区域。

11 要安装的集群网络插件。默认值 **OVNKubernetes** 是唯一支持的值。

12 可选：应该用来引导 control plane 和计算机器的自定义 Red Hat Enterprise Linux CoreOS (RHCOS) 镜像。**platform.azure.defaultMachinePlatform.osImage** 下的 **publisher**, **offer**, **sku**, 和 **version** 参数应用到 control plane 和计算机器。如果设置了 **controlPlane.platform.azure.osImage** 或 **compute.platform.azure.osImage** 下的参数，它们会覆盖 **platform.azure.defaultMachinePlatform.osImage** 参数。

13 指定包含基域的 DNS 区的资源组的名称。

14 指定要安装集群的现有资源组的名称。如果未定义，则会为集群创建新的资源组。

15 如果使用现有的 VNet，请指定包含它的资源组的名称。

16 如果使用现有的 VNet，请指定其名称。

17 如果使用现有的 VNet，请指定托管 control plane 机器的子网名称。

18 如果使用现有的 VNet，请指定托管计算机器的子网名称。

19 您可以自定义自己的出站路由。配置用户定义的路由可防止在集群中公开外部端点。出口的用户定义路由需要将集群部署到现有的 VNet。

20 指定要将集群部署到的 Azure 云环境的名称。将 **AzureUS GovernmentCloud** 设置为部署到 Microsoft Azure Government(MAG)区域。默认值为 **AzurePublicCloud**。

22 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 Red Hat Enterprise Linux (RHEL) 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅在 FIPS 模式中安装该系统。

当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS)时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。

23 您可以选择提供您用来访问集群中机器的 **sshKey** 值。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

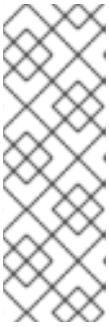
24 如何发布集群的面向用户的端点。将 **publish** 设置为 **Internal** 以部署一个私有集群，它不能被互联网访问。默认值为 **External**。

7.9.8.6. 在安装过程中配置集群范围的代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 **Proxy** 对象的 **spec.noProxy** 字段中添加站点来绕过代理。



注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，**Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 **http**。

2 用于创建集群外 HTTPS 连接的代理 URL。

3

要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 **.** 以仅匹配子域。例如，**.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代

4 如果提供，安装程序会在 **openshift-config** 命名空间中生成名为 **user-ca-bundle** 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 **trusted-ca-bundle** 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， **Proxy** 对象的 **trustedCA** 字段中也会引用此配置映射。**additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

5 可选：决定 **Proxy** 对象的配置以引用 **trustedCA** 字段中 **user-ca-bundle** 配置映射的策略。允许的值是 **Proxyonly** 和 **Always**。仅在配置了 **http/https** 代理时，使用 **Proxyonly** 引用 **user-ca-bundle** 配置映射。使用 **Always** 始终引用 **user-ca-bundle** 配置映射。默认值为 **Proxyonly**。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。

注意

如果安装程序超时，重启并使用安装程序的 **wait-for** 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. 保存该文件并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 cluster 的集群范围代理，该代理 **使用** 提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

其他资源

- 有关加速网络的详情，请参阅 Microsoft Azure 虚拟机的加速网络。

7.9.9. 部署集群

您可以在兼容云平台上安装 OpenShift Container Platform。

重要

在初始安装过程中，您只能运行安装程序的 **create cluster** 命令一次。

先决条件

- 您已使用托管集群的云平台配置了帐户。
- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。

- 您有一个 Azure 订阅 ID 和租户 ID。
- 如果要使用服务主体安装集群，则有其应用程序 ID 和密码。
- 如果您要使用系统分配的受管身份安装集群，需要在您要从其中运行安装程序的虚拟机上启用它。
- 如果您要使用用户分配的受管身份安装集群，需要满足以下先决条件：
  - 您有它的客户端 ID。
  - 您已将其分配给您要从其运行安装程序的虚拟机。

流程

1. 可选：如果您之前在这个计算机上运行安装程序，并希望使用替代的服务主体或受管身份，请进入 **~/.azure/** 目录并删除 **osServicePrincipal.json** 配置文件。
   删除此文件可防止安装程序自动重复使用之前安装中的订阅和验证值。
2. 进入包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1. 对于 **<installation_directory>**，请指定自定义 **./install-config.yaml** 文件的位置。
2. 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不是 **info**。

   如果安装程序无法找到之前安装中的 **osServicePrincipal.json** 配置文件，会提示您输入 Azure 订阅和验证值。

3. 为您的订阅输入以下 Azure 参数值：
   - **Azure subscription id**：输入用于集群的订阅 ID。
   - **Azure 租户 id**：输入租户 ID。
4. 根据您用来部署集群的 Azure 身份，在提示输入 **azure 服务主体客户端 id**时执行以下操作之一：
   - 如果您使用服务主体，请输入其应用程序 ID。
   - 如果您使用系统分配的受管身份，请将此值设为空白。
   - 如果您使用用户分配的受管身份，请指定其客户端 ID。
5. 根据您用来部署集群的 Azure 身份，在提示输入 **azure 服务主体客户端 secret**时执行以下操作之一：
   - 如果您使用服务主体，请输入其密码。
   - 如果您使用系统分配的受管身份，请将此值设为空白。
   - 如果您使用用户分配的受管身份，请将此值设为空白。

在以前的版本中，安装程序会创建一个 **osServicePrincipal.json** 配置文件，并将此文件存储在计算机上的 **~/.azure/** 目录中。这样可确保安装程序在目标平台上创建 OpenShift Container Platform 集群时可以加载配置集。

验证

当集群部署成功完成时：

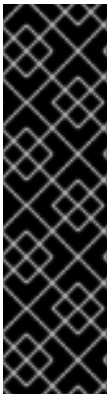
- 终端会显示用于访问集群的说明，包括指向 Web 控制台和 **kubeadmin** 用户的凭证的链接。
- 凭证信息还会输出到 **<installation_directory>/.openshift_install.log**.

重要

不要删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

7.9.10. 安装 OpenShift CLI

您可以安装 OpenShift CLI(**oc**)来使用命令行界面与 OpenShift Container Platform 进行交互。您可以在 Linux、Windows 或 macOS 上安装 **oc**。

重要

如果安装了旧版本的 **oc**，则可能无法使用 OpenShift Container Platform 4.16 中的所有命令。下载并安装新版本的 **oc**。

在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **产品变体** 下拉列表中选择架构。
3. 从 **版本** 下拉列表中选择适当的版本。
4. 点 **OpenShift v4.16 Linux Client** 条目旁的 **Download Now** 来保存文件。
5. 解包存档：

```
$ tar xvf <file>
```

6. 将 **oc** 二进制文件放到 **PATH 中的目录中**。
要查看您的 **PATH**，请执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

在 Windows 上安装 OpenShift CLI
您可以按照以下流程在 Windows 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 Windows Client** 条目旁的 **Download Now** 来保存文件。
4. 使用 ZIP 程序解压存档。
5. 将 **oc** 二进制文件移到 **PATH 中的目录中**。
要查看您的 **PATH**，请打开命令提示并执行以下命令：

```
C:\> path
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

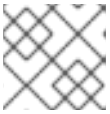
```
C:\> oc <command>
```

在 macOS 上安装 OpenShift CLI
您可以按照以下流程在 macOS 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。

2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 macOS Client** 条目旁的 **Download Now** 来保存文件。



注意

对于 macOS arm64，请选择 **OpenShift v4.16 macOS arm64 Client** 条目。

4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
要查看您的 **PATH**，请打开终端并执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

7.9.11. 使用 CLI 登录集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

其他资源

- 如需有关 访问和了解 OpenShift Container Platform Web 控制台的更多详情，请参阅 访问 Web 控制台。

7.9.12. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于 远程健康监控

7.9.13. 后续步骤

- 自定义集群。
- 如果需要，您可以选择 不使用远程健康报告。

7.10. 在使用用户置备的受限网络中的 AZURE 上安装集群

在 OpenShift Container Platform 中，您可以使用您提供的基础架构在 Microsoft Azure 上安装集群。

提供的几个 Azure Resource Manager（ARM） 模板可协助完成这些步骤，也可帮助您自行建模。

重要

进行用户置备的基础架构安装的步骤仅作为示例。使用您提供的基础架构安装集群需要了解云供应商和 OpenShift Container Platform 安装过程。提供的几个 ARM 模板可帮助完成这些步骤，或帮助您自行建模。您还可以自由选择通过其他方法创建所需的资源。

先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读选择集群安装方法并为用户准备它的文档。
- 您已将 Azure 帐户配置为托管集群，并决定要将集群部署到的已测试和验证的区域。
- 您已 将断开连接的安装的镜像镜像 到 registry，并获取了 OpenShift Container Platform 版本的 **imageContentSources** 数据。

重要

因为安装介质位于镜像主机上，因此您必须使用该计算机完成所有安装步骤。

- 如果使用防火墙，将其配置为允许集群需要访问的站点。
- 如果环境中无法访问云身份和访问管理(IAM) API，或者不想将管理员级别的凭证 secret 存储在 **kube-system** 命名空间中，您可以手动创建长期凭证。

- 如果您使用客户管理的加密密钥，准备了用于加密的 Azure 环境。

7.10.1. 关于在受限网络中安装

在 OpenShift Container Platform 4.16 中，可以执行不需要有效的互联网连接来获取软件组件的安装。受限网络安装可以使用安装程序置备的基础架构或用户置备的基础架构完成，具体取决于您要安装集群的云平台。

如果您选择在云平台中执行受限网络安装，您仍需要访问其云 API。有些云功能，比如 Amazon Web Service 的 Route 53 DNS 和 IAM 服务，需要访问互联网。根据您的网络，在裸机硬件、Nutanix 或 VMware vSphere 上安装可能需要较少的互联网访问。

要完成受限网络安装，您必须创建一个 registry，以镜像 OpenShift 镜像 registry 的内容并包含安装介质。您可以在镜像主机上创建此 registry，该主机可同时访问互联网和您的封闭网络，也可以使用满足您的限制条件的其他方法。

重要

由于用户置备安装配置的复杂性，在尝试使用用户置备的基础架构受限网络安装前，请考虑完成标准用户置备的基础架构安装。完成此测试安装后，您可以更轻松地隔离和排除在受限网络中安装过程中可能出现的任何问题。

7.10.1.1. 其他限制

受限网络中的集群有以下额外限制和限制：

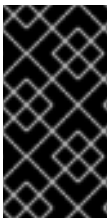
- **ClusterVersion** 状态包含一个 **Unable to retrieve available updates** 错误。
- 默认情况下，您无法使用 Developer Catalog 的内容，因为您无法访问所需的镜像流标签。

7.10.1.2. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

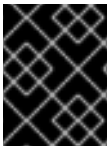


重要

如果您的集群无法直接访问互联网，则可以在置备的某些类型的基础架构上执行受限网络安装。在此过程中，您可以下载所需的内容，并使用它为镜像 registry 填充安装软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

7.10.2. 配置 Azure 项目

在安装 OpenShift Container Platform 之前，您必须配置 Azure 项目来托管它。

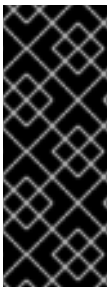


重要

所有通过公共端点提供的 Azure 资源均存在资源名称的限制，您无法创建使用某些名称的资源。如需 Azure 限制词语列表，请参阅 Azure 文档中的解决保留资源名称错误。

7.10.2.1. Azure 帐户限值

OpenShift Container Platform 集群使用诸多 Microsoft Azure 组件，默认的 Azure 订阅和服务限值、配额和约束会影响您安装 OpenShift Container Platform 集群的能力。



重要

默认的限制因服务类别的不同（如 Free Trial 或 Pay-As-You-Go）以及系列的不同（如 Dv2、F 或 G）而有所不同。例如，对于 Enterprise Agreement 订阅的默认限制是 350 个内核。

在 Azure 上安装默认集群前，请检查您的订阅类型的限制，如有必要，请提高帐户的配额限制。

下表总结了 Azure 组件，它们的限值会影响您安装和运行 OpenShift Container Platform 集群的能力。

组件	默认所需的组件数	默认 Azure 限值	描述
vCPU	44	每个区域 20 个	默认集群需要 44 个 vCPU，因此您必须提高帐户限值。 默认情况下，每个集群创建以下实例： • 一台 Bootstrap 机器，在安装后删除 • 三个 control plane 机器 • 三个计算（compute）机器 因为 bootstrap 和 control plane 机器使用 **Standard_D8s_v3** 虚拟机 (使用 8 个 vCPU)，因此计算机器使用 **Standard_D4s_v3** 虚拟机（使用 4 个 vCPU），所以默认集群需要 44 个 vCPU。bootstrap 节点 VM（使用 8 个 vCPU）只在安装过程中使用。 若要部署更多 worker 节点、启用自动扩展、部署大型工作负载或使用不同的实例类型，您必须进一步提高帐户的 vCPU 限值，以确保集群可以部署您需要的机器。
OS Disk	7		每个集群机器必须至少有 100 GB 存储和 300 IOPS。虽然这些值是最低支持的值，但对于具有密集型工作负载的生产环境集群和集群，建议使用更快的存储。有关优化性能存储的更多信息，请参阅"扩展和性能"部分中的页面标题为"优化存储"。

组件	默认所需的组件数	默认 Azure 限值	描述
VNet	1	每个区域 1000 个	每个默认集群都需要一个虚拟网络 (VNet)，此网络包括两个子网。
网络接口	7	每个区域 65,536 个	每个默认集群都需要 7 个网络接口。如果您要创建更多机器或者您部署的工作负载要创建负载均衡器，则集群会使用更多的网络接口。
网络安全组	2	5000	每个集群为 VNet 中的每个子网创建网络安全组。默认集群为 control plane 和计算节点子网创建网络安全组： **controlplane**：允许从任何位置通过端口 6443 访问 control plane 机器 **node**：允许从互联网通过端口 80 和 443 访问 worker 节点
网络负载均衡器	3	每个区域 1000 个	每个集群都会创建以下负载均衡器： **default**：用于在 worker 机器之间对端口 80 和 443 的请求进行负载均衡的公共 IP 地址 **internal**：用于在 control plane 机器之间对端口 6443 和 22623 的请求进行负载均衡的专用 IP 地址 **external**：用于在 control plane 机器之间对端口 6443 的请求进行负载均衡的公共 IP 地址 如果您的应用程序创建了更多的 Kubernetes **LoadBalancer** 服务对象，您的集群会使用更多的负载均衡器。
公共 IP 地址	3		两个公共负载均衡器各自使用一个公共 IP 地址。bootstrap 机器也使用一个公共 IP 地址，以便您可以在安装期间通过 SSH 连接到该机器来进行故障排除。bootstrap 节点的 IP 地址仅在安装过程中使用。
专用 IP 地址	7		内部负载均衡器、三台 control plane 机器中的每一台以及三台 worker 机器中的每一台各自使用一个专用 IP 地址。

组件	默认所需的组件数	默认 Azure 限值	描述
Spot VM vCPU（可选）	0 如果配置 spot 虚拟机，您的集群必须为每个计算节点有两个 spot VM vCPU。	每个区域 20 个	这是可选组件。要使用 spot 虚拟机，您必须将 Azure 默认限值增加到集群中至少有两倍的计算节点数量。  **注意** 不建议将 spot 虚拟机用于 control plane 节点。

其他资源

- 优化存储

7.10.2.2. 在 Azure 中配置公共 DNS 区

要安装 OpenShift Container Platform，您使用的 Microsoft Azure 帐户必须在帐户中具有一个专用的公共托管 DNS 区。此区域必须对域具有权威。此服务为集群外部连接提供集群 DNS 解析和名称查询。

流程

1. 标识您的域或子域，以及注册商（registrar）。您可以转移现有的域和注册商，或通过 Azure 或其他来源获取新的域和注册商。

注意

如需通过 Azure 购买域的更多信息，请参阅 Azure 文档中的购买 Azure 应用服务的自定义域名。

2. 如果您使用现有的域和注册商，请将其 DNS 迁移到 Azure。请参阅 Azure 文档中的将活动 DNS 名称迁移到 Azure 应用服务。

3. 为您的域配置 DNS。按照 Azure 文档中教程：在 Azure DNS 中托管域部分里的步骤，为您的域或子域创建一个公共托管区，提取新的权威名称服务器，并更新您的域使用的名称服务器的注册商记录。
使用合适的根域（如 **openshiftcorp.com**）或子域（如 **clusters.openshiftcorp.com**）。

4. 如果您使用子域，请按照您公司的流程将其委派记录添加到父域。

您可以通过访问此 示例来创建 DNS 区域来查看 Azure 的 DNS 解决方案 。

7.10.2.3. 提高 Azure 帐户限值

要提高帐户限值，请在 Azure 门户上提交支持请求。

注意

每一支持请求只能提高一种类型的配额。

流程

1. 从 Azure 门户，点击左下角的 **Help + suport**。
2. 点击 **New support request**，然后选择所需的值：
   a. 从 **Issue type** 列表中，选择 **Service and subscription limits (quotas)**。
   b. 从 **Subscription** 列表中，选择要修改的订阅。
   c. 从 **Quota type** 列表中，选择要提高的配额。例如，选择 **Compute-VM (cores-vCPUs) subscription limit increases** 以增加 vCPU 的数量，这是安装集群所必须的。
   d. 点击 **Next: Solutions**。
3. 在 **Problem Details** 页面中，提供您要提高配额所需的信息：
   a. 点击 **Provide details**，然后在 **Quota details** 窗口中提供所需的详情。
   b. 在 SUPPORT METHOD 和 CONTACT INFO 部分中，提供问题严重性和您的联系详情。
4. 点击 **Next: Review + create**，然后点击 **Create**。

7.10.2.4. 证书签名请求管理

在使用您置备的基础架构时，集群只能有限地访问自动机器管理，因此您必须提供一种在安装后批准集群证书签名请求 (CSR) 的机制。**kube-controller-manager** 只能批准 kubelet 客户端 CSR。**machine-approver** 无法保证使用 kubelet 凭证请求的提供证书的有效性，因为它不能确认是正确的机器发出了该请求。您必须决定并实施一种方法，以验证 kubelet 提供证书请求的有效性并进行批准。

7.10.2.5. 所需的 Azure 角色

OpenShift Container Platform 集群需要一个 Azure 身份来创建和管理 Azure 资源。在创建身份前，请验证您的环境是否满足以下要求：

- 用于创建身份的 Azure 帐户被分配 **User Access Administrator** 和 **Contributor** 角色。在以下情况下需要这些角色：
  - 创建服务主体或用户分配的受管身份。
  - 在虚拟机上启用系统分配的受管身份。
- 如果您要使用服务主体完成安装，请验证您用来创建身份的 Azure 帐户是否已被分配了 Microsoft Entra ID 中的 **microsoft.directory/servicePrincipals/createAsOwner** 权限。

要在 Azure 门户上设置角色，请参阅 Azure 文档中的使用 RBAC 和 Azure 门户管理对 Azure 资源的访问。

7.10.2.6. 用户置备的基础架构所需的 Azure 权限

安装程序需要访问具有所需权限的 Azure 服务主体或受管身份，以部署集群并维护其每日操作。这些权限必须授予与身份关联的 Azure 订阅。

以下选项可供您使用：

- 您可以为身份分配 **Contributor** 和 **User Access Administrator** 角色。分配这些角色是授予所有所需权限的最快速方法。
  有关分配角色的更多信息，请参阅 Azure 文档，使用 Azure 门户管理 Azure 资源的访问。

- 如果机构的安全策略需要更严格的权限集，您可以创建具有所需权限的自定义角色。

在 Microsoft Azure 上创建 OpenShift Container Platform 集群需要以下权限。

例 7.37. 创建授权资源所需的权限

- **Microsoft.Authorization/policies/audit/action**
- **Microsoft.Authorization/policies/auditIfNotExists/action**
- **Microsoft.Authorization/roleAssignments/read**
- **Microsoft.Authorization/roleAssignments/write**

例 7.38. 创建计算资源所需的权限

- **Microsoft.Compute/images/read**
- **Microsoft.Compute/images/write**
- **Microsoft.Compute/images/delete**
- **Microsoft.Compute/availabilitySets/read**
- **Microsoft.Compute/disks/beginGetAccess/action**
- **Microsoft.Compute/disks/delete**
- **Microsoft.Compute/disks/read**
- **Microsoft.Compute/disks/write**
- **Microsoft.Compute/galleries/images/read**
- **Microsoft.Compute/galleries/images/versions/read**
- **Microsoft.Compute/galleries/images/versions/write**
- **Microsoft.Compute/galleries/images/write**
- **Microsoft.Compute/galleries/read**
- **Microsoft.Compute/galleries/write**
- **Microsoft.Compute/snapshots/read**
- **Microsoft.Compute/snapshots/write**
- **Microsoft.Compute/snapshots/delete**
- **Microsoft.Compute/virtualMachines/delete**
- **Microsoft.Compute/virtualMachines/powerOff/action**
- **Microsoft.Compute/virtualMachines/read**

- **Microsoft.Compute/virtualMachines/write**
- **Microsoft.Compute/virtualMachines/deallocate/action**

例 7.39. 创建身份管理资源所需的权限

- **Microsoft.ManagedIdentity/userAssignedIdentities/assign/action**
- **Microsoft.ManagedIdentity/userAssignedIdentities/read**
- **Microsoft.ManagedIdentity/userAssignedIdentities/write**

例 7.40. 创建网络资源所需的权限

- **Microsoft.Network/dnsZones/A/write**
- **Microsoft.Network/dnsZones/CNAME/write**
- **Microsoft.Network/dnszones/CNAME/read**
- **Microsoft.Network/dnszones/read**
- **Microsoft.Network/loadBalancers/backendAddressPools/join/action**
- **Microsoft.Network/loadBalancers/backendAddressPools/read**
- **Microsoft.Network/loadBalancers/backendAddressPools/write**
- **Microsoft.Network/loadBalancers/read**
- **Microsoft.Network/loadBalancers/write**
- **Microsoft.Network/networkInterfaces/delete**
- **Microsoft.Network/networkInterfaces/join/action**
- **Microsoft.Network/networkInterfaces/read**
- **Microsoft.Network/networkInterfaces/write**
- **Microsoft.Network/networkSecurityGroups/join/action**
- **Microsoft.Network/networkSecurityGroups/read**
- **Microsoft.Network/networkSecurityGroups/securityRules/delete**
- **Microsoft.Network/networkSecurityGroups/securityRules/read**
- **Microsoft.Network/networkSecurityGroups/securityRules/write**
- **Microsoft.Network/networkSecurityGroups/write**
- **Microsoft.Network/privateDnsZones/A/read**
- **Microsoft.Network/privateDnsZones/A/write**

- **Microsoft.Network/privateDnsZones/A/delete**
- **Microsoft.Network/privateDnsZones/SOA/read**
- **Microsoft.Network/privateDnsZones/read**
- **Microsoft.Network/privateDnsZones/virtualNetworkLinks/read**
- **Microsoft.Network/privateDnsZones/virtualNetworkLinks/write**
- **Microsoft.Network/privateDnsZones/write**
- **Microsoft.Network/publicIPAddresses/delete**
- **Microsoft.Network/publicIPAddresses/join/action**
- **Microsoft.Network/publicIPAddresses/read**
- **Microsoft.Network/publicIPAddresses/write**
- **Microsoft.Network/virtualNetworks/join/action**
- **Microsoft.Network/virtualNetworks/read**
- **Microsoft.Network/virtualNetworks/subnets/join/action**
- **Microsoft.Network/virtualNetworks/subnets/read**
- **Microsoft.Network/virtualNetworks/subnets/write**
- **Microsoft.Network/virtualNetworks/write**

例 7.41. 检查资源健康状况所需的权限

- **Microsoft.Resourcehealth/healthevent/Activated/action**
- **Microsoft.Resourcehealth/healthevent/InProgress/action**
- **Microsoft.Resourcehealth/healthevent/Pending/action**
- **Microsoft.Resourcehealth/healthevent/Resolved/action**
- **Microsoft.Resourcehealth/healthevent/Updated/action**

例 7.42. 创建资源组所需的权限

- **Microsoft.Resources/subscriptions/resourceGroups/read**
- **Microsoft.Resources/subscriptions/resourcegroups/write**

例 7.43. 创建资源标签所需的权限

- **Microsoft.Resources/tags/write**

例 7.44. 创建存储资源所需的权限

- **Microsoft.Storage/storageAccounts/blobServices/read**
- **Microsoft.Storage/storageAccounts/blobServices/containers/write**
- **Microsoft.Storage/storageAccounts/fileServices/read**
- **Microsoft.Storage/storageAccounts/fileServices/shares/read**
- **Microsoft.Storage/storageAccounts/fileServices/shares/write**
- **Microsoft.Storage/storageAccounts/fileServices/shares/delete**
- **Microsoft.Storage/storageAccounts/listKeys/action**
- **Microsoft.Storage/storageAccounts/read**
- **Microsoft.Storage/storageAccounts/write**

例 7.45. 创建部署所需的权限

- **Microsoft.Resources/deployments/read**
- **Microsoft.Resources/deployments/write**
- **Microsoft.Resources/deployments/validate/action**
- **Microsoft.Resources/deployments/operationstatuses/read**

例 7.46. 创建计算资源的可选权限

- **Microsoft.Compute/availabilitySets/delete**
- **Microsoft.Compute/availabilitySets/write**

例 7.47. 创建 marketplace 虚拟机资源的可选权限

- **Microsoft.MarketplaceOrdering/offertypes/publishers/offers/plans/agreements/read**
- **Microsoft.MarketplaceOrdering/offertypes/publishers/offers/plans/agreements/write**

例 7.48. 启用用户管理加密的可选权限

- **Microsoft.Compute/diskEncryptionSets/read**
- **Microsoft.Compute/diskEncryptionSets/write**
- **Microsoft.Compute/diskEncryptionSets/delete**
- **Microsoft.KeyVault/vaults/read**

- **Microsoft.KeyVault/vaults/write**
- **Microsoft.KeyVault/vaults/delete**
- **Microsoft.KeyVault/vaults/deploy/action**
- **Microsoft.KeyVault/vaults/keys/read**
- **Microsoft.KeyVault/vaults/keys/write**
- **Microsoft.Features/providers/features/register/action**

删除 Microsoft Azure 上的 OpenShift Container Platform 集群需要以下权限。

例 7.49. 删除授权资源所需的权限

- **Microsoft.Authorization/roleAssignments/delete**

例 7.50. 删除计算资源所需的权限

- **Microsoft.Compute/disks/delete**
- **Microsoft.Compute/galleries/delete**
- **Microsoft.Compute/galleries/images/delete**
- **Microsoft.Compute/galleries/images/versions/delete**
- **Microsoft.Compute/virtualMachines/delete**
- **Microsoft.Compute/images/delete**

例 7.51. 删除身份管理资源所需的权限

- **Microsoft.ManagedIdentity/userAssignedIdentities/delete**

例 7.52. 删除网络资源所需的权限

- **Microsoft.Network/dnszones/read**
- **Microsoft.Network/dnsZones/A/read**
- **Microsoft.Network/dnsZones/A/delete**
- **Microsoft.Network/dnsZones/CNAME/read**
- **Microsoft.Network/dnsZones/CNAME/delete**
- **Microsoft.Network/loadBalancers/delete**
- **Microsoft.Network/networkInterfaces/delete**

- **Microsoft.Network/networkSecurityGroups/delete**
- **Microsoft.Network/privateDnsZones/read**
- **Microsoft.Network/privateDnsZones/A/read**
- **Microsoft.Network/privateDnsZones/delete**
- **Microsoft.Network/privateDnsZones/virtualNetworkLinks/delete**
- **Microsoft.Network/publicIPAddresses/delete**
- **Microsoft.Network/virtualNetworks/delete**

例 7.53. 检查资源健康状况所需的权限

- **Microsoft.Resourcehealth/healthevent/Activated/action**
- **Microsoft.Resourcehealth/healthevent/Resolved/action**
- **Microsoft.Resourcehealth/healthevent/Updated/action**

例 7.54. 删除资源组所需的权限

- **Microsoft.Resources/subscriptions/resourcegroups/delete**

例 7.55. 删除存储资源所需的权限

- **Microsoft.Storage/storageAccounts/delete**
- **Microsoft.Storage/storageAccounts/listKeys/action**

注意

要在 Azure 上安装 OpenShift Container Platform，您必须将资源组创建的权限范围到您的订阅。创建资源组后，您可以将剩余权限的范围限定到所创建的资源组。如果其他资源组中存在公共 DNS 区域，则必须始终将网络 DNS 区域相关权限应用到您的订阅。

在删除 OpenShift Container Platform 集群时，您可以将订阅的所有权限限定到您的订阅。

7.10.2.7. 创建服务主体

由于 OpenShift Container Platform 及其安装程序使用 Azure Resource Manager 创建 Microsoft Azure 资源，因此您必须创建一个服务主体来代表它。

先决条件

- 安装或更新 Azure CLI。
- 您的 Azure 帐户具有您所用订阅所需的角色。

- 如果要使用自定义角色，您已创建了带有在 *使用用户置备的基础架构所需的 Azure 权限*中列出的所需权限的一个自定义角色。

流程

1. 登录 Azure CLI：

```
$ az login
```

2. 如果您的 Azure 帐户使用订阅，请确定您使用正确的订阅：

   a. 查看可用帐户列表并记录您要用于集群的订阅的 **tenantId** 值：

```
$ az account list --refresh
```

   输出示例

```
[
  {
    "cloudName": "AzureCloud",
    "id": "9bab1460-96d5-40b3-a78e-17b15e978a80",
    "isDefault": true,
    "name": "Subscription Name",
    "state": "Enabled",
    "tenantId": "6057c7e9-b3ae-489d-a54e-de3f6bf6a8ee",
    "user": {
      "name": "you@example.com",
      "type": "user"
    }
  }
]
```

   b. 查看您的活跃帐户详情，确认 **tenantId** 值与您要使用的订阅匹配：

```
$ az account show
```

   输出示例

```
{
  "environmentName": "AzureCloud",
  "id": "9bab1460-96d5-40b3-a78e-17b15e978a80",
  "isDefault": true,
  "name": "Subscription Name",
  "state": "Enabled",
  "tenantId": "6057c7e9-b3ae-489d-a54e-de3f6bf6a8ee", (1)
  "user": {
    "name": "you@example.com",
    "type": "user"
  }
}
```

   (1) 确保 **tenantId** 参数的值是正确的订阅 ID。

c. 如果您使用的订阅不正确，请更改活跃的订阅：

```
$ az account set -s <subscription_id> 1
```

1 指定订阅 ID。

d. 验证订阅 ID 更新：

```
$ az account show
```

输出示例

```
{
  "environmentName": "AzureCloud",
  "id": "33212d16-bdf6-45cb-b038-f6565b61edda",
  "isDefault": true,
  "name": "Subscription Name",
  "state": "Enabled",
  "tenantId": "8049c7e9-c3de-762d-a54e-dc3f6be6a7ee",
  "user": {
    "name": "you@example.com",
    "type": "user"
  }
}
```

3. 记录输出中的 **tenantId** 和 **id** 参数值。OpenShift Container Platform 安装过程中需要这些值。

4. 为您的帐户创建服务主体：

```
$ az ad sp create-for-rbac --role <role_name> \ 1
    --name <service_principal> \ 2
    --scopes /subscriptions/<subscription_id> 3
```

1 定义角色名称。您可以使用 **Contributor** 角色，或者指定包含所需权限的自定义角色。

2 定义服务主体名称。

3 指定订阅 ID。

输出示例

```
Creating 'Contributor' role assignment under scope '/subscriptions/<subscription_id>'
The output includes credentials that you must protect. Be sure that you do not
include these credentials in your code or check the credentials into your source
control. For more information, see https://aka.ms/azadsp-cli
{
  "appId": "ac461d78-bf4b-4387-ad16-7e32e328aec6",
  "displayName": <service_principal>",
  "password": "00000000-0000-0000-0000-000000000000",
  "tenantId": "8049c7e9-c3de-762d-a54e-dc3f6be6a7ee"
}
```

5. 记录前面输出中 **appId** 和 **password** 参数的值。OpenShift Container Platform 安装过程中需要这些值。

6. 如果您将 **Contributor** 角色应用到服务主体，请运行以下命令来分配 **User Administrator Access** 角色：

```
$ az role assignment create --role "User Access Administrator" \
  --assignee-object-id $(az ad sp show --id <appId> --query id -o tsv) 1
```

1 为您的服务主体指定 **appId** 参数值。

其他资源

- 如需有关 CCO 模式的更多信息，请参阅 关于 Cloud Credential Operator。

7.10.2.8. 支持的 Azure 区域

安装程序会根据您的订阅动态地生成可用的 Microsoft Azure 区域列表。

支持的 Azure 公共区域

- **australiacentral** (Australia Central)
- **australiaeast** (Australia East)
- **australiasoutheast** (Australia South East)
- **brazilsouth** (Brazil South)
- **canadacentral** (Canada Central)
- **canadaeast** (Canada East)
- **centralindia** (Central India)
- **centralus** (Central US)
- **eastasia** (East Asia)
- **eastus** (East US)
- **eastus2** (East US 2)
- **francecentral** (France Central)
- **germanywestcentral** (Germany West Central)
- **israelcentral** (Israel Central)
- **italynorth** (Italy North)
- **japaneast** (Japan East)
- **japanwest** (Japan West)
- **koreacentral** (Korea Central)

- **koreasouth** (Korea South)
- **mexicocentral** (Mexico Central)
- **northcentralus** (North Central US)
- **northeurope** (North Europe)
- **norwayeast** (Norway East)
- **polandcentral** (Poland Central)
- **qatarcentral** (Qatar Central)
- **southafricanorth** (South Africa North)
- **southcentralus** (South Central US)
- **southeastasia** (Southeast Asia)
- **southindia** (South India)
- **swedencentral** (Sweden Central)
- **switzerlandnorth** (Switzerland North)
- **uaenorth** (UAE North)
- **uksouth** (UK South)
- **ukwest** (UK West)
- **westcentralus** (West Central US)
- **westeurope** (West Europe)
- **westindia** (West India)
- **westus** (West US)
- **westus2** (West US 2)
- **westus3** (West US 3)

支持的 Azure 政府区域
OpenShift Container Platform 4.6 添加了对以下 Microsoft Azure Government（MAG）区域的支持：

- **usgovtexas** (US Gov Texas)
- **usgovvirginia** (US Gov Virginia)

您可以参阅 Azure 文档来了解与所有可用 MAG 区域的信息。其他 MAG 区域应该可以与 OpenShift Container Platform 一起工作，但并没有经过测试。

7.10.3. 具有用户置备基础架构的集群的要求

对于包含用户置备的基础架构的集群，您必须部署所有所需的机器。

本节论述了在用户置备的基础架构上部署 OpenShift Container Platform 的要求。

7.10.3.1. 集群安装所需的机器

最小的 OpenShift Container Platform 集群需要以下主机：

表 7.20. 最低所需的主机

主机	描述
一个临时 bootstrap 机器	集群需要 bootstrap 机器在三台 control plane 机器上部署 OpenShift Container Platform 集群。您可在安装集群后删除 bootstrap 机器。
三台 control plane 机器	control plane 机器运行组成 control plane 的 Kubernetes 和 OpenShift Container Platform 服务。
至少两台计算机器，也称为 worker 机器。	OpenShift Container Platform 用户请求的工作负载在计算机器上运行。

重要

要保持集群的高可用性，请将独立的物理主机用于这些集群机器。

bootstrap 和 control plane 机器必须使用 Red Hat Enterprise Linux CoreOS(RHCOS)作为操作系统。但是，计算机器可以在 Red Hat Enterprise Linux CoreOS(RHCOS)、Red Hat Enterprise Linux(RHEL) 8.6 和更高的版本。

请注意，RHCOS 基于 Red Hat Enterprise Linux(RHEL) 9.2，并继承其所有硬件认证和要求。查看 红帽企业 Linux 技术功能和限制。

7.10.3.2. 集群安装的最低资源要求

每台集群机器都必须满足以下最低要求：

表 7.21. 最低资源要求

机器	操作系统	vCPU [1]	虚拟内存	Storage	每秒输入/输出 (IOPS) [2]
bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane（控制平面）	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS、RHEL 8.6 及更新版本 [3]	2	8 GB	100 GB	300

1. 当未启用并发多线程(SMT)或超线程时，一个 vCPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比例：（每个内核数的线程）× sockets = vCPU。

2. OpenShift Container Platform 和 Kubernetes 对磁盘性能非常敏感，建议使用更快的存储速度，特别是 control plane 节点上需要 10 ms p99 fsync 持续时间的 etcd。请注意，在许多云平台上，存储大小和 IOPS 可一起扩展，因此您可能需要过度分配存储卷来获取足够的性能。

3. 与所有用户置备的安装一样，如果您选择在集群中使用 RHEL 计算机器，则负责所有操作系统生命周期管理和维护，包括执行系统更新、应用补丁和完成所有其他必要的任务。RHEL 7 计算机器的使用已弃用，并已在 OpenShift Container Platform 4.10 及更新的版本中删除。

注意

从 OpenShift Container Platform 版本 4.13 开始，RHCOS 基于 RHEL 版本 9.2，它更新了微架构要求。以下列表包含每个架构需要的最小指令集架构 (ISA)：

- x86-64 体系结构需要 x86-64-v2 ISA
- ARM64 架构需要 ARMv8.0-A ISA
- IBM Power 架构需要 Power 9 ISA
- s390x 架构需要 z14 ISA

如需更多信息，请参阅 RHEL 架构。

重要

您需要使用将 **PremiumIO** 参数设置为 **true** 的 Azure 虚拟机。

如果平台的实例类型满足集群机器的最低要求，则 OpenShift Container Platform 支持使用它。

7.10.3.3. 为 Azure 测试的实例类型

以下 Microsoft Azure 实例类型已经 OpenShift Container Platform 测试。

例 7.56. 基于 64 位 x86 架构的机器类型

- **c4.***
- **c5.***
- **c5a.***
- **i3.***
- **m4.***
- **m5.***
- **m5a.***
- **m6a.***
- **m6i.***

- **r4.***
- **r5.***
- **r5a.***
- **r6i.***
- **t3.***
- **t3a.***

7.10.3.4. 在 64 位 ARM 基础架构上为 Azure 测试的实例类型

以下 Microsoft Azure ARM64 实例类型已使用 OpenShift Container Platform 测试。

例 7.57. 基于 64 位 ARM 架构的机器类型

- **c6g.***
- **m6g.***

7.10.4. 使用 Azure Marketplace 产品

使用 Azure Marketplace 产品可让您部署 OpenShift Container Platform 集群，该集群按照使用付费（按小时、每个内核）进行计费，同时仍由红帽直接支持。

要使用 Azure Marketplace 产品部署 OpenShift Container Platform 集群，您必须首先获取 Azure Marketplace 镜像。安装程序使用这个镜像来部署 worker 或 control plane 节点。在获取您的镜像时，请考虑以下事项：

- 虽然镜像相同，但 Azure Marketplace publisher 根据您的区域。如果您位于北美，请将 **redhat** 指定为发布者。如果您位于 EMEA，请将 **redhat-limited** 指定为发布者。
- 此项优惠包括 **rh-ocp-worker** SKU 和 **rh-ocp-worker-gen1** SKU。**rh-ocp-worker** SKU 代表 Hyper-V 生成版本 2 虚拟机镜像。OpenShift Container Platform 中使用的默认实例类型与版本 2 兼容。如果您计划使用与版本 1 兼容的实例类型，请使用与 **rh-ocp-worker-gen1** SKU 关联的镜像。**rh-ocp-worker-gen1** SKU 代表 Hyper-V 版本 1 虚拟机镜像。

重要

在使用 64 位 ARM 实例的集群上不支持使用 Azure marketplace 安装镜像。

先决条件

- 已安装 Azure CLI 客户端 **(az)**。
- 您的 Azure 帐户为产品授权，您使用 Azure CLI 客户端登录到此帐户。

流程

1. 运行以下命令之一，显示所有可用的 OpenShift Container Platform 镜像：

- 北美：

```
$  az vm image list --all --offer rh-ocp-worker --publisher redhat -o table
```

输出示例

```
Offer          Publisher       Sku                Urn                                                                  Version
-------------  --------------  -----------------  -------------------------------------------------------------------
----  -----------------
rh-ocp-worker  RedHat          rh-ocp-worker      RedHat:rh-ocp-worker:rh-ocp-
worker:413.92.2023101700           413.92.2023101700
rh-ocp-worker  RedHat          rh-ocp-worker-gen1  RedHat:rh-ocp-worker:rh-ocp-worker-
gen1:413.92.2023101700       413.92.2023101700
```

- 欧洲、中东和非洲地区：

```
$  az vm image list --all --offer rh-ocp-worker --publisher redhat-limited -o table
```

输出示例

```
Offer          Publisher       Sku                Urn
Version
-------------  --------------  -----------------  -------------------------------------------------------------------
----          -----------------
rh-ocp-worker  redhat-limited  rh-ocp-worker      redhat-limited:rh-ocp-worker:rh-ocp-
worker:413.92.2023101700           413.92.2023101700
rh-ocp-worker  redhat-limited  rh-ocp-worker-gen1  redhat-limited:rh-ocp-worker:rh-ocp-
worker-gen1:413.92.2023101700       413.92.2023101700
```

注意

使用可用于 compute 和 control plane 节点的最新镜像。如果需要，您的虚拟机会在安装过程中自动升级。

2. 运行以下命令之一检查您的所提供的镜像：

- 北美：

```
$ az vm image show --urn redhat:rh-ocp-worker:rh-ocp-worker:<version>
```

- 欧洲、中东和非洲地区：

```
$ az vm image show --urn redhat-limited:rh-ocp-worker:rh-ocp-worker:<version>
```

3. 运行以下命令之一查看提供的术语：

- 北美：

```
$ az vm image terms show --urn redhat:rh-ocp-worker:rh-ocp-worker:<version>
```

- 欧洲、中东和非洲地区：

```
$ az vm image terms show --urn redhat-limited:rh-ocp-worker:rh-ocp-worker:<version>
```

4. 运行以下命令之一接受产品条款：

   - 北美：

     ```
     $ az vm image terms accept --urn redhat:rh-ocp-worker:rh-ocp-worker:<version>
     ```

   - 欧洲、中东和非洲地区：

     ```
     $ az vm image terms accept --urn redhat-limited:rh-ocp-worker:rh-ocp-worker:<version>
     ```

5. 记录您的所提供的镜像详情。如果使用 Azure Resource Manager (ARM) 模板来部署计算节点：

   a. 您可以通过删除 **id** 参数，并使用您的值来添加 **offer**, **publisher**, **sku**, and **version** 参数来更新 **storageProfile.imageReference**。

   b. 为虚拟机 (VM) 指定一个计**划**。

   示例 06_workers.json ARM 模板带有一个更新的 storageProfile.imageReference 对象和一个特定的计划

```
...
  "plan" : {
    "name": "rh-ocp-worker",
    "product": "rh-ocp-worker",
    "publisher": "redhat"
  },
  "dependsOn" : [
    "[concat('Microsoft.Network/networkInterfaces/', concat(variables('vmNames')
[copyIndex()], '-nic'))]"
  ],
  "properties" : {
...
  "storageProfile": {
    "imageReference": {
    "offer": "rh-ocp-worker",
    "publisher": "redhat",
    "sku": "rh-ocp-worker",
    "version": "413.92.2023101700"
    }
    ...
   }
...
  }
```

7.10.4.1. 获取安装程序

在安装 OpenShift Container Platform 前，将安装文件下载到您用于安装的主机上。

先决条件

- 您有一台运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB。

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐户，请使用您的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入到安装类型的页面，下载与您的主机操作系统和架构对应的安装程序，并将该文件放在您要存储安装配置文件的目录中。

重要

安装程序会在用来安装集群的计算机上创建几个文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。

重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，请为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。此 pull secret 允许您与所含授权机构提供的服务进行身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

7.10.4.2. 为集群节点 SSH 访问生成密钥对

在 OpenShift Container Platform 安装过程中，您可以为安装程序提供 SSH 公钥。密钥通过它们的 Ignition 配置文件传递给 Red Hat Enterprise Linux CoreOS(RHCOS)节点，用于验证对节点的 SSH 访问。密钥添加到每个节点上 **core** 用户的 **~/.ssh/authorized_keys** 列表中，这将启用免密码身份验证。

将密钥传递给节点后，您可以使用密钥对作为用户 **核心** 通过 SSH 连接到 RHCOS 节点。若要通过 SSH 访问节点，必须由 SSH 为您的本地用户管理私钥身份。

如果要通过 SSH 连接到集群节点来执行安装调试或灾难恢复，则必须在安装过程中提供 SSH 公钥。**./openshift-install gather** 命令还需要在集群节点上设置 SSH 公钥。

重要

不要在生产环境中跳过这个过程，在生产环境中需要灾难恢复和调试。

注意

您必须使用本地密钥，而不是使用特定平台方法配置 的密钥，如 AWS 密钥对。

流程

1. 如果您在本地计算机上没有可用于在集群节点上进行身份验证的现有 SSH 密钥对，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 ~/**.ssh/id_ed25519**。如果您已有密钥对，请确保您的公钥位于 ~/**.ssh** 目录中。

注意

如果您计划在 **x86_64**、**ppc64le** 和 **s390x** 架构上安装使用 RHEL 加密库（这些加密库已提交给 NIST 用于 FIPS 140-2/140-3 验证）的 OpenShift Container Platform 集群，则不要创建使用 **ed25519** 算法的密钥。相反，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 查看公共 SSH 密钥：

```
$ cat <path>/<file_name>.pub
```

例如，运行以下命令来查看 ~/**.ssh/id_ed25519.pub** 公钥：

```
$ cat ~/.ssh/id_ed25519.pub
```

3. 将 SSH 私钥身份添加到本地用户的 SSH 代理（如果尚未添加）。在集群节点上，或者要使用 **./openshift-install gather** 命令，需要对该密钥进行 SSH 代理管理，才能在集群节点上进行免密码 SSH 身份验证。

注意

在某些发行版中，自动管理默认 SSH 私钥身份，如 ~/**.ssh/id_rsa** 和 ~/**.ssh/id_dsa**。

a. 如果 **ssh-agent** 进程尚未为您的本地用户运行，请将其作为后台任务启动：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果集群处于 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

4. 将 SSH 私钥添加到 **ssh-agent** ：

```
$ ssh-add <path>/<file_name> 1
```

1 指定 SSH 私钥的路径和文件名，如 ~/**.ssh/id_ed25519.pub**

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

后续步骤

- 安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。如果在您置备的基础架构上安装集群，则必须为安装程序提供密钥。

7.10.5. 创建用于 Azure 的安装文件

要使用用户置备的基础架构在 Microsoft Azure 上安装 OpenShift Container Platform，您必须生成并修改安装程序部署集群所需的文件，以便集群只创建要使用的机器。您要生成并自定义 **install-config.yaml** 文件、Kubernetes 清单和 Ignition 配置文件。您还可以选择在安装准备阶段首先设置独立 **var** 分区。

7.10.5.1. 可选：创建独立 /var 分区

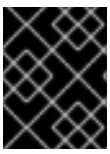
建议安装程序将 OpenShift Container Platform 的磁盘分区保留给安装程序。然而，在有些情况下您可能需要在文件系统的一部分中创建独立分区。

OpenShift Container Platform 支持添加单个分区来将存储附加到 **/var** 分区或 **/var** 的子目录中。例如：

- **/var/lib/containers** ：保存随着系统中添加更多镜像和容器而增长的容器相关内容。
- **/var/lib/etcd** ：保存您可能希望独立保留的数据，比如 etcd 存储的性能优化。
- **/var** ：保存您可能希望独立保留的数据，以满足审计等目的。

通过单独存储 **/var** 目录的内容，可以更轻松地根据需要为区域扩展存储，并在以后重新安装 OpenShift Container Platform，并保持该数据的完整性。使用这个方法，您不必再次拉取所有容器，在更新系统时也不必复制大量日志文件。

因为 **/var** 在进行一个全新的 Red Hat Enterprise Linux CoreOS（RHCOS）安装前必需存在，所以这个流程会在 OpenShift Container Platform 安装过程的 **openshift-install** 准备阶段插入一个创建的机器配置清单的机器配置来设置独立的 **/var** 分区。



重要

如果按照以下步骤在此流程中创建独立 **/var** 分区，则不需要再次创建 Kubernetes 清单和 Ignition 配置文件，如本节所述。

流程

1. 创建存放 OpenShift Container Platform 安装文件的目录：

```
$ mkdir $HOME/clusterconfig
```

2. 运行 **openshift-install**，以在 **manifest** 和 **openshift** 子目录中创建一组文件。在系统提示时回答系统问题：

```
$ openshift-install create manifests --dir $HOME/clusterconfig
```

输出示例

```
? SSH Public Key ...
```

```
INFO Credentials loaded from the "myprofile" profile in file "/home/myuser/.aws/credentials"
INFO Consuming Install Config from target directory
INFO Manifests created in: $HOME/clusterconfig/manifests and
$HOME/clusterconfig/openshift
```

3. 可选：确认安装程序在 **clusterconfig/openshift** 目录中创建了清单：

```
$ ls $HOME/clusterconfig/openshift/
```

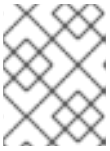
输出示例

```
99_kubeadmin-password-secret.yaml
99_openshift-cluster-api_master-machines-0.yaml
99_openshift-cluster-api_master-machines-1.yaml
99_openshift-cluster-api_master-machines-2.yaml
...
```

4. 创建用于配置额外分区的 Butane 配置。例如，将文件命名为 **$HOME/clusterconfig/98-var-partition.bu**，将磁盘设备名称改为 **worker** 系统上存储设备的名称，并根据情况设置存储大小。这个示例将 **/var** 目录放在一个单独的分区中：

```
variant: openshift
version: 4.16.0
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
storage:
  disks:
  - device: /dev/disk/by-id/<device_name> 1
    partitions:
    - label: var
      start_mib: <partition_start_offset> 2
      size_mib: <partition_size> 3
      number: 5
  filesystems:
    - device: /dev/disk/by-partlabel/var
      path: /var
      format: xfs
      mount_options: [defaults, prjquota] 4
      with_mount_unit: true
```

1. 要分区的磁盘的存储设备名称。
2. 在引导磁盘中添加数据分区时，推荐最少使用 25000 MiB(Mebibytes)。root 文件系统会自动调整大小以填充所有可用空间（最多到指定的偏移值）。如果没有指定值，或者指定的值小于推荐的最小值，则生成的 root 文件系统会太小，而在以后进行的 RHCOS 重新安装可能会覆盖数据分区的开始部分。
3. 以兆字节为单位的数据分区大小。
4. 对于用于容器存储的文件系统，必须启用 **prjquota** 挂载选项。



注意

当创建单独的 **/var** 分区时，如果不同的实例类型没有相同的设备名称，则无法为 worker 节点使用不同的实例类型。

5. 从 Butane 配置创建一个清单，并将它保存到 **clusterconfig/openshift** 目录中。例如，运行以下命令：

```
$ butane $HOME/clusterconfig/98-var-partition.bu -o $HOME/clusterconfig/openshift/98-var-partition.yaml
```

6. 再次运行 **openshift-install**，从 **manifest** 和 **openshift** 子目录中的一组文件创建 Ignition 配置：

```
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
auth  bootstrap.ign  master.ign  metadata.json  worker.ign
```

现在，您可以使用 Ignition 配置文件作为安装程序的输入来安装 Red Hat Enterprise Linux CoreOS(RHCOS)系统。

7.10.5.2. 创建安装配置文件

您可以自定义在 Microsoft Azure 上安装的 OpenShift Container Platform 集群。

先决条件

- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。对于受限网络安装，这些文件位于您的镜像主机上。
- 您有创建镜像 registry 期间生成的 **imageContentSources** 值。
- 您已获取了镜像 registry 的证书内容。
- 您已检索了 Red Hat Enterprise Linux CoreOS (RHCOS) 镜像，并将其上传到可访问的位置。
- 您有一个 Azure 订阅 ID 和租户 ID。
- 如果要使用服务主体安装集群，则有其应用程序 ID 和密码。
- 如果您要使用系统分配的受管身份安装集群，需要在您要从其中运行安装程序的虚拟机上启用它。
- 如果您要使用用户分配的受管身份安装集群，需要满足以下先决条件：
  - 您有它的客户端 ID。
  - 您已将其分配给您要从其运行安装程序的虚拟机。

流程

1. 可选：如果您之前在这个计算机上运行安装程序，并希望使用替代的服务主体或受管身份，请进入 **~/.azure/** 目录并删除 **osServicePrincipal.json** 配置文件。
   删除此文件可防止安装程序自动重复使用之前安装中的订阅和验证值。

2. 创建 **install-config.yaml** 文件。

a. 进入包含安装程序的目录并运行以下命令：

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定要存储安装程序创建的文件的目录名称。

在指定目录时：

- 验证该目录是否具有**执行**权限。在安装目录中运行 Terraform 二进制文件需要这个权限。
- 使用空目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

b. 在提示符处，提供云的配置详情：

i. 可选：选择用于访问集群机器的 SSH 密钥。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

ii. 选择 **azure** 作为目标平台。
如果安装程序无法找到之前安装中的 **osServicePrincipal.json** 配置文件，会提示您输入 Azure 订阅和验证值。

iii. 为您的订阅输入以下 Azure 参数值：

- **Azure subscription id**：输入用于集群的订阅 ID。
- **Azure 租户 id**：输入租户 ID。

iv. 根据您用来部署集群的 Azure 身份，在提示输入 **azure 服务主体客户端 id**时执行以下操作之一：

- 如果您使用服务主体，请输入其应用程序 ID。
- 如果您使用系统分配的受管身份，请将此值设为空白。
- 如果您使用用户分配的受管身份，请指定其客户端 ID。

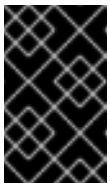
v. 根据您用来部署集群的 Azure 身份，在提示输入 **azure 服务主体客户端 secret**时执行以下操作之一：

- 如果您使用服务主体，请输入其密码。
- 如果您使用系统分配的受管身份，请将此值设为空白。
- 如果您使用用户分配的受管身份，请将此值设为空白。

vi. 选择要将集群部署到的区域。

vii. 选择集群要部署到的基域。基域与您为集群创建的 Azure DNS 区对应。

viii. 为集群输入一个描述性名称。



重要

所有通过公共端点提供的 Azure 资源均存在资源名称的限制，您无法创建使用某些名称的资源。如需 Azure 限制词语列表，请参阅 Azure 文档中的解决保留资源名称错误。

ix. 粘贴 Red Hat OpenShift Cluster Manager 中的 pull secret 。

3. 编辑 **install-config.yaml** 文件，以提供在受限网络中安装所需的额外信息。

a. 更新 **pullSecret** 值，使其包含 registry 的身份验证信息：

```
pullSecret: '{"auths":{"<mirror_host_name>:5000": {"auth": "<credentials>","email":
"you@example.com"}}}'
```

对于 **<mirror_host_name>**，请指定 您在镜像 registry 证书中指定的 registry 域名 ；对于 **<credentials>**， 请指定您的镜像 registry 的 base64 编码用户名和密码。

b. 添加 **additionalTrustBundle** 参数和值。

```
additionalTrustBundle: |
  -----BEGIN CERTIFICATE-----

ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
  -----END CERTIFICATE-----
```

该值必须是您用于镜像 registry 的证书文件内容。证书文件可以是现有的可信证书颁发机构，也可以是您为镜像 registry 生成的自签名证书。

c. 在 **platform.azure** 字段中定义 VNet 的网络和子网，以安装集群：

```
networkResourceGroupName: <vnet_resource_group> 1
virtualNetwork: <vnet> 2
controlPlaneSubnet: <control_plane_subnet> 3
computeSubnet: <compute_subnet> 4
```

1. 将 **<vnet_resource_group>** 替换为包含现有虚拟网络 (VNet)的资源组名称。
2. 将 **<vnet>** 替换为现有的虚拟网络名称。
3. 将 **<control_plane_subnet>** 替换为部署 control plane 机器的现有子网名称。
4. 将 **<compute_subnet>** 替换为部署计算机器的现有子网名称。

d. 添加镜像内容资源，类似于以下 YAML 摘录：

```
imageContentSources:
- mirrors:
  - <mirror_host_name>:5000/<repo_name>/release
  source: quay.io/openshift-release-dev/ocp-release
```

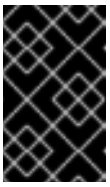
```
- mirrors:
  - <mirror_host_name>:5000/<repo_name>/release
  source: registry.redhat.io/ocp/release
```

对于这些值，请使用您在创建镜像 registry 时记录的 **imageContentSources**。

e. 可选：将发布策略设置为 **Internal**：

```
publish: Internal
```

通过设置这个选项，您可以创建一个内部 Ingress Controller 和一个私有负载均衡器。



重要

Azure Firewall 无法与 Azure 公共负载均衡器无缝工作。因此，当使用 Azure Firewall 来限制互联网访问时，**install-config.yaml** 中的 **publish** 字段应设置为 **Internal**。

4. 对您需要的 **install-config.yaml** 文件进行任何其他修改。
有关参数的更多信息，请参阅"安装配置参数"。

5. 备份 **install-config.yaml** 文件，以便您可以使用它安装多个集群。

重要

install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用该文件，您必须立即备份该文件。

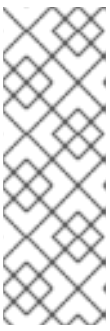
在以前的版本中，安装程序会创建一个 **osServicePrincipal.json** 配置文件，并将此文件存储在计算机上的 **~/.azure/** 目录中。这样可确保安装程序在目标平台上创建 OpenShift Container Platform 集群时可以加载配置集。

7.10.5.3. 在安装过程中配置集群范围的代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 **Proxy** 对象的 **spec.noProxy** 字段中添加站点来绕过代理。



注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，**Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

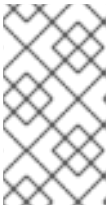
1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 **.** 以仅匹配子域。例如，**.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代理。
4. 如果提供，安装程序会在 **openshift-config** 命名空间中生成名为 **user-ca-bundle** 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 **trusted-ca-bundle** 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， **Proxy** 对象的 **trustedCA** 字段中也会引用此配置映射。**additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。
5. 可选：决定 **Proxy** 对象的配置以引用 **trustedCA** 字段中 **user-ca-bundle** 配置映射的策略。允许的值是 **Proxyonly** 和 **Always**。仅在配置了 **http/https** 代理时，使用 **Proxyonly** 引用 **user-ca-bundle** 配置映射。使用 **Always** 始终引用 **user-ca-bundle** 配置映射。默认值为 **Proxyonly**。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。



注意

如果安装程序超时，重启并使用安装程序的 **wait-for** 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. 保存该文件并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 cluster 的集群范围代理，该代理 **使用** 提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

7.10.5.4. 为 ARM 模板导出常用变量

您必须导出与提供的 Azure Resource Manager（ARM）模板搭配使用的一组常用变量，它们有助于在 Microsoft Azure 上完成用户提供基础架构安装。

注意

特定的 ARM 模板可能还需要其他导出变量，这些变量在相关的程序中详细介绍。

先决条件

- 获取 OpenShift Container Platform 安装程序和集群的 pull secret。

流程

1. 导出 **install-config.yaml** 中由提供的 ARM 模板使用的通用变量：

```
$ export CLUSTER_NAME=<cluster_name> (1)
$ export AZURE_REGION=<azure_region> (2)
$ export SSH_KEY=<ssh_key> (3)
$ export BASE_DOMAIN=<base_domain> (4)
$ export BASE_DOMAIN_RESOURCE_GROUP=<base_domain_resource_group> (5)
```

(1) **install-config.yaml** 文件中的**.metadata.name** 属性的值。

(2) 集群要部署到的区域，如 **centralus** 。这是来自 **install-config.yaml** 文件中的 **.platform.azure.region** 属性的值。

(3) 作为字符串的 SSH RSA 公钥文件。您必须使用引号包括 SSH 密钥，因为它包含空格。这是来自 **install-config.yaml** 文件中的 **.sshKey** 属性的值。

(4) 集群要部署到的基域。基域与您为集群创建的公共 DNS 区对应。这是来自 **install-config.yaml** 文件中的 **.baseDomain** 属性的值。

(5) 公共 DNS 区所在的资源组。这是来自 **install-config.yaml** 文件中的 **.platform.azure.baseDomainResourceGroupName** 属性的值。

例如：

```
$ export CLUSTER_NAME=test-cluster
$ export AZURE_REGION=centralus
$ export SSH_KEY="ssh-rsa xxx/xxx/xxx= user@email.com"
$ export BASE_DOMAIN=example.com
$ export BASE_DOMAIN_RESOURCE_GROUP=ocp-cluster
```

2. 导出 kubeadmin 凭证：

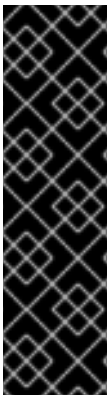
```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig (1)
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

7.10.5.5. 创建 Kubernetes 清单和 Ignition 配置文件

由于您必须修改一些集群定义文件并手动启动集群机器，因此您必须生成 Kubernetes 清单和 Ignition 配置文件来配置机器。

安装配置文件转换为 Kubernetes 清单。清单嵌套到 Ignition 配置文件中，稍后用于配置集群机器。



重要

- OpenShift Container Platform 安装程序生成的 Ignition 配置文件包含 24 小时后过期的证书，然后在该时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

先决条件

- 已获得 OpenShift Container Platform 安装程序。
- 已创建 **install-config.yaml** 安装配置文件。

流程

1. 进入包含 OpenShift Container Platform 安装程序的目录，并为集群生成 Kubernetes 清单：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定包含您创建的 **install-config.yaml** 文件的安装目录。

2. 删除定义 control plane 机器的 Kubernetes 清单文件：

```
$ rm -f <installation_directory>/openshift/99_openshift-cluster-api_master-machines-*.yaml
```

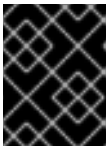
通过删除这些文件，您可以防止集群自动生成 control plane 机器。

3. 删除定义 control plane 机器集的 Kubernetes 清单文件：

```
$ rm -f <installation_directory>/openshift/99_openshift-machine-api_master-control-plane-machine-set.yaml
```

4. 删除定义 worker 机器的 Kubernetes 清单文件：

```
$ rm -f <installation_directory>/openshift/99_openshift-cluster-api_worker-machineset-*.yaml
```



重要

如果在用户置备的基础架构上安装集群时禁用了 **MachineAPI** 功能，则必须删除定义 worker 机器的 Kubernetes 清单文件。否则，集群将无法安装。

由于您要自行创建和管理 worker 机器，因此不需要初始化这些机器。

5. 检查 **<installation_directory>/manifests/cluster-scheduler-02-config.yml Kubernetes 清单文件中的 mastersSchedulable** 参数是否已设置为 **false**。此设置可防止在 control plane 机器上调度 pod：

   a. 打开 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** 文件。

   b. 找到 **mastersSchedulable** 参数，并确保它被设置为 **false**。

   c. 保存并退出 文件。

6. 可选：如果您不希望 Ingress Operator 代表您创建 DNS 记录，请删除 **<installation_directory>/manifests/cluster-dns-02-config.yml** DNS 配置文件中的 **privateZone 和 publicZone** 部分：

```
apiVersion: config.openshift.io/v1
kind: DNS
metadata:
  creationTimestamp: null
  name: cluster
spec:
  baseDomain: example.openshift.com
  privateZone: 1
    id: mycluster-100419-private-zone
  publicZone: 2
    id: example.openshift.com
status: {}
```

1 2 完全删除此部分。

如果您这样做，后续步骤中必须手动添加入口 DNS 记录。

7. 在用户置备的基础架构上配置 Azure 时，您必须导出清单文件中定义的一些常见变量，以备稍后在 Azure Resource Manager（ARM）模板中使用：

   a. 使用以下命令导出基础架构 ID:

```
$ export INFRA_ID=<infra_id> 1
```

   1 OpenShift Container Platform 集群被分配了一个标识符（**INFRA_ID**），其格式为 **<cluster_name>-<random_string>**。这将作为使用提供的 ARM 模板创建的大部分资源的基本名称。这是来自 **manifests/cluster-infrastructure-02-config.yml** 文件中的 **.status.infrastructureName** 属性的值。

   b. 使用以下命令导出资源组：

```
$ export RESOURCE_GROUP=<resource_group> 1
```

1 此 Azure 部署中创建的所有资源都作为资源组的一部分。资源组名称还基于 **INFRA_ID**，格式为 **<cluster_name>-<random_string>-rg**。这是来自

8. 要创建 Ignition 配置文件，从包含安装程序的目录运行以下命令：

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定相同的安装目录。

为安装目录中的 bootstrap、control plane 和计算节点创建 Ignition 配置文件。**kubeadmin-password** 和 **kubeconfig** 文件是在 **./<installation_directory>/auth** 目录中创建的：

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

7.10.6. 创建 Azure 资源组

您必须创建一个 Microsoft Azure 资源组以及该资源组的身份。它们都用于在 Azure 上安装 OpenShift Container Platform 集群。

先决条件

- 配置 Azure 帐户。
- 为集群生成 Ignition 配置文件。

流程

1. 在受支持的 Azure 区域中创建资源组：

```
$ az group create --name ${RESOURCE_GROUP} --location ${AZURE_REGION}
```

2. 为资源组创建 Azure 身份：

```
$ az identity create -g ${RESOURCE_GROUP} -n ${INFRA_ID}-identity
```

这用于授予集群中 Operator 所需的访问权限。例如，这允许 Ingress Operator 创建公共 IP 及其负载均衡器。您必须将 Azure 身份分配给角色。

3. 将 Contributor 角色授予 Azure 身份：

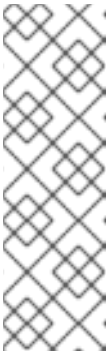
   a. 导出 Azure 角色分配所需的以下变量：

```
$ export PRINCIPAL_ID=`az identity show -g ${RESOURCE_GROUP} -n ${INFRA_ID}-identity --query principalId --out tsv`
```

```
$ export RESOURCE_GROUP_ID=`az group show -g ${RESOURCE_GROUP} --query
id --out tsv`
```

b. 将 Contributor 角色分配给身份：

```
$ az role assignment create --assignee "${PRINCIPAL_ID}" --role 'Contributor' --scope
"${RESOURCE_GROUP_ID}"
```



注意

如果要为身份分配具有所有所需权限的自定义角色，请运行以下命令：

```
$ az role assignment create --assignee "${PRINCIPAL_ID}" --role
<custom_role> \ 1
--scope "${RESOURCE_GROUP_ID}"
```

1 指定自定义角色名称。

7.10.7. 上传 RHCOS 集群镜像和 bootstrap Ignition 配置文件

Azure 客户端不支持基于本地现有文件进行部署。您必须复制 RHCOS 虚拟硬盘(VHD)集群镜像，并将 bootstrap Ignition 配置文件存储在存储容器中，以便在部署过程中访问它们。

先决条件

- 配置 Azure 帐户。
- 为集群生成 Ignition 配置文件。

流程

1. 创建 Azure 存储帐户以存储 VHD 集群镜像：

```
$ az storage account create -g ${RESOURCE_GROUP} --location ${AZURE_REGION} --
name ${CLUSTER_NAME}sa --kind Storage --sku Standard_LRS
```

警告

Azure 存储帐户名称的长度必须在 3 到 24 个字符之，且只使用数字和小写字母。如果您的 **CLUSTER_NAME** 变量没有遵循这些限制，您必须手动定义 Azure 存储帐户名称。如需有关 Azure 存储帐户名称限制的更多信息，请参阅 Azure 文档中的解决存储帐户名称的错误。

2. 将存储帐户密钥导出为环境变量：

```
$ export ACCOUNT_KEY=`az storage account keys list -g ${RESOURCE_GROUP} --
account-name ${CLUSTER_NAME}sa --query "[0].value" -o tsv`
```

3. 将 RHCOS VHD 的 URL 导出为环境变量：

```
$ export VHD_URL=`openshift-install coreos print-stream-json | jq -r '.architectures.
<architecture>."rhel-coreos-extensions"."azure-disk".url'`
```

其中：

<architecture>

指定架构，有效值包括 **x86_64** 或 **aarch64**。

重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每个发行版本而改变。您必须指定一个最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。如果可用，请使用与 OpenShift Container Platform 版本匹配的镜像版本。

4. 为 VHD 创建存储容器：

```
$ az storage container create --name vhd --account-name ${CLUSTER_NAME}sa --account-
key ${ACCOUNT_KEY}
```

5. 将本地 VHD 复制为一个 blob:

```
$ az storage blob copy start --account-name ${CLUSTER_NAME}sa --account-key
${ACCOUNT_KEY} --destination-blob "rhcos.vhd" --destination-container vhd --source-uri
"${VHD_URL}"
```

6. 创建 blob 存储容器并上传生成的 **bootstrap.ign** 文件：

```
$ az storage container create --name files --account-name ${CLUSTER_NAME}sa --
account-key ${ACCOUNT_KEY}
```

```
$ az storage blob upload --account-name ${CLUSTER_NAME}sa --account-key
${ACCOUNT_KEY} -c "files" -f "<installation_directory>/bootstrap.ign" -n "bootstrap.ign"
```

7.10.8. 创建 DNS 区示例

使用用户置备的基础架构的集群需要 DNS 记录。您应该选择适合您的场景的 DNS 策略。

在本例中，使用了 Azure 的 DNS 解决方案，因此您将为外部（内部网络）可见性创建一个新的公共 DNS 区域，并为内部集群解析创建一个私有 DNS 区域。

注意

公共 DNS 区域不需要与集群部署位于同一个资源组中，且可能已在您的机构中为所需基域存在。如果情况如此，您可以跳过创建公共 DNS 区这一步；请确定您之前生成的安装配置反映了这种情况。

先决条件

- 配置 Azure 帐户。
- 为集群生成 Ignition 配置文件。

流程

1. 在 **BASE_DOMAIN_RESOURCE_GROUP** 环境变量中导出的资源组中创建新的公共 DNS 区域：

```
$ az network dns zone create -g ${BASE_DOMAIN_RESOURCE_GROUP} -n
${CLUSTER_NAME}.${BASE_DOMAIN}
```

   如果您使用的是公共 DNS 区域，可以跳过这一步。

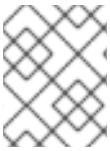
2. 在与这个部署的其余部分相同的资源组中创建私有 DNS 区域：

```
$ az network private-dns zone create -g ${RESOURCE_GROUP} -n
${CLUSTER_NAME}.${BASE_DOMAIN}
```

如需了解更多信息，请参阅在 Azure 中配置公共 DNS 的信息。

7.10.9. 在 Azure 中创建 VNet

您必须在 Microsoft Azure 中创建虚拟网络（VNet），供您的 OpenShift Container Platform 集群使用。您可以对 VNet 进行定制来满足您的要求。创建 VNet 的一种方法是修改提供的 Azure Resource Manager（ARM）模板。



注意

如果不使用提供的 ARM 模板来创建 Azure 基础架构，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 Azure 帐户。
- 为集群生成 Ignition 配置文件。

流程

1. 复制 **VNet 的 ARM 模板** 一节中的模板，并将它以 **01_vnet.json** 保存到集群的安装目录中。此模板描述了集群所需的 VNet。
2. 使用 **az** CLI 创建部署：

```
$ az deployment group create -g ${RESOURCE_GROUP} \
  --template-file "<installation_directory>/01_vnet.json" \
  --parameters baseName="${INFRA_ID}" 1
```

   1 资源名称使用的基本名称；这通常是集群的基础架构 ID。

3. 将 VNet 模板链接到私有 DNS 区域：

```
$ az network private-dns link vnet create -g ${RESOURCE_GROUP} -z
${CLUSTER_NAME}.${BASE_DOMAIN} -n ${INFRA_ID}-network-link -v "${INFRA_ID}-vnet"
-e false
```

7.10.9.1. VNet 的 ARM 模板

您可以使用以下 Azure Resource Manager（ARM）模板来部署 OpenShift Container Platform 集群所需的 VPC：

例 7.58. 01_vnet.json ARM 模板

```
{
  "$schema" : "https://schema.management.azure.com/schemas/2015-01-
01/deploymentTemplate.json#",
  "contentVersion" : "1.0.0.0",
  "parameters" : {
    "baseName" : {
      "type" : "string",
      "minLength" : 1,
      "metadata" : {
        "description" : "Base name to be used in resource names (usually the cluster's Infra ID)"
      }
    }
  },
  "variables" : {
    "location" : "[resourceGroup().location]",
    "virtualNetworkName" : "[concat(parameters('baseName'), '-vnet')]",
    "addressPrefix" : "10.0.0.0/16",
    "masterSubnetName" : "[concat(parameters('baseName'), '-master-subnet')]",
    "masterSubnetPrefix" : "10.0.0.0/24",
    "nodeSubnetName" : "[concat(parameters('baseName'), '-worker-subnet')]",
    "nodeSubnetPrefix" : "10.0.1.0/24",
    "clusterNsgName" : "[concat(parameters('baseName'), '-nsg')]"
  },
  "resources" : [
    {
      "apiVersion" : "2018-12-01",
      "type" : "Microsoft.Network/virtualNetworks",
      "name" : "[variables('virtualNetworkName')]",
      "location" : "[variables('location')]",
      "dependsOn" : [
        "[concat('Microsoft.Network/networkSecurityGroups/', variables('clusterNsgName'))]"
      ],
      "properties" : {
        "addressSpace" : {
          "addressPrefixes" : [
            "[variables('addressPrefix')]"
          ]
        },
        "subnets" : [
          {
            "name" : "[variables('masterSubnetName')]",
            "properties" : {
              "addressPrefix" : "[variables('masterSubnetPrefix')]",
              "serviceEndpoints": [],
```

```
            "networkSecurityGroup" : {
              "id" : "[resourceId('Microsoft.Network/networkSecurityGroups',
variables('clusterNsgName'))]"
            }
          }
        },
        {
          "name" : "[variables('nodeSubnetName')]",
          "properties" : {
            "addressPrefix" : "[variables('nodeSubnetPrefix')]",
            "serviceEndpoints": [],
            "networkSecurityGroup" : {
              "id" : "[resourceId('Microsoft.Network/networkSecurityGroups',
variables('clusterNsgName'))]"
            }
          }
        }
      ]
    }
  },
  {
    "type" : "Microsoft.Network/networkSecurityGroups",
    "name" : "[variables('clusterNsgName')]",
    "apiVersion" : "2018-10-01",
    "location" : "[variables('location')]",
    "properties" : {
      "securityRules" : [
        {
          "name" : "apiserver_in",
          "properties" : {
            "protocol" : "Tcp",
            "sourcePortRange" : "*",
            "destinationPortRange" : "6443",
            "sourceAddressPrefix" : "*",
            "destinationAddressPrefix" : "*",
            "access" : "Allow",
            "priority" : 101,
            "direction" : "Inbound"
          }
        }
      ]
    }
  }
 ]
}
```

7.10.10. 为 Azure 基础架构创建 RHCOS 集群镜像

您必须对 OpenShift Container Platform 节点的 Microsoft Azure 使用有效的 Red Hat Enterprise Linux CoreOS（RHCOS）镜像。

先决条件

- 配置 Azure 帐户。

- 为集群生成 Ignition 配置文件。
- 将 RHCOS 虚拟硬盘（VHD）集群镜像存储在 Azure 存储容器中。
- 在 Azure 存储容器中存储 bootstrap Ignition 配置文件。

流程

1. 复制**镜像存储的 ARM 模板** 部分中的模板，并将它以 **02_storage.json** 保存到集群的安装目录中。此模板描述了集群所需的镜像存储。
2. 以一个变量的形式将 RHCOS VHD blob URL 导出：

```
$ export VHD_BLOB_URL=`az storage blob url --account-name ${CLUSTER_NAME}sa --
account-key ${ACCOUNT_KEY} -c vhd -n "rhcos.vhd" -o tsv`
```

3. 部署集群镜像

```
$ az deployment group create -g ${RESOURCE_GROUP} \
  --template-file "<installation_directory>/02_storage.json" \
  --parameters vhdBlobURL="${VHD_BLOB_URL}" \ 1
  --parameters baseName="${INFRA_ID}" \ 2
  --parameters storageAccount="${CLUSTER_NAME}sa" \ 3
  --parameters architecture="<architecture>" 4
```

1. 用于创建 master 和 worker 机器的 RHCOS VHD 的 blob URL。
2. 资源名称使用的基本名称；这通常是集群的基础架构 ID。
3. Azure 存储帐户的名称。
4. 指定系统架构。有效值为 **x64** （默认）或 **Arm64**。

7.10.10.1. 镜像存储的 ARM 模板

您可以使用以下 Azure Resource Manager（ARM）模板来部署 OpenShift Container Platform 集群所需的存储的 Red Hat Enterprise Linux CoreOS（RHCOS）镜像：

例 7.59. 02_storage.json ARM 模板

```
{
  "$schema": "https://schema.management.azure.com/schemas/2019-04-
01/deploymentTemplate.json#",
  "contentVersion": "1.0.0.0",
  "parameters": {
    "architecture": {
      "type": "string",
      "metadata": {
        "description": "The architecture of the Virtual Machines"
      },
      "defaultValue": "x64",
      "allowedValues": [
        "Arm64",
```

```
        "x64"
      ]
    },
    "baseName": {
      "type": "string",
      "minLength": 1,
      "metadata": {
        "description": "Base name to be used in resource names (usually the cluster's Infra ID)"
      }
    },
    "storageAccount": {
      "type": "string",
      "metadata": {
        "description": "The Storage Account name"
      }
    },
    "vhdBlobURL": {
      "type": "string",
      "metadata": {
        "description": "URL pointing to the blob where the VHD to be used to create master and
worker machines is located"
      }
    }
  },
  "variables": {
    "location": "[resourceGroup().location]",
    "galleryName": "[concat('gallery_', replace(parameters('baseName'), '-', '_'))]",
    "imageName": "[parameters('baseName')]",
    "imageNameGen2": "[concat(parameters('baseName'), '-gen2')]",
    "imageRelease": "1.0.0"
  },
  "resources": [
    {
      "apiVersion": "2021-10-01",
      "type": "Microsoft.Compute/galleries",
      "name": "[variables('galleryName')]",
      "location": "[variables('location')]",
      "resources": [
        {
          "apiVersion": "2021-10-01",
          "type": "images",
          "name": "[variables('imageName')]",
          "location": "[variables('location')]",
          "dependsOn": [
            "[variables('galleryName')]"
          ],
          "properties": {
            "architecture": "[parameters('architecture')]",
            "hyperVGeneration": "V1",
            "identifier": {
              "offer": "rhcos",
              "publisher": "RedHat",
              "sku": "basic"
            },
            "osState": "Generalized",
            "osType": "Linux"
```

```
        },
        "resources": [
          {
            "apiVersion": "2021-10-01",
            "type": "versions",
            "name": "[variables('imageRelease')]",
            "location": "[variables('location')]",
            "dependsOn": [
              "[variables('imageName')]"
            ],
            "properties": {
              "publishingProfile": {
                "storageAccountType": "Standard_LRS",
                "targetRegions": [
                  {
                    "name": "[variables('location')]",
                    "regionalReplicaCount": "1"
                  }
                ]
              },
              "storageProfile": {
                "osDiskImage": {
                  "source": {
                    "id": "[resourceId('Microsoft.Storage/storageAccounts',
parameters('storageAccount'))]",
                    "uri": "[parameters('vhdBlobURL')]"
                  }
                }
              }
            }
          }
        ]
      },
      {
        "apiVersion": "2021-10-01",
        "type": "images",
        "name": "[variables('imageNameGen2')]",
        "location": "[variables('location')]",
        "dependsOn": [
          "[variables('galleryName')]"
        ],
        "properties": {
          "architecture": "[parameters('architecture')]",
          "hyperVGeneration": "V2",
          "identifier": {
            "offer": "rhcos-gen2",
            "publisher": "RedHat-gen2",
            "sku": "gen2"
          },
          "osState": "Generalized",
          "osType": "Linux"
        },
        "resources": [
          {
            "apiVersion": "2021-10-01",
            "type": "versions",
```

```
            "name": "[variables('imageRelease')]",
            "location": "[variables('location')]",
            "dependsOn": [
              "[variables('imageNameGen2')]"
            ],
            "properties": {
              "publishingProfile": {
                "storageAccountType": "Standard_LRS",
                "targetRegions": [
                  {
                    "name": "[variables('location')]",
                    "regionalReplicaCount": "1"
                  }
                ]
              },
              "storageProfile": {
                "osDiskImage": {
                  "source": {
                    "id": "[resourceId('Microsoft.Storage/storageAccounts',
parameters('storageAccount'))]",
                    "uri": "[parameters('vhdBlobURL')]"
                  }
                }
              }
            }
          }
        ]
      }
    ]
  }
}
```

7.10.11. 用户置备的基础架构对网络的要求

所有 Red Hat Enterprise Linux CoreOS（RHCOS）机器需要在启动过程中在 **initramfs** 中配置网络，以获取其 Ignition 配置文件。

7.10.11.1. 网络连接要求

您必须配置机器之间的网络连接，以允许 OpenShift Container Platform 集群组件进行通信。每台机器都必须能够解析集群中所有其他机器的主机名。

本节详细介绍了所需的端口。

重要

在连接的 OpenShift Container Platform 环境中，所有节点都需要访问互联网才能为平台容器拉取镜像，并向红帽提供遥测数据。

表 7.22. 用于全机器到所有机器通信的端口

协议	port	描述
ICMP	N/A	网络可访问性测试
TCP	**1936**	指标
	9000-9999	主机级别的服务，包括端口 9 **100-9101 上的节点导出器**，以及端口 **9099** 上的 Cluster Version Operator。
	10250-10259	Kubernetes 保留的默认端口
UDP	**4789**	VXLAN
	6081	Geneve
	9000-9999	主机级别的服务，包括端口 **9100**到**9101** 上的节点导出器。
	500	IPsec IKE 数据包
	4500	IPsec NAT-T 数据包
	123	UDP 端口 **123** 上的网络时间协议 (NTP) 如果配置了外部 NTP 时间服务器，需要打开 UDP 端口 **123**。
TCP/UDP	**30000-32767**	Kubernetes 节点端口
ESP	N/A	IPsec Encapsulating Security Payload(ESP)

表 7.23. 用于所有机器控制平面通信的端口

协议	port	描述
TCP	**6443**	Kubernetes API

表 7.24. control plane 机器用于 control plane 机器通信的端口

协议	port	描述
TCP	**2379-2380**	etcd 服务器和对等端口

7.10.12. 在 Azure 中创建网络和负载均衡组件

您必须在 Microsoft Azure 中配置网络和负载均衡，供您的 OpenShift Container Platform 集群使用。创建这些组件的一种方法是修改提供的 Azure Resource Manager（ARM）模板。

注意

如果不使用提供的 ARM 模板来创建 Azure 基础架构，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 Azure 帐户。
- 为集群生成 Ignition 配置文件。
- 在 Azure 中创建和配置 VNet 及相关子网。

流程

1. 复制 **网络和负载均衡器的 ARM 模板** 一节中的模板，并将它以 **03_infra.json** 保存到集群的安装目录中。此模板描述了集群所需的网络和负载均衡对象。
2. 使用 **az** CLI 创建部署：

```
$ az deployment group create -g ${RESOURCE_GROUP} \
  --template-file "<installation_directory>/03_infra.json" \
  --parameters privateDNSZoneName="${CLUSTER_NAME}.${BASE_DOMAIN}" \ 1
  --parameters baseName="${INFRA_ID}" 2
```

1. 私有 DNS 区的名称。
2. 资源名称使用的基本名称；这通常是集群的基础架构 ID。

3. 在公共区为 API 公共负载均衡器创建一个 **api** DNS 记录。**${BASE_DOMAIN_RESOURCE_GROUP}** 变量必须指向存在公共 DNS 区的资源组。

   a. 导出以下变量：

```
$ export PUBLIC_IP=`az network public-ip list -g ${RESOURCE_GROUP} --query "[?
name=='${INFRA_ID}-master-pip'] | [0].ipAddress" -o tsv`
```

   b. 在新的公共区中创建 **api** DNS 记录：

```
$ az network dns record-set a add-record -g ${BASE_DOMAIN_RESOURCE_GROUP} -
z ${CLUSTER_NAME}.${BASE_DOMAIN} -n api -a ${PUBLIC_IP} --ttl 60
```

   如果要将集群添加到现有的公共区，您可以在其中创建 **api** DNS 记录：

```
$ az network dns record-set a add-record -g ${BASE_DOMAIN_RESOURCE_GROUP} -
z ${BASE_DOMAIN} -n api.${CLUSTER_NAME} -a ${PUBLIC_IP} --ttl 60
```

7.10.12.1. 网络和负载均衡器的 ARM 模板

您可以使用以下 Azure Resource Manager(ARM)模板来部署 OpenShift Container Platform 集群所需的网络对象和负载均衡器：

例 7.60. 03_infra.json ARM 模板

```
{
  "$schema" : "https://schema.management.azure.com/schemas/2015-01-
01/deploymentTemplate.json#",
  "contentVersion" : "1.0.0.0",
  "parameters" : {
    "baseName" : {
      "type" : "string",
      "minLength" : 1,
      "metadata" : {
        "description" : "Base name to be used in resource names (usually the cluster's Infra ID)"
      }
    },
    "vnetBaseName": {
      "type": "string",
      "defaultValue": "",
      "metadata" : {
        "description" : "The specific customer vnet's base name (optional)"
      }
    },
    "privateDNSZoneName" : {
      "type" : "string",
      "metadata" : {
        "description" : "Name of the private DNS zone"
      }
    }
  },
  "variables" : {
    "location" : "[resourceGroup().location]",
    "virtualNetworkName" : "[concat(if(not(empty(parameters('vnetBaseName'))),
parameters('vnetBaseName'), parameters('baseName')), '-vnet')]",
    "virtualNetworkID" : "[resourceId('Microsoft.Network/virtualNetworks',
variables('virtualNetworkName'))]",
    "masterSubnetName" : "[concat(if(not(empty(parameters('vnetBaseName'))),
parameters('vnetBaseName'), parameters('baseName')), '-master-subnet')]",
    "masterSubnetRef" : "[concat(variables('virtualNetworkID'), '/subnets/',
variables('masterSubnetName'))]",
    "masterPublicIpAddressName" : "[concat(parameters('baseName'), '-master-pip')]",
    "masterPublicIpAddressID" : "[resourceId('Microsoft.Network/publicIPAddresses',
variables('masterPublicIpAddressName'))]",
    "masterLoadBalancerName" : "[parameters('baseName')]",
    "masterLoadBalancerID" : "[resourceId('Microsoft.Network/loadBalancers',
variables('masterLoadBalancerName'))]",
    "internalLoadBalancerName" : "[concat(parameters('baseName'), '-internal-lb')]",
    "internalLoadBalancerID" : "[resourceId('Microsoft.Network/loadBalancers',
variables('internalLoadBalancerName'))]",
    "skuName": "Standard"
  },
  "resources" : [
    {
      "apiVersion" : "2018-12-01",
      "type" : "Microsoft.Network/publicIPAddresses",
      "name" : "[variables('masterPublicIpAddressName')]",
      "location" : "[variables('location')]",
      "sku": {
        "name": "[variables('skuName')]"
      },
```

```
    "properties" : {
      "publicIPAllocationMethod" : "Static",
      "dnsSettings" : {
        "domainNameLabel" : "[variables('masterPublicIpAddressName')]"
      }
    }
  },
  {
    "apiVersion" : "2018-12-01",
    "type" : "Microsoft.Network/loadBalancers",
    "name" : "[variables('masterLoadBalancerName')]",
    "location" : "[variables('location')]",
    "sku": {
      "name": "[variables('skuName')]"
    },
    "dependsOn" : [
      "[concat('Microsoft.Network/publicIPAddresses/', variables('masterPublicIpAddressName'))]"
    ],
    "properties" : {
      "frontendIPConfigurations" : [
        {
          "name" : "public-lb-ip-v4",
          "properties" : {
            "publicIPAddress" : {
              "id" : "[variables('masterPublicIpAddressID')]"
            }
          }
        }
      ],
      "backendAddressPools" : [
        {
          "name" : "[variables('masterLoadBalancerName')]"
        }
      ],
      "loadBalancingRules" : [
        {
          "name" : "api-internal",
          "properties" : {
            "frontendIPConfiguration" : {
              "id" :"[concat(variables('masterLoadBalancerID'), '/frontendIPConfigurations/public-lb-ip-
v4')]"
            },
            "backendAddressPool" : {
              "id" : "[concat(variables('masterLoadBalancerID'), '/backendAddressPools/',
variables('masterLoadBalancerName'))]"
            },
            "protocol" : "Tcp",
            "loadDistribution" : "Default",
            "idleTimeoutInMinutes" : 30,
            "frontendPort" : 6443,
            "backendPort" : 6443,
            "probe" : {
              "id" : "[concat(variables('masterLoadBalancerID'), '/probes/api-internal-probe')]"
            }
          }
        }
```

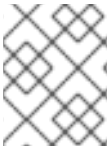
```
        ],
        "probes" : [
          {
            "name" : "api-internal-probe",
            "properties" : {
              "protocol" : "Https",
              "port" : 6443,
              "requestPath": "/readyz",
              "intervalInSeconds" : 10,
              "numberOfProbes" : 3
            }
          }
        ]
      }
    },
    {
      "apiVersion" : "2018-12-01",
      "type" : "Microsoft.Network/loadBalancers",
      "name" : "[variables('internalLoadBalancerName')]",
      "location" : "[variables('location')]",
      "sku": {
        "name": "[variables('skuName')]"
      },
      "properties" : {
        "frontendIPConfigurations" : [
          {
            "name" : "internal-lb-ip",
            "properties" : {
              "privateIPAllocationMethod" : "Dynamic",
              "subnet" : {
                "id" : "[variables('masterSubnetRef')]"
              },
              "privateIPAddressVersion" : "IPv4"
            }
          }
        ],
        "backendAddressPools" : [
          {
            "name" : "internal-lb-backend"
          }
        ],
        "loadBalancingRules" : [
          {
            "name" : "api-internal",
            "properties" : {
              "frontendIPConfiguration" : {
                "id" : "[concat(variables('internalLoadBalancerID'), '/frontendIPConfigurations/internal-lb-
ip')]"
              },
              "frontendPort" : 6443,
              "backendPort" : 6443,
              "enableFloatingIP" : false,
              "idleTimeoutInMinutes" : 30,
              "protocol" : "Tcp",
              "enableTcpReset" : false,
              "loadDistribution" : "Default",
```

```
            "backendAddressPool" : {
              "id" : "[concat(variables('internalLoadBalancerID'), '/backendAddressPools/internal-lb-
backend')]"
            },
            "probe" : {
              "id" : "[concat(variables('internalLoadBalancerID'), '/probes/api-internal-probe')]"
            }
          }
        },
        {
          "name" : "sint",
          "properties" : {
            "frontendIPConfiguration" : {
              "id" : "[concat(variables('internalLoadBalancerID'), '/frontendIPConfigurations/internal-lb-
ip')]"
            },
            "frontendPort" : 22623,
            "backendPort" : 22623,
            "enableFloatingIP" : false,
            "idleTimeoutInMinutes" : 30,
            "protocol" : "Tcp",
            "enableTcpReset" : false,
            "loadDistribution" : "Default",
            "backendAddressPool" : {
              "id" : "[concat(variables('internalLoadBalancerID'), '/backendAddressPools/internal-lb-
backend')]"
            },
            "probe" : {
              "id" : "[concat(variables('internalLoadBalancerID'), '/probes/sint-probe')]"
            }
          }
        }
      ],
      "probes" : [
        {
          "name" : "api-internal-probe",
          "properties" : {
            "protocol" : "Https",
            "port" : 6443,
            "requestPath": "/readyz",
            "intervalInSeconds" : 10,
            "numberOfProbes" : 3
          }
        },
        {
          "name" : "sint-probe",
          "properties" : {
            "protocol" : "Https",
            "port" : 22623,
            "requestPath": "/healthz",
            "intervalInSeconds" : 10,
            "numberOfProbes" : 3
          }
        }
      ]
    }
```

```
    },
    {
      "apiVersion": "2018-09-01",
      "type": "Microsoft.Network/privateDnsZones/A",
      "name": "[concat(parameters('privateDNSZoneName'), '/api')]",
      "location" : "[variables('location')]",
      "dependsOn" : [
        "[concat('Microsoft.Network/loadBalancers/', variables('internalLoadBalancerName'))]"
      ],
      "properties": {
        "ttl": 60,
        "aRecords": [
          {
            "ipv4Address": "
[reference(variables('internalLoadBalancerName')).frontendIPConfigurations[0].properties.privateIP
Address]"
          }
        ]
      }
    },
    {
      "apiVersion": "2018-09-01",
      "type": "Microsoft.Network/privateDnsZones/A",
      "name": "[concat(parameters('privateDNSZoneName'), '/api-int')]",
      "location" : "[variables('location')]",
      "dependsOn" : [
        "[concat('Microsoft.Network/loadBalancers/', variables('internalLoadBalancerName'))]"
      ],
      "properties": {
        "ttl": 60,
        "aRecords": [
          {
            "ipv4Address": "
[reference(variables('internalLoadBalancerName')).frontendIPConfigurations[0].properties.privateIP
Address]"
          }
        ]
      }
    }
  ]
}
```

7.10.13. 在 Azure 中创建 bootstrap 机器

您必须在 Microsoft Azure 中创建 bootstrap 机器，以便在 OpenShift Container Platform 集群初始化过程中使用。创建此机器的一种方法是修改提供的 Azure Resource Manager（ARM）模板。



注意

如果不使用提供的 ARM 模板来创建 bootstrap 机器，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 Azure 帐户。
- 为集群生成 Ignition 配置文件。
- 在 Azure 中创建和配置 VNet 及相关子网。
- 在 Azure 中创建和配置联网及负载均衡器。
- 创建 control plane 和计算角色。

流程

1. 复制 **bootstrap 机器的 ARM 模板一节中的模板**，并将它以 **04_bootstrap.json** 保存到集群的安装目录中。此模板描述了集群所需的 bootstrap 机器。
2. 导出 bootstrap URL 变量：

```
$ bootstrap_url_expiry=`date -u -d "10 hours" '+%Y-%m-%dT%H:%MZ'`
```

```
$ export BOOTSTRAP_URL=`az storage blob generate-sas -c 'files' -n 'bootstrap.ign' --https-only --full-uri --permissions r --expiry $bootstrap_url_expiry --account-name ${CLUSTER_NAME}sa --account-key ${ACCOUNT_KEY} -o tsv`
```

3. 导出 bootstrap ignition 变量：

```
$ export BOOTSTRAP_IGNITION=`jq -rcnM --arg v "3.2.0" --arg url ${BOOTSTRAP_URL} '{ignition:{version:$v,config:{replace:{source:$url}}}}' | base64 | tr -d '\n'`
```

4. 使用 **az** CLI 创建部署：

```
$ az deployment group create -g ${RESOURCE_GROUP} \
  --template-file "<installation_directory>/04_bootstrap.json" \
  --parameters bootstrapIgnition="${BOOTSTRAP_IGNITION}" \ 1
  --parameters baseName="${INFRA_ID}" \ 2
  --parameter bootstrapVMSize="Standard_D4s_v3" 3
```

1. bootstrap 集群的 bootstrap Ignition 内容。
2. 资源名称使用的基本名称；这通常是集群的基础架构 ID。
3. 可选：指定 bootstrap 虚拟机的大小。使用与指定架构兼容的虚拟机大小。如果未定义这个值，则会设置模板中的默认值。

7.10.13.1. bootstrap 机器的 ARM 模板

您可以使用以下 Azure Resource Manager(ARM)模板来部署 OpenShift Container Platform 集群所需的 bootstrap 机器：

例 7.61. 04_bootstrap.json ARM 模板

```
{
  "$schema" : "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",
```

```
  "contentVersion" : "1.0.0.0",
  "parameters" : {
    "baseName" : {
      "type" : "string",
      "minLength" : 1,
      "metadata" : {
        "description" : "Base name to be used in resource names (usually the cluster's Infra ID)"
      }
    },
    "vnetBaseName": {
      "type": "string",
      "defaultValue": "",
      "metadata" : {
        "description" : "The specific customer vnet's base name (optional)"
      }
    },
    "bootstrapIgnition" : {
      "type" : "string",
      "minLength" : 1,
      "metadata" : {
        "description" : "Bootstrap ignition content for the bootstrap cluster"
      }
    },
    "sshKeyData" : {
      "type" : "securestring",
      "defaultValue" : "Unused",
      "metadata" : {
        "description" : "Unused"
      }
    },
    "bootstrapVMSize" : {
      "type" : "string",
      "defaultValue" : "Standard_D4s_v3",
      "metadata" : {
        "description" : "The size of the Bootstrap Virtual Machine"
      }
    },
    "hyperVGen": {
      "type": "string",
      "metadata": {
        "description": "VM generation image to use"
      },
      "defaultValue": "V2",
      "allowedValues": [
        "V1",
        "V2"
      ]
    }
  },
  "variables" : {
    "location" : "[resourceGroup().location]",
    "virtualNetworkName" : "[concat(if(not(empty(parameters('vnetBaseName'))),
parameters('vnetBaseName'), parameters('baseName')), '-vnet')]",
    "virtualNetworkID" : "[resourceId('Microsoft.Network/virtualNetworks',
variables('virtualNetworkName'))]",
    "masterSubnetName" : "[concat(if(not(empty(parameters('vnetBaseName'))),
```

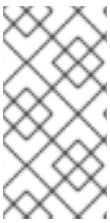
```
parameters('vnetBaseName'), parameters('baseName')), '-master-subnet')]",
    "masterSubnetRef" : "[concat(variables('virtualNetworkID'), '/subnets/',
variables('masterSubnetName'))]",
    "masterLoadBalancerName" : "[parameters('baseName')]",
    "internalLoadBalancerName" : "[concat(parameters('baseName'), '-internal-lb')]",
    "sshKeyPath" : "/home/core/.ssh/authorized_keys",
    "identityName" : "[concat(parameters('baseName'), '-identity')]",
    "vmName" : "[concat(parameters('baseName'), '-bootstrap')]",
    "nicName" : "[concat(variables('vmName'), '-nic')]",
    "galleryName": "[concat('gallery_', replace(parameters('baseName'), '-', '_'))]",
    "imageName" : "[concat(parameters('baseName'), if(equals(parameters('hyperVGen'), 'V2'), '-
gen2', ''))]",
    "clusterNsgName" : "[concat(if(not(empty(parameters('vnetBaseName'))),
parameters('vnetBaseName'), parameters('baseName')), '-nsg')]",
    "sshPublicIpAddressName" : "[concat(variables('vmName'), '-ssh-pip')]"
  },
  "resources" : [
    {
      "apiVersion" : "2018-12-01",
      "type" : "Microsoft.Network/publicIPAddresses",
      "name" : "[variables('sshPublicIpAddressName')]",
      "location" : "[variables('location')]",
      "sku": {
        "name": "Standard"
      },
      "properties" : {
        "publicIPAllocationMethod" : "Static",
        "dnsSettings" : {
          "domainNameLabel" : "[variables('sshPublicIpAddressName')]"
        }
      }
    },
    {
      "apiVersion" : "2018-06-01",
      "type" : "Microsoft.Network/networkInterfaces",
      "name" : "[variables('nicName')]",
      "location" : "[variables('location')]",
      "dependsOn" : [
        "[resourceId('Microsoft.Network/publicIPAddresses', variables('sshPublicIpAddressName'))]"
      ],
      "properties" : {
        "ipConfigurations" : [
          {
            "name" : "pipConfig",
            "properties" : {
              "privateIPAllocationMethod" : "Dynamic",
              "publicIPAddress": {
                "id": "[resourceId('Microsoft.Network/publicIPAddresses',
variables('sshPublicIpAddressName'))]"
              },
              "subnet" : {
                "id" : "[variables('masterSubnetRef')]"
              },
              "loadBalancerBackendAddressPools" : [
                {
                  "id" : "[concat('/subscriptions/', subscription().subscriptionId, '/resourceGroups/',
```

```
resourceGroup().name, '/providers/Microsoft.Network/loadBalancers/',
variables('masterLoadBalancerName'), '/backendAddressPools/',
variables('masterLoadBalancerName'))]"
              },
              {
                "id" : "[concat('/subscriptions/', subscription().subscriptionId, '/resourceGroups/',
resourceGroup().name, '/providers/Microsoft.Network/loadBalancers/',
variables('internalLoadBalancerName'), '/backendAddressPools/internal-lb-backend')]"
              }
            ]
          }
        }
      ]
    }
  },
  {
    "apiVersion" : "2018-06-01",
    "type" : "Microsoft.Compute/virtualMachines",
    "name" : "[variables('vmName')]",
    "location" : "[variables('location')]",
    "identity" : {
      "type" : "userAssigned",
      "userAssignedIdentities" : {
        "[resourceID('Microsoft.ManagedIdentity/userAssignedIdentities/',
variables('identityName'))]" : {}
      }
    },
    "dependsOn" : [
      "[concat('Microsoft.Network/networkInterfaces/', variables('nicName'))]"
    ],
    "properties" : {
      "hardwareProfile" : {
        "vmSize" : "[parameters('bootstrapVMSize')]"
      },
      "osProfile" : {
        "computerName" : "[variables('vmName')]",
        "adminUsername" : "core",
        "adminPassword" : "NotActuallyApplied!",
        "customData" : "[parameters('bootstrapIgnition')]",
        "linuxConfiguration" : {
          "disablePasswordAuthentication" : false
        }
      },
      "storageProfile" : {
        "imageReference": {
          "id": "[resourceId('Microsoft.Compute/galleries/images', variables('galleryName'),
variables('imageName'))]"
        },
        "osDisk" : {
          "name": "[concat(variables('vmName'),'_OSDisk')]",
          "osType" : "Linux",
          "createOption" : "FromImage",
          "managedDisk": {
            "storageAccountType": "Premium_LRS"
          },
          "diskSizeGB" : 100
```

```
        }
       },
       "networkProfile" : {
        "networkInterfaces" : [
         {
          "id" : "[resourceId('Microsoft.Network/networkInterfaces', variables('nicName'))]"
         }
        ]
       }
      }
     },
     {
      "apiVersion" : "2018-06-01",
      "type": "Microsoft.Network/networkSecurityGroups/securityRules",
      "name" : "[concat(variables('clusterNsgName'), '/bootstrap_ssh_in')]",
      "location" : "[variables('location')]",
      "dependsOn" : [
       "[resourceId('Microsoft.Compute/virtualMachines', variables('vmName'))]"
      ],
      "properties": {
       "protocol" : "Tcp",
       "sourcePortRange" : "*",
       "destinationPortRange" : "22",
       "sourceAddressPrefix" : "*",
       "destinationAddressPrefix" : "*",
       "access" : "Allow",
       "priority" : 100,
       "direction" : "Inbound"
      }
     }
    ]
}
```

7.10.14. 在 Azure 中创建 control plane 机器

您必须在 Microsoft Azure 中创建 control plane 机器，供您的集群使用。创建这些机器的一种方法是修改提供的 Azure Resource Manager（ARM）模板。



注意

默认情况下，Microsoft Azure 将 control plane 机器和计算机器放在预先设置的可用区中。您可以为计算节点或 control plane 节点手动设置可用区。要做到这一点，通过在虚拟机资源的 **zones** 参数中指定每个可用区来修改供应商的 Azure Resource Manager (ARM) 模板。

如果不使用提供的 ARM 模板来创建 control plane 机器，您必须检查提供的信息并手动创建基础架构。如果您的集群没有正确初始化，请考虑与安装日志联系红帽支持。

先决条件

- 配置 Azure 帐户。
- 为集群生成 Ignition 配置文件。

- 在 Azure 中创建和配置 VNet 及相关子网。
- 在 Azure 中创建和配置联网及负载均衡器。
- 创建 control plane 和计算角色。
- 创建 bootstrap 机器。

流程

1. 复制 **control plane 机器的 ARM 模板**一节中的模板，并将它以 **05_masters.json** 保存到集群的安装目录中。此模板描述了集群所需的 control plane 机器。
2. 导出 control plane 机器部署所需的以下变量：

```
$ export MASTER_IGNITION=`cat <installation_directory>/master.ign | base64 | tr -d '\n'`
```

3. 使用 **az** CLI 创建部署：

```
$ az deployment group create -g ${RESOURCE_GROUP} \
  --template-file "<installation_directory>/05_masters.json" \
  --parameters masterIgnition="${MASTER_IGNITION}" \ 1
  --parameters baseName="${INFRA_ID}" \ 2
  --parameters masterVMSize="Standard_D8s_v3" 3
```

1. control plane 节点的 Ignition 内容。
2. 资源名称使用的基本名称；这通常是集群的基础架构 ID。
3. 可选：指定 Control Plane 虚拟机的大小。使用与指定架构兼容的虚拟机大小。如果未定义这个值，则会设置模板中的默认值。

7.10.14.1. control plane 机器的 ARM 模板

您可以使用以下 Azure Resource Manager(ARM)模板来部署 OpenShift Container Platform 集群所需的 control plane 机器：

例 7.62. 05_masters.json ARM 模板

```
{
  "$schema" : "https://schema.management.azure.com/schemas/2015-01-
01/deploymentTemplate.json#",
  "contentVersion" : "1.0.0.0",
  "parameters" : {
    "baseName" : {
      "type" : "string",
      "minLength" : 1,
      "metadata" : {
        "description" : "Base name to be used in resource names (usually the cluster's Infra ID)"
      }
    },
    "vnetBaseName": {
      "type": "string",
      "defaultValue": "",
```

```
    "metadata" : {
      "description" : "The specific customer vnet's base name (optional)"
    }
  },
  "masterIgnition" : {
    "type" : "string",
    "metadata" : {
      "description" : "Ignition content for the master nodes"
    }
  },
  "numberOfMasters" : {
    "type" : "int",
    "defaultValue" : 3,
    "minValue" : 2,
    "maxValue" : 30,
    "metadata" : {
      "description" : "Number of OpenShift masters to deploy"
    }
  },
  "sshKeyData" : {
    "type" : "securestring",
    "defaultValue" : "Unused",
    "metadata" : {
      "description" : "Unused"
    }
  },
  "privateDNSZoneName" : {
    "type" : "string",
    "defaultValue" : "",
    "metadata" : {
      "description" : "unused"
    }
  },
  "masterVMSize" : {
    "type" : "string",
    "defaultValue" : "Standard_D8s_v3",
    "metadata" : {
      "description" : "The size of the Master Virtual Machines"
    }
  },
  "diskSizeGB" : {
    "type" : "int",
    "defaultValue" : 1024,
    "metadata" : {
      "description" : "Size of the Master VM OS disk, in GB"
    }
  },
  "hyperVGen": {
    "type": "string",
    "metadata": {
      "description": "VM generation image to use"
    },
    "defaultValue": "V2",
    "allowedValues": [
      "V1",
      "V2"
```

```
      ]
    }
  },
  "variables" : {
    "location" : "[resourceGroup().location]",
    "virtualNetworkName" : "[concat(if(not(empty(parameters('vnetBaseName'))),
parameters('vnetBaseName'), parameters('baseName')), '-vnet')]",
    "virtualNetworkID" : "[resourceId('Microsoft.Network/virtualNetworks',
variables('virtualNetworkName'))]",
    "masterSubnetName" : "[concat(if(not(empty(parameters('vnetBaseName'))),
parameters('vnetBaseName'), parameters('baseName')), '-master-subnet')]",
    "masterSubnetRef" : "[concat(variables('virtualNetworkID'), '/subnets/',
variables('masterSubnetName'))]",
    "masterLoadBalancerName" : "[parameters('baseName')]",
    "internalLoadBalancerName" : "[concat(parameters('baseName'), '-internal-lb')]",
    "sshKeyPath" : "/home/core/.ssh/authorized_keys",
    "identityName" : "[concat(parameters('baseName'), '-identity')]",
    "galleryName": "[concat('gallery_', replace(parameters('baseName'), '-', '_'))]",
    "imageName" : "[concat(parameters('baseName'), if(equals(parameters('hyperVGen'), 'V2'), '-
gen2', ''))]",
    "copy" : [
     {
       "name" : "vmNames",
       "count" :  "[parameters('numberOfMasters')]",
       "input" : "[concat(parameters('baseName'), '-master-', copyIndex('vmNames'))]"
     }
    ]
  },
  "resources" : [
    {
      "apiVersion" : "2018-06-01",
      "type" : "Microsoft.Network/networkInterfaces",
      "copy" : {
        "name" : "nicCopy",
        "count" : "[length(variables('vmNames'))]"
      },
      "name" : "[concat(variables('vmNames')[copyIndex()], '-nic')]",
      "location" : "[variables('location')]",
      "properties" : {
        "ipConfigurations" : [
          {
            "name" : "pipConfig",
            "properties" : {
              "privateIPAllocationMethod" : "Dynamic",
              "subnet" : {
                "id" : "[variables('masterSubnetRef')]"
              },
              "loadBalancerBackendAddressPools" : [
                {
                  "id" : "[concat('/subscriptions/', subscription().subscriptionId, '/resourceGroups/',
resourceGroup().name, '/providers/Microsoft.Network/loadBalancers/',
variables('masterLoadBalancerName'), '/backendAddressPools/',
variables('masterLoadBalancerName'))]"
                },
                {
                  "id" : "[concat('/subscriptions/', subscription().subscriptionId, '/resourceGroups/',
```

```
resourceGroup().name, '/providers/Microsoft.Network/loadBalancers/',
variables('internalLoadBalancerName'), '/backendAddressPools/internal-lb-backend')]"
              }
            ]
          }
        }
      ]
    }
  },
  {
    "apiVersion" : "2018-06-01",
    "type" : "Microsoft.Compute/virtualMachines",
    "copy" : {
      "name" : "vmCopy",
      "count" : "[length(variables('vmNames'))]"
    },
    "name" : "[variables('vmNames')[copyIndex()]]",
    "location" : "[variables('location')]",
    "identity" : {
      "type" : "userAssigned",
      "userAssignedIdentities" : {
        "[resourceID('Microsoft.ManagedIdentity/userAssignedIdentities/',
variables('identityName'))]" : {}
      }
    },
    "dependsOn" : [
      "[concat('Microsoft.Network/networkInterfaces/', concat(variables('vmNames')[copyIndex()], '-
nic'))]"
    ],
    "properties" : {
      "hardwareProfile" : {
        "vmSize" : "[parameters('masterVMSize')]"
      },
      "osProfile" : {
        "computerName" : "[variables('vmNames')[copyIndex()]]",
        "adminUsername" : "core",
        "adminPassword" : "NotActuallyApplied!",
        "customData" : "[parameters('masterIgnition')]",
        "linuxConfiguration" : {
          "disablePasswordAuthentication" : false
        }
      },
      "storageProfile" : {
        "imageReference": {
          "id": "[resourceId('Microsoft.Compute/galleries/images', variables('galleryName'),
variables('imageName'))]"
        },
        "osDisk" : {
          "name": "[concat(variables('vmNames')[copyIndex()], '_OSDisk')]",
          "osType" : "Linux",
          "createOption" : "FromImage",
          "caching": "ReadOnly",
          "writeAcceleratorEnabled": false,
          "managedDisk": {
            "storageAccountType": "Premium_LRS"
          },
```

```
            "diskSizeGB" : "[parameters('diskSizeGB')]"
          }
        },
        "networkProfile" : {
          "networkInterfaces" : [
            {
              "id" : "[resourceId('Microsoft.Network/networkInterfaces', concat(variables('vmNames')
[copyIndex()], '-nic'))]",
              "properties": {
                "primary": false
              }
            }
          ]
        }
      }
    }
  ]
}
```

7.10.15. 等待 bootstrap 完成并删除 Azure 中的 bootstrap 资源

在 Microsoft Azure 中创建所有所需的基础架构后，请等待您通过安装程序生成的 Ignition 配置文件所置备的机器上完成 bootstrap 过程。

先决条件

- 配置 Azure 帐户。
- 为集群生成 Ignition 配置文件。
- 在 Azure 中创建和配置 VNet 及相关子网。
- 在 Azure 中创建和配置联网及负载均衡器。
- 创建 control plane 和计算角色。
- 创建 bootstrap 机器。
- 创建 control plane 机器。

流程

1. 进入包含安装程序的目录并运行以下命令：

```
$ ./openshift-install wait-for bootstrap-complete --dir <installation_directory> \ 1
    --log-level info 2
```

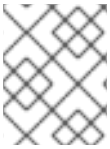
1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不是 **info**。

如果命令退出时没有 **FATAL** 警告，则您的生产环境 control plane 已被初始化。

2. 删除 bootstrap 资源：

```
$ az network nsg rule delete -g ${RESOURCE_GROUP} --nsg-name ${INFRA_ID}-nsg --
name bootstrap_ssh_in
$ az vm stop -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap
$ az vm deallocate -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap
$ az vm delete -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap --yes
$ az disk delete -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap_OSDisk --no-
wait --yes
$ az network nic delete -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap-nic --no-
wait
$ az storage blob delete --account-key ${ACCOUNT_KEY} --account-name
${CLUSTER_NAME}sa --container-name files --name bootstrap.ign
$ az network public-ip delete -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap-
ssh-pip
```



注意

如果没有删除 bootstrap 服务器，因为 API 流量会路由到 bootstrap 服务器，所以安装可能无法成功。

7.10.16. 在 Azure 中创建额外的 worker 机器

您可以通过分散启动各个实例或利用集群外自动化流程（如自动缩放组），在 Microsoft Azure 中为您的集群创建 worker 机器。您还可以利用 OpenShift Container Platform 中的内置集群扩展机制和机器 API。

在本例中，您要使用 Azure Resource Manager(ARM)模板手动启动一个实例。通过在 文件中包括类型为 **06_workers.json** 的其他资源，即可启动其他实例。

注意

默认情况下，Microsoft Azure 将 control plane 机器和计算机器放在预先设置的可用区中。您可以为计算节点或 control plane 节点手动设置可用区。要做到这一点，通过在虚拟机资源的 **zones** 参数中指定每个可用区来修改供应商的 ARM 模板。

如果不使用提供的 ARM 模板来创建 control plane 机器，您必须检查提供的信息并手动创建基础架构。如果您的集群没有正确初始化，请考虑与安装日志联系红帽支持。

先决条件

- 配置 Azure 帐户。
- 为集群生成 Ignition 配置文件。
- 在 Azure 中创建和配置 VNet 及相关子网。
- 在 Azure 中创建和配置联网及负载均衡器。
- 创建 control plane 和计算角色。
- 创建 bootstrap 机器。
- 创建 control plane 机器。

流程

1. 复制 **worker 机器的 ARM 模板一节中的模板**，并将它以 **06_workers.json** 保存到集群的安装目录中。此模板描述了集群所需的 worker 机器。

2. 导出 worker 机器部署所需的以下变量：

```
$ export WORKER_IGNITION=`cat <installation_directory>/worker.ign | base64 | tr -d '\n'`
```

3. 使用 **az** CLI 创建部署：

```
$ az deployment group create -g ${RESOURCE_GROUP} \
  --template-file "<installation_directory>/06_workers.json" \
  --parameters workerIgnition="${WORKER_IGNITION}" \ 1
  --parameters baseName="${INFRA_ID}" \ 2
  --parameters nodeVMSize="Standard_D4s_v3" 3
```

1. worker 节点的 Ignition 内容。
2. 资源名称使用的基本名称；这通常是集群的基础架构 ID。
3. 可选：指定计算节点虚拟机的大小。使用与指定架构兼容的虚拟机大小。如果未定义这个值，则会设置模板中的默认值。

7.10.16.1. worker 机器的 ARM 模板

您可以使用以下 Azure Resource Manager(ARM)模板来部署 OpenShift Container Platform 集群所需的 worker 机器：

例 7.63. 06_workers.json ARM template

```
{
  "$schema" : "https://schema.management.azure.com/schemas/2015-01-
01/deploymentTemplate.json#",
  "contentVersion" : "1.0.0.0",
  "parameters" : {
    "baseName" : {
      "type" : "string",
      "minLength" : 1,
      "metadata" : {
        "description" : "Base name to be used in resource names (usually the cluster's Infra ID)"
      }
    },
    "vnetBaseName": {
      "type": "string",
      "defaultValue": "",
      "metadata" : {
        "description" : "The specific customer vnet's base name (optional)"
      }
    },
    "workerIgnition" : {
      "type" : "string",
      "metadata" : {
        "description" : "Ignition content for the worker nodes"
```

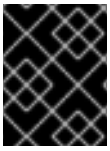
```
      }
    },
    "numberOfNodes" : {
      "type" : "int",
      "defaultValue" : 3,
      "minValue" : 2,
      "maxValue" : 30,
      "metadata" : {
        "description" : "Number of OpenShift compute nodes to deploy"
      }
    },
    "sshKeyData" : {
      "type" : "securestring",
      "defaultValue" : "Unused",
      "metadata" : {
        "description" : "Unused"
      }
    },
    "nodeVMSize" : {
      "type" : "string",
      "defaultValue" : "Standard_D4s_v3",
      "metadata" : {
        "description" : "The size of the each Node Virtual Machine"
      }
    },
    "hyperVGen": {
      "type": "string",
      "metadata": {
        "description": "VM generation image to use"
      },
      "defaultValue": "V2",
      "allowedValues": [
        "V1",
        "V2"
      ]
    }
  },
  "variables" : {
    "location" : "[resourceGroup().location]",
    "virtualNetworkName" : "[concat(if(not(empty(parameters('vnetBaseName'))),
parameters('vnetBaseName'), parameters('baseName')), '-vnet')]",
    "virtualNetworkID" : "[resourceId('Microsoft.Network/virtualNetworks',
variables('virtualNetworkName'))]",
    "nodeSubnetName" : "[concat(if(not(empty(parameters('vnetBaseName'))),
parameters('vnetBaseName'), parameters('baseName')), '-worker-subnet')]",
    "nodeSubnetRef" : "[concat(variables('virtualNetworkID'), '/subnets/',
variables('nodeSubnetName'))]",
    "infraLoadBalancerName" : "[parameters('baseName')]",
    "sshKeyPath" : "/home/capi/.ssh/authorized_keys",
    "identityName" : "[concat(parameters('baseName'), '-identity')]",
    "galleryName": "[concat('gallery_', replace(parameters('baseName'), '-', '_'))]",
    "imageName" : "[concat(parameters('baseName'), if(equals(parameters('hyperVGen'), 'V2'), '-
gen2', ''))]",
    "copy" : [
      {
        "name" : "vmNames",
```

```
        "count" :  "[parameters('numberOfNodes')]",
        "input" : "[concat(parameters('baseName'), '-worker-', variables('location'), '-',
copyIndex('vmNames', 1))]"
      }
    ]
  },
  "resources" : [
    {
      "apiVersion" : "2019-05-01",
      "name" : "[concat('node', copyIndex())]",
      "type" : "Microsoft.Resources/deployments",
      "copy" : {
        "name" : "nodeCopy",
        "count" : "[length(variables('vmNames'))]"
      },
      "properties" : {
        "mode" : "Incremental",
        "template" : {
          "$schema" : "http://schema.management.azure.com/schemas/2015-01-
01/deploymentTemplate.json#",
          "contentVersion" : "1.0.0.0",
          "resources" : [
            {
              "apiVersion" : "2018-06-01",
              "type" : "Microsoft.Network/networkInterfaces",
              "name" : "[concat(variables('vmNames')[copyIndex()], '-nic')]",
              "location" : "[variables('location')]",
              "properties" : {
                "ipConfigurations" : [
                  {
                    "name" : "pipConfig",
                    "properties" : {
                      "privateIPAllocationMethod" : "Dynamic",
                      "subnet" : {
                        "id" : "[variables('nodeSubnetRef')]"
                      }
                    }
                  }
                ]
              }
            },
            {
              "apiVersion" : "2018-06-01",
              "type" : "Microsoft.Compute/virtualMachines",
              "name" : "[variables('vmNames')[copyIndex()]]",
              "location" : "[variables('location')]",
              "tags" : {
                "kubernetes.io-cluster-ffranzupi": "owned"
              },
              "identity" : {
                "type" : "userAssigned",
                "userAssignedIdentities" : {
                  "[resourceID('Microsoft.ManagedIdentity/userAssignedIdentities/',
variables('identityName'))]" : {}
                }
              },
```

```
            "dependsOn" : [
              "[concat('Microsoft.Network/networkInterfaces/', concat(variables('vmNames')
[copyIndex()], '-nic'))]"
            ],
            "properties" : {
              "hardwareProfile" : {
                "vmSize" : "[parameters('nodeVMSize')]"
              },
              "osProfile" : {
                "computerName" : "[variables('vmNames')[copyIndex()]]",
                "adminUsername" : "capi",
                "adminPassword" : "NotActuallyApplied!",
                "customData" : "[parameters('workerIgnition')]",
                "linuxConfiguration" : {
                  "disablePasswordAuthentication" : false
                }
              },
              "storageProfile" : {
                "imageReference": {
                  "id": "[resourceId('Microsoft.Compute/galleries/images', variables('galleryName'),
variables('imageName'))]"
                },
                "osDisk" : {
                  "name": "[concat(variables('vmNames')[copyIndex()],'_OSDisk')]",
                  "osType" : "Linux",
                  "createOption" : "FromImage",
                  "managedDisk": {
                    "storageAccountType": "Premium_LRS"
                  },
                  "diskSizeGB": 128
                }
              },
              "networkProfile" : {
                "networkInterfaces" : [
                  {
                    "id" : "[resourceId('Microsoft.Network/networkInterfaces',
concat(variables('vmNames')[copyIndex()], '-nic'))]",
                    "properties": {
                      "primary": true
                    }
                  }
                ]
              }
            }
          }
        ]
      }
    }
  ]
}
```

7.10.17. 安装 OpenShift CLI

您可以安装 OpenShift CLI(**oc**)来使用命令行界面与 OpenShift Container Platform 进行交互。您可以在 Linux、Windows 或 macOS 上安装 **oc**。



重要

如果安装了旧版本的 **oc**，则可能无法使用 OpenShift Container Platform 4.16 中的所有命令。下载并安装新版本的 **oc**。

在 Linux 上安装 OpenShift CLI
您可以按照以下流程在 Linux 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **产品变体** 下拉列表中选择架构。
3. 从 **版本** 下拉列表中选择适当的版本。
4. 点 **OpenShift v4.16 Linux Client** 条目旁的 **Download Now** 来保存文件。
5. 解包存档：

```
$ tar xvf <file>
```

6. 将 **oc** 二进制文件放到 **PATH 中的目录中**。
要查看您的 **PATH**，请执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

在 Windows 上安装 OpenShift CLI
您可以按照以下流程在 Windows 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 Windows Client** 条目旁的 **Download Now** 来保存文件。
4. 使用 ZIP 程序解压存档。
5. 将 **oc** 二进制文件移到 **PATH 中的目录中**。
要查看您的 **PATH**，请打开命令提示并执行以下命令：

```
C:\> path
```

验证

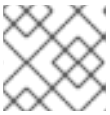
- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

在 macOS 上安装 OpenShift CLI
您可以按照以下流程在 macOS 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 macOS Client** 条目旁的 **Download Now** 来保存文件。



注意

对于 macOS arm64，请选择 **OpenShift v4.16 macOS arm64 Client** 条目。

4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
要查看您的 **PATH**，请打开终端并执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

7.10.18. 使用 CLI 登录集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

7.10.19. 批准机器的证书签名请求

当您将机器添加到集群时，会为您添加的每台机器生成两个待处理证书签名请求(CSR)。您必须确认这些 CSR 已获得批准，或根据需要自行批准。必须首先批准客户端请求，然后批准服务器请求。

先决条件

- 您已将机器添加到集群中。

流程

1. 确认集群可以识别这些机器：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0 Ready    master  63m  v1.29.4
master-1 Ready    master  63m  v1.29.4
master-2 Ready    master  64m  v1.29.4
```

输出中列出了您创建的所有机器。

注意

在有些 CSR 被批准前，前面的输出可能不包括计算节点（也称为 worker 节点）。

2. 检查待处理的 CSR，并确保添加到集群中的每台机器都有 **Pending** 或 **Approved** 状态的客户端请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE    REQUESTOR                                                   CONDITION
csr-8b2br   15m    system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper   Pending
csr-8vnps   15m    system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper   Pending
...
```

在本例中，两台机器加入集群。您可能会在列表中看到更多已批准的 CSR。

3. 如果 CSR 没有获得批准，在您添加的机器的所有待处理 CSR 都处于 **Pending 状态** 后，请批准集群机器的 CSR：

注意

由于 CSR 会自动轮转，因此请在将机器添加到集群后一小时内批准您的 CSR。如果没有在一小时内批准它们，证书将会轮转，每个节点会存在多个证书。您必须批准所有这些证书。批准客户端 CSR 后，Kubelet 为服务证书创建一个二级 CSR，这需要手动批准。然后，如果 Kubelet 请求具有相同参数的新证书，则后续提供证书续订请求由 **machine-approver** 自动批准。

注意

对于在未启用机器 API 的平台上运行的集群，如裸机和其他用户置备的基础架构，您必须实施一种方法来自动批准 kubelet 提供证书请求(CSR)。如果没有批准请求，则 **oc exec**、oc**rsh** 和 **oc logs** 命令将无法成功，因为 API 服务器连接到 kubelet 时需要服务证书。与 Kubelet 端点联系的任何操作都需要此证书批准。该方法必须监视新的 CSR，确认 CSR 由 system: **node** 或 **system:admin** 组中的 **node-bootstrapper** 服务帐户提交，并确认节点的身份。

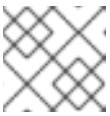
- 要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```



注意

在有些 CSR 被批准前，一些 Operator 可能无法使用。

4. 现在，您的客户端请求已被批准，您必须查看添加到集群中的每台机器的服务器请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE     REQUESTOR                                                   CONDITION
csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

5. 如果剩余的 CSR 没有被批准，且处于 **Pending** 状态，请批准集群机器的 CSR：

   - 要单独批准，请对每个有效的 CSR 运行以下命令：

     ```
     $ oc adm certificate approve <csr_name> 1
     ```

     1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

   - 要批准所有待处理的 CSR，请运行以下命令：

     ```
     $ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
     {{end}}{{end}}' | xargs oc adm certificate approve
     ```

6. 批准所有客户端和服务器 CSR 后，机器将 处于 **Ready 状态**。运行以下命令验证：

   ```
   $ oc get nodes
   ```

   输出示例

   ```
   NAME      STATUS    ROLES   AGE  VERSION
   master-0  Ready     master  73m  v1.29.4
   master-1  Ready     master  73m  v1.29.4
   master-2  Ready     master  74m  v1.29.4
   worker-0  Ready     worker  11m  v1.29.4
   worker-1  Ready     worker  11m  v1.29.4
   ```

   注意

   批准服务器 CSR 后可能需要几分钟时间让机器过渡到 **Ready 状态**。

其他信息

- 如需有关 CSR 的更多信息，请参阅 证书签名请求。

7.10.20. 添加 Ingress DNS 记录

如果在创建 Kubernetes 清单并生成 Ignition 配置时删除了 DNS 区配置，您必须手动创建指向 Ingress 负载均衡器的 DNS 记录。您可以创建一个通配符 ***.apps.{baseDomain}.** 或特定的记录。您可以根据要求使用 A、CNAME 和其他记录。

先决条件

- 已使用您置备的基础架构在 Microsoft Azure 上安装了 OpenShift Container Platform 集群。
- 安装 OpenShift CLI（**oc**）。
- 安装或更新 Azure CLI。

流程

1. 确认 Ingress 路由器已创建了负载均衡器并填充 **EXTERNAL-IP** 字段：

```
$ oc -n openshift-ingress get service router-default
```

输出示例

```
NAME            TYPE           CLUSTER-IP     EXTERNAL-IP     PORT(S)                      AGE
router-default  LoadBalancer  172.30.20.10  35.130.120.110
80:32288/TCP,443:31215/TCP  20
```

2. 将 Ingress 路由器 IP 导出作为变量：

```
$ export PUBLIC_IP_ROUTER=`oc -n openshift-ingress get service router-default --no-
headers | awk '{print $4}'`
```

3. 在公共 DNS 区域中添加 ***.apps** 记录。

   a. 如果您要将此集群添加到新的公共区，请运行：

   ```
   $ az network dns record-set a add-record -g ${BASE_DOMAIN_RESOURCE_GROUP} -
   z ${CLUSTER_NAME}.${BASE_DOMAIN} -n *.apps -a ${PUBLIC_IP_ROUTER} --ttl 300
   ```

   b. 如果您要将此集群添加到已经存在的公共区中，请运行：

   ```
   $ az network dns record-set a add-record -g ${BASE_DOMAIN_RESOURCE_GROUP} -
   z ${BASE_DOMAIN} -n *.apps.${CLUSTER_NAME} -a ${PUBLIC_IP_ROUTER} --ttl 300
   ```

4. 在私有 DNS 区域中添加 ***.apps** 记录：

   a. 使用以下命令创建 ***.apps** 记录：

   ```
   $ az network private-dns record-set a create -g ${RESOURCE_GROUP} -z
   ${CLUSTER_NAME}.${BASE_DOMAIN} -n *.apps --ttl 300
   ```

   b. 使用以下命令在专用 DNS 区域中添加 ***.apps** 记录：

   ```
   $ az network private-dns record-set a add-record -g ${RESOURCE_GROUP} -z
   ${CLUSTER_NAME}.${BASE_DOMAIN} -n *.apps -a ${PUBLIC_IP_ROUTER}
   ```

如果需要添加特定域而不使用通配符，可以为集群的每个当前路由创建条目：

```
$ oc get --all-namespaces -o jsonpath='{range .items[*]}{range .status.ingress[*]}{.host}{"\n"}{end}
{end}' routes
```

输出示例

```
oauth-openshift.apps.cluster.basedomain.com
console-openshift-console.apps.cluster.basedomain.com
downloads-openshift-console.apps.cluster.basedomain.com
alertmanager-main-openshift-monitoring.apps.cluster.basedomain.com
prometheus-k8s-openshift-monitoring.apps.cluster.basedomain.com
```

7.10.21. 在用户置备的基础架构上完成 Azure 安装

在 Microsoft Azure 用户置备的基础架构上启动 OpenShift Container Platform 安装后，您可以监控集群事件，直到集群就绪可用。

先决条件

- 在用户置备的 Azure 基础架构上为 OpenShift Container Platform 集群部署 bootstrap 机器。
- 安装 **oc** CLI 并登录。

流程

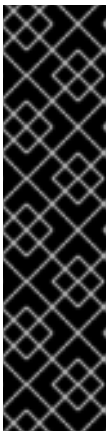
- 完成集群安装：

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

输出示例

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。



重要

- 安装程序生成的 Ignition 配置文件包含 24 小时后过期的证书，然后在该时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

7.10.22. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

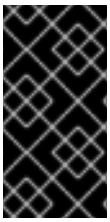
其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于 远程健康监控

7.11. 使用 ARM 模板在 AZURE 上安装集群

在 OpenShift Container Platform 版本 4.16 中，您可以使用您提供的基础架构在 Microsoft Azure 上安装集群。

提供的几个 Azure Resource Manager（ARM） 模板可协助完成这些步骤，也可帮助您自行建模。



重要

进行用户置备的基础架构安装的步骤仅作为示例。使用您提供的基础架构安装集群需要了解云供应商和 OpenShift Container Platform 安装过程。提供的几个 ARM 模板可帮助完成这些步骤，或帮助您自行建模。您也可以自由选择通过其他方法创建所需的资源；模板仅作示例之用。

7.11.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读选择集群安装方法并为用户准备它的文档。
- 已将 Azure 帐户配置为 托管集群。
- 您下载了 Azure CLI 并安装到您的计算机上。请参阅 Azure 文档中的安装 Azure CLI。以下文档最近使用 Azure CLI 的版本 **2.38.0** 测试。Azure CLI 命令可能会根据您使用的版本的不同而不同。
- 如果环境中无法访问云身份和访问管理 (IAM) API，或者不想将管理员级别的凭证 secret 存储在 **kube-system** 命名空间中，请参阅在 kube-system 项目中存储管理员级别的 secret 以了解其他选项。
- 如果您使用防火墙并计划使用 Telemetry 服务，则将防火墙配置为允许集群需要访问的站点。

注意

如果您要配置代理，请务必也要查看此站点列表。

7.11.2. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类型的基础架构上执行受限网络安装。在此过程中，您可以下载所需的内容，并使用它为镜像 registry 填充安装软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

7.11.3. 配置 Azure 项目

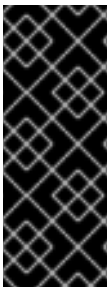
在安装 OpenShift Container Platform 之前，您必须配置 Azure 项目来托管它。

重要

所有通过公共端点提供的 Azure 资源均存在资源名称的限制，您无法创建使用某些名称的资源。如需 Azure 限制词语列表，请参阅 Azure 文档中的解决保留资源名称错误。

7.11.3.1. Azure 帐户限值

OpenShift Container Platform 集群使用诸多 Microsoft Azure 组件，默认的 Azure 订阅和服务限值、配额和约束会影响您安装 OpenShift Container Platform 集群的能力。



重要

默认的限制因服务类别的不同（如 Free Trial 或 Pay-As-You-Go）以及系列的不同（如 Dv2、F 或 G）而有所不同。例如，对于 Enterprise Agreement 订阅的默认限制是 350 个内核。

在 Azure 上安装默认集群前，请检查您的订阅类型的限制，如有必要，请提高帐户的配额限制。

下表总结了 Azure 组件，它们的限值会影响您安装和运行 OpenShift Container Platform 集群的能力。

组件	默认所需的组件数	默认 Azure 限值	描述
vCPU	40	每个区域 20 个	默认集群需要 40 个 vCPU，因此您必须提高帐户限值。 默认情况下，每个集群创建以下实例： • 一台 Bootstrap 机器，在安装后删除 • 三个 control plane 机器 • 三个计算（compute）机器 由于 Bootstrap 机器使用 **Standard_D4s_v3** 机器（使用 4 个 vCPU），control plane 机器使用 **Standard_D8s_v3** 虚拟机（8 个 vCPU），并且 worker 机器使用 **Standard_D4s_v3** 虚拟机（4 个 vCPU），因此默认集群需要 40 个 vCPU。bootstrap 节点 VM（使用 4 个 vCPU）只在安装过程中使用。 若要部署更多 worker 节点、启用自动扩展、部署大型工作负载或使用不同的实例类型，您必须进一步提高帐户的 vCPU 限值，以确保集群可以部署您需要的机器。
OS Disk	7		每个集群机器必须至少有 100 GB 存储和 300 IOPS。虽然这些值是最低支持的值，但对于具有密集型工作负载的生产环境集群和集群，建议使用更快的存储。有关优化性能存储的更多信息，请参阅"扩展和性能"部分中的页面标题为"优化存储"。

组件	默认所需的组件数	默认 Azure 限值	描述
VNet	1	每个区域 1000 个	每个默认集群都需要一个虚拟网络 (VNet)，此网络包括两个子网。
网络接口	7	每个区域 65,536 个	每个默认集群都需要 7 个网络接口。如果您要创建更多机器或者您部署的工作负载要创建负载均衡器，则集群会使用更多的网络接口。
网络安全组	2	5000	每个集群为 VNet 中的每个子网创建网络安全组。默认集群为 control plane 和计算节点子网创建网络安全组： **controlplane**：允许从任何位置通过端口 6443 访问 control plane 机器 **node**：允许从互联网通过端口 80 和 443 访问 worker 节点
网络负载均衡器	3	每个区域 1000 个	每个集群都会创建以下负载均衡器： **default**：用于在 worker 机器之间对端口 80 和 443 的请求进行负载均衡的公共 IP 地址 **internal**：用于在 control plane 机器之间对端口 6443 和 22623 的请求进行负载均衡的专用 IP 地址 **external**：用于在 control plane 机器之间对端口 6443 的请求进行负载均衡的公共 IP 地址 如果您的应用程序创建了更多的 Kubernetes **LoadBalancer** 服务对象，您的集群会使用更多的负载均衡器。
公共 IP 地址	3		两个公共负载均衡器各自使用一个公共 IP 地址。bootstrap 机器也使用一个公共 IP 地址，以便您可以在安装期间通过 SSH 连接到该机器来进行故障排除。bootstrap 节点的 IP 地址仅在安装过程中使用。
专用 IP 地址	7		内部负载均衡器、三台 control plane 机器中的每一台以及三台 worker 机器中的每一台各自使用一个专用 IP 地址。

组件	默认所需的组件数	默认 Azure 限值	描述
Spot VM vCPU（可选）	0 如果配置 spot 虚拟机，您的集群必须为每个计算节点有两个 spot VM vCPU。	每个区域 20 个	这是可选组件。要使用 spot 虚拟机，您必须将 Azure 默认限值增加到集群中至少有两倍的计算节点数量。  **注意** 不建议将 spot 虚拟机用于 control plane 节点。

其他资源

- 优化存储

7.11.3.2. 在 Azure 中配置公共 DNS 区

要安装 OpenShift Container Platform，您使用的 Microsoft Azure 帐户必须在帐户中具有一个专用的公共托管 DNS 区。此区域必须对域具有权威。此服务为集群外部连接提供集群 DNS 解析和名称查询。

流程

1. 标识您的域或子域，以及注册商（registrar）。您可以转移现有的域和注册商，或通过 Azure 或其他来源获取新的域和注册商。

注意

如需通过 Azure 购买域的更多信息，请参阅 Azure 文档中的购买 Azure 应用服务的自定义域名。

2. 如果您使用现有的域和注册商，请将其 DNS 迁移到 Azure。请参阅 Azure 文档中的将活动 DNS 名称迁移到 Azure 应用服务。

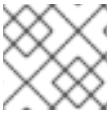
3. 为您的域配置 DNS。按照 Azure 文档中教程：在 Azure DNS 中托管域部分里的步骤，为您的域或子域创建一个公共托管区，提取新的权威名称服务器，并更新您的域使用的名称服务器的注册商记录。
   使用合适的根域（如 **openshiftcorp.com**）或子域（如 **clusters.openshiftcorp.com**）。

4. 如果您使用子域，请按照您公司的流程将其委派记录添加到父域。

您可以通过访问此 示例来创建 DNS 区域来查看 Azure 的 DNS 解决方案 。

7.11.3.3. 提高 Azure 帐户限值

要提高帐户限值，请在 Azure 门户上提交支持请求。



注意

每一支持请求只能提高一种类型的配额。

流程

1. 从 Azure 门户，点击左下角的 **Help + suport**。
2. 点击 **New support request**，然后选择所需的值：
   a. 从 **Issue type** 列表中，选择 **Service and subscription limits (quotas)**。
   b. 从 **Subscription** 列表中，选择要修改的订阅。
   c. 从 **Quota type** 列表中，选择要提高的配额。例如，选择 **Compute-VM (cores-vCPUs) subscription limit increases** 以增加 vCPU 的数量，这是安装集群所必须的。
   d. 点击 **Next: Solutions**。
3. 在 **Problem Details** 页面中，提供您要提高配额所需的信息：
   a. 点击 **Provide details**，然后在 **Quota details** 窗口中提供所需的详情。
   b. 在 SUPPORT METHOD 和 CONTACT INFO 部分中，提供问题严重性和您的联系详情。
4. 点击 **Next: Review + create**，然后点击 **Create**。

7.11.3.4. 证书签名请求管理

在使用您置备的基础架构时，集群只能有限地访问自动机器管理，因此您必须提供一种在安装后批准集群证书签名请求 (CSR) 的机制。**kube-controller-manager** 只能批准 kubelet 客户端 CSR。**machine-approver** 无法保证使用 kubelet 凭证请求的提供证书的有效性，因为它不能确认是正确的机器发出了该请求。您必须决定并实施一种方法，以验证 kubelet 提供证书请求的有效性并进行批准。

7.11.3.5. 记录下订阅和租户 ID

安装程序需要与 Azure 帐户关联的订阅和租户 ID。您可以使用 Azure CLI 收集此信息。

先决条件

- 已安装或更新 Azure CLI。

流程

1. 运行以下命令登录到 Azure CLI：

```
$ az login
```

2. 确保您使用正确的订阅：
   a. 运行以下命令，查看可用订阅列表：

```
$ az account list --refresh
```

输出示例

```
[
  {
    "cloudName": "AzureCloud",
    "id": "8xxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
    "isDefault": true,
```

```
    "name": "Subscription Name 1",
    "state": "Enabled",
    "tenantId": "6xxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
    "user": {
      "name": "you@example.com",
      "type": "user"
    }
  },
  {
    "cloudName": "AzureCloud",
    "id": "9xxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
    "isDefault": false,
    "name": "Subscription Name 2",
    "state": "Enabled",
    "tenantId": "7xxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
    "user": {
      "name": "you2@example.com",
      "type": "user"
    }
  }
]
```

b. 运行以下命令，查看活跃帐户的详情，并确认这是您要使用的订阅：

```
$ az account show
```

输出示例

```
{
  "environmentName": "AzureCloud",
  "id": "8xxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
  "isDefault": true,
  "name": "Subscription Name 1",
  "state": "Enabled",
  "tenantId": "6xxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
  "user": {
    "name": "you@example.com",
    "type": "user"
  }
}
```

3. 如果您没有使用正确的订阅：

a. 运行以下命令来更改活跃订阅：

```
$ az account set -s <subscription_id>
```

b. 运行以下命令验证您是否正在使用所需的订阅：

```
$ az account show
```

输出示例

```
{
```

```
  "environmentName": "AzureCloud",
  "id": "9xxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
  "isDefault": true,
  "name": "Subscription Name 2",
  "state": "Enabled",
  "tenantId": "7xxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
  "user": {
    "name": "you2@example.com",
    "type": "user"
  }
}
```


4. 记录输出中的 **id** 和 **tenantId** 参数值。您需要这些值来安装 OpenShift Container Platform 集群。

7.11.3.6. 支持访问 Azure 资源的身份

OpenShift Container Platform 集群需要一个 Azure 身份来创建和管理 Azure 资源。因此，您需要以下身份之一来完成安装：

- 服务主体
- 系统分配的管理身份
- 用户分配的受管身份

7.11.3.7. 用户置备的基础架构所需的 Azure 权限

安装程序需要访问具有所需权限的 Azure 服务主体或受管身份，以部署集群并维护其每日操作。这些权限必须授予与身份关联的 Azure 订阅。

以下选项可供您使用：

- 您可以为身份分配 **Contributor** 和 **User Access Administrator** 角色。分配这些角色是授予所有所需权限的最快速方法。
  有关分配角色的更多信息，请参阅 Azure 文档，使用 Azure 门户管理 Azure 资源的访问。
- 如果机构的安全策略需要更严格的权限集，您可以创建具有所需权限的自定义角色。

在 Microsoft Azure 上创建 OpenShift Container Platform 集群需要以下权限。

例 7.64. 创建授权资源所需的权限

- **Microsoft.Authorization/policies/audit/action**
- **Microsoft.Authorization/policies/auditIfNotExists/action**
- **Microsoft.Authorization/roleAssignments/read**
- **Microsoft.Authorization/roleAssignments/write**

例 7.65. 创建计算资源所需的权限

- **Microsoft.Compute/images/read**

- **Microsoft.Compute/images/write**
- **Microsoft.Compute/images/delete**
- **Microsoft.Compute/availabilitySets/read**
- **Microsoft.Compute/disks/beginGetAccess/action**
- **Microsoft.Compute/disks/delete**
- **Microsoft.Compute/disks/read**
- **Microsoft.Compute/disks/write**
- **Microsoft.Compute/galleries/images/read**
- **Microsoft.Compute/galleries/images/versions/read**
- **Microsoft.Compute/galleries/images/versions/write**
- **Microsoft.Compute/galleries/images/write**
- **Microsoft.Compute/galleries/read**
- **Microsoft.Compute/galleries/write**
- **Microsoft.Compute/snapshots/read**
- **Microsoft.Compute/snapshots/write**
- **Microsoft.Compute/snapshots/delete**
- **Microsoft.Compute/virtualMachines/delete**
- **Microsoft.Compute/virtualMachines/powerOff/action**
- **Microsoft.Compute/virtualMachines/read**
- **Microsoft.Compute/virtualMachines/write**
- **Microsoft.Compute/virtualMachines/deallocate/action**

例 7.66. 创建身份管理资源所需的权限

- **Microsoft.ManagedIdentity/userAssignedIdentities/assign/action**
- **Microsoft.ManagedIdentity/userAssignedIdentities/read**
- **Microsoft.ManagedIdentity/userAssignedIdentities/write**

例 7.67. 创建网络资源所需的权限

- **Microsoft.Network/dnsZones/A/write**
- **Microsoft.Network/dnsZones/CNAME/write**

- **Microsoft.Network/dnszones/CNAME/read**
- **Microsoft.Network/dnszones/read**
- **Microsoft.Network/loadBalancers/backendAddressPools/join/action**
- **Microsoft.Network/loadBalancers/backendAddressPools/read**
- **Microsoft.Network/loadBalancers/backendAddressPools/write**
- **Microsoft.Network/loadBalancers/read**
- **Microsoft.Network/loadBalancers/write**
- **Microsoft.Network/networkInterfaces/delete**
- **Microsoft.Network/networkInterfaces/join/action**
- **Microsoft.Network/networkInterfaces/read**
- **Microsoft.Network/networkInterfaces/write**
- **Microsoft.Network/networkSecurityGroups/join/action**
- **Microsoft.Network/networkSecurityGroups/read**
- **Microsoft.Network/networkSecurityGroups/securityRules/delete**
- **Microsoft.Network/networkSecurityGroups/securityRules/read**
- **Microsoft.Network/networkSecurityGroups/securityRules/write**
- **Microsoft.Network/networkSecurityGroups/write**
- **Microsoft.Network/privateDnsZones/A/read**
- **Microsoft.Network/privateDnsZones/A/write**
- **Microsoft.Network/privateDnsZones/A/delete**
- **Microsoft.Network/privateDnsZones/SOA/read**
- **Microsoft.Network/privateDnsZones/read**
- **Microsoft.Network/privateDnsZones/virtualNetworkLinks/read**
- **Microsoft.Network/privateDnsZones/virtualNetworkLinks/write**
- **Microsoft.Network/privateDnsZones/write**
- **Microsoft.Network/publicIPAddresses/delete**
- **Microsoft.Network/publicIPAddresses/join/action**
- **Microsoft.Network/publicIPAddresses/read**
- **Microsoft.Network/publicIPAddresses/write**

- **Microsoft.Network/virtualNetworks/join/action**
- **Microsoft.Network/virtualNetworks/read**
- **Microsoft.Network/virtualNetworks/subnets/join/action**
- **Microsoft.Network/virtualNetworks/subnets/read**
- **Microsoft.Network/virtualNetworks/subnets/write**
- **Microsoft.Network/virtualNetworks/write**

例 7.68. 检查资源健康状况所需的权限

- **Microsoft.Resourcehealth/healthevent/Activated/action**
- **Microsoft.Resourcehealth/healthevent/InProgress/action**
- **Microsoft.Resourcehealth/healthevent/Pending/action**
- **Microsoft.Resourcehealth/healthevent/Resolved/action**
- **Microsoft.Resourcehealth/healthevent/Updated/action**

例 7.69. 创建资源组所需的权限

- **Microsoft.Resources/subscriptions/resourceGroups/read**
- **Microsoft.Resources/subscriptions/resourcegroups/write**

例 7.70. 创建资源标签所需的权限

- **Microsoft.Resources/tags/write**

例 7.71. 创建存储资源所需的权限

- **Microsoft.Storage/storageAccounts/blobServices/read**
- **Microsoft.Storage/storageAccounts/blobServices/containers/write**
- **Microsoft.Storage/storageAccounts/fileServices/read**
- **Microsoft.Storage/storageAccounts/fileServices/shares/read**
- **Microsoft.Storage/storageAccounts/fileServices/shares/write**
- **Microsoft.Storage/storageAccounts/fileServices/shares/delete**
- **Microsoft.Storage/storageAccounts/listKeys/action**
- **Microsoft.Storage/storageAccounts/read**

- **Microsoft.Storage/storageAccounts/write**

例 7.72. 创建部署所需的权限

- **Microsoft.Resources/deployments/read**
- **Microsoft.Resources/deployments/write**
- **Microsoft.Resources/deployments/validate/action**
- **Microsoft.Resources/deployments/operationstatuses/read**

例 7.73. 创建计算资源的可选权限

- **Microsoft.Compute/availabilitySets/delete**
- **Microsoft.Compute/availabilitySets/write**

例 7.74. 创建 marketplace 虚拟机资源的可选权限

- **Microsoft.MarketplaceOrdering/offertypes/publishers/offers/plans/agreements/read**
- **Microsoft.MarketplaceOrdering/offertypes/publishers/offers/plans/agreements/write**

例 7.75. 启用用户管理加密的可选权限

- **Microsoft.Compute/diskEncryptionSets/read**
- **Microsoft.Compute/diskEncryptionSets/write**
- **Microsoft.Compute/diskEncryptionSets/delete**
- **Microsoft.KeyVault/vaults/read**
- **Microsoft.KeyVault/vaults/write**
- **Microsoft.KeyVault/vaults/delete**
- **Microsoft.KeyVault/vaults/deploy/action**
- **Microsoft.KeyVault/vaults/keys/read**
- **Microsoft.KeyVault/vaults/keys/write**
- **Microsoft.Features/providers/features/register/action**

删除 Microsoft Azure 上的 OpenShift Container Platform 集群需要以下权限。

例 7.76. 删除授权资源所需的权限

- **Microsoft.Authorization/roleAssignments/delete**

例 7.77. 删除计算资源所需的权限

- **Microsoft.Compute/disks/delete**
- **Microsoft.Compute/galleries/delete**
- **Microsoft.Compute/galleries/images/delete**
- **Microsoft.Compute/galleries/images/versions/delete**
- **Microsoft.Compute/virtualMachines/delete**
- **Microsoft.Compute/images/delete**

例 7.78. 删除身份管理资源所需的权限

- **Microsoft.ManagedIdentity/userAssignedIdentities/delete**

例 7.79. 删除网络资源所需的权限

- **Microsoft.Network/dnszones/read**
- **Microsoft.Network/dnsZones/A/read**
- **Microsoft.Network/dnsZones/A/delete**
- **Microsoft.Network/dnsZones/CNAME/read**
- **Microsoft.Network/dnsZones/CNAME/delete**
- **Microsoft.Network/loadBalancers/delete**
- **Microsoft.Network/networkInterfaces/delete**
- **Microsoft.Network/networkSecurityGroups/delete**
- **Microsoft.Network/privateDnsZones/read**
- **Microsoft.Network/privateDnsZones/A/read**
- **Microsoft.Network/privateDnsZones/delete**
- **Microsoft.Network/privateDnsZones/virtualNetworkLinks/delete**
- **Microsoft.Network/publicIPAddresses/delete**
- **Microsoft.Network/virtualNetworks/delete**

例 7.80. 检查资源健康状况所需的权限

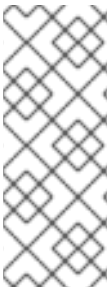
- **Microsoft.Resourcehealth/healthevent/Activated/action**
- **Microsoft.Resourcehealth/healthevent/Resolved/action**
- **Microsoft.Resourcehealth/healthevent/Updated/action**

例 7.81. 删除资源组所需的权限

- **Microsoft.Resources/subscriptions/resourcegroups/delete**

例 7.82. 删除存储资源所需的权限

- **Microsoft.Storage/storageAccounts/delete**
- **Microsoft.Storage/storageAccounts/listKeys/action**



注意

要在 Azure 上安装 OpenShift Container Platform，您必须将资源组创建的权限范围到您的订阅。创建资源组后，您可以将剩余权限的范围限定到所创建的资源组。如果其他资源组中存在公共 DNS 区域，则必须始终将网络 DNS 区域相关权限应用到您的订阅。

在删除 OpenShift Container Platform 集群时，您可以将订阅的所有权限限定到您的订阅。

7.11.3.8. 使用 Azure 管理的身份

安装程序需要一个 Azure 身份来完成安装。您可以使用系统分配或用户分配的受管身份。

如果无法使用受管身份，您可以使用服务主体。

流程

1. 如果您使用系统分配的受管身份，请在您要从其运行安装程序的虚拟机上启用它。
2. 如果您使用用户分配的受管身份：
   a. 将它分配给您要从中运行安装程序的虚拟机。
   b. 记录其客户端 ID。安装集群时需要这个值。
   有关查看用户分配受管身份的详情的更多信息，请参阅 Microsoft Azure 文档来列出用户分配的管理身份。
3. 验证是否为受管身份分配了所需的权限。

7.11.3.9. 创建服务主体

安装程序需要一个 Azure 身份来完成安装。您可以使用服务主体。

如果无法使用服务主体，您可以使用受管身份。

先决条件

- 已安装或更新 Azure CLI。
- 您有一个 Azure 订阅 ID。
- 如果您没有将 **Contributor** 和 **User Administrator Access** 角色分配给服务主体，您已创建了具有所需 Azure 权限的自定义角色。

流程

1. 运行以下命令，为您的帐户创建服务主体：

```
$ az ad sp create-for-rbac --role <role_name> \ 1
    --name <service_principal> \ 2
    --scopes /subscriptions/<subscription_id> 3
```

1. 定义角色名称。您可以使用 **Contributor** 角色，或者指定包含所需权限的自定义角色。
2. 定义服务主体名称。
3. 指定订阅 ID。

输出示例

```
Creating 'Contributor' role assignment under scope '/subscriptions/<subscription_id>'
The output includes credentials that you must protect. Be sure that you do not
include these credentials in your code or check the credentials into your source
control. For more information, see https://aka.ms/azadsp-cli
{
  "appId": "axxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx",
  "displayName": <service_principal>",
  "password": "00000000-0000-0000-0000-000000000000",
  "tenantId": "8xxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx"
}
```

2. 记录输出中的 **appId** 和 **password** 参数的值。安装集群时需要这些值。
3. 如果您将 **Contributor** 角色应用到服务主体，请运行以下命令来分配 **User Administrator Access** 角色：

```
$ az role assignment create --role "User Access Administrator" \
  --assignee-object-id $(az ad sp show --id <appId> --query id -o tsv) 1
  --scope /subscriptions/<subscription_id> 2
```

1. 为您的服务主体指定 **appId** 参数值。
2. 指定订阅 ID。

其他资源

- 如需有关 CCO 模式的更多信息，请参阅 关于 Cloud Credential Operator。

7.11.3.10. 支持的 Azure 区域

安装程序会根据您的订阅动态地生成可用的 Microsoft Azure 区域列表。

支持的 Azure 公共区域

- **australiacentral** (Australia Central)
- **australiaeast** (Australia East)
- **australiasoutheast** (Australia South East)
- **brazilsouth** (Brazil South)
- **canadacentral** (Canada Central)
- **canadaeast** (Canada East)
- **centralindia** (Central India)
- **centralus** (Central US)
- **eastasia** (East Asia)
- **eastus** (East US)
- **eastus2** (East US 2)
- **francecentral** (France Central)
- **germanywestcentral** (Germany West Central)
- **israelcentral** (Israel Central)
- **italynorth** (Italy North)
- **japaneast** (Japan East)
- **japanwest** (Japan West)
- **koreacentral** (Korea Central)
- **koreasouth** (Korea South)
- **mexicocentral** (Mexico Central)
- **northcentralus** (North Central US)
- **northeurope** (North Europe)
- **norwayeast** (Norway East)
- **polandcentral** (Poland Central)
- **qatarcentral** (Qatar Central)
- **southafricanorth** (South Africa North)

- **southcentralus** (South Central US)
- **southeastasia** (Southeast Asia)
- **southindia** (South India)
- **swedencentral** (Sweden Central)
- **switzerlandnorth** (Switzerland North)
- **uaenorth** (UAE North)
- **uksouth** (UK South)
- **ukwest** (UK West)
- **westcentralus** (West Central US)
- **westeurope** (West Europe)
- **westindia** (West India)
- **westus** (West US)
- **westus2** (West US 2)
- **westus3** (West US 3)

支持的 Azure 政府区域
OpenShift Container Platform 4.6 添加了对以下 Microsoft Azure Government（MAG）区域的支持：

- **usgovtexas** (US Gov Texas)
- **usgovvirginia** (US Gov Virginia)

您可以参阅 Azure 文档来了解与所有可用 MAG 区域的信息。其他 MAG 区域应该可以与 OpenShift Container Platform 一起工作，但并没有经过测试。

7.11.4. 具有用户置备基础架构的集群的要求

对于包含用户置备的基础架构的集群，您必须部署所有所需的机器。

本节论述了在用户置备的基础架构上部署 OpenShift Container Platform 的要求。

7.11.4.1. 集群安装所需的机器

最小的 OpenShift Container Platform 集群需要以下主机：

表 7.25. 最低所需的主机

主机	描述
一个临时 bootstrap 机器	集群需要 bootstrap 机器在三台 control plane 机器上部署 OpenShift Container Platform 集群。您可在安装集群后删除 bootstrap 机器。

主机	描述
三台 control plane 机器	control plane 机器运行组成 control plane 的 Kubernetes 和 OpenShift Container Platform 服务。
至少两台计算机器，也称为 worker 机器。	OpenShift Container Platform 用户请求的工作负载在计算机器上运行。

重要

要保持集群的高可用性，请将独立的物理主机用于这些集群机器。

bootstrap 和 control plane 机器必须使用 Red Hat Enterprise Linux CoreOS(RHCOS)作为操作系统。但是，计算机器可以在 Red Hat Enterprise Linux CoreOS(RHCOS)、Red Hat Enterprise Linux(RHEL) 8.6 和更高的版本。

请注意，RHCOS 基于 Red Hat Enterprise Linux(RHEL) 9.2，并继承其所有硬件认证和要求。查看 红帽企业 Linux 技术功能和限制。

7.11.4.2. 集群安装的最低资源要求

每台集群机器都必须满足以下最低要求：

表 7.26. 最低资源要求

机器	操作系统	vCPU [1]	虚拟内存	Storage	每秒输入/输出(IOPS) [2]
bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane（控制平面）	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS、RHEL 8.6 及更新版本 [3]	2	8 GB	100 GB	300

1. 当未启用并发多线程(SMT)或超线程时，一个 vCPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比例：（每个内核数的线程）× sockets = vCPU。
2. OpenShift Container Platform 和 Kubernetes 对磁盘性能非常敏感，建议使用更快的存储速度，特别是 control plane 节点上需要 10 ms p99 fsync 持续时间的 etcd。请注意，在许多云平台上，存储大小和 IOPS 可一起扩展，因此您可能需要过度分配存储卷来获取足够的性能。
3. 与所有用户置备的安装一样，如果您选择在集群中使用 RHEL 计算机器，则负责所有操作系统生命周期管理和维护，包括执行系统更新、应用补丁和完成所有其他必要的任务。RHEL 7 计算机器的使用已弃用，并已在 OpenShift Container Platform 4.10 及更新的版本中删除。

注意

从 OpenShift Container Platform 版本 4.13 开始，RHCOS 基于 RHEL 版本 9.2，它更新了微架构要求。以下列表包含每个架构需要的最小指令集架构 (ISA)：

- x86-64 体系结构需要 x86-64-v2 ISA
- ARM64 架构需要 ARMv8.0-A ISA
- IBM Power 架构需要 Power 9 ISA
- s390x 架构需要 z14 ISA

如需更多信息，请参阅 RHEL 架构。

重要

您需要使用将 **PremiumIO** 参数设置为 **true** 的 Azure 虚拟机。

如果平台的实例类型满足集群机器的最低要求，则 OpenShift Container Platform 支持使用它。

其他资源

- 优化存储

7.11.4.3. 为 Azure 测试的实例类型

以下 Microsoft Azure 实例类型已经 OpenShift Container Platform 测试。

例 7.83. 基于 64 位 x86 架构的机器类型

- **c4.***
- **c5.***
- **c5a.***
- **i3.***
- **m4.***
- **m5.***
- **m5a.***
- **m6a.***
- **m6i.***
- **r4.***
- **r5.***
- **r5a.***
- **r6i.***

- **t3.***
- **t3a.***

7.11.4.4. 在 64 位 ARM 基础架构上为 Azure 测试的实例类型

以下 Microsoft Azure ARM64 实例类型已使用 OpenShift Container Platform 测试。

例 7.84. 基于 64 位 ARM 架构的机器类型

- **c6g.***
- **m6g.***

7.11.5. 使用 Azure Marketplace 产品

使用 Azure Marketplace 产品可让您部署 OpenShift Container Platform 集群，该集群按照使用付费（按小时、每个内核）进行计费，同时仍由红帽直接支持。

要使用 Azure Marketplace 产品部署 OpenShift Container Platform 集群，您必须首先获取 Azure Marketplace 镜像。安装程序使用这个镜像来部署 worker 或 control plane 节点。在获取您的镜像时，请考虑以下事项：

- 虽然镜像相同，但 Azure Marketplace publisher 根据您的区域。如果您位于北美，请将 **redhat** 指定为发布者。如果您位于 EMEA，请将 **redhat-limited** 指定为发布者。
- 此项优惠包括 **rh-ocp-worker** SKU 和 **rh-ocp-worker-gen1** SKU。**rh-ocp-worker** SKU 代表 Hyper-V 生成版本 2 虚拟机镜像。OpenShift Container Platform 中使用的默认实例类型与版本 2 兼容。如果您计划使用与版本 1 兼容的实例类型，请使用与 **rh-ocp-worker-gen1** SKU 关联的镜像。**rh-ocp-worker-gen1** SKU 代表 Hyper-V 版本 1 虚拟机镜像。

重要

在使用 64 位 ARM 实例的集群上不支持使用 Azure marketplace 安装镜像。

先决条件

- 已安装 Azure CLI 客户端 **(az)**。
- 您的 Azure 帐户为产品授权，您使用 Azure CLI 客户端登录到此帐户。

流程

1. 运行以下命令之一，显示所有可用的 OpenShift Container Platform 镜像：
   - 北美：

     ```
     $ az vm image list --all --offer rh-ocp-worker --publisher redhat -o table
     ```

     输出示例

     ```
     Offer          Publisher       Sku                 Urn                                                  Version
     ```

```
------------  -------------  -----------------  ------------------------------------------------------------
----  -----------------
rh-ocp-worker  RedHat         rh-ocp-worker       RedHat:rh-ocp-worker:rh-ocp-
worker:413.92.2023101700          413.92.2023101700
rh-ocp-worker  RedHat         rh-ocp-worker-gen1  RedHat:rh-ocp-worker:rh-ocp-worker-
gen1:413.92.2023101700       413.92.2023101700
```

- 欧洲、中东和非洲地区：

```
$  az vm image list --all --offer rh-ocp-worker --publisher redhat-limited -o table
```

输出示例

```
Offer          Publisher       Sku                 Urn
Version
-------------  -------------  -----------------  ------------------------------------------------------------
----          -----------------
rh-ocp-worker  redhat-limited  rh-ocp-worker       redhat-limited:rh-ocp-worker:rh-ocp-
worker:413.92.2023101700          413.92.2023101700
rh-ocp-worker  redhat-limited  rh-ocp-worker-gen1  redhat-limited:rh-ocp-worker:rh-ocp-
worker-gen1:413.92.2023101700       413.92.2023101700
```

注意

使用可用于 compute 和 control plane 节点的最新镜像。如果需要，您的虚拟机会在安装过程中自动升级。

2. 运行以下命令之一检查您的所提供的镜像：

- 北美：

```
$ az vm image show --urn redhat:rh-ocp-worker:rh-ocp-worker:<version>
```

- 欧洲、中东和非洲地区：

```
$ az vm image show --urn redhat-limited:rh-ocp-worker:rh-ocp-worker:<version>
```

3. 运行以下命令之一查看提供的术语：

- 北美：

```
$ az vm image terms show --urn redhat:rh-ocp-worker:rh-ocp-worker:<version>
```

- 欧洲、中东和非洲地区：

```
$ az vm image terms show --urn redhat-limited:rh-ocp-worker:rh-ocp-worker:<version>
```

4. 运行以下命令之一接受产品条款：

- 北美：

```
$ az vm image terms accept --urn redhat:rh-ocp-worker:rh-ocp-worker:<version>
```

- 欧洲、中东和非洲地区：

```
$ az vm image terms accept --urn redhat-limited:rh-ocp-worker:rh-ocp-worker:<version>
```

5. 记录您的所提供的镜像详情。如果使用 Azure Resource Manager (ARM) 模板来部署计算节点：

   a. 您可以通过删除 **id** 参数，并使用您的值来添加 **offer**, **publisher**, **sku**, and **version** 参数来更新 **storageProfile.imageReference**。

   b. 为虚拟机 (VM) 指定一个计**划**。

   示例 06_workers.json ARM 模板带有一个更新的 storageProfile.imageReference 对象和一个特定的计划

```
...
  "plan" : {
    "name": "rh-ocp-worker",
    "product": "rh-ocp-worker",
    "publisher": "redhat"
  },
  "dependsOn" : [
    "[concat('Microsoft.Network/networkInterfaces/', concat(variables('vmNames')
[copyIndex()], '-nic'))]"
  ],
  "properties" : {
...
  "storageProfile": {
    "imageReference": {
    "offer": "rh-ocp-worker",
    "publisher": "redhat",
    "sku": "rh-ocp-worker",
    "version": "413.92.2023101700"
    }
    ...
  }
...
  }
```

7.11.6. 获取安装程序

在安装 OpenShift Container Platform 前，将安装文件下载到您用于安装的主机上。

先决条件

- 您有一台运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB。

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐户，请使用您的凭证登录。如果没有，请创建一个帐户。

2. 选择您的基础架构供应商。

3. 进入到安装类型的页面，下载与您的主机操作系统和架构对应的安装程序，并将该文件放在您要存储安装配置文件的目录中。

重要

安装程序会在用来安装集群的计算机上创建几个文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。

重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，请为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。此 pull secret 允许您与所含授权机构提供的服务进行身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

7.11.7. 为集群节点 SSH 访问生成密钥对

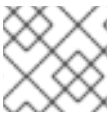
在 OpenShift Container Platform 安装过程中，您可以为安装程序提供 SSH 公钥。密钥通过它们的 Ignition 配置文件传递给 Red Hat Enterprise Linux CoreOS(RHCOS)节点，用于验证对节点的 SSH 访问。密钥添加到每个节点上 **core** 用户的 **~/.ssh/authorized_keys** 列表中，这将启用免密码身份验证。

将密钥传递给节点后，您可以使用密钥对作为用户 **核心** 通过 SSH 连接到 RHCOS 节点。若要通过 SSH 访问节点，必须由 SSH 为您的本地用户管理私钥身份。

如果要通过 SSH 连接到集群节点来执行安装调试或灾难恢复，则必须在安装过程中提供 SSH 公钥。**./openshift-install gather** 命令还需要在集群节点上设置 SSH 公钥。

重要

不要在生产环境中跳过这个过程，在生产环境中需要灾难恢复和调试。



注意

您必须使用本地密钥，而不是使用特定平台方法配置 的密钥，如 AWS 密钥对。

流程

1. 如果您在本地计算机上没有可用于在集群节点上进行身份验证的现有 SSH 密钥对，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_ed25519**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

注意

如果您计划在 **x86_64**、**ppc64le** 和 **s390x** 架构上安装使用 RHEL 加密库（这些加密库已提交给 NIST 用于 FIPS 140-2/140-3 验证）的 OpenShift Container Platform 集群，则不要创建使用 **ed25519** 算法的密钥。相反，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 查看公共 SSH 密钥：

```
$ cat <path>/<file_name>.pub
```

例如，运行以下命令来查看 **~/.ssh/id_ed25519.pub** 公钥：

```
$ cat ~/.ssh/id_ed25519.pub
```

3. 将 SSH 私钥身份添加到本地用户的 SSH 代理（如果尚未添加）。在集群节点上，或者要使用 **./openshift-install gather** 命令，需要对该密钥进行 SSH 代理管理，才能在集群节点上进行免密码 SSH 身份验证。

注意

在某些发行版中，自动管理默认 SSH 私钥身份，如 **~/.ssh/id_rsa** 和 **~/.ssh/id_dsa**。

a. 如果 **ssh-agent** 进程尚未为您的本地用户运行，请将其作为后台任务启动：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果集群处于 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

4. 将 SSH 私钥添加到 **ssh-agent** ：

```
$ ssh-add <path>/<file_name> (1)
```

(1) 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_ed25519.pub**

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

后续步骤

- 安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。如果在您置备的基础架构上安装集群，则必须为安装程序提供密钥。

7.11.8. 创建用于 Azure 的安装文件

要使用用户置备的基础架构在 Microsoft Azure 上安装 OpenShift Container Platform，您必须生成并修改安装程序部署集群所需的文件，以便集群只创建要使用的机器。您要生成并自定义 **install-config.yaml** 文件、Kubernetes 清单和 Ignition 配置文件。您还可以选择在安装准备阶段首先设置独立 **var** 分区。

7.11.8.1. 可选：创建独立 /var 分区

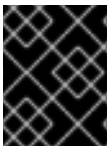
建议安装程序将 OpenShift Container Platform 的磁盘分区保留给安装程序。然而，在有些情况下您可能需要在文件系统的一部分中创建独立分区。

OpenShift Container Platform 支持添加单个分区来将存储附加到 **/var** 分区或 **/var** 的子目录中。例如：

- **/var/lib/containers** ：保存随着系统中添加更多镜像和容器而增长的容器相关内容。
- **/var/lib/etcd** ：保存您可能希望独立保留的数据，比如 etcd 存储的性能优化。
- **/var** ：保存您可能希望独立保留的数据，以满足审计等目的。

通过单独存储 **/var** 目录的内容，可以更轻松地根据需要为区域扩展存储，并在以后重新安装 OpenShift Container Platform，并保持该数据的完整性。使用这个方法，您不必再次拉取所有容器，在更新系统时也不必复制大量日志文件。

因为 **/var** 在进行一个全新的 Red Hat Enterprise Linux CoreOS（RHCOS）安装前必需存在，所以这个流程会在 OpenShift Container Platform 安装过程的 **openshift-install** 准备阶段插入一个创建的机器配置清单的机器配置来设置独立的 **/var** 分区。



重要

如果按照以下步骤在此流程中创建独立 **/var** 分区，则不需要再次创建 Kubernetes 清单和 Ignition 配置文件，如本节所述。

流程

1. 创建存放 OpenShift Container Platform 安装文件的目录：

```
$ mkdir $HOME/clusterconfig
```

2. 运行 **openshift-install**，以在 **manifest** 和 **openshift** 子目录中创建一组文件。在系统提示时回答系统问题：

```
$ openshift-install create manifests --dir $HOME/clusterconfig
```

输出示例

```
? SSH Public Key ...
INFO Credentials loaded from the "myprofile" profile in file "/home/myuser/.aws/credentials"
INFO Consuming Install Config from target directory
INFO Manifests created in: $HOME/clusterconfig/manifests and
$HOME/clusterconfig/openshift
```

3. 可选：确认安装程序在 **clusterconfig/openshift** 目录中创建了清单：

```
$ ls $HOME/clusterconfig/openshift/
```

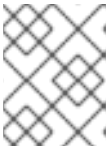
输出示例

```
99_kubeadmin-password-secret.yaml
99_openshift-cluster-api_master-machines-0.yaml
99_openshift-cluster-api_master-machines-1.yaml
99_openshift-cluster-api_master-machines-2.yaml
...
```

4. 创建用于配置额外分区的 Butane 配置。例如，将文件命名为 **$HOME/clusterconfig/98-var-partition.bu**，将磁盘设备名称改为 **worker** 系统上存储设备的名称，并根据情况设置存储大小。这个示例将 **/var** 目录放在一个单独的分区中：

```
variant: openshift
version: 4.16.0
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
storage:
  disks:
  - device: /dev/disk/by-id/<device_name> 1
    partitions:
    - label: var
      start_mib: <partition_start_offset> 2
      size_mib: <partition_size> 3
      number: 5
  filesystems:
    - device: /dev/disk/by-partlabel/var
      path: /var
      format: xfs
      mount_options: [defaults, prjquota] 4
      with_mount_unit: true
```

1. 要分区的磁盘的存储设备名称。
2. 在引导磁盘中添加数据分区时，推荐最少使用 25000 MiB(Mebibytes)。root 文件系统会自动调整大小以填充所有可用空间（最多到指定的偏移值）。如果没有指定值，或者指定的值小于推荐的最小值，则生成的 root 文件系统会太小，而在以后进行的 RHCOS 重新安装可能会覆盖数据分区的开始部分。
3. 以兆字节为单位的数据分区大小。
4. 对于用于容器存储的文件系统，必须启用 **prjquota** 挂载选项。



注意

当创建单独的 **/var** 分区时，如果不同的实例类型没有相同的设备名称，则无法为 worker 节点使用不同的实例类型。

5. 从 Butane 配置创建一个清单，并将它保存到 **clusterconfig/openshift** 目录中。例如，运行以下命令：

```
$ butane $HOME/clusterconfig/98-var-partition.bu -o $HOME/clusterconfig/openshift/98-var-partition.yaml
```

6. 再次运行 **openshift-install**，从 **manifest** 和 **openshift** 子目录中的一组文件创建 Ignition 配置：

```
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
auth  bootstrap.ign  master.ign  metadata.json  worker.ign
```

现在，您可以使用 Ignition 配置文件作为安装程序的输入来安装 Red Hat Enterprise Linux CoreOS(RHCOS)系统。

7.11.8.2. 创建安装配置文件

您可以自定义在 Microsoft Azure 上安装的 OpenShift Container Platform 集群。

先决条件

- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。
- 您有一个 Azure 订阅 ID 和租户 ID。
- 如果要使用服务主体安装集群，则有其应用程序 ID 和密码。
- 如果您要使用系统分配的受管身份安装集群，需要在您要从其中运行安装程序的虚拟机上启用它。
- 如果您要使用用户分配的受管身份安装集群，需要满足以下先决条件：
  - 您有它的客户端 ID。
  - 您已将其分配给您要从其运行安装程序的虚拟机。

流程

1. 可选：如果您之前在这个计算机上运行安装程序，并希望使用替代的服务主体或受管身份，请进入 **~/.azure/** 目录并删除 **osServicePrincipal.json** 配置文件。
   删除此文件可防止安装程序自动重复使用之前安装中的订阅和验证值。

2. 创建 **install-config.yaml** 文件。

   a. 进入包含安装程序的目录并运行以下命令：

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

   1 对于 **<installation_directory>**，请指定要存储安装程序创建的文件的目录名称。

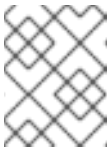
   在指定目录时：

   - 验证该目录是否具有**执行**权限。在安装目录中运行 Terraform 二进制文件需要这个权限。

- 使用空目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

b. 在提示符处，提供云的配置详情：

i. 可选： 选择用于访问集群机器的 SSH 密钥。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

ii. 选择 **azure** 作为目标平台。
如果安装程序无法找到之前安装中的 **osServicePrincipal.json** 配置文件，会提示您输入 Azure 订阅和验证值。

iii. 为您的订阅输入以下 Azure 参数值：

- **Azure subscription id** ：输入用于集群的订阅 ID。
- **Azure 租户 id** ：输入租户 ID。

iv. 根据您用来部署集群的 Azure 身份，在提示输入 **azure 服务主体客户端 id**时执行以下操作之一：

- 如果您使用服务主体，请输入其应用程序 ID。
- 如果您使用系统分配的受管身份，请将此值设为空白。
- 如果您使用用户分配的受管身份，请指定其客户端 ID。

v. 根据您用来部署集群的 Azure 身份，在提示输入 **azure 服务主体客户端 secret**时执行以下操作之一：

- 如果您使用服务主体，请输入其密码。
- 如果您使用系统分配的受管身份，请将此值设为空白。
- 如果您使用用户分配的受管身份，请将此值设为空白。

vi. 选择要将集群部署到的区域。

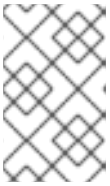
vii. 选择集群要部署到的基域。基域与您为集群创建的 Azure DNS 区对应。

viii. 为集群输入一个描述性名称。

重要

所有通过公共端点提供的 Azure 资源均存在资源名称的限制，您无法创建使用某些名称的资源。如需 Azure 限制词语的列表，请参阅 Azure 文档中的解决预留资源名称错误。

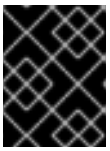
3. 修改 **install-config.yaml** 文件。您可以在"安装配置参数"部分找到有关可用参数的更多信息。



注意

如果要安装三节点集群，请确保将 **compute.replicas** 参数设置为 **0**。这样可确保集群的 control plane 可以调度。如需更多信息，请参阅"在 Azure 上安装三节点集群"。

4. 备份 **install-config.yaml** 文件，以便您可以使用它安装多个集群。



重要

install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用该文件，您必须立即备份该文件。

在以前的版本中，安装程序会创建一个 **osServicePrincipal.json** 配置文件，并将此文件存储在计算机上的 **~/.azure/** 目录中。这样可确保安装程序在目标平台上创建 OpenShift Container Platform 集群时可以加载配置集。

7.11.8.3. 在安装过程中配置集群范围的代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 **Proxy** 对象的 **spec.noProxy** 字段中添加站点来绕过代理。

注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，**Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

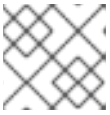
流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
```

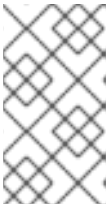
```
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> (5)
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 **.** 以仅匹配子域。例如，**.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代理。
4. 如果提供，安装程序会在 **openshift-config** 命名空间中生成名为 **user-ca-bundle** 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 **trusted-ca-bundle** 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， **Proxy** 对象的 **trustedCA** 字段中也会引用此配置映射。**additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。
5. 可选：决定 **Proxy** 对象的配置以引用 **trustedCA** 字段中 **user-ca-bundle** 配置映射的策略。允许的值是 **Proxyonly** 和 **Always**。仅在配置了 **http/https** 代理时，使用 **Proxyonly** 引用 **user-ca-bundle** 配置映射。使用 **Always** 始终引用 **user-ca-bundle** 配置映射。默认值为 **Proxyonly**。



注意

安装程序不支持代理的 **readinessEndpoints** 字段。



注意

如果安装程序超时，重启并使用安装程序的 **wait-for** 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. 保存该文件并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 cluster 的集群范围代理，该代理 **使用** 提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

7.11.8.4. 为 ARM 模板导出常用变量

您必须导出与提供的 Azure Resource Manager（ARM）模板搭配使用的一组常用变量，它们有助于在 Microsoft Azure 上完成用户提供基础架构安装。

注意

特定的 ARM 模板可能还需要其他导出变量，这些变量在相关的程序中详细介绍。

先决条件

- 获取 OpenShift Container Platform 安装程序和集群的 pull secret。

流程

1. 导出 **install-config.yaml** 中由提供的 ARM 模板使用的通用变量：

```
$ export CLUSTER_NAME=<cluster_name> (1)
$ export AZURE_REGION=<azure_region> (2)
$ export SSH_KEY=<ssh_key> (3)
$ export BASE_DOMAIN=<base_domain> (4)
$ export BASE_DOMAIN_RESOURCE_GROUP=<base_domain_resource_group> (5)
```

(1) **install-config.yaml** 文件中的**.metadata.name** 属性的值。

(2) 集群要部署到的区域，如 **centralus** 。这是来自 **install-config.yaml** 文件中的 **.platform.azure.region** 属性的值。

(3) 作为字符串的 SSH RSA 公钥文件。您必须使用引号包括 SSH 密钥，因为它包含空格。这是来自 **install-config.yaml** 文件中的 **.sshKey** 属性的值。

(4) 集群要部署到的基域。基域与您为集群创建的公共 DNS 区对应。这是来自 **install-config.yaml** 文件中的 **.baseDomain** 属性的值。

(5) 公共 DNS 区所在的资源组。这是来自 **install-config.yaml** 文件中的 **.platform.azure.baseDomainResourceGroupName** 属性的值。

例如：

```
$ export CLUSTER_NAME=test-cluster
$ export AZURE_REGION=centralus
$ export SSH_KEY="ssh-rsa xxx/xxx/xxx= user@email.com"
$ export BASE_DOMAIN=example.com
$ export BASE_DOMAIN_RESOURCE_GROUP=ocp-cluster
```

2. 导出 kubeadmin 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig (1)
```

(1) 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

7.11.8.5. 创建 Kubernetes 清单和 Ignition 配置文件

由于您必须修改一些集群定义文件并手动启动集群机器，因此您必须生成 Kubernetes 清单和 Ignition 配置文件来配置机器。

安装配置文件转换为 Kubernetes 清单。清单嵌套到 Ignition 配置文件中，稍后用于配置集群机器。

重要

- OpenShift Container Platform 安装程序生成的 Ignition 配置文件包含 24 小时后过期的证书，然后在该时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

先决条件

- 已获得 OpenShift Container Platform 安装程序。
- 已创建 **install-config.yaml** 安装配置文件。

流程

1. 进入包含 OpenShift Container Platform 安装程序的目录，并为集群生成 Kubernetes 清单：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定包含您创建的 **install-config.yaml** 文件的安装目录。

2. 删除定义 control plane 机器的 Kubernetes 清单文件：

```
$ rm -f <installation_directory>/openshift/99_openshift-cluster-api_master-machines-*.yaml
```

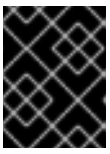
通过删除这些文件，您可以防止集群自动生成 control plane 机器。

3. 删除定义 control plane 机器集的 Kubernetes 清单文件：

```
$ rm -f <installation_directory>/openshift/99_openshift-machine-api_master-control-plane-machine-set.yaml
```

4. 删除定义 worker 机器的 Kubernetes 清单文件：

```
$ rm -f <installation_directory>/openshift/99_openshift-cluster-api_worker-machineset-*.yaml
```



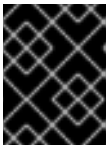
重要

如果在用户置备的基础架构上安装集群时禁用了 **MachineAPI** 功能，则必须删除定义 worker 机器的 Kubernetes 清单文件。否则，集群将无法安装。

由于您要自行创建和管理 worker 机器，因此不需要初始化这些机器。

警告

如果您要安装一个三节点集群，请跳过以下步骤，以便可以调度 control plane 节点。



重要

当您将 control plane 节点从默认的不可调度配置为可以调度时，需要额外的订阅。这是因为 control plane 节点变为计算节点。

5. 检查 **<installation_directory>/manifests/cluster-scheduler-02-config.yml Kubernetes 清单文件中的 mastersSchedulable** 参数是否已设置为 **false**。此设置可防止在 control plane 机器上调度 pod：

   a. 打开 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** 文件。

   b. 找到 **mastersSchedulable** 参数，并确保它被设置为 **false**。

   c. 保存并退出 文件。

6. 可选：如果您不希望 Ingress Operator 代表您创建 DNS 记录，请删除 **<installation_directory>/manifests/cluster-dns-02-config.yml** DNS 配置文件中的 **privateZone 和 publicZone** 部分：

```
apiVersion: config.openshift.io/v1
kind: DNS
metadata:
  creationTimestamp: null
  name: cluster
spec:
  baseDomain: example.openshift.com
  privateZone: 1
    id: mycluster-100419-private-zone
  publicZone: 2
    id: example.openshift.com
status: {}
```

1 2 完全删除此部分。

   如果您这样做，后续步骤中必须手动添加入口 DNS 记录。

7. 在用户置备的基础架构上配置 Azure 时，您必须导出清单文件中定义的一些常见变量，以备稍后在 Azure Resource Manager（ARM）模板中使用：

   a. 使用以下命令导出基础架构 ID:

```
$ export INFRA_ID=<infra_id> 1
```

1 OpenShift Container Platform 集群被分配了一个标识符（**INFRA_ID**），其格式为 **<cluster_name>-<random_string>**。这将作为使用提供的 ARM 模板创建的大部分资

b. 使用以下命令导出资源组：

```
$ export RESOURCE_GROUP=<resource_group> 1
```

1 此 Azure 部署中创建的所有资源都作为资源组的一部分。资源组名称还基于 **INFRA_ID**，格式为 **<cluster_name>-<random_string>-rg**。这是来自 **manifests/cluster-infrastructure-02-config.yml** 文件中的 **.status.platformStatus.azure.resourceGroupName** 属性的值。

8. 要创建 Ignition 配置文件，从包含安装程序的目录运行以下命令：

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定相同的安装目录。

为安装目录中的 bootstrap、control plane 和计算节点创建 Ignition 配置文件。**kubeadmin-password** 和 **kubeconfig** 文件是在 **./<installation_directory>/auth** 目录中创建的：

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

7.11.9. 创建 Azure 资源组

您必须创建一个 Microsoft Azure 资源组以及该资源组的身份。它们都用于在 Azure 上安装 OpenShift Container Platform 集群。

先决条件

- 配置 Azure 帐户。
- 为集群生成 Ignition 配置文件。

流程

1. 在受支持的 Azure 区域中创建资源组：

```
$ az group create --name ${RESOURCE_GROUP} --location ${AZURE_REGION}
```

2. 为资源组创建 Azure 身份：

```
$ az identity create -g ${RESOURCE_GROUP} -n ${INFRA_ID}-identity
```

这用于授予集群中 Operator 所需的访问权限。例如，这允许 Ingress Operator 创建公共 IP 及其负载均衡器。您必须将 Azure 身份分配给角色。

3. 将 Contributor 角色授予 Azure 身份：

   a. 导出 Azure 角色分配所需的以下变量：

```
$ export PRINCIPAL_ID=`az identity show -g ${RESOURCE_GROUP} -n ${INFRA_ID}-identity --query principalId --out tsv`
```

```
$ export RESOURCE_GROUP_ID=`az group show -g ${RESOURCE_GROUP} --query id --out tsv`
```

   b. 将 Contributor 角色分配给身份：

```
$ az role assignment create --assignee "${PRINCIPAL_ID}" --role 'Contributor' --scope "${RESOURCE_GROUP_ID}"
```

注意

如果要为身份分配具有所有所需权限的自定义角色，请运行以下命令：

```
$ az role assignment create --assignee "${PRINCIPAL_ID}" --role <custom_role> \ 1
--scope "${RESOURCE_GROUP_ID}"
```

1 指定自定义角色名称。

7.11.10. 上传 RHCOS 集群镜像和 bootstrap Ignition 配置文件

Azure 客户端不支持基于本地现有文件进行部署。您必须复制 RHCOS 虚拟硬盘(VHD)集群镜像，并将 bootstrap Ignition 配置文件存储在存储容器中，以便在部署过程中访问它们。

先决条件

- 配置 Azure 帐户。
- 为集群生成 Ignition 配置文件。

流程

1. 创建 Azure 存储帐户以存储 VHD 集群镜像：

```
$ az storage account create -g ${RESOURCE_GROUP} --location ${AZURE_REGION} --name ${CLUSTER_NAME}sa --kind Storage --sku Standard_LRS
```

警告

Azure 存储帐户名称的长度必须在 3 到 24 个字符之，且只使用数字和小写字母。如果您的 **CLUSTER_NAME** 变量没有遵循这些限制，您必须手动定义 Azure 存储帐户名称。如需有关 Azure 存储帐户名称限制的更多信息，请参阅 Azure 文档中的解决存储帐户名称的错误。

2. 将存储帐户密钥导出为环境变量：

```
$ export ACCOUNT_KEY=`az storage account keys list -g ${RESOURCE_GROUP} --
account-name ${CLUSTER_NAME}sa --query "[0].value" -o tsv`
```

3. 将 RHCOS VHD 的 URL 导出为环境变量：

```
$ export VHD_URL=`openshift-install coreos print-stream-json | jq -r '.architectures.
<architecture>."rhel-coreos-extensions"."azure-disk".url'`
```

其中：

<architecture>

指定架构，有效值包括 **x86_64** 或 **aarch64**。

重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每个发行版本而改变。您必须指定一个最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。如果可用，请使用与 OpenShift Container Platform 版本匹配的镜像版本。

4. 为 VHD 创建存储容器：

```
$ az storage container create --name vhd --account-name ${CLUSTER_NAME}sa --account-
key ${ACCOUNT_KEY}
```

5. 将本地 VHD 复制为一个 blob:

```
$ az storage blob copy start --account-name ${CLUSTER_NAME}sa --account-key
${ACCOUNT_KEY} --destination-blob "rhcos.vhd" --destination-container vhd --source-uri
"${VHD_URL}"
```

6. 创建 blob 存储容器并上传生成的 **bootstrap.ign** 文件：

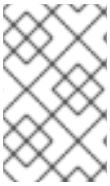
```
$ az storage container create --name files --account-name ${CLUSTER_NAME}sa --
account-key ${ACCOUNT_KEY}
```

```
$ az storage blob upload --account-name ${CLUSTER_NAME}sa --account-key
${ACCOUNT_KEY} -c "files" -f "<installation_directory>/bootstrap.ign" -n "bootstrap.ign"
```

7.11.11. 创建 DNS 区示例

使用用户置备的基础架构的集群需要 DNS 记录。您应该选择适合您的场景的 DNS 策略。

在本例中，使用了 Azure 的 DNS 解决方案，因此您将为外部（内部网络）可见性创建一个新的公共 DNS 区域，并为内部集群解析创建一个私有 DNS 区域。



注意

公共 DNS 区域不需要与集群部署位于同一个资源组中，且可能已在您的机构中为所需基域存在。如果情况如此，您可以跳过创建公共 DNS 区这一步；请确定您之前生成的安装配置反映了这种情况。

先决条件

- 配置 Azure 帐户。
- 为集群生成 Ignition 配置文件。

流程

1. 在 **BASE_DOMAIN_RESOURCE_GROUP** 环境变量中导出的资源组中创建新的公共 DNS 区域：

```
$ az network dns zone create -g ${BASE_DOMAIN_RESOURCE_GROUP} -n
${CLUSTER_NAME}.${BASE_DOMAIN}
```

   如果您使用的是公共 DNS 区域，可以跳过这一步。

2. 在与这个部署的其余部分相同的资源组中创建私有 DNS 区域：

```
$ az network private-dns zone create -g ${RESOURCE_GROUP} -n
${CLUSTER_NAME}.${BASE_DOMAIN}
```

如需了解更多信息，请参阅在 Azure 中配置公共 DNS 的信息。

7.11.12. 在 Azure 中创建 VNet

您必须在 Microsoft Azure 中创建虚拟网络（VNet），供您的 OpenShift Container Platform 集群使用。您可以对 VNet 进行定制来满足您的要求。创建 VNet 的一种方法是修改提供的 Azure Resource Manager（ARM）模板。

注意

如果不使用提供的 ARM 模板来创建 Azure 基础架构，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 Azure 帐户。
- 为集群生成 Ignition 配置文件。

流程

1. 复制 **VNet 的 ARM 模板** 一节中的模板，并将它以 **01_vnet.json** 保存到集群的安装目录中。此模板描述了集群所需的 VNet。

2. 使用 **az** CLI 创建部署：

```
$ az deployment group create -g ${RESOURCE_GROUP} \
  --template-file "<installation_directory>/01_vnet.json" \
  --parameters baseName="${INFRA_ID}" 1
```

1 资源名称使用的基本名称 ; 这通常是集群的基础架构 ID。

3. 将 VNet 模板链接到私有 DNS 区域：

```
$ az network private-dns link vnet create -g ${RESOURCE_GROUP} -z
${CLUSTER_NAME}.${BASE_DOMAIN} -n ${INFRA_ID}-network-link -v "${INFRA_ID}-vnet"
-e false
```

7.11.12.1. VNet 的 ARM 模板

您可以使用以下 Azure Resource Manager（ARM）模板来部署 OpenShift Container Platform 集群所需的 VPC：

例 7.85. 01_vnet.json ARM 模板

```
{
  "$schema" : "https://schema.management.azure.com/schemas/2015-01-
01/deploymentTemplate.json#",
  "contentVersion" : "1.0.0.0",
  "parameters" : {
    "baseName" : {
      "type" : "string",
      "minLength" : 1,
      "metadata" : {
        "description" : "Base name to be used in resource names (usually the cluster's Infra ID)"
      }
    }
  },
  "variables" : {
    "location" : "[resourceGroup().location]",
    "virtualNetworkName" : "[concat(parameters('baseName'), '-vnet')]",
    "addressPrefix" : "10.0.0.0/16",
    "masterSubnetName" : "[concat(parameters('baseName'), '-master-subnet')]",
    "masterSubnetPrefix" : "10.0.0.0/24",
    "nodeSubnetName" : "[concat(parameters('baseName'), '-worker-subnet')]",
    "nodeSubnetPrefix" : "10.0.1.0/24",
    "clusterNsgName" : "[concat(parameters('baseName'), '-nsg')]"
  },
  "resources" : [
    {
      "apiVersion" : "2018-12-01",
      "type" : "Microsoft.Network/virtualNetworks",
      "name" : "[variables('virtualNetworkName')]",
      "location" : "[variables('location')]",
      "dependsOn" : [
```

```
        "[concat('Microsoft.Network/networkSecurityGroups/', variables('clusterNsgName'))]"
      ],
      "properties" : {
        "addressSpace" : {
          "addressPrefixes" : [
            "[variables('addressPrefix')]"
          ]
        },
        "subnets" : [
          {
            "name" : "[variables('masterSubnetName')]",
            "properties" : {
              "addressPrefix" : "[variables('masterSubnetPrefix')]",
              "serviceEndpoints": [],
              "networkSecurityGroup" : {
                "id" : "[resourceId('Microsoft.Network/networkSecurityGroups',
variables('clusterNsgName'))]"
              }
            }
          },
          {
            "name" : "[variables('nodeSubnetName')]",
            "properties" : {
              "addressPrefix" : "[variables('nodeSubnetPrefix')]",
              "serviceEndpoints": [],
              "networkSecurityGroup" : {
                "id" : "[resourceId('Microsoft.Network/networkSecurityGroups',
variables('clusterNsgName'))]"
              }
            }
          }
        ]
      }
    },
    {
      "type" : "Microsoft.Network/networkSecurityGroups",
      "name" : "[variables('clusterNsgName')]",
      "apiVersion" : "2018-10-01",
      "location" : "[variables('location')]",
      "properties" : {
        "securityRules" : [
          {
            "name" : "apiserver_in",
            "properties" : {
              "protocol" : "Tcp",
              "sourcePortRange" : "*",
              "destinationPortRange" : "6443",
              "sourceAddressPrefix" : "*",
              "destinationAddressPrefix" : "*",
              "access" : "Allow",
              "priority" : 101,
              "direction" : "Inbound"
            }
          }
        ]
      }
```

```
    }
  ]
}
```

7.11.13. 为 Azure 基础架构创建 RHCOS 集群镜像

您必须对 OpenShift Container Platform 节点的 Microsoft Azure 使用有效的 Red Hat Enterprise Linux CoreOS（RHCOS）镜像。

先决条件

- 配置 Azure 帐户。
- 为集群生成 Ignition 配置文件。
- 将 RHCOS 虚拟硬盘（VHD）集群镜像存储在 Azure 存储容器中。
- 在 Azure 存储容器中存储 bootstrap Ignition 配置文件。

流程

1. 复制**镜像存储的 ARM 模板** 部分中的模板，并将它以 **02_storage.json** 保存到集群的安装目录中。此模板描述了集群所需的镜像存储。
2. 以一个变量的形式将 RHCOS VHD blob URL 导出：

```
$ export VHD_BLOB_URL=`az storage blob url --account-name ${CLUSTER_NAME}sa --account-key ${ACCOUNT_KEY} -c vhd -n "rhcos.vhd" -o tsv`
```

3. 部署集群镜像

```
$ az deployment group create -g ${RESOURCE_GROUP} \
  --template-file "<installation_directory>/02_storage.json" \
  --parameters vhdBlobURL="${VHD_BLOB_URL}" \ 1
  --parameters baseName="${INFRA_ID}" \ 2
  --parameters storageAccount="${CLUSTER_NAME}sa" \ 3
  --parameters architecture="<architecture>" 4
```

1. 用于创建 master 和 worker 机器的 RHCOS VHD 的 blob URL。
2. 资源名称使用的基本名称；这通常是集群的基础架构 ID。
3. Azure 存储帐户的名称。
4. 指定系统架构。有效值为 **x64** （默认）或 **Arm64**。

7.11.13.1. 镜像存储的 ARM 模板

您可以使用以下 Azure Resource Manager（ARM）模板来部署 OpenShift Container Platform 集群所需的存储的 Red Hat Enterprise Linux CoreOS（RHCOS）镜像：

例 7.86. 02_storage.json ARM 模板

```
{
  "$schema": "https://schema.management.azure.com/schemas/2019-04-
01/deploymentTemplate.json#",
  "contentVersion": "1.0.0.0",
  "parameters": {
    "architecture": {
      "type": "string",
      "metadata": {
        "description": "The architecture of the Virtual Machines"
      },
      "defaultValue": "x64",
      "allowedValues": [
        "Arm64",
        "x64"
      ]
    },
    "baseName": {
      "type": "string",
      "minLength": 1,
      "metadata": {
        "description": "Base name to be used in resource names (usually the cluster's Infra ID)"
      }
    },
    "storageAccount": {
      "type": "string",
      "metadata": {
        "description": "The Storage Account name"
      }
    },
    "vhdBlobURL": {
      "type": "string",
      "metadata": {
        "description": "URL pointing to the blob where the VHD to be used to create master and
worker machines is located"
      }
    }
  },
  "variables": {
    "location": "[resourceGroup().location]",
    "galleryName": "[concat('gallery_', replace(parameters('baseName'), '-', '_'))]",
    "imageName": "[parameters('baseName')]",
    "imageNameGen2": "[concat(parameters('baseName'), '-gen2')]",
    "imageRelease": "1.0.0"
  },
  "resources": [
    {
      "apiVersion": "2021-10-01",
      "type": "Microsoft.Compute/galleries",
      "name": "[variables('galleryName')]",
      "location": "[variables('location')]",
      "resources": [
        {
          "apiVersion": "2021-10-01",
          "type": "images",
```

```
        "name": "[variables('imageName')]",
        "location": "[variables('location')]",
        "dependsOn": [
          "[variables('galleryName')]"
        ],
        "properties": {
          "architecture": "[parameters('architecture')]",
          "hyperVGeneration": "V1",
          "identifier": {
            "offer": "rhcos",
            "publisher": "RedHat",
            "sku": "basic"
          },
          "osState": "Generalized",
          "osType": "Linux"
        },
        "resources": [
          {
            "apiVersion": "2021-10-01",
            "type": "versions",
            "name": "[variables('imageRelease')]",
            "location": "[variables('location')]",
            "dependsOn": [
              "[variables('imageName')]"
            ],
            "properties": {
              "publishingProfile": {
                "storageAccountType": "Standard_LRS",
                "targetRegions": [
                  {
                    "name": "[variables('location')]",
                    "regionalReplicaCount": "1"
                  }
                ]
              },
              "storageProfile": {
                "osDiskImage": {
                  "source": {
                    "id": "[resourceId('Microsoft.Storage/storageAccounts',
parameters('storageAccount'))]",
                    "uri": "[parameters('vhdBlobURL')]"
                  }
                }
              }
            }
          }
        ]
      },
      {
        "apiVersion": "2021-10-01",
        "type": "images",
        "name": "[variables('imageNameGen2')]",
        "location": "[variables('location')]",
        "dependsOn": [
          "[variables('galleryName')]"
        ],
```

```
          "properties": {
            "architecture": "[parameters('architecture')]",
            "hyperVGeneration": "V2",
            "identifier": {
              "offer": "rhcos-gen2",
              "publisher": "RedHat-gen2",
              "sku": "gen2"
            },
            "osState": "Generalized",
            "osType": "Linux"
          },
          "resources": [
            {
              "apiVersion": "2021-10-01",
              "type": "versions",
              "name": "[variables('imageRelease')]",
              "location": "[variables('location')]",
              "dependsOn": [
                "[variables('imageNameGen2')]"
              ],
              "properties": {
                "publishingProfile": {
                  "storageAccountType": "Standard_LRS",
                  "targetRegions": [
                    {
                      "name": "[variables('location')]",
                      "regionalReplicaCount": "1"
                    }
                  ]
                },
                "storageProfile": {
                  "osDiskImage": {
                    "source": {
                      "id": "[resourceId('Microsoft.Storage/storageAccounts',
parameters('storageAccount'))]",
                      "uri": "[parameters('vhdBlobURL')]"
                    }
                  }
                }
              }
            }
          ]
        }
      ]
    }
  ]
}
```

7.11.14. 用户置备的基础架构对网络的要求

所有 Red Hat Enterprise Linux CoreOS（RHCOS）机器需要在启动过程中在 **initramfs** 中配置网络，以获取其 Ignition 配置文件。

7.11.14.1. 网络连接要求

您必须配置机器之间的网络连接，以允许 OpenShift Container Platform 集群组件进行通信。每台机器都必须能够解析集群中所有其他机器的主机名。

本节详细介绍了所需的端口。

重要

在连接的 OpenShift Container Platform 环境中，所有节点都需要访问互联网才能为平台容器拉取镜像，并向红帽提供遥测数据。

表 7.27. 用于全机器到所有机器通信的端口

协议	port	描述
ICMP	N/A	网络可访问性测试
TCP	**1936**	指标
	9000-**9999**	主机级别的服务，包括端口 9**100**-**9101** 上的节点导出器，以及端口 **9099** 上的 Cluster Version Operator。
	10250-**10259**	Kubernetes 保留的默认端口
UDP	**4789**	VXLAN
	6081	Geneve
	9000-**9999**	主机级别的服务，包括端口 **9100**到**9101** 上的节点导出器。
	500	IPsec IKE 数据包
	4500	IPsec NAT-T 数据包
	123	UDP 端口 **123** 上的网络时间协议 (NTP) 如果配置了外部 NTP 时间服务器，需要打开 UDP 端口 **123**。
TCP/UDP	**30000**-**32767**	Kubernetes 节点端口
ESP	N/A	IPsec Encapsulating Security Payload(ESP)

表 7.28. 用于所有机器控制平面通信的端口

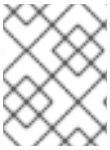
协议	port	描述
TCP	**6443**	Kubernetes API

表 7.29. control plane 机器用于 control plane 机器通信的端口

协议	port	描述
TCP	**2379-2380**	etcd 服务器和对等端口

7.11.15. 在 Azure 中创建网络和负载均衡组件

您必须在 Microsoft Azure 中配置网络和负载均衡，供您的 OpenShift Container Platform 集群使用。创建这些组件的一种方法是修改提供的 Azure Resource Manager（ARM）模板。



注意

如果不使用提供的 ARM 模板来创建 Azure 基础架构，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 Azure 帐户。
- 为集群生成 Ignition 配置文件。
- 在 Azure 中创建和配置 VNet 及相关子网。

流程

1. 复制 **网络和负载均衡器的 ARM 模板** 一节中的模板，并将它以 **03_infra.json** 保存到集群的安装目录中。此模板描述了集群所需的网络和负载均衡对象。
2. 使用 **az** CLI 创建部署：

```
$ az deployment group create -g ${RESOURCE_GROUP} \
  --template-file "<installation_directory>/03_infra.json" \
  --parameters privateDNSZoneName="${CLUSTER_NAME}.${BASE_DOMAIN}" \ 1
  --parameters baseName="${INFRA_ID}" 2
```

1. 私有 DNS 区的名称。
2. 资源名称使用的基本名称；这通常是集群的基础架构 ID。

3. 在公共区为 API 公共负载均衡器创建一个 **api** DNS 记录。**${BASE_DOMAIN_RESOURCE_GROUP}** 变量必须指向存在公共 DNS 区的资源组。
   a. 导出以下变量：

```
$ export PUBLIC_IP=`az network public-ip list -g ${RESOURCE_GROUP} --query "[?
name=='${INFRA_ID}-master-pip'] | [0].ipAddress" -o tsv`
```

   b. 在新的公共区中创建 **api** DNS 记录：

```
$ az network dns record-set a add-record -g ${BASE_DOMAIN_RESOURCE_GROUP} -
z ${CLUSTER_NAME}.${BASE_DOMAIN} -n api -a ${PUBLIC_IP} --ttl 60
```

   如果要将集群添加到现有的公共区，您可以在其中创建 **api** DNS 记录：

```
$ az network dns record-set a add-record -g ${BASE_DOMAIN_RESOURCE_GROUP} -
z ${BASE_DOMAIN} -n api.${CLUSTER_NAME} -a ${PUBLIC_IP} --ttl 60
```

7.11.15.1. 网络和负载均衡器的 ARM 模板

您可以使用以下 Azure Resource Manager(ARM)模板来部署 OpenShift Container Platform 集群所需的网络对象和负载均衡器：

例 7.87. 03_infra.json ARM 模板

```
{
  "$schema" : "https://schema.management.azure.com/schemas/2015-01-
01/deploymentTemplate.json#",
  "contentVersion" : "1.0.0.0",
  "parameters" : {
    "baseName" : {
      "type" : "string",
      "minLength" : 1,
      "metadata" : {
        "description" : "Base name to be used in resource names (usually the cluster's Infra ID)"
      }
    },
    "vnetBaseName": {
      "type": "string",
      "defaultValue": "",
      "metadata" : {
        "description" : "The specific customer vnet's base name (optional)"
      }
    },
    "privateDNSZoneName" : {
      "type" : "string",
      "metadata" : {
        "description" : "Name of the private DNS zone"
      }
    }
  },
  "variables" : {
    "location" : "[resourceGroup().location]",
    "virtualNetworkName" : "[concat(if(not(empty(parameters('vnetBaseName'))),
parameters('vnetBaseName'), parameters('baseName')), '-vnet')]",
    "virtualNetworkID" : "[resourceId('Microsoft.Network/virtualNetworks',
variables('virtualNetworkName'))]",
    "masterSubnetName" : "[concat(if(not(empty(parameters('vnetBaseName'))),
parameters('vnetBaseName'), parameters('baseName')), '-master-subnet')]",
    "masterSubnetRef" : "[concat(variables('virtualNetworkID'), '/subnets/',
variables('masterSubnetName'))]",
    "masterPublicIpAddressName" : "[concat(parameters('baseName'), '-master-pip')]",
    "masterPublicIpAddressID" : "[resourceId('Microsoft.Network/publicIPAddresses',
variables('masterPublicIpAddressName'))]",
    "masterLoadBalancerName" : "[parameters('baseName')]",
    "masterLoadBalancerID" : "[resourceId('Microsoft.Network/loadBalancers',
variables('masterLoadBalancerName'))]",
    "internalLoadBalancerName" : "[concat(parameters('baseName'), '-internal-lb')]",
    "internalLoadBalancerID" : "[resourceId('Microsoft.Network/loadBalancers',
variables('internalLoadBalancerName'))]",
```

```
    "skuName": "Standard"
  },
  "resources" : [
    {
      "apiVersion" : "2018-12-01",
      "type" : "Microsoft.Network/publicIPAddresses",
      "name" : "[variables('masterPublicIpAddressName')]",
      "location" : "[variables('location')]",
      "sku": {
        "name": "[variables('skuName')]"
      },
      "properties" : {
        "publicIPAllocationMethod" : "Static",
        "dnsSettings" : {
          "domainNameLabel" : "[variables('masterPublicIpAddressName')]"
        }
      }
    },
    {
      "apiVersion" : "2018-12-01",
      "type" : "Microsoft.Network/loadBalancers",
      "name" : "[variables('masterLoadBalancerName')]",
      "location" : "[variables('location')]",
      "sku": {
        "name": "[variables('skuName')]"
      },
      "dependsOn" : [
        "[concat('Microsoft.Network/publicIPAddresses/', variables('masterPublicIpAddressName'))]"
      ],
      "properties" : {
        "frontendIPConfigurations" : [
          {
            "name" : "public-lb-ip-v4",
            "properties" : {
              "publicIPAddress" : {
                "id" : "[variables('masterPublicIpAddressID')]"
              }
            }
          }
        ],
        "backendAddressPools" : [
          {
            "name" : "[variables('masterLoadBalancerName')]"
          }
        ],
        "loadBalancingRules" : [
          {
            "name" : "api-internal",
            "properties" : {
              "frontendIPConfiguration" : {
                "id" :"[concat(variables('masterLoadBalancerID'), '/frontendIPConfigurations/public-lb-ip-
v4')]"
              },
              "backendAddressPool" : {
                "id" : "[concat(variables('masterLoadBalancerID'), '/backendAddressPools/',
variables('masterLoadBalancerName'))]"
```

```
            },
            "protocol" : "Tcp",
            "loadDistribution" : "Default",
            "idleTimeoutInMinutes" : 30,
            "frontendPort" : 6443,
            "backendPort" : 6443,
            "probe" : {
              "id" : "[concat(variables('masterLoadBalancerID'), '/probes/api-internal-probe')]"
            }
          }
        }
      ],
      "probes" : [
        {
          "name" : "api-internal-probe",
          "properties" : {
            "protocol" : "Https",
            "port" : 6443,
            "requestPath": "/readyz",
            "intervalInSeconds" : 10,
            "numberOfProbes" : 3
          }
        }
      ]
    }
  },
  {
    "apiVersion" : "2018-12-01",
    "type" : "Microsoft.Network/loadBalancers",
    "name" : "[variables('internalLoadBalancerName')]",
    "location" : "[variables('location')]",
    "sku": {
      "name": "[variables('skuName')]"
    },
    "properties" : {
      "frontendIPConfigurations" : [
        {
          "name" : "internal-lb-ip",
          "properties" : {
            "privateIPAllocationMethod" : "Dynamic",
            "subnet" : {
              "id" : "[variables('masterSubnetRef')]"
            },
            "privateIPAddressVersion" : "IPv4"
          }
        }
      ],
      "backendAddressPools" : [
        {
          "name" : "internal-lb-backend"
        }
      ],
      "loadBalancingRules" : [
        {
          "name" : "api-internal",
          "properties" : {
```

```
            "frontendIPConfiguration" : {
              "id" : "[concat(variables('internalLoadBalancerID'), '/frontendIPConfigurations/internal-lb-ip')]"
            },
            "frontendPort" : 6443,
            "backendPort" : 6443,
            "enableFloatingIP" : false,
            "idleTimeoutInMinutes" : 30,
            "protocol" : "Tcp",
            "enableTcpReset" : false,
            "loadDistribution" : "Default",
            "backendAddressPool" : {
              "id" : "[concat(variables('internalLoadBalancerID'), '/backendAddressPools/internal-lb-backend')]"
            },
            "probe" : {
              "id" : "[concat(variables('internalLoadBalancerID'), '/probes/api-internal-probe')]"
            }
          }
        },
        {
          "name" : "sint",
          "properties" : {
            "frontendIPConfiguration" : {
              "id" : "[concat(variables('internalLoadBalancerID'), '/frontendIPConfigurations/internal-lb-ip')]"
            },
            "frontendPort" : 22623,
            "backendPort" : 22623,
            "enableFloatingIP" : false,
            "idleTimeoutInMinutes" : 30,
            "protocol" : "Tcp",
            "enableTcpReset" : false,
            "loadDistribution" : "Default",
            "backendAddressPool" : {
              "id" : "[concat(variables('internalLoadBalancerID'), '/backendAddressPools/internal-lb-backend')]"
            },
            "probe" : {
              "id" : "[concat(variables('internalLoadBalancerID'), '/probes/sint-probe')]"
            }
          }
        }
      ],
      "probes" : [
        {
          "name" : "api-internal-probe",
          "properties" : {
            "protocol" : "Https",
            "port" : 6443,
            "requestPath": "/readyz",
            "intervalInSeconds" : 10,
            "numberOfProbes" : 3
          }
        },
        {
```

```
            "name" : "sint-probe",
            "properties" : {
              "protocol" : "Https",
              "port" : 22623,
              "requestPath": "/healthz",
              "intervalInSeconds" : 10,
              "numberOfProbes" : 3
            }
          }
        ]
      }
    },
    {
      "apiVersion": "2018-09-01",
      "type": "Microsoft.Network/privateDnsZones/A",
      "name": "[concat(parameters('privateDNSZoneName'), '/api')]",
      "location" : "[variables('location')]",
      "dependsOn" : [
        "[concat('Microsoft.Network/loadBalancers/', variables('internalLoadBalancerName'))]"
      ],
      "properties": {
        "ttl": 60,
        "aRecords": [
          {
            "ipv4Address": "
[reference(variables('internalLoadBalancerName')).frontendIPConfigurations[0].properties.privateIP
Address]"
          }
        ]
      }
    },
    {
      "apiVersion": "2018-09-01",
      "type": "Microsoft.Network/privateDnsZones/A",
      "name": "[concat(parameters('privateDNSZoneName'), '/api-int')]",
      "location" : "[variables('location')]",
      "dependsOn" : [
        "[concat('Microsoft.Network/loadBalancers/', variables('internalLoadBalancerName'))]"
      ],
      "properties": {
        "ttl": 60,
        "aRecords": [
          {
            "ipv4Address": "
[reference(variables('internalLoadBalancerName')).frontendIPConfigurations[0].properties.privateIP
Address]"
          }
        ]
      }
    }
  ]
}
```

7.11.16. 在 Azure 中创建 bootstrap 机器

您必须在 Microsoft Azure 中创建 bootstrap 机器，以便在 OpenShift Container Platform 集群初始化过程中使用。创建此机器的一种方法是修改提供的 Azure Resource Manager（ARM）模板。

注意

如果不使用提供的 ARM 模板来创建 bootstrap 机器，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 Azure 帐户。
- 为集群生成 Ignition 配置文件。
- 在 Azure 中创建和配置 VNet 及相关子网。
- 在 Azure 中创建和配置联网及负载均衡器。
- 创建 control plane 和计算角色。

流程

1. 复制 **bootstrap 机器的 ARM 模板一节中的模板**，并将它以 **04_bootstrap.json** 保存到集群的安装目录中。此模板描述了集群所需的 bootstrap 机器。
2. 导出 bootstrap URL 变量：

```
$ bootstrap_url_expiry=`date -u -d "10 hours" '+%Y-%m-%dT%H:%MZ'`
```

```
$ export BOOTSTRAP_URL=`az storage blob generate-sas -c 'files' -n 'bootstrap.ign' --https-only --full-uri --permissions r --expiry $bootstrap_url_expiry --account-name ${CLUSTER_NAME}sa --account-key ${ACCOUNT_KEY} -o tsv`
```

3. 导出 bootstrap ignition 变量：

```
$ export BOOTSTRAP_IGNITION=`jq -rcnM --arg v "3.2.0" --arg url ${BOOTSTRAP_URL} '{ignition:{version:$v,config:{replace:{source:$url}}}}' | base64 | tr -d '\n'`
```

4. 使用 **az** CLI 创建部署：

```
$ az deployment group create -g ${RESOURCE_GROUP} \
  --template-file "<installation_directory>/04_bootstrap.json" \
  --parameters bootstrapIgnition="${BOOTSTRAP_IGNITION}" \ 1
  --parameters baseName="${INFRA_ID}" \ 2
  --parameter bootstrapVMSize="Standard_D4s_v3" 3
```

1. bootstrap 集群的 bootstrap Ignition 内容。
2. 资源名称使用的基本名称；这通常是集群的基础架构 ID。
3. 可选：指定 bootstrap 虚拟机的大小。使用与指定架构兼容的虚拟机大小。如果未定义这个值，则会设置模板中的默认值。

7.11.16.1. bootstrap 机器的 ARM 模板

您可以使用以下 Azure Resource Manager(ARM)模板来部署 OpenShift Container Platform 集群所需的 bootstrap 机器：

例 7.88. 04_bootstrap.json ARM 模板

```
{
  "$schema" : "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",
  "contentVersion" : "1.0.0.0",
  "parameters" : {
    "baseName" : {
      "type" : "string",
      "minLength" : 1,
      "metadata" : {
        "description" : "Base name to be used in resource names (usually the cluster's Infra ID)"
      }
    },
    "vnetBaseName": {
      "type": "string",
      "defaultValue": "",
      "metadata" : {
        "description" : "The specific customer vnet's base name (optional)"
      }
    },
    "bootstrapIgnition" : {
      "type" : "string",
      "minLength" : 1,
      "metadata" : {
        "description" : "Bootstrap ignition content for the bootstrap cluster"
      }
    },
    "sshKeyData" : {
      "type" : "securestring",
      "defaultValue" : "Unused",
      "metadata" : {
        "description" : "Unused"
      }
    },
    "bootstrapVMSize" : {
      "type" : "string",
      "defaultValue" : "Standard_D4s_v3",
      "metadata" : {
        "description" : "The size of the Bootstrap Virtual Machine"
      }
    },
    "hyperVGen": {
      "type": "string",
      "metadata": {
        "description": "VM generation image to use"
      },
      "defaultValue": "V2",
      "allowedValues": [
        "V1",
        "V2"
```

```
      ]
    }
  },
  "variables" : {
    "location" : "[resourceGroup().location]",
    "virtualNetworkName" : "[concat(if(not(empty(parameters('vnetBaseName'))),
parameters('vnetBaseName'), parameters('baseName')), '-vnet')]",
    "virtualNetworkID" : "[resourceId('Microsoft.Network/virtualNetworks',
variables('virtualNetworkName'))]",
    "masterSubnetName" : "[concat(if(not(empty(parameters('vnetBaseName'))),
parameters('vnetBaseName'), parameters('baseName')), '-master-subnet')]",
    "masterSubnetRef" : "[concat(variables('virtualNetworkID'), '/subnets/',
variables('masterSubnetName'))]",
    "masterLoadBalancerName" : "[parameters('baseName')]",
    "internalLoadBalancerName" : "[concat(parameters('baseName'), '-internal-lb')]",
    "sshKeyPath" : "/home/core/.ssh/authorized_keys",
    "identityName" : "[concat(parameters('baseName'), '-identity')]",
    "vmName" : "[concat(parameters('baseName'), '-bootstrap')]",
    "nicName" : "[concat(variables('vmName'), '-nic')]",
    "galleryName": "[concat('gallery_', replace(parameters('baseName'), '-', '_'))]",
    "imageName" : "[concat(parameters('baseName'), if(equals(parameters('hyperVGen'), 'V2'), '-
gen2', ''))]",
    "clusterNsgName" : "[concat(if(not(empty(parameters('vnetBaseName'))),
parameters('vnetBaseName'), parameters('baseName')), '-nsg')]",
    "sshPublicIpAddressName" : "[concat(variables('vmName'), '-ssh-pip')]"
  },
  "resources" : [
    {
      "apiVersion" : "2018-12-01",
      "type" : "Microsoft.Network/publicIPAddresses",
      "name" : "[variables('sshPublicIpAddressName')]",
      "location" : "[variables('location')]",
      "sku": {
        "name": "Standard"
      },
      "properties" : {
        "publicIPAllocationMethod" : "Static",
        "dnsSettings" : {
          "domainNameLabel" : "[variables('sshPublicIpAddressName')]"
        }
      }
    },
    {
      "apiVersion" : "2018-06-01",
      "type" : "Microsoft.Network/networkInterfaces",
      "name" : "[variables('nicName')]",
      "location" : "[variables('location')]",
      "dependsOn" : [
        "[resourceId('Microsoft.Network/publicIPAddresses', variables('sshPublicIpAddressName'))]"
      ],
      "properties" : {
        "ipConfigurations" : [
          {
            "name" : "pipConfig",
            "properties" : {
              "privateIPAllocationMethod" : "Dynamic",
```

```
            "publicIPAddress": {
              "id": "[resourceId('Microsoft.Network/publicIPAddresses',
variables('sshPublicIpAddressName'))]"
            },
            "subnet" : {
              "id" : "[variables('masterSubnetRef')]"
            },
            "loadBalancerBackendAddressPools" : [
              {
                "id" : "[concat('/subscriptions/', subscription().subscriptionId, '/resourceGroups/',
resourceGroup().name, '/providers/Microsoft.Network/loadBalancers/',
variables('masterLoadBalancerName'), '/backendAddressPools/',
variables('masterLoadBalancerName'))]"
              },
              {
                "id" : "[concat('/subscriptions/', subscription().subscriptionId, '/resourceGroups/',
resourceGroup().name, '/providers/Microsoft.Network/loadBalancers/',
variables('internalLoadBalancerName'), '/backendAddressPools/internal-lb-backend')]"
              }
            ]
          }
        }
      ]
    }
  },
  {
    "apiVersion" : "2018-06-01",
    "type" : "Microsoft.Compute/virtualMachines",
    "name" : "[variables('vmName')]",
    "location" : "[variables('location')]",
    "identity" : {
      "type" : "userAssigned",
      "userAssignedIdentities" : {
        "[resourceID('Microsoft.ManagedIdentity/userAssignedIdentities/',
variables('identityName'))]" : {}
      }
    },
    "dependsOn" : [
      "[concat('Microsoft.Network/networkInterfaces/', variables('nicName'))]"
    ],
    "properties" : {
      "hardwareProfile" : {
        "vmSize" : "[parameters('bootstrapVMSize')]"
      },
      "osProfile" : {
        "computerName" : "[variables('vmName')]",
        "adminUsername" : "core",
        "adminPassword" : "NotActuallyApplied!",
        "customData" : "[parameters('bootstrapIgnition')]",
        "linuxConfiguration" : {
          "disablePasswordAuthentication" : false
        }
      },
      "storageProfile" : {
        "imageReference": {
          "id": "[resourceId('Microsoft.Compute/galleries/images', variables('galleryName'),
```

```
variables('imageName'))]"
          },
          "osDisk" : {
            "name": "[concat(variables('vmName'),'_OSDisk')]",
            "osType" : "Linux",
            "createOption" : "FromImage",
            "managedDisk": {
              "storageAccountType": "Premium_LRS"
            },
            "diskSizeGB" : 100
          }
        },
        "networkProfile" : {
          "networkInterfaces" : [
            {
              "id" : "[resourceId('Microsoft.Network/networkInterfaces', variables('nicName'))]"
            }
          ]
        }
      }
    },
    {
      "apiVersion" : "2018-06-01",
      "type": "Microsoft.Network/networkSecurityGroups/securityRules",
      "name" : "[concat(variables('clusterNsgName'), '/bootstrap_ssh_in')]",
      "location" : "[variables('location')]",
      "dependsOn" : [
        "[resourceId('Microsoft.Compute/virtualMachines', variables('vmName'))]"
      ],
      "properties": {
        "protocol" : "Tcp",
        "sourcePortRange" : "*",
        "destinationPortRange" : "22",
        "sourceAddressPrefix" : "*",
        "destinationAddressPrefix" : "*",
        "access" : "Allow",
        "priority" : 100,
        "direction" : "Inbound"
      }
    }
  ]
}
```

7.11.17. 在 Azure 中创建 control plane 机器

您必须在 Microsoft Azure 中创建 control plane 机器，供您的集群使用。创建这些机器的一种方法是修改提供的 Azure Resource Manager（ARM）模板。

注意

默认情况下，Microsoft Azure 将 control plane 机器和计算机器放在预先设置的可用区中。您可以为计算节点或 control plane 节点手动设置可用区。要做到这一点，通过在虚拟机资源的 **zones** 参数中指定每个可用区来修改供应商的 Azure Resource Manager (ARM) 模板。

如果不使用提供的 ARM 模板来创建 control plane 机器，您必须检查提供的信息并手动创建基础架构。如果您的集群没有正确初始化，请考虑与安装日志联系红帽支持。

先决条件

- 配置 Azure 帐户。
- 为集群生成 Ignition 配置文件。
- 在 Azure 中创建和配置 VNet 及相关子网。
- 在 Azure 中创建和配置联网及负载均衡器。
- 创建 control plane 和计算角色。
- 创建 bootstrap 机器。

流程

1. 复制 **control plane 机器的 ARM 模板** 一节中的模板，并将它以 **05_masters.json** 保存到集群的安装目录中。此模板描述了集群所需的 control plane 机器。
2. 导出 control plane 机器部署所需的以下变量：

```
$ export MASTER_IGNITION=`cat <installation_directory>/master.ign | base64 | tr -d '\n'`
```

3. 使用 **az** CLI 创建部署：

```
$ az deployment group create -g ${RESOURCE_GROUP} \
  --template-file "<installation_directory>/05_masters.json" \
  --parameters masterIgnition="${MASTER_IGNITION}" \ 1
  --parameters baseName="${INFRA_ID}" \ 2
  --parameters masterVMSize="Standard_D8s_v3" 3
```

1. control plane 节点的 Ignition 内容。
2. 资源名称使用的基本名称；这通常是集群的基础架构 ID。
3. 可选：指定 Control Plane 虚拟机的大小。使用与指定架构兼容的虚拟机大小。如果未定义这个值，则会设置模板中的默认值。

7.11.17.1. control plane 机器的 ARM 模板

您可以使用以下 Azure Resource Manager(ARM)模板来部署 OpenShift Container Platform 集群所需的 control plane 机器：

例 7.89. 05_masters.json ARM 模板

```
{
  "$schema" : "https://schema.management.azure.com/schemas/2015-01-
01/deploymentTemplate.json#",
  "contentVersion" : "1.0.0.0",
  "parameters" : {
```

```
"baseName" : {
  "type" : "string",
  "minLength" : 1,
  "metadata" : {
    "description" : "Base name to be used in resource names (usually the cluster's Infra ID)"
  }
},
"vnetBaseName": {
  "type": "string",
  "defaultValue": "",
  "metadata" : {
    "description" : "The specific customer vnet's base name (optional)"
  }
},
"masterIgnition" : {
  "type" : "string",
  "metadata" : {
    "description" : "Ignition content for the master nodes"
  }
},
"numberOfMasters" : {
  "type" : "int",
  "defaultValue" : 3,
  "minValue" : 2,
  "maxValue" : 30,
  "metadata" : {
    "description" : "Number of OpenShift masters to deploy"
  }
},
"sshKeyData" : {
  "type" : "securestring",
  "defaultValue" : "Unused",
  "metadata" : {
    "description" : "Unused"
  }
},
"privateDNSZoneName" : {
  "type" : "string",
  "defaultValue" : "",
  "metadata" : {
    "description" : "unused"
  }
},
"masterVMSize" : {
  "type" : "string",
  "defaultValue" : "Standard_D8s_v3",
  "metadata" : {
    "description" : "The size of the Master Virtual Machines"
  }
},
"diskSizeGB" : {
  "type" : "int",
  "defaultValue" : 1024,
  "metadata" : {
    "description" : "Size of the Master VM OS disk, in GB"
  }
```

```
    },
    "hyperVGen": {
      "type": "string",
      "metadata": {
        "description": "VM generation image to use"
      },
      "defaultValue": "V2",
      "allowedValues": [
        "V1",
        "V2"
      ]
    }
  },
  "variables" : {
    "location" : "[resourceGroup().location]",
    "virtualNetworkName" : "[concat(if(not(empty(parameters('vnetBaseName'))),
parameters('vnetBaseName'), parameters('baseName')), '-vnet')]",
    "virtualNetworkID" : "[resourceId('Microsoft.Network/virtualNetworks',
variables('virtualNetworkName'))]",
    "masterSubnetName" : "[concat(if(not(empty(parameters('vnetBaseName'))),
parameters('vnetBaseName'), parameters('baseName')), '-master-subnet')]",
    "masterSubnetRef" : "[concat(variables('virtualNetworkID'), '/subnets/',
variables('masterSubnetName'))]",
    "masterLoadBalancerName" : "[parameters('baseName')]",
    "internalLoadBalancerName" : "[concat(parameters('baseName'), '-internal-lb')]",
    "sshKeyPath" : "/home/core/.ssh/authorized_keys",
    "identityName" : "[concat(parameters('baseName'), '-identity')]",
    "galleryName": "[concat('gallery_', replace(parameters('baseName'), '-', '_'))]",
    "imageName" : "[concat(parameters('baseName'), if(equals(parameters('hyperVGen'), 'V2'), '-
gen2', ''))]",
    "copy" : [
      {
        "name" : "vmNames",
        "count" :  "[parameters('numberOfMasters')]",
        "input" : "[concat(parameters('baseName'), '-master-', copyIndex('vmNames'))]"
      }
    ]
  },
  "resources" : [
    {
      "apiVersion" : "2018-06-01",
      "type" : "Microsoft.Network/networkInterfaces",
      "copy" : {
        "name" : "nicCopy",
        "count" : "[length(variables('vmNames'))]"
      },
      "name" : "[concat(variables('vmNames')[copyIndex()], '-nic')]",
      "location" : "[variables('location')]",
      "properties" : {
        "ipConfigurations" : [
          {
            "name" : "pipConfig",
            "properties" : {
              "privateIPAllocationMethod" : "Dynamic",
              "subnet" : {
                "id" : "[variables('masterSubnetRef')]"
```

```
            },
            "loadBalancerBackendAddressPools" : [
              {
                "id" : "[concat('/subscriptions/', subscription().subscriptionId, '/resourceGroups/',
resourceGroup().name, '/providers/Microsoft.Network/loadBalancers/',
variables('masterLoadBalancerName'), '/backendAddressPools/',
variables('masterLoadBalancerName'))]"
              },
              {
                "id" : "[concat('/subscriptions/', subscription().subscriptionId, '/resourceGroups/',
resourceGroup().name, '/providers/Microsoft.Network/loadBalancers/',
variables('internalLoadBalancerName'), '/backendAddressPools/internal-lb-backend')]"
              }
            ]
          }
        }
      ]
    },
    {
      "apiVersion" : "2018-06-01",
      "type" : "Microsoft.Compute/virtualMachines",
      "copy" : {
        "name" : "vmCopy",
        "count" : "[length(variables('vmNames'))]"
      },
      "name" : "[variables('vmNames')[copyIndex()]]",
      "location" : "[variables('location')]",
      "identity" : {
        "type" : "userAssigned",
        "userAssignedIdentities" : {
          "[resourceID('Microsoft.ManagedIdentity/userAssignedIdentities/',
variables('identityName'))]" : {}
        }
      },
      "dependsOn" : [
        "[concat('Microsoft.Network/networkInterfaces/', concat(variables('vmNames')[copyIndex()], '-
nic'))]"
      ],
      "properties" : {
        "hardwareProfile" : {
          "vmSize" : "[parameters('masterVMSize')]"
        },
        "osProfile" : {
          "computerName" : "[variables('vmNames')[copyIndex()]]",
          "adminUsername" : "core",
          "adminPassword" : "NotActuallyApplied!",
          "customData" : "[parameters('masterIgnition')]",
          "linuxConfiguration" : {
            "disablePasswordAuthentication" : false
          }
        },
        "storageProfile" : {
          "imageReference": {
            "id": "[resourceId('Microsoft.Compute/galleries/images', variables('galleryName'),
variables('imageName'))]"
```

```
        },
        "osDisk" : {
          "name": "[concat(variables('vmNames')[copyIndex()], '_OSDisk')]",
          "osType" : "Linux",
          "createOption" : "FromImage",
          "caching": "ReadOnly",
          "writeAcceleratorEnabled": false,
          "managedDisk": {
            "storageAccountType": "Premium_LRS"
          },
          "diskSizeGB" : "[parameters('diskSizeGB')]"
        }
      },
      "networkProfile" : {
        "networkInterfaces" : [
          {
            "id" : "[resourceId('Microsoft.Network/networkInterfaces', concat(variables('vmNames')
[copyIndex()], '-nic'))]",
            "properties": {
              "primary": false
            }
          }
        ]
      }
    }
  ]
}
```

7.11.18. 等待 bootstrap 完成并删除 Azure 中的 bootstrap 资源

在 Microsoft Azure 中创建所有所需的基础架构后，请等待您通过安装程序生成的 Ignition 配置文件所置备的机器上完成 bootstrap 过程。

先决条件

- 配置 Azure 帐户。
- 为集群生成 Ignition 配置文件。
- 在 Azure 中创建和配置 VNet 及相关子网。
- 在 Azure 中创建和配置联网及负载均衡器。
- 创建 control plane 和计算角色。
- 创建 bootstrap 机器。
- 创建 control plane 机器。

流程

1. 进入包含安装程序的目录并运行以下命令：

```
$ ./openshift-install wait-for bootstrap-complete --dir <installation_directory> \ 1
    --log-level info 2
```

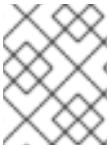
1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不是 **info**。

如果命令退出时没有 **FATAL** 警告，则您的生产环境 control plane 已被初始化。

2. 删除 bootstrap 资源：

```
$ az network nsg rule delete -g ${RESOURCE_GROUP} --nsg-name ${INFRA_ID}-nsg --
name bootstrap_ssh_in
$ az vm stop -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap
$ az vm deallocate -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap
$ az vm delete -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap --yes
$ az disk delete -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap_OSDisk --no-
wait --yes
$ az network nic delete -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap-nic --no-
wait
$ az storage blob delete --account-key ${ACCOUNT_KEY} --account-name
${CLUSTER_NAME}sa --container-name files --name bootstrap.ign
$ az network public-ip delete -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap-
ssh-pip
```



注意

如果没有删除 bootstrap 服务器，因为 API 流量会路由到 bootstrap 服务器，所以安装可能无法成功。

7.11.19. 在 Azure 中创建额外的 worker 机器

您可以通过分散启动各个实例或利用集群外自动化流程（如自动缩放组），在 Microsoft Azure 中为您的集群创建 worker 机器。您还可以利用 OpenShift Container Platform 中的内置集群扩展机制和机器 API。

注意

如果您要安装三节点集群，请跳过这一步。三节点集群包含三个 control plane 机器，它们也可以充当计算机器。

在本例中，您要使用 Azure Resource Manager(ARM)模板手动启动一个实例。通过在 文件中包括类型为 **06_workers.json** 的其他资源，即可启动其他实例。

注意

默认情况下，Microsoft Azure 将 control plane 机器和计算机器放在预先设置的可用区中。您可以为计算节点或 control plane 节点手动设置可用区。要做到这一点，通过在虚拟机资源的 **zones** 参数中指定每个可用区来修改供应商的 ARM 模板。

如果不使用提供的 ARM 模板来创建 control plane 机器，您必须检查提供的信息并手动创建基础架构。如果您的集群没有正确初始化，请考虑与安装日志联系红帽支持。

先决条件

- 配置 Azure 帐户。
- 为集群生成 Ignition 配置文件。
- 在 Azure 中创建和配置 VNet 及相关子网。
- 在 Azure 中创建和配置联网及负载均衡器。
- 创建 control plane 和计算角色。
- 创建 bootstrap 机器。
- 创建 control plane 机器。

流程

1. 复制 **worker 机器的 ARM 模板一节中的模板**，并将它以 **06_workers.json** 保存到集群的安装目录中。此模板描述了集群所需的 worker 机器。
2. 导出 worker 机器部署所需的以下变量：

```
$ export WORKER_IGNITION=`cat <installation_directory>/worker.ign | base64 | tr -d '\n'`
```

3. 使用 **az** CLI 创建部署：

```
$ az deployment group create -g ${RESOURCE_GROUP} \
  --template-file "<installation_directory>/06_workers.json" \
  --parameters workerIgnition="${WORKER_IGNITION}" \ 1
  --parameters baseName="${INFRA_ID}" \ 2
  --parameters nodeVMSize="Standard_D4s_v3" 3
```

1. worker 节点的 Ignition 内容。
2. 资源名称使用的基本名称；这通常是集群的基础架构 ID。
3. 可选：指定计算节点虚拟机的大小。使用与指定架构兼容的虚拟机大小。如果未定义这个值，则会设置模板中的默认值。

7.11.19.1. worker 机器的 ARM 模板

您可以使用以下 Azure Resource Manager(ARM)模板来部署 OpenShift Container Platform 集群所需的 worker 机器：

例 7.90. 06_workers.json ARM template

```
{
  "$schema" : "https://schema.management.azure.com/schemas/2015-01-
01/deploymentTemplate.json#",
  "contentVersion" : "1.0.0.0",
  "parameters" : {
    "baseName" : {
      "type" : "string",
```

```
      "minLength" : 1,
      "metadata" : {
        "description" : "Base name to be used in resource names (usually the cluster's Infra ID)"
      }
    },
    "vnetBaseName": {
      "type": "string",
      "defaultValue": "",
      "metadata" : {
        "description" : "The specific customer vnet's base name (optional)"
      }
    },
    "workerIgnition" : {
      "type" : "string",
      "metadata" : {
        "description" : "Ignition content for the worker nodes"
      }
    },
    "numberOfNodes" : {
      "type" : "int",
      "defaultValue" : 3,
      "minValue" : 2,
      "maxValue" : 30,
      "metadata" : {
        "description" : "Number of OpenShift compute nodes to deploy"
      }
    },
    "sshKeyData" : {
      "type" : "securestring",
      "defaultValue" : "Unused",
      "metadata" : {
        "description" : "Unused"
      }
    },
    "nodeVMSize" : {
      "type" : "string",
      "defaultValue" : "Standard_D4s_v3",
      "metadata" : {
        "description" : "The size of the each Node Virtual Machine"
      }
    },
    "hyperVGen": {
      "type": "string",
      "metadata": {
        "description": "VM generation image to use"
      },
      "defaultValue": "V2",
      "allowedValues": [
        "V1",
        "V2"
      ]
    }
  },
  "variables" : {
    "location" : "[resourceGroup().location]",
    "virtualNetworkName" : "[concat(if(not(empty(parameters('vnetBaseName'))),
```

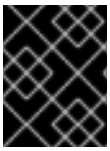
```
parameters('vnetBaseName'), parameters('baseName')), '-vnet')]",
    "virtualNetworkID" : "[resourceId('Microsoft.Network/virtualNetworks',
variables('virtualNetworkName'))]",
    "nodeSubnetName" : "[concat(if(not(empty(parameters('vnetBaseName'))),
parameters('vnetBaseName'), parameters('baseName')), '-worker-subnet')]",
    "nodeSubnetRef" : "[concat(variables('virtualNetworkID'), '/subnets/',
variables('nodeSubnetName'))]",
    "infraLoadBalancerName" : "[parameters('baseName')]",
    "sshKeyPath" : "/home/capi/.ssh/authorized_keys",
    "identityName" : "[concat(parameters('baseName'), '-identity')]",
    "galleryName": "[concat('gallery_', replace(parameters('baseName'), '-', '_'))]",
    "imageName" : "[concat(parameters('baseName'), if(equals(parameters('hyperVGen'), 'V2'), '-
gen2', ''))]",
    "copy" : [
      {
        "name" : "vmNames",
        "count" :  "[parameters('numberOfNodes')]",
        "input" : "[concat(parameters('baseName'), '-worker-', variables('location'), '-',
copyIndex('vmNames', 1))]"
      }
    ]
  },
  "resources" : [
    {
      "apiVersion" : "2019-05-01",
      "name" : "[concat('node', copyIndex())]",
      "type" : "Microsoft.Resources/deployments",
      "copy" : {
        "name" : "nodeCopy",
        "count" : "[length(variables('vmNames'))]"
      },
      "properties" : {
        "mode" : "Incremental",
        "template" : {
          "$schema" : "http://schema.management.azure.com/schemas/2015-01-
01/deploymentTemplate.json#",
          "contentVersion" : "1.0.0.0",
          "resources" : [
            {
              "apiVersion" : "2018-06-01",
              "type" : "Microsoft.Network/networkInterfaces",
              "name" : "[concat(variables('vmNames')[copyIndex()], '-nic')]",
              "location" : "[variables('location')]",
              "properties" : {
                "ipConfigurations" : [
                  {
                    "name" : "pipConfig",
                    "properties" : {
                      "privateIPAllocationMethod" : "Dynamic",
                      "subnet" : {
                        "id" : "[variables('nodeSubnetRef')]"
                      }
                    }
                  }
                ]
              }
```

```
            },
            {
              "apiVersion" : "2018-06-01",
              "type" : "Microsoft.Compute/virtualMachines",
              "name" : "[variables('vmNames')[copyIndex()]]",
              "location" : "[variables('location')]",
              "tags" : {
                "kubernetes.io-cluster-ffranzupi": "owned"
              },
              "identity" : {
                "type" : "userAssigned",
                "userAssignedIdentities" : {
                  "[resourceID('Microsoft.ManagedIdentity/userAssignedIdentities/',
variables('identityName'))]" : {}
                }
              },
              "dependsOn" : [
                "[concat('Microsoft.Network/networkInterfaces/', concat(variables('vmNames')
[copyIndex()], '-nic'))]"
              ],
              "properties" : {
                "hardwareProfile" : {
                  "vmSize" : "[parameters('nodeVMSize')]"
                },
                "osProfile" : {
                  "computerName" : "[variables('vmNames')[copyIndex()]]",
                  "adminUsername" : "capi",
                  "adminPassword" : "NotActuallyApplied!",
                  "customData" : "[parameters('workerIgnition')]",
                  "linuxConfiguration" : {
                    "disablePasswordAuthentication" : false
                  }
                },
                "storageProfile" : {
                  "imageReference": {
                    "id": "[resourceId('Microsoft.Compute/galleries/images', variables('galleryName'),
variables('imageName'))]"
                  },
                  "osDisk" : {
                    "name": "[concat(variables('vmNames')[copyIndex()],'_OSDisk')]",
                    "osType" : "Linux",
                    "createOption" : "FromImage",
                    "managedDisk": {
                      "storageAccountType": "Premium_LRS"
                    },
                    "diskSizeGB": 128
                  }
                },
                "networkProfile" : {
                  "networkInterfaces" : [
                    {
                      "id" : "[resourceId('Microsoft.Network/networkInterfaces',
concat(variables('vmNames')[copyIndex()], '-nic'))]",
                      "properties": {
                        "primary": true
                      }
```

```
            }
          ]
        }
      }
    }
  ]
 }
}
```

7.11.20. 安装 OpenShift CLI

您可以安装 OpenShift CLI(**oc**)来使用命令行界面与 OpenShift Container Platform 进行交互。您可以在 Linux、Windows 或 macOS 上安装 **oc**。



重要

如果安装了旧版本的 **oc**，则可能无法使用 OpenShift Container Platform 4.16 中的所有命令。下载并安装新版本的 **oc**。

在 Linux 上安装 OpenShift CLI
您可以按照以下流程在 Linux 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **产品变体** 下拉列表中选择架构。
3. 从 **版本** 下拉列表中选择适当的版本。
4. 点 **OpenShift v4.16 Linux Client** 条目旁的 **Download Now** 来保存文件。
5. 解包存档：

```
$ tar xvf <file>
```

6. 将 **oc** 二进制文件放到 **PATH 中的目录中**。
   要查看您的 **PATH**，请执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

在 Windows 上安装 OpenShift CLI
您可以按照以下流程在 Windows 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 Windows Client** 条目旁的 **Download Now** 来保存文件。
4. 使用 ZIP 程序解压存档。
5. 将 **oc** 二进制文件移到 **PATH 中的目录中**。
   要查看您的 **PATH**，请打开命令提示并执行以下命令：

```
C:\> path
```

验证

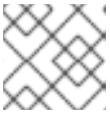
- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

在 macOS 上安装 OpenShift CLI
您可以按照以下流程在 macOS 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 macOS Client** 条目旁的 **Download Now** 来保存文件。



注意

对于 macOS arm64，请选择 **OpenShift v4.16 macOS arm64 Client** 条目。

4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
   要查看您的 **PATH**，请打开终端并执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

7.11.21. 使用 CLI 登录集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

7.11.22. 批准机器的证书签名请求

当您将机器添加到集群时，会为您添加的每台机器生成两个待处理证书签名请求(CSR)。您必须确认这些 CSR 已获得批准，或根据需要自行批准。必须首先批准客户端请求，然后批准服务器请求。

先决条件

- 您已将机器添加到集群中。

流程

1. 确认集群可以识别这些机器：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0 Ready    master  63m  v1.29.4
master-1 Ready    master  63m  v1.29.4
master-2 Ready    master  64m  v1.29.4
```

输出中列出了您创建的所有机器。

注意

在有些 CSR 被批准前，前面的输出可能不包括计算节点（也称为 worker 节点）。

2. 检查待处理的 CSR，并确保添加到集群中的每台机器都有 **Pending** 或 **Approved** 状态的客户端请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE    REQUESTOR                                                   CONDITION
csr-8b2br   15m    system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
csr-8vnps   15m    system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
...
```

在本例中，两台机器加入集群。您可能会在列表中看到更多已批准的 CSR。

3. 如果 CSR 没有获得批准，在您添加的机器的所有待处理 CSR 都处于 **Pending 状态** 后，请批准集群机器的 CSR：

注意

由于 CSR 会自动轮转，因此请在将机器添加到集群后一小时内批准您的 CSR。如果没有在一小时内批准它们，证书将会轮转，每个节点会存在多个证书。您必须批准所有这些证书。批准客户端 CSR 后，Kubelet 为服务证书创建一个二级 CSR，这需要手动批准。然后，如果 Kubelet 请求具有相同参数的新证书，则后续提供证书续订请求由 **machine-approver** 自动批准。

注意

对于在未启用机器 API 的平台上运行的集群，如裸机和其他用户置备的基础架构，您必须实施一种方法来自动批准 kubelet 提供证书请求(CSR)。如果没有批准请求，则 **oc exec**、oc**rsh** 和 **oc logs** 命令将无法成功，因为 API 服务器连接到 kubelet 时需要服务证书。与 Kubelet 端点联系的任何操作都需要此证书批准。该方法必须监视新的 CSR，确认 CSR 由 system: **node** 或 **system:admin** 组中的 **node-bootstrapper** 服务帐户提交，并确认节点的身份。

- 要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```

注意

在有些 CSR 被批准前，一些 Operator 可能无法使用。

4. 现在，您的客户端请求已被批准，您必须查看添加到集群中的每台机器的服务器请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE     REQUESTOR                                                   CONDITION
csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal       Pending
csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal       Pending
...
```

5. 如果剩余的 CSR 没有被批准，且处于 **Pending** 状态，请批准集群机器的 CSR：

- 要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs oc adm certificate approve
```

6. 批准所有客户端和服务器 CSR 后，机器将 处于 **Ready 状态**。运行以下命令验证：

```
$ oc get nodes
```

输出示例

```
NAME      STATUS   ROLES   AGE  VERSION
master-0  Ready    master  73m  v1.29.4
master-1  Ready    master  73m  v1.29.4
master-2  Ready    master  74m  v1.29.4
worker-0  Ready    worker  11m  v1.29.4
worker-1  Ready    worker  11m  v1.29.4
```

注意

批准服务器 CSR 后可能需要几分钟时间让机器过渡到 **Ready 状态**。

其他信息

- 如需有关 CSR 的更多信息，请参阅 证书签名请求。

7.11.23. 添加 Ingress DNS 记录

如果在创建 Kubernetes 清单并生成 Ignition 配置时删除了 DNS 区配置，您必须手动创建指向 Ingress 负载均衡器的 DNS 记录。您可以创建一个通配符 ***.apps.{baseDomain}.** 或特定的记录。您可以根据要求使用 A、CNAME 和其他记录。

先决条件

- 已使用您置备的基础架构在 Microsoft Azure 上安装了 OpenShift Container Platform 集群。
- 安装 OpenShift CLI（**oc**）。
- 安装或更新 Azure CLI。

流程

1. 确认 Ingress 路由器已创建了负载均衡器并填充 **EXTERNAL-IP** 字段：

```
$ oc -n openshift-ingress get service router-default
```

输出示例

```
NAME          TYPE           CLUSTER-IP     EXTERNAL-IP     PORT(S)                      AGE
router-default   LoadBalancer   172.30.20.10   35.130.120.110
80:32288/TCP,443:31215/TCP   20
```

2. 将 Ingress 路由器 IP 导出作为变量：

```
$ export PUBLIC_IP_ROUTER=`oc -n openshift-ingress get service router-default --no-
headers | awk '{print $4}'`
```

3. 在公共 DNS 区域中添加 ***.apps** 记录。

   a. 如果您要将此集群添加到新的公共区，请运行：

```
$ az network dns record-set a add-record -g ${BASE_DOMAIN_RESOURCE_GROUP} -
z ${CLUSTER_NAME}.${BASE_DOMAIN} -n *.apps -a ${PUBLIC_IP_ROUTER} --ttl 300
```

   b. 如果您要将此集群添加到已经存在的公共区中，请运行：

```
$ az network dns record-set a add-record -g ${BASE_DOMAIN_RESOURCE_GROUP} -
z ${BASE_DOMAIN} -n *.apps.${CLUSTER_NAME} -a ${PUBLIC_IP_ROUTER} --ttl 300
```

4. 在私有 DNS 区域中添加 ***.apps** 记录：

   a. 使用以下命令创建 ***.apps** 记录：

```
$ az network private-dns record-set a create -g ${RESOURCE_GROUP} -z
${CLUSTER_NAME}.${BASE_DOMAIN} -n *.apps --ttl 300
```

   b. 使用以下命令在专用 DNS 区域中添加 ***.apps** 记录：

```
$ az network private-dns record-set a add-record -g ${RESOURCE_GROUP} -z
${CLUSTER_NAME}.${BASE_DOMAIN} -n *.apps -a ${PUBLIC_IP_ROUTER}
```

如果需要添加特定域而不使用通配符，可以为集群的每个当前路由创建条目：

```
$ oc get --all-namespaces -o jsonpath='{range .items[*]}{range .status.ingress[*]}{.host}{"\n"}{end}
{end}' routes
```

输出示例

```
oauth-openshift.apps.cluster.basedomain.com
console-openshift-console.apps.cluster.basedomain.com
downloads-openshift-console.apps.cluster.basedomain.com
alertmanager-main-openshift-monitoring.apps.cluster.basedomain.com
prometheus-k8s-openshift-monitoring.apps.cluster.basedomain.com
```

7.11.24. 在用户置备的基础架构上完成 Azure 安装

在 Microsoft Azure 用户置备的基础架构上启动 OpenShift Container Platform 安装后，您可以监控集群事件，直到集群就绪可用。

先决条件

- 在用户置备的 Azure 基础架构上为 OpenShift Container Platform 集群部署 bootstrap 机器。
- 安装 **oc** CLI 并登录。

流程

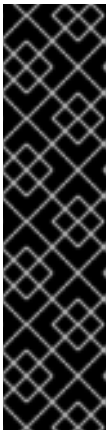
- 完成集群安装：

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

输出示例

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。



重要

- 安装程序生成的 Ignition 配置文件包含 24 小时后过期的证书，然后在该时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

7.11.25. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于 远程健康监控

7.12. 在受限网络中的 AZURE 上安装集群

在 OpenShift Container Platform 版本 4.16 中，您可以通过在现有 Azure Virtual Network (VNet)上创建安装发行内容的内部镜像在受限网络中的 Microsoft Azure 上安装集群。

重要

您可以使用镜像安装发行内容安装 OpenShift Container Platform 集群，但集群需要访问互联网才能使用 Azure API。

7.12.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读选择集群安装方法并为用户准备它的文档。
- 您已将 Azure 帐户配置为托管集群，并决定要将集群部署到的已测试和验证的区域。
- 您已 将断开连接的安装的镜像镜像 到 registry，并获取了 OpenShift Container Platform 版本的 **imageContentSources** 数据。

  重要

  由于安装介质位于镜像主机上，因此您可以使用该计算机完成所有安装步骤。

- Azure 中有一个现有的 VNet。在使用安装程序置备的基础架构的受限网络中安装集群时，您无法使用安装程序置备的 VNet。您必须使用用户置备的 VNet 来满足以下要求之一：
  - VNet 包含镜像 registry
  - VNet 有防火墙规则或对等连接来访问其他位置托管的镜像 registry
- 如果使用防火墙，将其配置为允许集群需要访问的站点。
- 如果您使用客户管理的加密密钥，准备了用于加密的 Azure 环境。

7.12.2. 关于在受限网络中安装

在 OpenShift Container Platform 4.16 中，可以执行不需要有效的互联网连接来获取软件组件的安装。受限网络安装可以使用安装程序置备的基础架构或用户置备的基础架构完成，具体取决于您要安装集群的云平台。

如果您选择在云平台中执行受限网络安装，您仍需要访问其云 API。有些云功能，比如 Amazon Web Service 的 Route 53 DNS 和 IAM 服务，需要访问互联网。根据您的网络，在裸机硬件、Nutanix 或 VMware vSphere 上安装可能需要较少的互联网访问。

要完成受限网络安装，您必须创建一个 registry，以镜像 OpenShift 镜像 registry 的内容并包含安装介质。您可以在镜像主机上创建此 registry，该主机可同时访问互联网和您的封闭网络，也可以使用满足您的限制条件的其他方法。

7.12.2.1. 其他限制

受限网络中的集群有以下额外限制和限制：

- **ClusterVersion** 状态包含一个 **Unable to retrieve available updates** 错误。
- 默认情况下，您无法使用 Developer Catalog 的内容，因为您无法访问所需的镜像流标签。

7.12.2.2. 用户定义的出站路由

在 OpenShift Container Platform 中，您可以选择自己的出站路由来连接到互联网。这可让您跳过创建公共 IP 地址和公共负载均衡器的步骤。

您可在安装集群前修改 **install-config.yaml** 文件中的参数来配置用户定义的路由。安装集群时，需要一个已存在的 VNet 来使用出站路由，安装程序不负责配置它。

当将集群配置为使用用户定义的路由时，安装程序不会创建以下资源：

- 用于访问互联网的出站规则。
- 公共负载均衡器的公共 IP。
- Kubernetes Service 对象，为出站请求将集群机器添加到公共负载均衡器中。

在设置用户定义的路由前，您必须确保以下项目可用：

- 出口到互联网可以拉取容器镜像，除非使用 OpenShift image registry 镜像。
- 集群可以访问 Azure API。
- 配置了各种允许列表端点。您可以在 *配置防火墙* 部分引用这些端点。

支持一些已存在的网络设置，使用用户定义的路由访问互联网。

使用 Azure Firewall 的受限集群
您可以使用 Azure Firewall 来限制用于安装 OpenShift Container Platform 集群的虚拟网络 (VNet) 的出站路由。如需更多信息，请参阅使用 Azure Firewall 提供用户定义的路由。您可以使用 Azure Firewall 的 VNet 并配置用户定义的路由，在受限网络中创建 OpenShift Container Platform 集群。

重要

如果使用 Azure Firewall 来限制互联网访问，您必须在 **install-config.yaml** 文件中将 **publish** 字段设置为 **Internal**。这是因为 Azure Firewall 无法用于 Azure 公共负载均衡器。

7.12.3. 关于为 OpenShift Container Platform 集群重复使用 VNet

在 OpenShift Container Platform 4.16 中，您可以在 Microsoft Azure 中将集群部署到现有的 Azure Virtual Network(VNet)中。如果您这样做，还必须在 VNet 和路由规则中使用现有子网。

通过将 OpenShift Container Platform 部署到现有的 Azure VNet 中，您可以避免新帐户中的服务限制，或者更容易地利用公司所设置的操作限制。如果您无法获得创建 VNet 所需的基础架构创建权限，则可以使用这个选项。

7.12.3.1. 使用 VNet 的要求

当使用现有 VNet 部署集群时，必须在安装集群前执行额外的网络配置。在安装程序置备的基础架构集群中，安装程序通常会创建以下组件，但在安装到现有 VNet 时不会创建它们：

- 子网
- 路由表
- VNets
- 网络安全组

注意

安装程序要求您使用由云提供的 DNS 服务器。不支持使用自定义 DNS 服务器，并导致安装失败。

如果使用自定义 VNet，您必须正确配置它及其子网，供安装程序和集群使用。安装程序不能为集群分配要使用的网络范围，为子网设置路由表，或者设置类似 DHCP 的 VNet 选项，因此您必须在安装集群前这样做。

集群必须能够访问包含现有 VNet 和子网的资源组。虽然集群创建的所有资源都放在它创建的单独资源组中，但有些网络资源则从单独的组中使用。有些集群 Operator 必须能够访问这两个资源组中的资源。例如，Machine API 控制器会为它创建的虚拟机附加 NICS，以便从网络资源组中进行子网。

您的 VNet 必须满足以下特征：

- VNet 的 CIDR 块必须包含 **Networking.MachineCIDR** 范围，它是集群机器的 IP 地址池。
- VNet 及其子网必须属于同一资源组，子网必须配置为使用 Azure 分配的 DHCP IP 地址，而不是静态 IP 地址。

您必须在 VNet 中提供两个子网，一个用于 control plane 机器，一个用于计算机器。因为 Azure 在您指定的区域内的不同可用区中分发机器，所以集群将默认具有高可用性。

注意

默认情况下，如果您在 **install-config.yaml** 文件中指定可用区，安装程序会在一个区（region）内的这些可用区间分发 control plane 机器和计算机器。要确保集群的高可用性，请选择至少含有三个可用区的区域。如果您的区域包含的可用区少于三个，安装程序将在可用区中放置多台 control plane 机器。

为确保您提供的子网适合，安装程序会确认以下数据：

- 所有指定的子网都存在。

- 有两个专用子网，一个用于 control plane 机器，一个用于计算机器。
- 子网 CIDR 属于您指定的机器 CIDR。机器不会在您不为其提供私有子网的可用区中置备。如果需要，安装程序会创建管理 control plane 和 worker 节点的公共负载均衡器，Azure 会为其分配一个公共 IP 地址。

注意

如果您销毁了使用现有 VNet 的集群，则不会删除 VNet。

7.12.3.1.1. 网络安全组要求

托管 compute 和 control plane 机器的子网的网络安全组需要特定的访问权限，以确保集群通信正确。您必须创建规则以允许访问所需的集群通信端口。

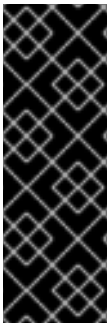
重要

在安装集群前，必须先设置网络安全组规则。如果您试图在没有所需访问权限的情况下安装集群，安装程序无法访问 Azure API，安装会失败。

表 7.30. 所需端口

port	描述	Control plane（控制平面）	Compute
80	允许 HTTP 流量		x
443	允许 HTTPS 流量		x
6443	允许与 control plane 机器通信	x	
22623	允许内部与机器配置服务器通信以用于置备机器	x	
*	允许连接到 Azure API。您必须将 Destination Service Tag 设置为 **AzureCloud**。[1]	x	x
*	拒绝连接到互联网。您必须将目标服务标签设置为 **Internet**。[1]	x	x

1. 如果使用 Azure Firewall 来限制互联网访问，您可以将 Azure Firewall 配置为允许 Azure API。不需要网络安全组规则。



重要

目前，不支持阻止或限制机器配置服务器端点。机器配置服务器必须公开给网络，以便新置备的机器没有现有配置或状态，才能获取其配置。在这个模型中，信任的根是证书签名请求 (CSR) 端点，即 kubelet 发送其证书签名请求以批准加入集群。因此，机器配置不应用于分发敏感信息，如 secret 和证书。

为确保机器配置服务器端点，端口 22623 和 22624 在裸机场景中是安全的，客户必须配置正确的网络策略。

由于集群组件不会修改 Kubernetes 控制器更新的用户提供的网络安全组，因此为 Kubernetes 控制器在不影响其余环境的情况下创建一个伪网络安全组。

其他资源

- 关于 OpenShift SDN 网络插件
- 配置防火墙

7.12.3.2. 权限划分

从 OpenShift Container Platform 4.3 开始，您不需要安装程序置备的基础架构集群部署所需的所有权限。这与您所在机构可能拥有的权限划分类似：一些个人可以在您的云中创建不同的资源。例如，您可以创建特定于应用程序的对象，如实例、存储和负载均衡器，但不能创建与网络相关的组件，如 VNets、子网或入站规则。

您在创建集群时使用的 Azure 凭证不需要 VNets 和核心网络组件（如子网、路由表、互联网网关、NAT 和 VPN）所需的网络权限。您仍然需要获取集群中的机器需要的应用程序资源的权限，如负载均衡器、安全组、存储帐户和节点。

7.12.3.3. 集群间隔离

因为集群无法修改现有子网中的网络安全组，所以无法在 VNet 中相互隔离集群。

7.12.4. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来获得用来安装集群的镜像。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

7.12.5. 为集群节点 SSH 访问生成密钥对

在 OpenShift Container Platform 安装过程中，您可以为安装程序提供 SSH 公钥。密钥通过它们的 Ignition 配置文件传递给 Red Hat Enterprise Linux CoreOS(RHCOS)节点，用于验证对节点的 SSH 访问。密钥添加到每个节点上 **core** 用户的 **~/.ssh/authorized_keys** 列表中，这将启用免密码身份验证。

将密钥传递给节点后，您可以使用密钥对作为用户 **核心** 通过 SSH 连接到 RHCOS 节点。若要通过 SSH 访问节点，必须由 SSH 为您的本地用户管理私钥身份。

如果要通过 SSH 连接到集群节点来执行安装调试或灾难恢复，则必须在安装过程中提供 SSH 公钥。**./openshift-install gather** 命令还需要在集群节点上设置 SSH 公钥。

重要

不要在生产环境中跳过这个过程，在生产环境中需要灾难恢复和调试。

注意

您必须使用本地密钥，而不是使用特定平台方法配置 的密钥，如 AWS 密钥对。

流程

1. 如果您在本地计算机上没有可用于在集群节点上进行身份验证的现有 SSH 密钥对，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_ed25519**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

注意

如果您计划在 **x86_64**、**ppc64le** 和 **s390x** 架构上安装使用 RHEL 加密库（这些加密库已提交给 NIST 用于 FIPS 140-2/140-3 验证）的 OpenShift Container Platform 集群，则不要创建使用 **ed25519** 算法的密钥。相反，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 查看公共 SSH 密钥：

```
$ cat <path>/<file_name>.pub
```

例如，运行以下命令来查看 **~/.ssh/id_ed25519.pub** 公钥：

```
$ cat ~/.ssh/id_ed25519.pub
```

3. 将 SSH 私钥身份添加到本地用户的 SSH 代理（如果尚未添加）。在集群节点上，或者要使用 **./openshift-install gather** 命令，需要对该密钥进行 SSH 代理管理，才能在集群节点上进行免密码 SSH 身份验证。

注意

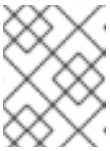
在某些发行版中，自动管理默认 SSH 私钥身份，如 **~/.ssh/id_rsa** 和 **~/.ssh/id_dsa**。

a. 如果 **ssh-agent** 进程尚未为您的本地用户运行，请将其作为后台任务启动：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```



注意

如果集群处于 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

4. 将 SSH 私钥添加到 **ssh-agent** ：

```
$ ssh-add <path>/<file_name> 1
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_ed25519.pub**

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

后续步骤

- 安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

7.12.6. 创建安装配置文件

您可以自定义在 Microsoft Azure 上安装的 OpenShift Container Platform 集群。

先决条件

- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。对于受限网络安装，这些文件位于您的镜像主机上。
- 您有创建镜像 registry 期间生成的 **imageContentSources** 值。
- 您已获取了镜像 registry 的证书内容。
- 您已检索了 Red Hat Enterprise Linux CoreOS (RHCOS) 镜像，并将其上传到可访问的位置。
- 您有一个 Azure 订阅 ID 和租户 ID。
- 如果要使用服务主体安装集群，则有其应用程序 ID 和密码。
- 如果您要使用系统分配的受管身份安装集群，需要在您要从其中运行安装程序的虚拟机上启用它。
- 如果您要使用用户分配的受管身份安装集群，需要满足以下先决条件：
  - 您有它的客户端 ID。
  - 您已将其分配给您要从其运行安装程序的虚拟机。

流程

1. 可选：如果您之前在这个计算机上运行安装程序，并希望使用替代的服务主体或受管身份，请进入 ~/**.azure**/ 目录并删除 **osServicePrincipal.json** 配置文件。
删除此文件可防止安装程序自动重复使用之前安装中的订阅和验证值。

2. 创建 **install-config.yaml** 文件。

   a. 进入包含安装程序的目录并运行以下命令：

   ```
   $ ./openshift-install create install-config --dir <installation_directory> 1
   ```

   1 对于 **<installation_directory>**，请指定要存储安装程序创建的文件的目录名称。

   在指定目录时：

   - 验证该目录是否具有**执行**权限。在安装目录中运行 Terraform 二进制文件需要这个权限。
   - 使用空目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

   b. 在提示符处，提供云的配置详情：

      i. 可选： 选择用于访问集群机器的 SSH 密钥。

      注意

      对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

      ii. 选择 **azure** 作为目标平台。
      如果安装程序无法找到之前安装中的 **osServicePrincipal.json** 配置文件，会提示您输入 Azure 订阅和验证值。

      iii. 为您的订阅输入以下 Azure 参数值：

      - **Azure subscription id**：输入用于集群的订阅 ID。
      - **Azure 租户 id**：输入租户 ID。

      iv. 根据您用来部署集群的 Azure 身份，在提示输入 **azure 服务主体客户端 id**时执行以下操作之一：

      - 如果您使用服务主体，请输入其应用程序 ID。
      - 如果您使用系统分配的受管身份，请将此值设为空白。
      - 如果您使用用户分配的受管身份，请指定其客户端 ID。

      v. 根据您用来部署集群的 Azure 身份，在提示输入 **azure 服务主体客户端 secret**时执行以下操作之一：

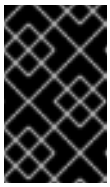
      - 如果您使用服务主体，请输入其密码。
      - 如果您使用系统分配的受管身份，请将此值设为空白。

- 如果您使用用户分配的受管身份，请将此值设为空白。

vi. 选择要将集群部署到的区域。

vii. 选择集群要部署到的基域。基域与您为集群创建的 Azure DNS 区对应。

viii. 为集群输入一个描述性名称。



重要

所有通过公共端点提供的 Azure 资源均存在资源名称的限制，您无法创建使用某些名称的资源。如需 Azure 限制词语列表，请参阅 Azure 文档中的解决保留资源名称错误。

ix. 粘贴 Red Hat OpenShift Cluster Manager 中的 pull secret 。

3. 编辑 **install-config.yaml** 文件，以提供在受限网络中安装所需的额外信息。

a. 更新 **pullSecret** 值，使其包含 registry 的身份验证信息：

```
pullSecret: '{"auths":{"<mirror_host_name>:5000": {"auth": "<credentials>","email":
"you@example.com"}}}'
```

对于 **<mirror_host_name>，请指定** 您在镜像 registry 证书中指定的 registry 域名 **；对于 <credentials>，** 请指定您的镜像 registry 的 base64 编码用户名和密码。

b. 添加 **additionalTrustBundle** 参数和值。

```
additionalTrustBundle: |
 -----BEGIN CERTIFICATE-----

ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
 -----END CERTIFICATE-----
```

该值必须是您用于镜像 registry 的证书文件内容。证书文件可以是现有的可信证书颁发机构，也可以是您为镜像 registry 生成的自签名证书。

c. 在 **platform.azure** 字段中定义 VNet 的网络和子网，以安装集群：

```
networkResourceGroupName: <vnet_resource_group> 1
virtualNetwork: <vnet> 2
controlPlaneSubnet: <control_plane_subnet> 3
computeSubnet: <compute_subnet> 4
```

1. 将 **<vnet_resource_group>** 替换为包含现有虚拟网络 (VNet)的资源组名称。
2. 将 **<vnet>** 替换为现有的虚拟网络名称。
3. 将 **<control_plane_subnet>** 替换为部署 control plane 机器的现有子网名称。
4. 将 **<compute_subnet>** 替换为部署计算机器的现有子网名称。

d. 添加镜像内容资源，类似于以下 YAML 摘录：

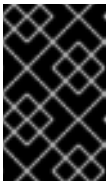
```
imageContentSources:
- mirrors:
  - <mirror_host_name>:5000/<repo_name>/release
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <mirror_host_name>:5000/<repo_name>/release
  source: registry.redhat.io/ocp/release
```

对于这些值，请使用您在创建镜像 registry 时记录的 **imageContentSources**。

e. 可选：将发布策略设置为 **Internal**：

```
publish: Internal
```

通过设置这个选项，您可以创建一个内部 Ingress Controller 和一个私有负载均衡器。

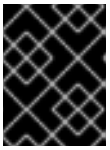


重要

Azure Firewall 无法与 Azure 公共负载均衡器无缝工作。因此，当使用 Azure Firewall 来限制互联网访问时，**install-config.yaml** 中的 **publish** 字段应设置为 **Internal**。

4. 对您需要的 **install-config.yaml** 文件进行任何其他修改。
有关参数的更多信息，请参阅"安装配置参数"。

5. 备份 **install-config.yaml** 文件，以便您可以使用它安装多个集群。



重要

install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用该文件，您必须立即备份该文件。

在以前的版本中，安装程序会创建一个 **osServicePrincipal.json** 配置文件，并将此文件存储在计算机上的 **~/.azure/** 目录中。这样可确保安装程序在目标平台上创建 OpenShift Container Platform 集群时可以加载配置集。

其他资源

- Azure 的安装配置参数

7.12.6.1. 集群安装的最低资源要求

每台集群机器都必须满足以下最低要求：

表 7.31. 最低资源要求

机器	操作系统	vCPU [1]	虚拟内存	Storage	每秒输入/输出 (IOPS) [2]
bootstrap	RHCOS	4	16 GB	100 GB	300

机器	操作系统	vCPU [1]	虚拟内存	Storage	每秒输入/输出 (IOPS) [2]
Control plane（控制平面）	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS、RHEL 8.6 及更新版本 [3]	2	8 GB	100 GB	300

1. 当未启用并发多线程(SMT)或超线程时，一个 vCPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比例：（每个内核数的线程）× sockets = vCPU。

2. OpenShift Container Platform 和 Kubernetes 对磁盘性能非常敏感，建议使用更快的存储速度，特别是 control plane 节点上需要 10 ms p99 fsync 持续时间的 etcd。请注意，在许多云平台上，存储大小和 IOPS 可一起扩展，因此您可能需要过度分配存储卷来获取足够的性能。

3. 与所有用户置备的安装一样，如果您选择在集群中使用 RHEL 计算机器，则负责所有操作系统生命周期管理和维护，包括执行系统更新、应用补丁和完成所有其他必要的任务。RHEL 7 计算机器的使用已弃用，并已在 OpenShift Container Platform 4.10 及更新的版本中删除。

注意

从 OpenShift Container Platform 版本 4.13 开始，RHCOS 基于 RHEL 版本 9.2，它更新了微架构要求。以下列表包含每个架构需要的最小指令集架构 (ISA)：

- x86-64 体系结构需要 x86-64-v2 ISA
- ARM64 架构需要 ARMv8.0-A ISA
- IBM Power 架构需要 Power 9 ISA
- s390x 架构需要 z14 ISA

如需更多信息，请参阅 RHEL 架构。

重要

您需要使用将 **PremiumIO** 参数设置为 **true** 的 Azure 虚拟机。

如果平台的实例类型满足集群机器的最低要求，则 OpenShift Container Platform 支持使用它。

7.12.6.2. 为 Azure 测试的实例类型

以下 Microsoft Azure 实例类型已经 OpenShift Container Platform 测试。

例 7.91. 基于 64 位 x86 架构的机器类型

- **c4.***
- **c5.***

- **c5a.***
- **i3.***
- **m4.***
- **m5.***
- **m5a.***
- **m6a.***
- **m6i.***
- **r4.***
- **r5.***
- **r5a.***
- **r6i.***
- **t3.***
- **t3a.***

7.12.6.3. 在 64 位 ARM 基础架构上为 Azure 测试的实例类型

以下 Microsoft Azure ARM64 实例类型已使用 OpenShift Container Platform 测试。

例 7.92. 基于 64 位 ARM 架构的机器类型

- **c6g.***
- **m6g.***

7.12.6.4. 为 Azure 虚拟机启用可信启动

在 Azure 上安装集群时，您可以启用两个可信启动功能： 安全引导 和虚拟化可信平台模块。

请参阅 Azure 文档中有关 虚拟机大小 的信息，以了解虚拟机支持这些功能的大小。

重要

可信启动只是一个技术预览功能。技术预览功能不受红帽产品服务等级协议（SLA）支持，且功能可能并不完整。红帽不推荐在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。

有关红帽技术预览功能支持范围的更多信息，请参阅技术预览功能支持范围。

先决条件

- 您已创建了 **install-config.yaml** 文件。

流程

- 在部署集群前，使用文本编辑器编辑 **install-config.yaml** 文件并添加以下小节：

```
controlPlane: 1
  platform:
    azure:
      settings:
        securityType: TrustedLaunch 2
        trustedLaunch:
          uefiSettings:
            secureBoot: Enabled 3
            virtualizedTrustedPlatformModule: Enabled 4
```

1. 指定 **controlPlane.platform.azure** 或 **compute.platform.azure**，以分别在 control plane 或计算节点上启用可信启动。指定 **platform.azure.defaultMachinePlatform**，以便在所有节点上启用可信启动。
2. 启用可信启动功能。
3. 启用安全引导。如需更多信息，请参阅 Azure 文档中有关安全引导的内容。
4. 启用虚拟化可信平台模块。如需更多信息，请参阅 Azure 文档中有关 虚拟化可信平台模块的文档。

7.12.6.5. 启用机密虚拟机

您可在安装集群前启用机密虚拟机。您可以为计算节点、control plane 节点或所有节点启用机密虚拟机。

重要

使用机密虚拟机只是一个技术预览功能。技术预览功能不受红帽产品服务等级协议（SLA）支持，且功能可能并不完整。红帽不推荐在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。

有关红帽技术预览功能支持范围的更多信息，请参阅技术预览功能支持范围。

您可以使用带有以下虚拟机大小的机密虚拟机：

- DCasv5-series
- DCadsv5-series
- ECasv5-series
- ECadsv5-series

重要

64 位 ARM 架构目前不支持机密虚拟机。

先决条件

- 您已创建了 **install-config.yaml** 文件。

流程

- 在部署集群前，使用文本编辑器编辑 **install-config.yaml** 文件并添加以下小节：

```
controlPlane: 1
  platform:
    azure:
      settings:
        securityType: ConfidentialVM 2
        confidentialVM:
          uefiSettings:
            secureBoot: Enabled 3
            virtualizedTrustedPlatformModule: Enabled 4
      osDisk:
        securityProfile:
          securityEncryptionType: VMGuestStateOnly 5
```

1. 指定 **controlPlane.platform.azure** 或 **compute.platform.azure**，以分别在 control plane 或计算节点上部署机密虚拟机。指定 **platform.azure.defaultMachinePlatform** 以在所有节点上部署机密虚拟机。
2. 启用机密虚拟机。
3. 启用安全引导。如需更多信息，请参阅 Azure 文档中有关安全引导的内容。
4. 启用虚拟化可信平台模块。如需更多信息，请参阅 Azure 文档中有关 虚拟化可信平台模块的文档。
5. 指定 **VMGuestStateOnly** 来加密虚拟机客户端状态。

7.12.6.6. Azure 的自定义 install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多详情，或修改所需参数的值。

重要

此示例 YAML 文件仅供参考。您必须使用安装程序来获取 **install-config.yaml** 文件，并进行修改。

```
apiVersion: v1
baseDomain: example.com 1
controlPlane: 2
  hyperthreading: Enabled 3 4
  name: master
  platform:
    azure:
      encryptionAtHost: true
      ultraSSDCapability: Enabled
      osDisk:
```

```
        diskSizeGB: 1024 5
        diskType: Premium_LRS
        diskEncryptionSet:
          resourceGroup: disk_encryption_set_resource_group
          name: disk_encryption_set_name
          subscriptionId: secondary_subscription_id
      osImage:
        publisher: example_publisher_name
        offer: example_image_offer
        sku: example_offer_sku
        version: example_image_version
      type: Standard_D8s_v3
  replicas: 3
compute: 6
- hyperthreading: Enabled 7
  name: worker
  platform:
    azure:
      ultraSSDCapability: Enabled
      type: Standard_D2s_v3
      encryptionAtHost: true
      osDisk:
        diskSizeGB: 512 8
        diskType: Standard_LRS
        diskEncryptionSet:
          resourceGroup: disk_encryption_set_resource_group
          name: disk_encryption_set_name
          subscriptionId: secondary_subscription_id
      osImage:
        publisher: example_publisher_name
        offer: example_image_offer
        sku: example_offer_sku
        version: example_image_version
      zones: 9
      - "1"
      - "2"
      - "3"
  replicas: 5
metadata:
  name: test-cluster 10
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 11
  serviceNetwork:
  - 172.30.0.0/16
platform:
  azure:
    defaultMachinePlatform:
      osImage: 12
        publisher: example_publisher_name
        offer: example_image_offer
```

```
        sku: example_offer_sku
        version: example_image_version
      ultraSSDCapability: Enabled
    baseDomainResourceGroupName: resource_group 13
    region: centralus 14
    resourceGroupName: existing_resource_group 15
    networkResourceGroupName: vnet_resource_group 16
    virtualNetwork: vnet 17
    controlPlaneSubnet: control_plane_subnet 18
    computeSubnet: compute_subnet 19
    outboundType: UserDefinedRouting 20
    cloudName: AzurePublicCloud
pullSecret: '{"auths": ...}' 21
fips: false 22
sshKey: ssh-ed25519 AAAA... 23
additionalTrustBundle: | 24
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
imageContentSources: 25
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-v4.0-art-dev
publish: Internal 26
```

1 10 14 21 必需。安装程序会提示您输入这个值。

2 6 如果没有提供这些参数和值，安装程序会提供默认值。

3 7 **controlPlane** 部分是一个单个映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，controlPlane **部分** 的第一行则不以连字符开头。仅使用一个 control plane 池。

4 是否要启用或禁用并发多线程或 **超线程**。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将 参数值设置为 **Disabled** 来禁用它。如果在某些集群机器中禁用并发多线程，则必须在所有集群机器中禁用它。

重要

如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。如果您禁用并发多线程，请为您的机器使用较大的虚拟机类型，如 **Standard_D8s_v3**。

5 8 您可以指定要使用的磁盘大小（以 GB 为单位）。control plane 节点的最低推荐值为 1024 GB。

9 指定要将机器部署到的区域列表。如需高可用性，请至少指定两个区域。

11 要安装的集群网络插件。默认值 **OVNKubernetes** 是唯一支持的值。

12 可选：应该用来引导 control plane 和计算机器的自定义 Red Hat Enterprise Linux CoreOS (RHCOS) 镜像。**platform.azure.defaultMachinePlatform.osImage** 下的 **publisher**, **offer**, **sku**, 和 **version** 参数应用到 control plane 和计算机器。如果设置了 **controlPlane.platform.azure.osImage**

或 **compute.platform.azure.osImage** 下的参数，它们会覆盖 **platform.azure.defaultMachinePlatform.osImage** 参数。

13 指定包含基域的 DNS 区的资源组的名称。

15 指定要安装集群的现有资源组的名称。如果未定义，则会为集群创建新的资源组。

16 如果使用现有的 VNet，请指定包含它的资源组的名称。

17 如果使用现有的 VNet，请指定其名称。

18 如果使用现有的 VNet，请指定托管 control plane 机器的子网名称。

19 如果使用现有的 VNet，请指定托管计算机器的子网名称。

20 当使用 Azure Firewall 来限制互联网访问时，您必须配置出站路由来通过 Azure Firewall 发送流量。配置用户定义的路由可防止在集群中公开外部端点。

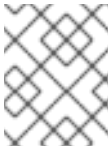
22 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 Red Hat Enterprise Linux (RHEL) 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅在 FIPS 模式中安装该系统。

当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS)时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。

23 您可以选择提供您用来访问集群中机器的 **sshKey** 值。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

24 提供用于镜像 registry 的证书文件内容。

25 提供命令输出中的 **imageContentSources 部分来** 镜像存储库。

26 如何发布集群的面向用户的端点。当使用 Azure Firewall 来限制互联网访问时，请将 **publish** 设置为 **Internal** 以部署私有集群。然后，无法从互联网访问面向用户的端点。默认值为 **External**。

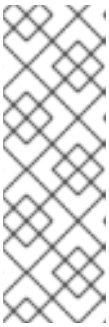
7.12.6.7. 在安装过程中配置集群范围的代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。

- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 **Proxy** 对象的 **spec.noProxy** 字段中添加站点来绕过代理。



注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，**Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

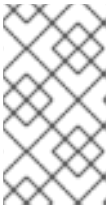
1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 **.** 以仅匹配子域。例如，**.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代理。
4. 如果提供，安装程序会在 **openshift-config** 命名空间中生成名为 **user-ca-bundle** 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 **trusted-ca-bundle** 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， **Proxy** 对象的 **trustedCA** 字段中也会引用此配置映射。**additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。
5. 可选：决定 **Proxy** 对象的配置以引用 **trustedCA** 字段中 **user-ca-bundle** 配置映射的策略。允许的值是 **Proxyonly** 和 **Always**。仅在配置了 **http/https** 代理时，使用 **Proxyonly** 引用 **user-ca-bundle** 配置映射。使用 **Always** 始终引用 **user-ca-bundle** 配置映射。默认值为 **Proxyonly**。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。



注意

如果安装程序超时，重启并使用安装程序的 **wait-for** 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. 保存该文件并在安装 OpenShift Container Platform 时引用。

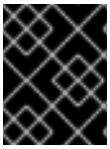
安装程序会创建一个名为 cluster 的集群范围代理，该代理 **使用** 提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

7.12.7. 安装 OpenShift CLI

您可以安装 OpenShift CLI(**oc**)来使用命令行界面与 OpenShift Container Platform 进行交互。您可以在 Linux、Windows 或 macOS 上安装 **oc**。



重要

如果安装了旧版本的 **oc**，则可能无法使用 OpenShift Container Platform 4.16 中的所有命令。下载并安装新版本的 **oc**。

在 Linux 上安装 OpenShift CLI
您可以按照以下流程在 Linux 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **产品变体** 下拉列表中选择架构。
3. 从 **版本** 下拉列表中选择适当的版本。
4. 点 **OpenShift v4.16 Linux Client** 条目旁的 **Download Now** 来保存文件。
5. 解包存档：

```
$ tar xvf <file>
```

6. 将 **oc** 二进制文件放到 **PATH** 中的目录中。
要查看您的 **PATH**，请执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 Windows Client** 条目旁的 **Download Now** 来保存文件。
4. 使用 ZIP 程序解压存档。
5. 将 **oc** 二进制文件移到 **PATH 中的目录中**。
   要查看您的 **PATH**，请打开命令提示并执行以下命令：

```
C:\> path
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 macOS Client** 条目旁的 **Download Now** 来保存文件。

注意

对于 macOS arm64，请选择 **OpenShift v4.16 macOS arm64 Client** 条目。

4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
   要查看您的 **PATH**，请打开终端并执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

7.12.8. 在 kube-system 项目中存储管理员级别的 secret 的替代方案

默认情况下，管理员 secret 存储在 **kube-system** 项目中。如果您在 **install-config.yaml** 文件中将 **credentialsMode** 参数配置为 **Manual**，则必须使用以下替代方案之一：

- 要手动管理长期云凭证，请按照手动创建长期凭证中的步骤操作。
- 要实现在集群外为各个组件管理的短期凭证，请按照配置 Azure 集群以使用短期凭证中的步骤操作。

7.12.8.1. 手动创建长期凭证

在无法访问云身份和访问管理(IAM)API 的环境中，或者管理员更不希望将管理员级别的凭证 secret 存储在集群 **kube-system** 命名空间中时，可以在安装前将 Cloud Credential Operator(CCO)放入手动模式。

流程

1. 如果您没有将 **install-config.yaml** 配置文件中的 **credentialsMode** 参数设置为 **Manual**，请修改值，如下所示：

   配置文件片段示例

   ```
   apiVersion: v1
   baseDomain: example.com
   credentialsMode: Manual
   # ...
   ```

2. 如果您之前还没有创建安装清单文件，请运行以下命令：

   ```
   $ openshift-install create manifests --dir <installation_directory>
   ```

   其中 **<installation_directory>** 是安装程序在其中创建文件的目录。

3. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

   ```
   $ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
   ```

4. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 自定义资源 (CR) 列表：

   ```
   $ oc adm release extract \
     --from=$RELEASE_IMAGE \
     --credentials-requests \
     --included \ 1
     --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
     --to=<path_to_directory_for_credentials_requests> 3
   ```

   1. **--included** 参数仅包含特定集群配置所需的清单。
   2. 指定 **install-config.yaml** 文件的位置。
   3. 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

此命令为每个 **CredentialsRequest** 对象创建一个 YAML 文件。

CredentialsRequest 对象示例

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component_credentials_request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: AzureProviderSpec
    roleBindings:
    - role: Contributor
  ...
```

5. 在之前生成的 **openshift-install** 清单目录中为 secret 创建 YAML 文件。secret 必须使用在 **spec.secretRef** 中为每个 **CredentialsRequest** 定义的命名空间和 secret 名称存储。

带有 secret 的 CredentialsRequest 对象示例

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component_credentials_request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: AzureProviderSpec
    roleBindings:
    - role: Contributor
      ...
  secretRef:
    name: <component_secret>
    namespace: <component_namespace>
  ...
```

Secret 对象示例

```
apiVersion: v1
kind: Secret
metadata:
  name: <component_secret>
  namespace: <component_namespace>
data:
  azure_subscription_id: <base64_encoded_azure_subscription_id>
  azure_client_id: <base64_encoded_azure_client_id>
  azure_client_secret: <base64_encoded_azure_client_secret>
  azure_tenant_id: <base64_encoded_azure_tenant_id>
```

```
azure_resource_prefix: <base64_encoded_azure_resource_prefix>
azure_resourcegroup: <base64_encoded_azure_resourcegroup>
azure_region: <base64_encoded_azure_region>
```

重要

在升级使用手动维护凭证的集群前，您必须确保 CCO 处于可升级状态。

7.12.8.2. 配置 Azure 集群以使用短期凭证

要安装使用 Microsoft Entra Workload ID 的集群，您必须配置 Cloud Credential Operator 工具，并为集群创建所需的 Azure 资源。

7.12.8.2.1. 配置 Cloud Credential Operator 工具

当 Cloud Credential Operator(CCO)以手动模式运行时，要从集群外部创建和管理云凭证，提取并准备 CCO 实用程序(**ccoctl**)二进制文件。

注意

ccoctl 工具是在 Linux 环境中运行的 Linux 二进制文件。

先决条件

- 您可以访问具有集群管理员权限的 OpenShift Container Platform 帐户。
- 已安装 OpenShift CLI(**oc**)。
- 您已为 **ccoctl** 工具创建了全局 Microsoft Azure 帐户，用于以下权限：

  例 7.93. 所需的 Azure 权限

  - Microsoft.Resources/subscriptions/resourceGroups/read
  - Microsoft.Resources/subscriptions/resourceGroups/write
  - Microsoft.Resources/subscriptions/resourceGroups/delete
  - Microsoft.Authorization/roleAssignments/read
  - Microsoft.Authorization/roleAssignments/delete
  - Microsoft.Authorization/roleAssignments/write
  - Microsoft.Authorization/roleDefinitions/read
  - Microsoft.Authorization/roleDefinitions/write
  - Microsoft.Authorization/roleDefinitions/delete
  - Microsoft.Storage/storageAccounts/listkeys/action
  - Microsoft.Storage/storageAccounts/delete
  - Microsoft.Storage/storageAccounts/read

- Microsoft.Storage/storageAccounts/write
- Microsoft.Storage/storageAccounts/blobServices/containers/write
- Microsoft.Storage/storageAccounts/blobServices/containers/delete
- Microsoft.Storage/storageAccounts/blobServices/containers/read
- Microsoft.ManagedIdentity/userAssignedIdentities/delete
- Microsoft.ManagedIdentity/userAssignedIdentities/read
- Microsoft.ManagedIdentity/userAssignedIdentities/write
- Microsoft.ManagedIdentity/userAssignedIdentities/federatedIdentityCredentials/read
- Microsoft.ManagedIdentity/userAssignedIdentities/federatedIdentityCredentials/write
- Microsoft.ManagedIdentity/userAssignedIdentities/federatedIdentityCredentials/delete
- Microsoft.Storage/register/action
- Microsoft.ManagedIdentity/register/action

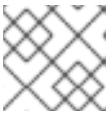
流程

1. 运行以下命令，为 OpenShift Container Platform 发行镜像设置变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. 运行以下命令，从 OpenShift Container Platform 发行镜像获取 CCO 容器镜像：

```
$ CCO_IMAGE=$(oc adm release info --image-for='cloud-credential-operator'
$RELEASE_IMAGE -a ~/.pull-secret)
```



注意

确保 **$RELEASE_IMAGE** 的架构与将使用 **ccoctl** 工具的环境架构相匹配。

3. 运行以下命令，将 CCO 容器镜像中的 **ccoctl** 二进制文件提取到 OpenShift Container Platform 发行镜像中：

```
$ oc image extract $CCO_IMAGE \
  --file="/usr/bin/ccoctl.<rhel_version>" \ 1
  -a ~/.pull-secret
```

1 对于 **<rhel_version>**，请指定与主机使用的 Red Hat Enterprise Linux (RHEL) 版本对应的值。如果没有指定值，则默认使用 **ccoctl.rhel8**。以下值有效：

- **rhel8**: 为使用 RHEL 8 的主机指定这个值。
- **rhel9** ：为使用 RHEL 9 的主机指定这个值。

4. 运行以下命令更改权限以使 **ccoctl** 可执行：

```
$ chmod 775 ccoctl.<rhel_version>
```

验证

- 要验证 **ccoctl** 是否可使用，请运行以下命令来显示帮助文件：

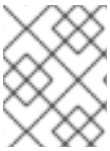
```
$ ccoctl --help
```

ccoctl --help 的输出

```
OpenShift credentials provisioning tool

Usage:
  ccoctl [command]

Available Commands:
  aws         Manage credentials objects for AWS cloud
  azure       Manage credentials objects for Azure
  gcp         Manage credentials objects for Google cloud
  help        Help about any command
  ibmcloud    Manage credentials objects for IBM Cloud
  nutanix     Manage credentials objects for Nutanix

Flags:
  -h, --help   help for ccoctl

Use "ccoctl [command] --help" for more information about a command.
```

7.12.8.2.2. 使用 Cloud Credential Operator 实用程序创建 Azure 资源

您可以使用 **ccoctl azure create-all** 命令自动创建 Azure 资源。



注意

默认情况下，**ccoctl** 在运行命令的目录中创建对象。要在其他目录中创建对象，请使用 **--output-dir** 标志。此流程使用 **<path_to_ccoctl_output_dir>** 来引用这个目录。

先决条件

您必须：

- 提取并准备好 **ccoctl** 二进制文件。
- 使用 Azure CLI 访问 Microsoft Azure 帐户。

流程

1. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 对象列表：

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
  --included \ 1
  --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
  --to=<path_to_directory_for_credentials_requests> 3
```

1. **--included** 参数仅包含特定集群配置所需的清单。
2. 指定 **install-config.yaml** 文件的位置。
3. 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

注意

此命令可能需要一些时间才能运行。

3. 要启用 **ccoctl** 工具自动检测 Azure 凭证，请运行以下命令登录到 Azure CLI：

```
$ az login
```

4. 运行以下命令，使用 **ccoctl** 工具处理所有 **CredentialsRequest** 对象：

```
$ ccoctl azure create-all \
  --name=<azure_infra_name> \ 1
  --output-dir=<ccoctl_output_dir> \ 2
  --region=<azure_region> \ 3
  --subscription-id=<azure_subscription_id> \ 4
  --credentials-requests-dir=<path_to_credentials_requests_directory> \ 5
  --dnszone-resource-group-name=<azure_dns_zone_resource_group_name> \ 6
  --tenant-id=<azure_tenant_id> 7
```

1. 为用于跟踪的所有创建 Azure 资源指定用户定义的名称。
2. 可选：指定您希望 **ccoctl** 实用程序在其中创建对象的目录。默认情况下，实用程序在运行命令的目录中创建对象。
3. 指定在其中创建云资源的 Azure 区域。
4. 指定要使用的 Azure 订阅 ID。
5. 指定包含组件 **CredentialsRequest** 对象文件的目录。
6. 指定包含集群基域 Azure DNS 区的资源组名称。
7. 指定要使用的 Azure 租户 ID。

注意

如果您的集群使用 **TechPreviewNoUpgrade** 功能集启用的技术预览功能，则必须包含 **--enable-tech-preview** 参数。

要查看其他可选参数以及如何使用它们的说明，请运行 **azure create-all --help** 命令。

验证

- 要验证 OpenShift Container Platform secret 是否已创建，列出 **<path_to_ccoctl_output_dir>/manifests** 目录中的文件：

```
$ ls <path_to_ccoctl_output_dir>/manifests
```

输出示例

```
azure-ad-pod-identity-webhook-config.yaml
cluster-authentication-02-config.yaml
openshift-cloud-controller-manager-azure-cloud-credentials-credentials.yaml
openshift-cloud-network-config-controller-cloud-credentials-credentials.yaml
openshift-cluster-api-capz-manager-bootstrap-credentials-credentials.yaml
openshift-cluster-csi-drivers-azure-disk-credentials-credentials.yaml
openshift-cluster-csi-drivers-azure-file-credentials-credentials.yaml
openshift-image-registry-installer-cloud-credentials-credentials.yaml
openshift-ingress-operator-cloud-credentials-credentials.yaml
openshift-machine-api-azure-cloud-credentials-credentials.yaml
```

您可以验证 Microsoft Entra ID 服务账户通过查询 Azure 而创建。如需更多信息，请参阅 Azure 文档中有关列出 Entra ID 服务帐户的内容。

7.12.8.2.3. 整合 Cloud Credential Operator 实用程序清单

要为单个组件在集群外实现短期安全凭证，您必须将创建 Cloud Credential Operator 实用程序 (**ccoctl**) 的清单文件移到安装程序的正确目录中。

先决条件

- 您已使用托管集群的云平台配置了帐户。
- 您已配置了 Cloud Credential Operator 实用程序 (**ccoctl**)。
- 已使用 **ccoctl** 工具创建了集群所需的云供应商资源。

流程

1. 如果您没有将 **install-config.yaml** 配置文件中的 **credentialsMode** 参数设置为 **Manual**，请修改值，如下所示：

配置文件片段示例

```
apiVersion: v1
baseDomain: example.com
credentialsMode: Manual
```

```
# ...
```

2. 如果您使用 **ccoctl** 实用程序创建新的 Azure 资源组，而不是使用现有资源组，请修改 **install-config.yaml** 中的 **resourceGroupName** 参数，如下所示：

   配置文件片段示例

```
apiVersion: v1
baseDomain: example.com
# ...
platform:
  azure:
    resourceGroupName: <azure_infra_name> 1
# ...
```

   1 这个值必须与 **ccoctl azure create-all** 命令的 **--name** 参数指定的 Azure 资源用户定义的名称匹配。

3. 如果您之前还没有创建安装清单文件，请运行以下命令：

```
$ openshift-install create manifests --dir <installation_directory>
```

   其中 **<installation_directory>** 是安装程序在其中创建文件的目录。

4. 运行以下命令，将 **ccoctl** 工具生成的清单复制到安装程序创建的 **manifests** 目录中：

```
$ cp /<path_to_ccoctl_output_dir>/manifests/* ./manifests/
```

5. 运行以下命令，将 **ccoctl** 工具在 **tls** 目录中生成的私钥复制到安装目录中：

```
$ cp -a /<path_to_ccoctl_output_dir>/tls .
```

7.12.9. 部署集群

您可以在兼容云平台上安装 OpenShift Container Platform。

重要

在初始安装过程中，您只能运行安装程序的 **create cluster** 命令一次。

先决条件

- 您已使用托管集群的云平台配置了帐户。
- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。
- 您有一个 Azure 订阅 ID 和租户 ID。

流程

- 进入包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定自定义 **./install-config.yaml** 文件的位置。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不是 **info**。

验证

当集群部署成功完成时：

- 终端会显示用于访问集群的说明，包括指向 Web 控制台和 **kubeadmin** 用户的凭证的链接。
- 凭证信息还会输出到 **<installation_directory>/.openshift_install.log**.

重要

不要删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

7.12.10. 使用 CLI 登录集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig
```
(1)

(1) 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

7.12.11. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于 远程健康监控

7.12.12. 后续步骤

- 自定义集群。
- 如果需要，您可以选择 不使用远程健康报告。

7.13. 在 AZURE 上安装三节点集群

在 OpenShift Container Platform 版本 4.16 中，您可以在 Microsoft Azure 上安装三节点集群。三节点集群包含三个 control plane 机器，它们也可以充当计算机器。这种类型的集群提供了一个较小的、效率更高的集群，供集群管理员和开发人员用于测试、开发和生产。

您可以使用安装程序置备或用户置备的基础架构安装三节点集群。

注意

不支持使用 Azure Marketplace 镜像部署三节点集群。

7.13.1. 配置三节点集群

在部署集群前，您可以通过将 **install-config.yaml** 文件中的 worker 节点数量设置为 **0** 来配置三节点集群。将 worker 节点数量设置为 **0** 可确保 control plane 机器可以调度。这允许调度应用程序工作负载从 control plane 节点运行。

注意

因为应用程序工作负载从 control plane 节点运行，所以需要额外的订阅，因为 control plane 节点被视为计算节点。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。

流程

1. 将 **install-config.yaml** 文件中的计算副本数量设置为 **0**，如以下 **compute** 小节中所示：

   三节点集群的 install-config.yaml 文件示例

   ```
   apiVersion: v1
   baseDomain: example.com
   compute:
   - name: worker
     platform: {}
     replicas: 0
   # ...
   ```

2. 如果您使用用户置备的基础架构部署集群：

   - 创建 Kubernetes 清单文件后，请确保在 **cluster-scheduler-02-config.yml** 文件中将 **spec.mastersSchedulable** 参数设置为 **true**。您可以在 **<installation_directory>/manifests** 中找到此文件。如需更多信息，请参阅"使用 ARM 模板在 Azure 上安装集群"中的"创建 Kubernetes 清单和 Ignition 配置文件"。

   - 不要创建额外的 worker 节点。

三节点集群的 cluster-scheduler-02-config.yml 文件示例

```
apiVersion: config.openshift.io/v1
kind: Scheduler
metadata:
  creationTimestamp: null
  name: cluster
spec:
  mastersSchedulable: true
  policy:
    name: ""
status: {}
```

7.13.2. 后续步骤

- 使用自定义在 Azure 上安装集群
- 使用 ARM 模板在 Azure 上安装集群

7.14. 在 AZURE 上卸载集群

您可以删除部署到 Microsoft Azure 的集群。

7.14.1. 删除使用安装程序置备的基础架构的集群

您可以从云中删除使用安装程序置备的基础架构的集群。

注意

卸载后，检查云供应商是否有未正确删除的资源，特别是在用户置备基础架构(UPI)集群中。可能存在安装程序未创建或安装程序无法访问的资源。

先决条件

- 有用于部署集群的安装程序副本。
- 有创建集群时安装程序生成的文件。

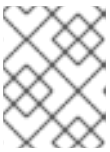
流程

1. 在用来安装集群的计算机中包含安装程序的目录中，运行以下命令：

```
$ ./openshift-install destroy cluster \
--dir <installation_directory> --log-level info 1 2
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2 要查看不同的详情，请指定 **warn**、**debug** 或 **error**，而不是 **info**。



注意

您必须为集群指定包含集群定义文件的目录。安装程序需要此目录中的 **metadata.json** 文件来删除集群。

2. 可选：删除 **<installation_directory>** 目录和 OpenShift Container Platform 安装程序。

7.14.2. 使用 Cloud Credential Operator 实用程序删除 Microsoft Azure 资源

卸载使用在集群外管理的短期凭证的 OpenShift Container Platform 集群后，您可以使用 CCO 实用程序 (**ccoctl**) 删除 **ccoctl** 在安装过程中创建的 Microsoft Azure (Azure)资源。

先决条件

- 提取并准备 **ccoctl** 二进制文件。
- 在 Azure 上卸载使用短期凭证的 OpenShift Container Platform 集群。

流程

- 运行以下命令，删除 **ccoctl** 创建的 Azure 资源：

```
$ ccoctl azure delete \
  --name=<name> \1
  --region=<azure_region> \2
  --subscription-id=<azure_subscription_id> \3
  --delete-oidc-resource-group
```

1 **<name>** 与最初用于创建和标记云资源的名称匹配。

2 **<azure_region>** 是要删除云资源的 Azure 区域。

3 **<azure_subscription_id>** 是要删除云资源的 Azure 订阅 ID。

验证

- 要验证资源是否已删除，请查询 Azure。如需更多信息，请参阅 Azure 文档。

7.15. AZURE 的安装配置参数

在 Microsoft Azure 上部署 OpenShift Container Platform 集群前，您可以提供参数来自定义集群以及托管它的平台。在创建 **install-config.yaml** 文件时，您可以通过命令行为所需参数提供值。然后，您可以修改 **install-config.yaml** 文件以进一步自定义集群。

7.15.1. Azure 可用的安装配置参数

下表指定您可以在安装过程中设置所需的、可选和特定于 Azure 的安装配置参数。

注意

安装后，您无法在 **install-config.yaml** 文件中修改这些参数。

7.15.1.1. 所需的配置参数

下表描述了所需的安装配置参数：

表 7.32. 所需的参数

参数	描述	值
apiVersion:	**install-config.yaml** 内容的 API 版本。当前版本为 **v1**。安装程序可能还支持旧的 API 版本。	字符串

参数	描述	值
baseDomain:	云供应商的基域。基域用于创建到 OpenShift Container Platform 集群组件的路由。集群的完整 DNS 名称是 **baseDomain** 和 **metadata.name** 参数值的组合，其格式为 **<metadata.name>.<baseDomain>**。	完全限定域名或子域名，如 **example.com**。
metadata:	Kubernetes 资源 **ObjectMeta**，其中只消耗 **name** 参数。	对象
metadata: name:	集群的名称。集群的 DNS 记录是 **{{.metadata.name}}.{{.baseDomain}}** 的子域。	小写字母、连字符(**-**)和句点(**.**)字符串，如 **dev**。
platform:	对于特定平台的配置取决于执行安装的环境： **aws**, **baremetal**, **azure**, **gcp**, **ibmcloud**, **nutanix**, **openstack**, **powervs**, **vsphere**, 或 **{}**.有关 **platform.<platform>** 参数的更多信息，请参考下表中您的特定平台。	对象

pullSecret:	从 Red Hat OpenShift Cluster Manager 获取 pull secret，验证从 Quay.io 等服务中下载 OpenShift Container Platform 组件的容器镜像。	<pre>{
		  "auths":{
		  "cloud.openshift.com":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  },
		  "quay.io":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  }
		  }
		}</pre>

7.15.1.2. 网络配置参数

您可以根据现有网络基础架构的要求自定义安装配置。例如，您可以扩展集群网络的 IP 地址块，或者提供不同于默认值的不同 IP 地址块。

仅支持 IPv4 地址。

注意

Red Hat OpenShift Data Foundation 灾难恢复解决方案不支持 Globalnet。对于区域灾难恢复场景，请确保为每个集群中的集群和服务网络使用非重叠的专用 IP 地址。

表 7.33. 网络参数

参数	描述	值
networking:	集群网络的配置。	对象  **注意** 您无法在安装后修改 **网络** 对象指定的参数。
networking: networkType:	要安装的 Red Hat OpenShift Networking 网络插件。	**OVNKubernetes**。**OVNKubernetes** 是 Linux 网络和包含 Linux 和 Windows 服务器的混合网络的 CNI 插件。默认值为 **OVNKubernetes**。
networking: clusterNetwork:	pod 的 IP 地址块。 默认值为 **10.128.0.0/14**，主机前缀为 **/23**。 如果您指定了多个 IP 地址块，块不得重叠。	对象数组。例如： networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23
networking: clusterNetwork: cidr:	使用 **networking.clusterNetwork** 时需要此项。IP 地址块。 IPv4 网络。	无类别域间路由(CIDR)表示法中的 IP 地址块。IPv4 块的前缀长度介于 **0 到 32** 之间。
networking: clusterNetwork: hostPrefix:	分配给每个节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从 given **cidr** 中分配 a/**23** 子网。**hostPrefix** 值 **23** 提供 510（2^(32 - 23)- 2）pod IP 地址。	子网前缀。 默认值为 **23**。
networking: serviceNetwork:	服务的 IP 地址块。默认值为 **172.30.0.0/16**。 OVN-Kubernetes 网络插件只支持服务网络的一个 IP 地址块。	CIDR 格式具有 IP 地址块的数组。例如： networking: serviceNetwork: - 172.30.0.0/16

参数	描述	值
networking: machineNetwork:	机器的 IP 地址块。 如果您指定了多个 IP 地址块，块不得重叠。	对象数组。例如： networking: machineNetwork: - cidr: 10.0.0.0/16
networking: machineNetwork: cidr:	使用 **networking.machineNetwork** 时需要此项。IP 地址块。对于 libvirt 和 IBM Power® Virtual Server 以外的所有平台，默认值为 **10.0.0.0/16**。对于 libvirt，默认值 为 **192.168.126.0/24**。对于 IBM Power® Virtual Server，默认值为 **192.168.0.0/24**。	CIDR 表示法中的 IP 网络块。 例如： **10.0.0.0/16**。 **注意** 将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

7.15.1.3. 可选的配置参数

下表描述了可选的安装配置参数：

表 7.34. 可选参数

参数	描述	值
additionalTrustBundle:	添加到节点可信证书存储中的 PEM 编码 X.509 证书捆绑包。配置了代理时，也可以使用此信任捆绑包。	字符串
capabilities:	控制可选核心组件的安装。您可以通过禁用可选组件来减少 OpenShift Container Platform 集群的空间。如需更多信息，请参阅 *安装*中的"集群功能"页面。	字符串数组
capabilities: baselineCapabilitySet:	选择要启用的一组初始可选功能。有效值为 **None**、**v4.11**、**v4.12** 和 **vCurrent**。默认值为 **vCurrent**。	字符串

参数	描述	值
capabilities: additionalEnabledCapabilities:	将可选功能集合扩展到您在 **baselineCapabilitySet** 中指定的范围。您可以在此参数中指定多个功能。	字符串数组
cpuPartitioningMode:	启用工作负载分区，它会隔离 OpenShift Container Platform 服务、集群管理工作负载和基础架构 pod，以便在保留的一组 CPU 上运行。工作负载分区只能在安装过程中启用，且在安装后无法禁用。虽然此字段启用工作负载分区，但它不会将工作负载配置为使用特定的 CPU。如需更多信息，请参阅 *Scalability and Performance* 部分中的 *Workload partitioning* 页面。	**None** 或 **AllNodes**.**None** 是默认值。
compute:	组成计算节点的机器的配置。	**MachinePool** 对象的数组。
compute: architecture:	决定池中机器的指令集合架构。目前，不支持具有不同架构的集群。所有池都必须指定相同的架构。有效值为 **amd64** 和 **arm64**。并非所有安装选项都支持 64 位 ARM 架构。要验证您的平台上是否支持您的安装选项，请参阅 *选择集群安装方法并为用户准备它*中的 *不同平台支持的安装方法*。	字符串
compute: hyperthreading:	是否在计算机器上启用或禁用并发多 **线程或超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。	**enabled** 或 **Disabled**
compute: name:	使用 **compute** 时需要此项。机器池的名称。	**worker**

参数	描述	值
compute: platform:	使用 **compute** 时需要此项。使用此参数指定托管 worker 机器的云供应商。此参数值必须与 **controlPlane.platform** 参数值匹配。	**aws**, **azure**, **gcp**, **ibmcloud**, **nutanix**, **openstack**, **powervs**, **vsphere**, 或 **{}**
compute: replicas:	要置备的计算机器数量，也称为 worker 机器。	大于或等于 **2** 的正整数。默认值为 **3**。
featureSet:	为功能集启用集群。功能集是 OpenShift Container Platform 功能的集合，默认情况下不启用。有关在安装过程中启用功能集的更多信息，请参阅"使用功能门启用功能"。	字符串.要启用的功能集的名称，如 **TechPreviewNoUpgrade**。
controlPlane:	组成 control plane 的机器的配置。	**MachinePool** 对象的数组。
controlPlane: architecture:	决定池中机器的指令集合架构。目前，不支持具有不同架构的集群。所有池都必须指定相同的架构。有效值为 **amd64** 和 **arm64**。并非所有安装选项都支持 64 位 ARM 架构。要验证您的平台上是否支持您的安装选项，请参阅 *选择集群安装方法并为用户准备它*中的*不同平台支持的安装方法*。	字符串
controlPlane: hyperthreading:	是否在 control plane 机器上启用或禁用并发多 **线程或超线程**。默认情况下，启用并发多线程以提高机器内核的性能。 **重要** 如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。	**enabled** 或 **Disabled**
controlPlane: name:	使用 **controlPlane** 时需要此项。机器池的名称。	**master**
controlPlane: platform:	使用 **controlPlane** 时需要此项。使用此参数指定托管 control plane 机器的云供应商。此参数值必须与 **compute.platform** 参数值匹配。	**aws**, **azure**, **gcp**, **ibmcloud**, **nutanix**, **openstack**, **powervs**, **vsphere**, 或 **{}**

参数	描述	值
controlPlane: replicas:	要置备的 control plane 机器数量。	部署单节点 OpenShift 时支持的值为 **3** 或 **1**。
credentialsMode:	Cloud Credential Operator(CCO)模式。如果没有指定模式，CCO 会动态尝试决定提供的凭证的功能，在支持多个模式的平台上首选 mint 模式。	**Mint**、**Passthrough**、**Manual** 或空字符串(**""**)。[1]
fips:	启用或禁用 FIPS 模式。默认值为 **false**（禁用）。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。  **重要** 要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 Red Hat Enterprise Linux (RHEL) 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅在 FIPS 模式中安装该系统。 当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS) 时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。  **注意** 如果使用 Azure File 存储，则无法启用 FIPS 模式。	**false** 或 **true**

参数	描述	值
imageContentSources:	release-image 内容的源和存储库。	对象数组。包括一个 **source** 以及可选的 **mirrors**，如本表的以下行所述。
imageContentSources: source:	使用 **imageContentSources** 时需要此项。指定用户在镜像拉取规格中引用的存储库。	字符串
imageContentSources: mirrors:	指定可能还包含同一镜像的一个或多个仓库。	字符串数组
publish:	如何发布或公开集群的面向用户的端点，如 Kubernetes API、OpenShift 路由。	**Internal**, **External**, 或 **Mixed**.要部署无法从互联网访问的私有集群，请将 **publish** 设置为 **Internal**。默认值为 **External**。要部署 API 和 ingress 服务器具有不同的发布策略的集群，请将 **publish** 设置为 **Mixed**，并使用 **operatorPublishingStrategy** 参数。
sshKey:	用于验证对集群机器的访问的 SSH 密钥。 **注意** 对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。	例如，**sshKey: ssh-ed25519 AAAA..**。

1. 不是所有 CCO 模式都支持所有云供应商。有关 CCO 模式的更多信息，请参阅*身份验证和授权*内容中的"管理云供应商凭证"条目。

重要

将此参数设置为 **Manual** 可启用在 **kube-system** 项目中存储管理员级别的 secret 的替代方案，这需要额外的配置步骤。如需更多信息，请参阅"在 kube-system 项目中存储管理员级别的 secret"。

7.15.1.4. 其他 Azure 配置参数

下表描述了其他 Azure 配置参数。

注意

默认情况下，如果您在 **install-config.yaml** 文件中指定可用区，安装程序会在一个区（region）内的这些可用区间分发 control plane 机器和计算机器。要确保集群的高可用性，请选择至少含有三个可用区的区域。如果您的区域包含的可用区少于三个，安装程序将在可用区中放置多台 control plane 机器。

表 7.35. 其他 Azure 参数

参数	描述	值
compute: platform: azure: encryptionAtHost:	为计算机器启用主机级别的加密。您可以启用这个加密，以及用户管理的服务器端加密。此功能加密虚拟机主机上的临时、临时、缓存和未管理的磁盘。这不是用户管理的服务器端加密的先决条件。	**true** 或 **false**。默认值为 **false**。
compute: platform: azure: osDisk: diskSizeGB:	虚拟机的 Azure 磁盘大小。	以 GB 为单位表示磁盘大小的整数。默认值为 **128**。
compute: platform: azure: osDisk: diskType:	定义磁盘的类型。	**standard_LRS**, **premium_LRS**, 或 **standardSSD_LRS**。默认值为 **Premium_LRS**。
compute: platform: azure: ultraSSDCapability:	启用 Azureultra 磁盘用于计算节点上的持久性存储。这需要您的 Azure 区域和区有可用的 ultra 磁盘。	**Enabled**, **Disabled**.默认值为 **Disabled**。
compute: platform: azure: osDisk: diskEncryptionSet: resourceGroup:	包含安装先决条件集中磁盘加密的 Azure 资源组名称。此资源组应和安装集群的资源组不同，以避免在集群销毁时删除 Azure 加密密钥。只有在打算使用用户管理的磁盘加密安装集群时才需要这个值。	字符串，如 **production_encryption_resource_group**。

参数	描述	值
compute: platform: azure: osDisk: diskEncryptionSet: name:	包含安装先决条件中加密密钥的磁盘加密集名称。	字符串，如 **production_disk_encryption_set**。
compute: platform: azure: osDisk: diskEncryptionSet: subscriptionId:	定义磁盘加密集所在的磁盘加密集的 Azure 订阅。此辅助磁盘加密集用于加密计算机器。	字符串，格式为 **00000000-0000-0000-0000-000000000000**。
compute: platform: azure: osImage: publisher:	可选。默认情况下，安装程序会下载并安装用于引导计算机器的 Red Hat Enterprise Linux CoreOS (RHCOS) 镜像。您可以使用 Azure Marketplace 中提供的自定义 RHCOS 镜像来覆盖默认行为。安装程序仅将此镜像用于计算机器。	字符串.镜像发布者的名称。
compute: platform: azure: osImage: offer:	与自定义 RHCOS 镜像关联的 Azure Marketplace 提供的名称。如果使用 **compute.platform.azure.osImage.publisher**，则需要此字段。	字符串.镜像提供的名称。
compute: platform: azure: osImage: sku:	Azure Marketplace 提供的实例。如果使用 **compute.platform.azure.osImage.publisher**，则需要此字段。	字符串.镜像提供的 SKU。
compute: platform: azure: osImage: version:	镜像 SKU 的版本号。如果使用 **compute.platform.azure.osImage.publisher**，则需要此字段。	字符串.要使用的镜像版本。

参数	描述	值
compute: platform: azure: vmNetworkingType: 	启用加速网络。加速网络可让单个根 I/O 虚拟化 (SR-IOV) 为虚拟机提高其网络性能。如果计算机器的实例类型支持 **Accelerated** 网络，安装程序会启用 **Accelerated** 网络，否则默认网络类型为 **Basic**。	**Accelerated** 或 **Basic**。
compute: platform: azure: type: 	为计算机器定义 Azure 实例类型。	字符串
compute: platform: azure: zones: 	安装程序在其中创建计算机器的可用区。	字符串列表
compute: platform: azure: settings: securityType: 	为计算节点启用机密虚拟机或可信启动。默认情况下不启用这个选项。	**ConfidentialVM** 或 **TrustedLaunch**.
compute: platform: azure: settings: confidentialVM: uefiSettings: secureBoot: 	如果使用机密虚拟机，则在计算节点上启用安全引导。	**Enabled** 或 **Disabled**。默认值为 **Disabled**。

参数	描述	值
compute: platform: azure: settings: confidentialVM: uefiSettings: virtualizedTrustedPlatformModule:	如果使用机密虚拟机，请在计算节点上启用虚拟化受信任的平台模块 (vTPM) 功能。	**Enabled** 或 **Disabled**。默认值为 **Disabled**。
compute: platform: azure: settings: trustedLaunch: uefiSettings: secureBoot:	如果使用可信启动，则在计算节点上启用安全引导。	**Enabled** 或 **Disabled**。默认值为 **Disabled**。
compute: platform: azure: settings: trustedLaunch: uefiSettings: virtualizedTrustedPlatformModule:	如果使用可信启动，则在计算节点上启用 vTPM 功能。	**Enabled** 或 **Disabled**。默认值为 **Disabled**。
compute: platform: azure: osDisk: securityProfile: securityEncryptionType:	为计算节点启用虚拟机客户机状态加密。只有在使用机密虚拟机时，才可以使用此参数。	**VMGuestStateOnly** 是唯一支持的值。

参数	描述	值
controlPlane: platform: azure: settings: securityType:	为 control plane 节点启用机密虚拟机或可信启动。默认情况下不启用这个选项。	**ConfidentialVM** 或 **TrustedLaunch**.
controlPlane: platform: azure: settings: confidentialVM: uefiSettings: secureBoot:	如果使用机密虚拟机，请在 control plane 节点上启用安全引导。	**Enabled** 或 **Disabled**。默认值为 **Disabled**。
controlPlane: platform: azure: settings: confidentialVM: uefiSettings: virtualizedTrustedPlatformModule:	如果使用机密虚拟机，请在 control plane 节点上启用 vTPM 功能。	**Enabled** 或 **Disabled**。默认值为 **Disabled**。
controlPlane: platform: azure: settings: trustedLaunch: uefiSettings: secureBoot:	如果使用可信启动，则在 control plane 节点上启用安全引导。	**Enabled** 或 **Disabled**。默认值为 **Disabled**。

参数	描述	值
controlPlane: platform: azure: settings: trustedLaunch: uefiSettings: virtualizedTrustedPlatformModule:	如果使用可信启动，请在 control plane 节点上启用 vTPM 功能。	**Enabled** 或 **Disabled**。默认值为 **Disabled**。
controlPlane: platform: azure: osDisk: securityProfile: securityEncryptionType:	为 control plane 节点启用虚拟机客户机状态加密。只有在使用机密虚拟机时，才可以使用此参数。	**VMGuestStateOnly** 是唯一支持的值。
controlPlane: platform: azure: type:	为 control plane 机器定义 Azure 实例类型。	字符串
controlPlane: platform: azure: zones:	安装程序在其中创建 control plane 机器的可用区。	字符串列表
platform: azure: defaultMachinePlatform: settings: securityType:	为所有节点启用机密虚拟机或可信启动。默认情况下不启用这个选项。	**ConfidentialVM** 或 **TrustedLaunch**.

参数	描述	值
platform: azure: defaultMachinePlatform: settings: confidentialVM: uefiSettings: secureBoot:	如果使用机密虚拟机，请在所有节点上启用安全引导。	**Enabled** 或 **Disabled**。默认值为 **Disabled**。
platform: azure: defaultMachinePlatform: settings: confidentialVM: uefiSettings: virtualizedTrustedPlatformModule:	如果您使用机密虚拟机，在所有节点上启用虚拟化受信任的平台模块 (vTPM) 功能。	**Enabled** 或 **Disabled**。默认值为 **Disabled**。
platform: azure: defaultMachinePlatform: settings: trustedLaunch: uefiSettings: secureBoot:	如果使用可信启动，则在所有节点上启用安全引导。	**Enabled** 或 **Disabled**。默认值为 **Disabled**。

参数	描述	值

platform:
  azure:
 	
defaultMachinePlatform:
  settings:
 	
trustedLaunch:
  uefiSettings:
 	
virtualizedTrustedPlatformModule:
	如果使用可信启动，则在所有节点上启用 vTPM 功能。	**Enabled** 或 **Disabled**。默认值为 **Disabled**。

platform:
  azure:
 	
defaultMachinePlatform:
  osDisk:
 	
securityProfile:
 	
securityEncryptionType:
	为所有节点启用虚拟机客户机状态加密。只有在使用机密虚拟机时，才可以使用此参数。	**VMGuestStateOnly** 是唯一支持的值。

platform: azure: defaultMachinePlatform: encryptionAtHost:	为计算机器启用主机级别的加密。您可以启用这个加密，以及用户管理的服务器端加密。此功能加密虚拟机主机上的临时、临时、缓存和未管理的磁盘。这个参数不是用户管理的服务器端加密的先决条件。	**true** 或 **false**。默认值为 **false**。

platform:
  azure:
 	
defaultMachinePlatform:
  osDisk:
 	
diskEncryptionSet:
  name:
	包含安装先决条件中加密密钥的磁盘加密集名称。	字符串，如 **production_disk_encryption_set**。

参数	描述	值

platform:
  azure:
 	
defaultMachinePlatform:
  osDisk:
 	
diskEncryptionSet:
 	
resourceGroup:
	包含安装先决条件集中磁盘加密的 Azure 资源组名称。为了避免在集群销毁时删除 Azure 加密密钥，此资源组必须与安装集群的资源组不同。只有在打算使用用户管理的磁盘加密安装集群时才需要这个值。	字符串，如 **production_encryption_resource_group**。

platform:
  azure:
 	
defaultMachinePlatform:
  osDisk:
 	
diskEncryptionSet:
 	
subscriptionId:
	定义磁盘加密集所在的磁盘加密集的 Azure 订阅。此辅助磁盘加密集用于加密计算机器。	字符串，格式为 **00000000-0000-0000-0000-000000000000**。

platform:
  azure:
 	
defaultMachinePlatform:
  osDisk:
  diskSizeGB:
	虚拟机的 Azure 磁盘大小。	以 GB 为单位表示磁盘大小的整数。默认值为 **128**。

platform:
  azure:
 	
defaultMachinePlatform:
  osDisk:
  diskType:
	定义磁盘的类型。	**premium_LRS** 或 **standardSSD_LRS**。默认值为 **Premium_LRS**。

参数	描述	值

platform:
  azure:
 	
defaultMachinePlatform:
  osImage:
  publisher:
	可选。默认情况下，安装程序会下载并安装用于引导 control plane 和计算机器的 Red Hat Enterprise Linux CoreOS (RHCOS) 镜像。您可以使用 Azure Marketplace 中提供的自定义 RHCOS 镜像来覆盖默认行为。安装程序将此镜像用于两种类型的机器。	字符串.镜像发布者的名称。

platform:
  azure:
 	
defaultMachinePlatform:
  osImage:
  offer:
	与自定义 RHCOS 镜像关联的 Azure Marketplace 提供的名称。如果您使用 **platform.azure.defaultMachinePlatform.osImage.publisher**，则需要此字段。	字符串.镜像提供的名称。

platform:
  azure:
 	
defaultMachinePlatform:
  osImage:
  sku:
	Azure Marketplace 提供的实例。如果您使用 **platform.azure.defaultMachinePlatform.osImage.publisher**，则需要此字段。	字符串.镜像提供的 SKU。

platform:
  azure:
 	
defaultMachinePlatform:
  osImage:
  version:
	镜像 SKU 的版本号。如果您使用 **platform.azure.defaultMachinePlatform.osImage.publisher**，则需要此字段。	字符串.要使用的镜像版本。

platform:
  azure:
 	
defaultMachinePlatform:
  type:
	control plane 和计算机器的 Azure 实例类型。	Azure 实例类型。

参数	描述	值

platform:
  azure:
 	
defaultMachinePlatform:
  zones:
	安装程序在其中创建计算和 control plane 机器的可用区。	字符串列表。

controlPlane:
  platform:
  azure:
 	
encryptionAtHost:
	为 control plane 机器启用主机级别的加密。您可以启用这个加密，以及用户管理的服务器端加密。此功能加密虚拟机主机上的临时、临时、缓存和未管理的磁盘。这不是用户管理的服务器端加密的先决条件。	**true** 或 **false**。默认值为 **false**。

controlPlane:
  platform:
  azure:
  osDisk:
 	
diskEncryptionSet:
 	
resourceGroup:
	包含安装先决条件集中磁盘加密的 Azure 资源组名称。此资源组应和安装集群的资源组不同，以避免在集群销毁时删除 Azure 加密密钥。只有在打算使用用户管理的磁盘加密安装集群时才需要这个值。	字符串，如 **production_encryption_resource_group**。

controlPlane:
  platform:
  azure:
  osDisk:
 	
diskEncryptionSet:
  name:
	包含安装先决条件中加密密钥的磁盘加密集名称。	字符串，如 **production_disk_encryption_set**。

controlPlane:
  platform:
  azure:
  osDisk:
 	
diskEncryptionSet:
 	
subscriptionId:
	定义磁盘加密集所在的磁盘加密集的 Azure 订阅。这个辅助磁盘加密集用于加密 control plane 机器。	字符串，格式为 **00000000-0000-0000-0000-000000000000**。

参数	描述	值
controlPlane: platform: azure: osDisk: diskSizeGB:	虚拟机的 Azure 磁盘大小。	以 GB 为单位表示磁盘大小的整数。默认值为 **1024**。
controlPlane: platform: azure: osDisk: diskType:	定义磁盘的类型。	**premium_LRS** 或 **standardSSD_LRS**。默认值为 **Premium_LRS**。
controlPlane: platform: azure: osImage: publisher:	可选。默认情况下，安装程序会下载并安装用于引导 control plane 机器的 Red Hat Enterprise Linux CoreOS (RHCOS)镜像。您可以使用 Azure Marketplace 中提供的自定义 RHCOS 镜像来覆盖默认行为。安装程序仅将此镜像用于 control plane 机器。	字符串.镜像发布者的名称。
controlPlane: platform: azure: osImage: offer:	与自定义 RHCOS 镜像关联的 Azure Marketplace 提供的名称。如果使用 **controlPlane.platform.azure.osImage.publisher**，则需要此字段。	字符串.镜像提供的名称。
controlPlane: platform: azure: osImage: sku:	Azure Marketplace 提供的实例。如果使用 **controlPlane.platform.azure.osImage.publisher**，则需要此字段。	字符串.镜像提供的 SKU。
controlPlane: platform: azure: osImage: version:	镜像 SKU 的版本号。如果使用 **controlPlane.platform.azure.osImage.publisher**，则需要此字段。	字符串.要使用的镜像版本。

参数	描述	值
controlPlane: platform: azure: ultraSSDCapability:	为 control plane 机器上的持久性存储启用 Azureultra 磁盘使用。这需要您的 Azure 区域和区有可用的 ultra 磁盘。	**Enabled**, **Disabled**.默认值为 **Disabled**。
controlPlane: platform: azure: vmNetworkingType: 	启用加速网络。加速网络可让单个根 I/O 虚拟化 (SR-IOV) 为虚拟机提高其网络性能。如果 control plane 机器的实例类型支持 **Accelerated** 网络，安装程序会启用 **Accelerated** 网络，否则默认网络类型为 **Basic**。	**Accelerated** 或 **Basic**。
platform: azure: baseDomainResourceGroupName:	包含基域的 DNS 区的资源组的名称。	字符串，如 **production_cluster**。
platform: azure: resourceGroupName:	集群要安装到的已有资源组的名称。此资源组必须为空，并且仅用于此特定群集；群集组件假定资源组中所有资源的所有权。如果您将安装程序的服务主体范围限制到这个资源组，您必须确保您的环境中安装程序使用的所有其他资源都有必要的权限，如公共 DNS 区和虚拟网络。使用安装程序销毁集群会删除此资源组。	字符串，如 **existing_resource_group**。

参数	描述	值
platform: azure: outboundType:	用于将集群连接到互联网的出站路由策略。如果您使用用户定义的路由，则必须在安装集群前配置出站路由。安装程序不负责配置用户定义的路由。如果您指定了 **NatGateway** 路由策略，安装程序将只创建一个 NAT 网关。如果指定了 **NatGateway** 路由策略，您的帐户必须具有 **Microsoft.Network/natGateways/read** 和 **Microsoft.Network/natGateways/write** 权限。 **重要** **NatGateway** 只是一个技术预览功能。技术预览功能不受红帽产品服务等级协议（SLA）支持，且功能可能并不完整。红帽不推荐在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。 有关红帽技术预览功能支持范围的更多信息，请参阅技术预览功能支持范围。	**LoadBalancer**、**UserDefinedRouting** 或 **NatGateway**。默认值为 **LoadBalancer**。
platform: azure: region:	托管集群的 Azure 区域的名称。	任何有效的区域名称，如 **centralus**。
platform: azure: zone:	要放入机器的可用区列表。如需高可用性，请至少指定两个区域。	区域列表，如 **["1", "2", "3"]**。

参数	描述	值
platform: azure: customerManagedKey: keyVault: name:	指定包含用于加密 Azure 存储的加密密钥的密钥 vault 名称。	字符串.
platform: azure: customerManagedKey: keyVault: keyName:	指定用于加密 Azure 存储的用户管理的加密密钥名称。	字符串.
platform: azure: customerManagedKey: keyVault: resourceGroup:	指定包含密钥 vault 和受管身份的资源组的名称。	字符串.
platform: azure: customerManagedKey: keyVault: subscriptionId:	指定与密钥 vault 关联的订阅 ID。	字符串，格式为 **00000000-0000-0000-0000-000000000000**。

参数	描述	值
platform: azure: customerManagedKey: userAssignedIdentityKey:	指定位于带有密钥 vault 资源组中的用户分配的受管身份的名称，并可访问用户管理的密钥。	字符串.
platform: azure: defaultMachinePlatform: ultraSSDCapability:	为 control plane 和计算机器上的持久性存储启用 Azureultra 磁盘使用。这需要您的 Azure 区域和区有可用的 ultra 磁盘。	**Enabled**, **Disabled**.默认值为 **Disabled**。
platform: azure: networkResourceGroupName:	包含您要将集群部署到的现有 VNet 的资源组的名称。这个名称不能与 platform.**azure.baseDomainResourceGroupName** 相同。	字符串.
platform: azure: virtualNetwork:	要将集群部署到的现有 VNet 的名称。	字符串.
platform: azure: controlPlaneSubnet:	要将 control plane 机器部署到的 VNet 中现有子网的名称。	有效的 CIDR，如 **10.0.0.0/16**。
platform: azure: computeSubnet:	要将计算机器部署到的 VNet 中现有子网的名称。	有效的 CIDR，如 **10.0.0.0/16**。

参数	描述	值
platform: azure: cloudName:	用于使用适当的 Azure API 端点配置 Azure SDK 的 Azure 云环境名称。如果为空，则使用默认值 **AzurePublicCloud**。	任何有效的云环境，如 **AzurePublicCloud** 或 **AzureUSrianCloud**。
platform: azure: defaultMachinePlatform: vmNetworkingType: 	启用加速网络。加速网络可让单个根 I/O 虚拟化 (SR-IOV) 为虚拟机提高其网络性能。	**Accelerated** 或 **Basic**。如果 control plane 和计算机器的实例类型支持 **Accelerated** 网络，安装程序会启用 **Accelerated**网络，否则默认网络类型为 **Basic**。
operatorPublishingStrategy: apiserver:	决定服务 API 的负载均衡器是公共还是私有的。将此参数设置为 **Internal**，以防止 API 服务器在 VNet 外部访问。将此参数设置为 **External**，使 API 服务器可以在 VNet 之外访问。如果设置此参数，您必须将 **publish** 参数设置为 **Mixed**。	**External** 或 **Internal**.默认值为 **External**。
operatorPublishingStrategy: ingress:	决定集群为入口流量创建的 DNS 资源是否公开可见。将此参数设置为 **Internal**，以防止公开访问入口 VIP。将此参数设置为 **External**，使入口 VIP 可被公开访问。如果设置此参数，您必须将 **publish** 参数设置为 **Mixed**。	**External** 或 **Internal**.默认值为 **External**。

注意

您无法自定义 Azure 可用区，也不能使用 标签来整理 Azure 集群的 Azure 资源。

第 8 章 在 AZURE STACK HUB 上安装

8.1. 准备在 AZURE STACK HUB 上安装

8.1.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读选择集群安装方法并为用户准备它的文档。
- 已安装 Azure Stack Hub 版本 2008 或更高版本。

8.1.2. 在 Azure Stack Hub 上安装 OpenShift Container Platform 的要求

在 Microsoft Azure Stack Hub 上安装 OpenShift Container Platform 前，您必须配置 Azure 帐户。

如需有关帐户配置、帐户限值、DNS 区配置、所需角色和创建服务主体的详细信息，请参阅配置 Azure Stack Hub 帐户。

8.1.3. 选择在 Azure Stack Hub 上安装 OpenShift Container Platform 的方法

您可以在安装程序置备或用户置备的基础架构上安装 OpenShift Container Platform。默认安装类型使用安装程序置备的基础架构，安装程序会在其中为集群置备底层基础架构。您还可以在您置备的基础架构上安装 OpenShift Container Platform。如果不使用安装程序置备的基础架构，您必须自己管理和维护集群资源。

如需有关安装程序置备和用户置备的安装过程的更多信息，请参阅 安装过程。

8.1.3.1. 在安装程序置备的基础架构上安装集群

您可以使用以下方法在 OpenShift Container Platform 安装程序置备的 Azure Stack Hub 基础架构上安装集群：

- **使用安装程序置备的基础架构在 Azure Stack Hub 上安装集群**：您可以在 OpenShift Container Platform 安装程序置备的 Azure Stack Hub 基础架构上安装 OpenShift Container Platform。

8.1.3.2. 在用户置备的基础架构上安装集群

您可以使用以下方法在您置备的 Azure Stack Hub 基础架构上安装集群：

- **使用 ARM 模板在 Azure Stack Hub 上安装集群**：您可以使用您提供的基础架构在 Azure Stack Hub 上安装 OpenShift Container Platform。您可以使用提供的 Azure Resource Manager（ARM）模板来协助安装。

8.1.4. 后续步骤

- 配置 Azure Stack Hub 帐户

8.2. 配置 AZURE STACK HUB 帐户

在安装 OpenShift Container Platform 之前，您必须配置 Microsoft Azure 帐户。

重要

所有通过公共端点提供的 Azure 资源均存在资源名称的限制，您无法创建使用某些名称的资源。如需 Azure 限制词语列表，请参阅 Azure 文档中的解决保留资源名称错误。

8.2.1. Azure Stack Hub 帐户限制

OpenShift Container Platform 集群使用多个 Microsoft Azure Stack Hub 组件，Azure Stack Hub 中的默认配额类型会影响您安装 OpenShift Container Platform 集群的能力。

下表总结了 Azure Stack Hub 组件，它们的限值会影响您安装和运行 OpenShift Container Platform 集群的能力。

组件	默认所需的组件数	描述
vCPU	56	默认集群需要 56 个 vCPU，因此您必须提高帐户限值。 默认情况下，每个集群创建以下实例： ● 一台 Bootstrap 机器，在安装后删除 ● 三个 control plane 机器 ● 三个计算（compute）机器 因为 bootstrap、control plane 和 worker 机器使用 **Standard_DS4_v2** 虚拟机（8 个 vCPU），所以默认集群需要 56 个 vCPU。bootstrap 节点虚拟机仅在安装过程中使用。 若要部署更多 worker 节点、启用自动扩展、部署大型工作负载或使用不同的实例类型，您必须进一步提高帐户的 vCPU 限值，以确保集群可以部署您需要的机器。
VNet	1	每个默认集群都需要一个虚拟网络 (VNet)，此网络包括两个子网。
网络接口	7	每个默认集群都需要 7 个网络接口。如果您要创建更多机器或者您部署的工作负载要创建负载均衡器，则集群会使用更多的网络接口。
网络安全组	2	每个集群为 VNet 中的每个子网创建网络安全组。默认集群为 control plane 和计算节点子网创建网络安全组： **controlplane**：允许从任何位置通过端口 6443 访问 control plane 机器 **node**：允许从互联网通过端口 80 和 443 访问 worker 节点

组件	默认所需的组件数	描述
网络负载均衡器	3	每个集群都会创建以下负载均衡器： **default** 用于在 worker 机器之间对端口 80 和 443 的请求进行负载均衡的公共 IP 地址 **internal** 用于在 control plane 机器之间对端口 6443 和 22623 的请求进行负载均衡的专用 IP 地址 **external** 用于在 control plane 机器之间对端口 6443 的请求进行负载均衡的公共 IP 地址 如果您的应用程序创建了更多的 Kubernetes **LoadBalancer** 服务对象，您的集群会使用更多的负载均衡器。
公共 IP 地址	2	公共负载均衡器使用一个公共 IP 地址。bootstrap 机器也使用一个公共 IP 地址，以便您可以在安装期间通过 SSH 连接到该机器来进行故障排除。bootstrap 节点的 IP 地址仅在安装过程中使用。
专用 IP 地址	7	内部负载均衡器、三台 control plane 机器中的每一台以及三台 worker 机器中的每一台各自使用一个专用 IP 地址。

其他资源

- 优化存储.

8.2.2. 在 Azure Stack Hub 中配置 DNS 区域

要在 Azure Stack Hub 上成功安装 OpenShift Container Platform，您必须在 Azure Stack Hub DNS 区域中创建 DNS 记录。DNS 区域必须对域具有权威。要将注册商的 DNS 区域委派给 Azure Stack Hub，请参阅 Microsoft 有关 Azure Stack Hub 数据中心 DNS 集成 的文档。

8.2.3. 所需的 Azure Stack Hub 角色

Microsoft Azure Stack Hub 帐户必须具有您使用的订阅的以下角色：

- **所有者**

要在 Azure 门户上设置角色，请参阅 Microsoft 文档中的通过基于角色的访问控制管理对 Azure Stack Hub 中资源的管理访问权限。

8.2.4. 创建服务主体

由于 OpenShift Container Platform 及其安装程序使用 Azure Resource Manager 创建 Microsoft Azure 资源，因此您必须创建一个服务主体来代表它。

先决条件

- 安装或更新 Azure CLI。

- 您的 Azure 帐户具有您所用订阅所需的角色。

流程

1. 注册您的环境：

```
$ az cloud register -n AzureStackCloud --endpoint-resource-manager <endpoint> 1
```

1 指定 Azure Resource Manager 端点 'https://management.<region>.<fqdn>/'。

详情请查看 Microsoft 文档。

2. 设置活跃环境：

```
$ az cloud set -n AzureStackCloud
```

3. 更新您的环境配置，以使用 Azure Stack Hub 的特定 API 版本：

```
$ az cloud update --profile 2019-03-01-hybrid
```

4. 登录 Azure CLI：

```
$ az login
```

如果您在多租户环境中，还必须提供租户 ID。

5. 如果您的 Azure 帐户使用订阅，请确定您使用正确的订阅：

a. 查看可用帐户列表并记录您要用于集群的订阅的 **tenantId** 值：

```
$ az account list --refresh
```

输出示例

```
[
  {
    "cloudName": AzureStackCloud",
    "id": "9bab1460-96d5-40b3-a78e-17b15e978a80",
    "isDefault": true,
    "name": "Subscription Name",
    "state": "Enabled",
    "tenantId": "6057c7e9-b3ae-489d-a54e-de3f6bf6a8ee",
    "user": {
      "name": "you@example.com",
      "type": "user"
    }
  }
]
```

b. 查看您的活跃帐户详情，确认 **tenantId** 值与您要使用的订阅匹配：

```
$ az account show
```

输出示例

```
{
  "environmentName": AzureStackCloud",
  "id": "9bab1460-96d5-40b3-a78e-17b15e978a80",
  "isDefault": true,
  "name": "Subscription Name",
  "state": "Enabled",
  "tenantId": "6057c7e9-b3ae-489d-a54e-de3f6bf6a8ee", (1)
  "user": {
    "name": "you@example.com",
    "type": "user"
  }
}
```

(1) 确保 **tenantId** 参数的值是正确的订阅 ID。

c. 如果您使用的订阅不正确，请更改活跃的订阅：

```
$ az account set -s <subscription_id> (1)
```

(1) 指定订阅 ID。

d. 验证订阅 ID 更新：

```
$ az account show
```

输出示例

```
{
  "environmentName": AzureStackCloud",
  "id": "33212d16-bdf6-45cb-b038-f6565b61edda",
  "isDefault": true,
  "name": "Subscription Name",
  "state": "Enabled",
  "tenantId": "8049c7e9-c3de-762d-a54e-dc3f6be6a7ee",
  "user": {
    "name": "you@example.com",
    "type": "user"
  }
}
```

6. 记录输出中的 **tenantId** 和 **id** 参数值。OpenShift Container Platform 安装过程中需要这些值。

7. 为您的帐户创建服务主体：

```
$ az ad sp create-for-rbac --role Contributor --name <service_principal> \ (1)
  --scopes /subscriptions/<subscription_id> (2)
  --years <years> (3)
```

(1) 指定服务主体名称。

2 指定订阅 ID。

3 指定年数。默认情况下，服务主体在一年后过期。通过使用 **--years** 选项，您可以扩展服务主体的有效性。

输出示例

```
Creating 'Contributor' role assignment under scope '/subscriptions/<subscription_id>'
The output includes credentials that you must protect. Be sure that you do not
include these credentials in your code or check the credentials into your source
control. For more information, see https://aka.ms/azadsp-cli
{
  "appId": "ac461d78-bf4b-4387-ad16-7e32e328aec6",
  "displayName": <service_principal>",
  "password": "00000000-0000-0000-0000-000000000000",
  "tenantId": "8049c7e9-c3de-762d-a54e-dc3f6be6a7ee"
}
```

8. 记录前面输出中 **appId** 和 **password** 参数的值。OpenShift Container Platform 安装过程中需要这些值。

其他资源

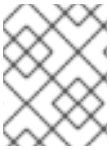
- 如需有关 CCO 模式的更多信息，请参阅 关于 Cloud Credential Operator。

8.2.5. 后续步骤

- 安装 OpenShift Container Platform 集群：
  - 在 Azure Stack Hub 上快速安装集群 。
  - 按照使用 ARM 模板在 Azure Stack Hub 上安装集群 ，使用用户置备的基础架构在 Azure Stack Hub 上安装 OpenShift Container Platform 集群。

8.3. 使用安装程序置备的基础架构在 AZURE STACK HUB 上安装集群

在 OpenShift Container Platform 版本 4.16 中，您可以使用安装程序置备的基础架构在 Microsoft Azure Stack Hub 上安装集群。但是，您必须手动配置 **install-config.yaml** 文件，以指定特定于 Azure Stack Hub 的值。



注意

虽然您可以在使用安装程序使用安装程序置备的基础架构部署集群时选择 **azure**，但这个选项只支持 Azure Public Cloud。

8.3.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读选择集群安装方法并为用户准备它的文档。
- 已将 Azure Stack Hub 帐户配置为 托管集群。

- 如果使用防火墙，则会 将其配置为允许集群需要访问的站点。
- 您确认有大约 16 GB 的本地磁盘空间。安装集群要求您下载 RHCOS 虚拟硬盘(VHD)集群镜像，并将其上传到 Azure Stack Hub 环境，以便在部署过程中访问它。解压缩 VHD 文件需要这个数量的本地磁盘空间。

8.3.2. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类型的基础架构上执行受限网络安装。在此过程中，您可以下载所需的内容，并使用它为镜像 registry 填充安装软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群前，您要更新镜像 registry 的内容。

8.3.3. 为集群节点 SSH 访问生成密钥对

在 OpenShift Container Platform 安装过程中，您可以为安装程序提供 SSH 公钥。密钥通过它们的 Ignition 配置文件传递给 Red Hat Enterprise Linux CoreOS(RHCOS)节点，用于验证对节点的 SSH 访问。密钥添加到每个节点上 **core** 用户的 **~/.ssh/authorized_keys** 列表中，这将启用免密码身份验证。

将密钥传递给节点后，您可以使用密钥对作为用户 **核心** 通过 SSH 连接到 RHCOS 节点。若要通过 SSH 访问节点，必须由 SSH 为您的本地用户管理私钥身份。

如果要通过 SSH 连接到集群节点来执行安装调试或灾难恢复，则必须在安装过程中提供 SSH 公钥。**./openshift-install gather** 命令还需要在集群节点上设置 SSH 公钥。

重要

不要在生产环境中跳过这个过程，在生产环境中需要灾难恢复和调试。

注意

您必须使用本地密钥，而不是使用特定平台方法配置 的密钥，如 AWS 密钥对。

流程

1. 如果您在本地计算机上没有可用于在集群节点上进行身份验证的现有 SSH 密钥对，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 ~/**.ssh/id_ed25519**。如果您已有密钥对，请确保您的公钥位于 ~/**.ssh** 目录中。

注意

如果您计划在 **x86_64**、**ppc64le** 和 **s390x** 架构上安装使用 RHEL 加密库（这些加密库已提交给 NIST 用于 FIPS 140-2/140-3 验证）的 OpenShift Container Platform 集群，则不要创建使用 **ed25519** 算法的密钥。相反，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 查看公共 SSH 密钥：

```
$ cat <path>/<file_name>.pub
```

例如，运行以下命令来查看 ~/**.ssh/id_ed25519.pub** 公钥：

```
$ cat ~/.ssh/id_ed25519.pub
```

3. 将 SSH 私钥身份添加到本地用户的 SSH 代理（如果尚未添加）。在集群节点上，或者要使用 **./openshift-install gather** 命令，需要对该密钥进行 SSH 代理管理，才能在集群节点上进行免密码 SSH 身份验证。

注意

在某些发行版中，自动管理默认 SSH 私钥身份，如 ~/**.ssh/id_rsa** 和 ~/**.ssh/id_dsa**。

a. 如果 **ssh-agent** 进程尚未为您的本地用户运行，请将其作为后台任务启动：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果集群处于 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

4. 将 SSH 私钥添加到 **ssh-agent** ：

```
$ ssh-add <path>/<file_name> 1
```

1 指定 SSH 私钥的路径和文件名，如 ~/**.ssh/id_ed25519.pub**

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

后续步骤

- 安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

8.3.4. 上传 RHCOS 集群镜像

您必须下载 RHCOS 虚拟硬盘(VHD)集群镜像，并将其上传到 Azure Stack Hub 环境，以便在部署过程中访问它。

先决条件

- 配置 Azure 帐户。

流程

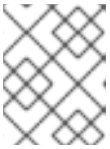
1. 获取 RHCOS VHD 集群镜像：
   a. 将 RHCOS VHD 的 URL 导出为环境变量。

```
$ export COMPRESSED_VHD_URL=$(openshift-install coreos print-stream-json | jq -r '.architectures.x86_64.artifacts.azurestack.formats."vhd.gz".disk.location')
```

   b. 本地下载压缩的 RHCOS VHD 文件。

```
$ curl -O -L ${COMPRESSED_VHD_URL}
```

2. 解压缩 VHD 文件。



注意

解压缩的 VHD 文件大约为 16 GB，因此请确保您的主机系统有 16 GB 的可用空间。上传后，可以删除 VHD 文件。

3. 将本地 VHD 上传到 Azure Stack Hub 环境，确保 blob 已公开可用。例如，您可以使用 **az** cli 或 web 门户将 VHD 上传到 blob。

8.3.5. 获取安装程序

在安装 OpenShift Container Platform 前，将安装文件下载到您用于安装的主机上。

先决条件

- 您有一台运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB。

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐户，请使用您的凭证登录。如果没有，请创建一个帐户。
2. 选择 **Azure** 作为云供应商。
3. 进入到安装类型的页面，下载与您的主机操作系统和架构对应的安装程序，并将该文件放在您要存储安装配置文件的目录中。

重要

安装程序会在用来安装集群的计算机上创建几个文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。

重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，请为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。此 pull secret 允许您与所含授权机构提供的服务进行身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

8.3.6. 手动创建安装配置文件

安装集群要求您手动创建安装配置文件。

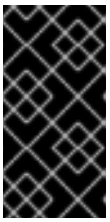
先决条件

- 您在本地机器上有一个 SSH 公钥来提供给安装程序。该密钥将用于在集群节点上进行 SSH 身份验证，以进行调试和灾难恢复。
- 已获取 OpenShift Container Platform 安装程序和集群的 pull secret。

流程

1. 创建一个安装目录来存储所需的安装资产：

```
$ mkdir <installation_directory>
```



重要

您必须创建一个目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

2. 自定义提供的 **install-config.yaml** 文件模板示例，并将其保存在 **<installation_directory>** 中。

注意

此配置文件必须命名为 **install-config.yaml**。

进行以下修改：

a. 指定所需的安装参数。

b. 更新 **platform.azure** 部分，以指定特定于 Azure Stack Hub 的参数。

c. 可选：更新一个或多个默认配置参数以自定义安装。
有关参数的更多信息，请参阅"安装配置参数"。

3. 备份 **install-config.yaml** 文件，以便您可以使用它安装多个集群。

重要

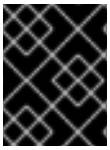
install-config.yaml 文件会在安装过程的下一步中使用。现在必须备份它。

其他资源

- Azure Stack Hub 的安装配置参数

8.3.6.1. Azure Stack Hub 的自定义 install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多详情，或修改所需参数的值。



重要

此示例 YAML 文件仅供参考。使用它作为资源，在您手动创建的安装配置文件中输入参数值。

```
apiVersion: v1
baseDomain: example.com (1)
credentialsMode: Manual
controlPlane: (2) (3)
  name: master
  platform:
    azure:
      osDisk:
        diskSizeGB: 1024 (4)
        diskType: premium_LRS
  replicas: 3
compute: (5)
- name: worker
  platform:
    azure:
      osDisk:
        diskSizeGB: 512 (6)
        diskType: premium_LRS
  replicas: 3
metadata:
  name: test-cluster (7) (8)
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OVNKubernetes (9)
```

```
  serviceNetwork:
  - 172.30.0.0/16
platform:
  azure:
    armEndpoint: azurestack_arm_endpoint (10) (11)
    baseDomainResourceGroupName: resource_group (12) (13)
    region: azure_stack_local_region (14) (15)
    resourceGroupName: existing_resource_group (16)
    outboundType: Loadbalancer
    cloudName: AzureStackCloud (17)
    clusterOSimage: https://vhdsa.blob.example.example.com/vhd/rhcos-410.84.202112040202-0-
azurestack.x86_64.vhd (18) (19)
pullSecret: '{"auths": ...}' (20) (21)
fips: false (22)
sshKey: ssh-ed25519 AAAA... (23)
additionalTrustBundle: | (24)
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
```

(1) (7) (10) (12) (14) (17) (18) (20) 必需。

(2) (5) 如果没有提供这些参数和值，安装程序会提供默认值。

(3) **controlPlane** 部分是一个单个映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，controlPlane **部分** 的第一行则不以连字符开头。虽然这两个部分目前都定义了单一机器池，但未来的 OpenShift Container Platform 版本可能在安装过程中支持定义多个计算池。仅使用一个 control plane 池。

(4) (6) 您可以指定要使用的磁盘大小（以 GB 为单位）。control plane 节点的最低推荐值为 1024 GB。

(8) 集群的名称。

(9) 要安装的集群网络插件。默认值 **OVNKubernetes** 是唯一支持的值。

(11) Azure Stack Hub Operator 提供的 Azure Resource Manager 端点。

(13) 包含基域的 DNS 区的资源组的名称。

(15) Azure Stack Hub 本地区域的名称。

(16) 安装集群的现有资源组的名称。如果未定义，则会为集群创建新的资源组。

(19) Azure Stack 环境中包含 RHCOS VHD 存储 blob 的 URL。

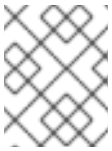
(21) 对集群进行身份验证所需的 pull secret。

(22) 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS)时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。

23 您可以选择提供您用来访问集群中机器的 **sshKey** 值。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

24 如果 Azure Stack Hub 环境使用内部证书颁发机构(CA)，则需要添加 CA 证书。

8.3.7. 手动管理云凭证

Cloud Credential Operator(CCO)只支持手动模式的云供应商。因此，您必须为云供应商指定身份和访问管理(IAM)secret。

流程

1. 如果您之前还没有创建安装清单文件，请运行以下命令：

```
$ openshift-install create manifests --dir <installation_directory>
```

其中 **<installation_directory>** 是安装程序在其中创建文件的目录。

2. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

3. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 自定义资源 (CR) 列表：

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
  --included \ 1
  --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
  --to=<path_to_directory_for_credentials_requests> 3
```

1 **--included** 参数仅包含特定集群配置所需的清单。

2 指定 **install-config.yaml** 文件的位置。

3 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

此命令为每个 **CredentialsRequest** 对象创建一个 YAML 文件。

CredentialsRequest 对象示例

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component_credentials_request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: AzureProviderSpec
    roleBindings:
    - role: Contributor
    ...
```

4. 在之前生成的 **openshift-install** 清单目录中为 secret 创建 YAML 文件。secret 必须使用在 **spec.secretRef** 中为每个 **CredentialsRequest** 定义的命名空间和 secret 名称存储。

带有 secret 的 CredentialsRequest 对象示例

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component_credentials_request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: AzureProviderSpec
    roleBindings:
    - role: Contributor
      ...
  secretRef:
    name: <component_secret>
    namespace: <component_namespace>
  ...
```

Secret 对象示例

```
apiVersion: v1
kind: Secret
metadata:
  name: <component_secret>
  namespace: <component_namespace>
data:
  azure_subscription_id: <base64_encoded_azure_subscription_id>
  azure_client_id: <base64_encoded_azure_client_id>
  azure_client_secret: <base64_encoded_azure_client_secret>
  azure_tenant_id: <base64_encoded_azure_tenant_id>
  azure_resource_prefix: <base64_encoded_azure_resource_prefix>
  azure_resourcegroup: <base64_encoded_azure_resourcegroup>
  azure_region: <base64_encoded_azure_region>
```

重要

在升级使用手动维护凭证的集群前，您必须确保 CCO 处于可升级状态。

其他资源

- 使用手动维护的凭证更新云供应商资源

8.3.8. 配置集群以使用内部 CA

如果 Azure Stack Hub 环境使用内部证书颁发机构(CA)，请更新 **cluster-proxy-01-config.yaml 文件** 将集群配置为使用内部 CA。

先决条件

- 创建 **install-config.yaml** 文件，并以 **.pem** 格式指定证书信任捆绑包。
- 创建集群清单。

流程

1. 从安装程序创建文件的目录中，前往 **manifests** 目录。
2. 将 **user-ca-bundle** 添加到 **spec.trustedCA.name** 字段中。

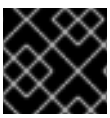
   cluster-proxy-01-config.yaml 文件示例

   ```
   apiVersion: config.openshift.io/v1
   kind: Proxy
   metadata:
     creationTimestamp: null
     name: cluster
   spec:
     trustedCA:
       name: user-ca-bundle
   status: {}
   ```

3. 可选：备份 **manifests/ cluster-proxy-01-config.yaml** 文件。在部署集群时，安装程序会消耗 **manifests/** 目录。

8.3.9. 部署集群

您可以在兼容云平台上安装 OpenShift Container Platform。



重要

在初始安装过程中，您只能运行安装程序的 **create cluster** 命令一次。

先决条件

- 您已使用托管集群的云平台配置了帐户。
- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。

- 已确认主机上的云供应商帐户具有部署集群的正确权限。权限不正确的帐户会导致安装过程失败，并显示包括缺失权限的错误消息。

流程

- 进入包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定自定义 **./install-config.yaml** 文件的位置。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不是 **info**。

验证

当集群部署成功完成时：

- 终端会显示用于访问集群的说明，包括指向 Web 控制台和 **kubeadmin** 用户的凭证的链接。
- 凭证信息还会输出到 **<installation_directory>/.openshift_install.log**.

重要

不要删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

8.3.10. 安装 OpenShift CLI

您可以安装 OpenShift CLI(**oc**)来使用命令行界面与 OpenShift Container Platform 进行交互。您可以在 Linux、Windows 或 macOS 上安装 **oc**。

重要

如果安装了旧版本的 **oc**，则可能无法使用 OpenShift Container Platform 4.16 中的所有命令。下载并安装新版本的 **oc**。

在 Linux 上安装 OpenShift CLI
您可以按照以下流程在 Linux 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **产品变体** 下拉列表中选择架构。
3. 从 **版本** 下拉列表中选择适当的版本。
4. 点 **OpenShift v4.16 Linux Client** 条目旁的 **Download Now** 来保存文件。
5. 解包存档：

```
$ tar xvf <file>
```

6. 将 **oc** 二进制文件放到 **PATH 中的目录中**。
要查看您的 **PATH**，请执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

在 Windows 上安装 OpenShift CLI
您可以按照以下流程在 Windows 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 Windows Client** 条目旁的 **Download Now** 来保存文件。
4. 使用 ZIP 程序解压存档。
5. 将 **oc** 二进制文件移到 **PATH 中的目录中**。
要查看您的 **PATH**，请打开命令提示并执行以下命令：

```
C:\> path
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

在 macOS 上安装 OpenShift CLI
您可以按照以下流程在 macOS 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 macOS Client** 条目旁的 **Download Now** 来保存文件。

注意

对于 macOS arm64，请选择 **OpenShift v4.16 macOS arm64 Client** 条目。

4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
要查看您的 **PATH**，请打开终端并执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

8.3.11. 使用 CLI 登录集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

8.3.12. 使用 Web 控制台登录到集群

kubeadmin 用户默认在 OpenShift Container Platform 安装后存在。您可以使用 OpenShift Container Platform Web 控制台以 **kubeadmin** 用户身份登录集群。

先决条件

- 有访问安装主机的访问权限。
- 您完成了集群安装，所有集群 Operator 都可用。

流程

1. 从安装主机上的 **kubeadmin -password** 文件中获取 kubeadmin 用户的密码：

```
$ cat <installation_directory>/auth/kubeadmin-password
```

注意

另外，您还可以从安装主机上的 **<installation_directory>/.openshift_install.log** 日志文件获取 **kubeadmin** 密码。

2. 列出 OpenShift Container Platform Web 控制台路由：

```
$ oc get routes -n openshift-console | grep 'console-openshift'
```

注意

另外，您还可以从安装主机上的 **<installation_directory>/.openshift_install.log** **日志** 文件获取 OpenShift Container Platform 路由。

输出示例

```
console     console-openshift-console.apps.<cluster_name>.<base_domain>          console
https   reencrypt/Redirect   None
```

3. 在 Web 浏览器中导航到上一命令输出中包括的路由，以 **kubeadmin** 用户身份登录。

其他资源

- 访问Web控制台

8.3.13. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

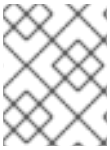
- 关于远程健康监控

8.3.14. 后续步骤

- 验证安装.
- 自定义集群。
- 如果需要，您可以选择 不使用远程健康报告。
- 如果需要，您可以删除云供应商凭证。

8.4. 使用网络自定义在 AZURE STACK HUB 上安装集群

在 OpenShift Container Platform 版本 4.16 中，您可以使用自定义的网络配置在安装程序在 Azure Stack Hub 上置备的基础架构上安装集群。通过自定义网络配置，您的集群可以与环境中现有的 IP 地址分配共存，并与现有的 MTU 和 VXLAN 配置集成。



注意

虽然您可以在使用安装程序使用安装程序置备的基础架构部署集群时选择 **azure**，但这个选项只支持 Azure Public Cloud。

8.4.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读选择集群安装方法并为用户准备它的文档。
- 已将 Azure Stack Hub 帐户配置为 托管集群。
- 如果使用防火墙，则会 将其配置为允许集群需要访问的站点。
- 您确认有大约 16 GB 的本地磁盘空间。安装集群要求您下载 RHCOS 虚拟硬盘(VHD)集群镜像，并将其上传到 Azure Stack Hub 环境，以便在部署过程中访问它。解压缩 VHD 文件需要这个数量的本地磁盘空间。

8.4.2. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类型的基础架构上执行受限网络安装。在此过程中，您可以下载所需的内容，并使用它为镜像 registry 填充安装软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群前，您要更新镜像 registry 的内容。

8.4.3. 为集群节点 SSH 访问生成密钥对

在 OpenShift Container Platform 安装过程中，您可以为安装程序提供 SSH 公钥。密钥通过它们的 Ignition 配置文件传递给 Red Hat Enterprise Linux CoreOS(RHCOS)节点，用于验证对节点的 SSH 访问。密钥添加到每个节点上 **core** 用户的 **~/.ssh/authorized_keys** 列表中，这将启用免密码身份验证。

将密钥传递给节点后，您可以使用密钥对作为用户 **核心** 通过 SSH 连接到 RHCOS 节点。若要通过 SSH 访问节点，必须由 SSH 为您的本地用户管理私钥身份。

如果要通过 SSH 连接到集群节点来执行安装调试或灾难恢复，则必须在安装过程中提供 SSH 公钥。**./openshift-install gather** 命令还需要在集群节点上设置 SSH 公钥。

重要

不要在生产环境中跳过这个过程，在生产环境中需要灾难恢复和调试。

注意

您必须使用本地密钥，而不是使用特定平台方法配置 的密钥，如 AWS 密钥对。

流程

1. 如果您在本地计算机上没有可用于在集群节点上进行身份验证的现有 SSH 密钥对，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_ed25519**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

注意

如果您计划在 **x86_64**、**ppc64le** 和 **s390x** 架构上安装使用 RHEL 加密库（这些加密库已提交给 NIST 用于 FIPS 140-2/140-3 验证）的 OpenShift Container Platform 集群，则不要创建使用 **ed25519** 算法的密钥。相反，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 查看公共 SSH 密钥：

```
$ cat <path>/<file_name>.pub
```

例如，运行以下命令来查看 ~/**.ssh/id_ed25519.pub** 公钥：

```
$ cat ~/.ssh/id_ed25519.pub
```

3. 将 SSH 私钥身份添加到本地用户的 SSH 代理（如果尚未添加）。在集群节点上，或者要使用 **./openshift-install gather** 命令，需要对该密钥进行 SSH 代理管理，才能在集群节点上进行免密码 SSH 身份验证。

注意

在某些发行版中，自动管理默认 SSH 私钥身份，如 ~/**.ssh/id_rsa** 和 ~/**.ssh/id_dsa**。

a. 如果 **ssh-agent** 进程尚未为您的本地用户运行，请将其作为后台任务启动：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果集群处于 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

4. 将 SSH 私钥添加到 **ssh-agent** ：

```
$ ssh-add <path>/<file_name> 1
```

1 指定 SSH 私钥的路径和文件名，如 ~/**.ssh/id_ed25519.pub**

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

后续步骤

- 安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

8.4.4. 上传 RHCOS 集群镜像

您必须下载 RHCOS 虚拟硬盘(VHD)集群镜像，并将其上传到 Azure Stack Hub 环境，以便在部署过程中访问它。

先决条件

- 配置 Azure 帐户。

流程

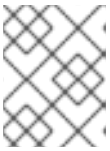
1. 获取 RHCOS VHD 集群镜像：
   a. 将 RHCOS VHD 的 URL 导出为环境变量。

```
$ export COMPRESSED_VHD_URL=$(openshift-install coreos print-stream-json | jq -r
'.architectures.x86_64.artifacts.azurestack.formats."vhd.gz".disk.location')
```

   b. 本地下载压缩的 RHCOS VHD 文件。

```
$ curl -O -L ${COMPRESSED_VHD_URL}
```

2. 解压缩 VHD 文件。



注意

解压缩的 VHD 文件大约为 16 GB，因此请确保您的主机系统有 16 GB 的可用空间。上传后，可以删除 VHD 文件。

3. 将本地 VHD 上传到 Azure Stack Hub 环境，确保 blob 已公开可用。例如，您可以使用 **az** cli 或 web 门户将 VHD 上传到 blob。

8.4.5. 获取安装程序

在安装 OpenShift Container Platform 前，将安装文件下载到您用于安装的主机上。

先决条件

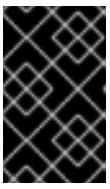
- 您有一台运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB。

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐户，请使用您的凭证登录。如果没有，请创建一个帐户。
2. 选择 **Azure** 作为云供应商。
3. 进入到安装类型的页面，下载与您的主机操作系统和架构对应的安装程序，并将该文件放在您要存储安装配置文件的目录中。

重要

安装程序会在用来安装集群的计算机上创建几个文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。



重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，请为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。此 pull secret 允许您与所含授权机构提供的服务进行身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

8.4.6. 手动创建安装配置文件

安装集群要求您手动创建安装配置文件。

先决条件

- 您在本地机器上有一个 SSH 公钥来提供给安装程序。该密钥将用于在集群节点上进行 SSH 身份验证，以进行调试和灾难恢复。
- 已获取 OpenShift Container Platform 安装程序和集群的 pull secret。

流程

1. 创建一个安装目录来存储所需的安装资产：

```
$ mkdir <installation_directory>
```

重要

您必须创建一个目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

2. 自定义提供的 **install-config.yaml** 文件模板示例，并将其保存在 **<installation_directory>** 中。

注意

此配置文件必须命名为 **install-config.yaml**。

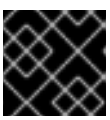
进行以下修改：

a. 指定所需的安装参数。

b. 更新 **platform.azure** 部分，以指定特定于 Azure Stack Hub 的参数。

c. 可选：更新一个或多个默认配置参数以自定义安装。
有关参数的更多信息，请参阅"安装配置参数"。

3. 备份 **install-config.yaml** 文件，以便您可以使用它安装多个集群。



重要

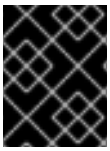
install-config.yaml 文件会在安装过程的下一步中使用。现在必须备份它。

其他资源

- Azure Stack Hub 的安装配置参数

8.4.6.1. Azure Stack Hub 的自定义 install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多详情，或修改所需参数的值。



重要

此示例 YAML 文件仅供参考。使用它作为资源，在您手动创建的安装配置文件中输入参数值。

```
apiVersion: v1
baseDomain: example.com (1)
credentialsMode: Manual
controlPlane: (2) (3)
  name: master
  platform:
    azure:
      osDisk:
        diskSizeGB: 1024 (4)
        diskType: premium_LRS
  replicas: 3
compute: (5)
- name: worker
  platform:
    azure:
      osDisk:
        diskSizeGB: 512 (6)
        diskType: premium_LRS
  replicas: 3
metadata:
  name: test-cluster (7) (8)
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OVNKubernetes (9)
  serviceNetwork:
  - 172.30.0.0/16
platform:
  azure:
    armEndpoint: azurestack_arm_endpoint (10) (11)
    baseDomainResourceGroupName: resource_group (12) (13)
    region: azure_stack_local_region (14) (15)
    resourceGroupName: existing_resource_group (16)
    outboundType: Loadbalancer
    cloudName: AzureStackCloud (17)
    clusterOSimage: https://vhdsa.blob.example.example.com/vhd/rhcos-410.84.202112040202-0-
azurestack.x86_64.vhd (18) (19)
pullSecret: '{"auths": ...}' (20) (21)
```

```
fips: false 22
sshKey: ssh-ed25519 AAAA... 23
additionalTrustBundle: | 24
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
```

1 7 10 12 14 17 18 20 必需。

2 5 如果没有提供这些参数和值，安装程序会提供默认值。

3 **controlPlane** 部分是一个单个映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，controlPlane **部分** 的第一行则不以连字符开头。虽然这两个部分目前都定义了单一机器池，但未来的 OpenShift Container Platform 版本可能在安装过程中支持定义多个计算池。仅使用一个 control plane 池。

4 6 您可以指定要使用的磁盘大小（以 GB 为单位）。control plane 节点的最低推荐值为 1024 GB。

8 集群的名称。

9 要安装的集群网络插件。默认值 **OVNKubernetes** 是唯一支持的值。

11 Azure Stack Hub Operator 提供的 Azure Resource Manager 端点。

13 包含基域的 DNS 区的资源组的名称。

15 Azure Stack Hub 本地区域的名称。

16 安装集群的现有资源组的名称。如果未定义，则会为集群创建新的资源组。

19 Azure Stack 环境中包含 RHCOS VHD 存储 blob 的 URL。

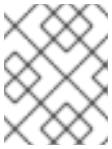
21 对集群进行身份验证所需的 pull secret。

22 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS)时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。

23 您可以选择提供您用来访问集群中机器的 **sshKey** 值。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

24 如果 Azure Stack Hub 环境使用内部证书颁发机构(CA)，则需要添加 CA 证书。

8.4.7. 手动管理云凭证

Cloud Credential Operator(CCO)只支持手动模式的云供应商。因此，您必须为云供应商指定身份和访问管理(IAM)secret。

流程

1. 如果您之前还没有创建安装清单文件，请运行以下命令：

```
$ openshift-install create manifests --dir <installation_directory>
```

其中 **<installation_directory>** 是安装程序在其中创建文件的目录。

2. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

3. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 自定义资源 (CR) 列表：

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
  --included \ 1
  --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
  --to=<path_to_directory_for_credentials_requests> 3
```

1. **--included** 参数仅包含特定集群配置所需的清单。
2. 指定 **install-config.yaml** 文件的位置。
3. 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

此命令为每个 **CredentialsRequest** 对象创建一个 YAML 文件。

CredentialsRequest 对象示例

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component_credentials_request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: AzureProviderSpec
    roleBindings:
    - role: Contributor
  ...
```

4. 在之前生成的 **openshift-install** 清单目录中为 secret 创建 YAML 文件。secret 必须使用在 **spec.secretRef** 中为每个 **CredentialsRequest** 定义的命名空间和 secret 名称存储。

   带有 secret 的 CredentialsRequest 对象示例

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component_credentials_request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: AzureProviderSpec
    roleBindings:
    - role: Contributor
      ...
  secretRef:
    name: <component_secret>
    namespace: <component_namespace>
  ...
```

   Secret 对象示例

```
apiVersion: v1
kind: Secret
metadata:
  name: <component_secret>
  namespace: <component_namespace>
data:
  azure_subscription_id: <base64_encoded_azure_subscription_id>
  azure_client_id: <base64_encoded_azure_client_id>
  azure_client_secret: <base64_encoded_azure_client_secret>
  azure_tenant_id: <base64_encoded_azure_tenant_id>
  azure_resource_prefix: <base64_encoded_azure_resource_prefix>
  azure_resourcegroup: <base64_encoded_azure_resourcegroup>
  azure_region: <base64_encoded_azure_region>
```

重要

在升级使用手动维护凭证的集群前，您必须确保 CCO 处于可升级状态。

其他资源

- 使用 Web 控制台更新集群
- 使用 CLI 更新集群

8.4.8. 配置集群以使用内部 CA

如果 Azure Stack Hub 环境使用内部证书颁发机构(CA)，请更新 **cluster-proxy-01-config.yaml 文件** 将集群配置为使用内部 CA。

先决条件

- 创建 **install-config.yaml** 文件，并以 **.pem** 格式指定证书信任捆绑包。
- 创建集群清单。

流程

1. 从安装程序创建文件的目录中，前往 **manifests** 目录。
2. 将 **user-ca-bundle** 添加到 **spec.trustedCA.name** 字段中。

   cluster-proxy-01-config.yaml 文件示例

```
apiVersion: config.openshift.io/v1
kind: Proxy
metadata:
  creationTimestamp: null
  name: cluster
spec:
  trustedCA:
    name: user-ca-bundle
status: {}
```

3. 可选：备份 **manifests/ cluster-proxy-01-config.yaml** 文件。在部署集群时，安装程序会消耗 **manifests/** 目录。

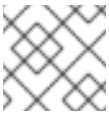
8.4.9. 网络配置阶段

OpenShift Container Platform 安装前有两个阶段，您可以在其中自定义网络配置。

第 1 阶段

在创建清单文件前，您可以自定义 **install-config.yaml** 文件中的以下与网络相关的字段：

- **networking.networkType**
- **networking.clusterNetwork**
- **networking.serviceNetwork**
- **networking.machineNetwork**
  有关这些字段的更多信息，请参阅 *安装配置参数*。



注意

将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

重要

CIDR 范围 **172.17.0.0/16** 由 libVirt 保留。对于集群中的任何网络，您无法使用此范围或与这个范围重叠的范围。

第 2 阶段

运行 **openshift-install create 清单创建** 清单文件后，您可以只使用您要修改的字段定义自定义 Cluster Network Operator 清单。您可以使用 清单指定高级网络配置。

您不能覆盖在 stage 2 阶段 1 中在 **install-config.yaml** 文件中指定的值。但是，您可以在第 2 阶段进一步自定义网络插件。

8.4.10. 指定高级网络配置

您可以使用网络插件的高级网络配置将集群集成到现有网络环境中。您只能在安装集群前指定高级网络配置。

重要

不支持通过修改安装程序创建的 OpenShift Container Platform 清单文件来自定义网络配置。支持应用您创建的清单文件，如以下流程中所示。

先决条件

- 您已创建 **install-config.yaml** 文件并完成对其所做的任何修改。

流程

1. 进入包含安装程序的目录并创建清单：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 **<installation_directory>** 指定包含集群的 **install-config.yaml** 文件的目录名称。

2. 在 **<installation_directory>/manifests/** 目录中 为**高级网络配置创建一个名为 cluster-network-03-config.yml** 的 stub 清单文件：

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
```

3. 在 **cluster-network-03-config.yml** 文件中指定集群的高级网络配置，如下例所示：

   为 OVN-Kubernetes 网络供应商启用 IPsec

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  defaultNetwork:
    ovnKubernetesConfig:
      ipsecConfig:
        mode: Full
```

4. 可选：备份 **manifests/cluster-network-03-config.yml** 文件。创建 Ignition 配置文件时，安装程序会使用 **manifests/** 目录。

8.4.11. Cluster Network Operator 配置

集群网络的配置作为 Cluster Network Operator(CNO)配置的一部分指定，并存储在名为 **cluster** 的自定义资源(CR)对象中。CR 指定 **operator.openshift.io** API 组中的 **Network** API 的字段。

CNO 配置在集群安装过程中从 **Network.config.openshift.io** API 组中的 **Network** API 继承以下字段：

clusterNetwork

从中分配 Pod IP 地址的 IP 地址池。

serviceNetwork

服务的 IP 地址池.

defaultNetwork.type

集群网络插件。**OVNKubernetes** 是安装期间唯一支持的插件。

您可以通过在名为 **cluster** 的 CNO 对象中设置 **defaultNetwork** 对象的字段来为集群指定集群网络插件配置。

8.4.11.1. Cluster Network Operator 配置对象

下表中描述了 Cluster Network Operator(CNO)的字段：

表 8.1. Cluster Network Operator 配置对象

字段	类型	描述
metadata.name	**字符串**	CNO 对象的名称。这个名称始终是 **集群**。

spec.clusterNetwork	**array**	用于指定从哪些 IP 地址块分配 Pod IP 地址以及集群中每个节点的子网前缀长度的列表。例如：
		`spec:`
		`  clusterNetwork:`
		`  - cidr: 10.128.0.0/19`
		`  hostPrefix: 23`
		`  - cidr: 10.128.32.0/19`
		`  hostPrefix: 23`

spec.serviceNetwork	**array**	服务的 IP 地址块。OpenShift SDN 和 OVN-Kubernetes 网络插件只支持服务网络的一个 IP 地址块。例如：
		`spec:`
		`  serviceNetwork:`
		`  - 172.30.0.0/14`
		您只能在创建清单前在 **install-config.yaml** 文件中自定义此字段。该值在清单文件中是只读的。

spec.defaultNetwork	**object**	为集群网络配置网络插件。

字段	类型	描述
spec.kubeProxyConfig	**object**	此对象的字段指定 kube-proxy 配置。如果使用 OVN-Kubernetes 集群网络供应商，则 kube-proxy 配置不会起作用。

defaultNetwork 对象配置
下表列出了 **defaultNetwork** 对象的值：

表 8.2. defaultNetwork 对象

字段	类型	描述
type	**字符串**	**OVNKubernetes**。Red Hat OpenShift Networking 网络插件在安装过程中被选择。此值在集群安装后无法更改。 **注意** OpenShift Container Platform 默认使用 OVN-Kubernetes 网络插件。OpenShift SDN 不再作为新集群的安装选择提供。
ovnKubernetesConfig	**object**	此对象仅对 OVN-Kubernetes 网络插件有效。

配置 OVN-Kubernetes 网络插件
下表描述了 OVN-Kubernetes 网络插件的配置字段：

表 8.3. ovnKubernetesConfig object

字段	类型	描述
mtu	**integer**	Geneve（通用网络虚拟化封装）覆盖网络的最大传输单元 (MTU)。这根据主网络接口的 MTU 自动探测。您通常不需要覆盖检测到的 MTU。 如果自动探测的值不是您期望的值，请确认节点上主网络接口上的 MTU 是否正确。您不能使用这个选项更改节点上主网络接口的 MTU 值。 如果集群中不同节点需要不同的 MTU 值，则必须将此值设置为**比** 集群中的最低 MTU 值小 100。例如，如果集群中的某些节点的 MTU 为 **9001**，而某些节点的 MTU 为 **1500**，则必须将此值设置为 **1400**。
genevePort	**integer**	用于所有 Geneve 数据包的端口。默认值为 **6081**。此值在集群安装后无法更改。
ipsecConfig	**object**	指定用于自定义 IPsec 配置的配置对象。

字段	类型	描述
ipv4	**object**	为 IPv4 设置指定配置对象。
ipv6	**object**	为 IPv6 设置指定配置对象。
policyAuditConfig	**object**	指定用于自定义网络策略审计日志的配置对象。如果未设置，则使用默认的审计日志设置。
gatewayConfig	**object**	可选：指定一个配置对象来自定义如何将出口流量发送到节点网关。 **注意** 在迁移出口流量时，工作负载和服务流量会受到一定影响，直到 Cluster Network Operator (CNO) 成功推出更改。

表 8.4. **ovnKubernetesConfig.ipv4** object

字段	类型	描述
internalTransitSwitchSubnet	字符串	如果您的现有网络基础架构与 **100.88.0.0/16** IPv4 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。启用东西流量的分布式传输交换机的子网。此子网不能与 OVN-Kubernetes 或主机本身使用的任何其他子网重叠。必须足够大，以适应集群中的每个节点一个 IP 地址。 默认值为 **100.88.0.0/16**。
internalJoinSubnet	字符串	如果您的现有网络基础架构与 **100.64.0.0/16** IPv4 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。您必须确保 IP 地址范围没有与 OpenShift Container Platform 安装使用的任何其他子网重叠。IP 地址范围必须大于可添加到集群的最大节点数。例如，如果 **clusterNetwork.cidr** 值为 **10.128.0.0/14**，并且 **clusterNetwork.hostPrefix** 值为 **/23**，则最大节点数量为 **2^(23-14)=512**。 默认值为 **100.64.0.0/16**。

表 8.5. **ovnKubernetesConfig.ipv6** object

字段	类型	描述

字段	类型	描述
internalTransitSwitchSubnet	字符串	如果您的现有网络基础架构与 **fd97::/64** IPv6 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。启用东西流量的分布式传输交换机的子网。此子网不能与 OVN-Kubernetes 或主机本身使用的任何其他子网重叠。必须足够大，以适应集群中的每个节点一个 IP 地址。 默认值为 **fd97::/64**。
internalJoinSubnet	字符串	如果您的现有网络基础架构与 **fd98::/64** IPv6 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。您必须确保 IP 地址范围没有与 OpenShift Container Platform 安装使用的任何其他子网重叠。IP 地址范围必须大于可添加到集群的最大节点数。 默认值为 **fd98::/64**。

表 8.6. **policyAuditConfig** object

字段	类型	描述
rateLimit	整数	每个节点每秒生成一次的消息数量上限。默认值为每秒 **20** 条消息。
maxFileSize	整数	审计日志的最大大小，以字节为单位。默认值为 **50000000** 或 50 MB。
maxLogFiles	整数	保留的日志文件的最大数量。
目的地	字符串	以下附加审计日志目标之一： **libc** 主机上的 journald 进程的 libc **syslog（）** 函数。 **UDP:<host>:<port>** 一个 syslog 服务器。将 **<host>:<port>** 替换为 **syslog 服务器的主机** 和端口。 **Unix:<file>** 由 **<file>** 指定的 Unix 域套接字文件。 **null** 不要将审计日志发送到任何其他目标。
syslogFacility	字符串	syslog 工具，如 as **kern**，如 RFC5424 定义。默认值为 **local0**。

表 8.7. **gatewayConfig** object

字段	类型	描述
routingViaHost	**布尔值**	将此字段设置为 **true**，将来自 pod 的出口流量发送到主机网络堆栈。对于依赖于在内核路由表中手动配置路由的高级别安装和应用程序，您可能需要将出口流量路由到主机网络堆栈。默认情况下，出口流量在 OVN 中进行处理以退出集群，不受内核路由表中的特殊路由的影响。默认值为 **false**。 此字段与 Open vSwitch 硬件卸载功能有交互。如果将此字段设置为 **true**，则不会获得卸载的性能优势，因为主机网络堆栈会处理出口流量。
ipForwarding	**object**	您可以使用 **Network** 资源中的 **ipForwarding** 规格来控制 OVN-Kubernetes 管理接口上所有流量的 IP 转发。指定 **Restricted** 只允许 Kubernetes 相关流量的 IP 转发。指定 **Global** 以允许转发所有 IP 流量。对于新安装，默认值为 **Restricted**。对于到 OpenShift Container Platform 4.14 或更高版本的更新，默认值为 **Global**。
ipv4	**object**	可选：指定一个对象来为主机配置内部 OVN-Kubernetes 伪装地址，以服务 IPv4 地址的流量。
ipv6	**object**	可选：指定一个对象来为主机配置内部 OVN-Kubernetes 伪装地址，以服务 IPv6 地址的流量。

表 8.8. **gatewayConfig.ipv4** 对象

字段	类型	描述
internalMasqueradeSubnet	**字符串**	内部使用的伪装 IPv4 地址，以启用主机服务流量。主机配置了这些 IP 地址和共享网关网桥接口。默认值为 **169.254.169.0/29**。

表 8.9. **gatewayConfig.ipv6** 对象

字段	类型	描述
internalMasqueradeSubnet	**字符串**	内部使用的伪装 IPv6 地址，以启用主机服务流量。主机配置了这些 IP 地址和共享网关网桥接口。默认值为 **fd69::/125**。

表 8.10. **ipsecConfig** 对象

字段	类型	描述

字段	类型	描述
模式	**字符串**	指定 IPsec 实现的行为。必须是以下值之一： • **Disabled**: 在集群节点上不启用 IPsec。 • **External** ：对于带有外部主机的网络流量，启用 IPsec。 • **Full**: IPsec 为带有外部主机的 pod 流量和网络流量启用 IPsec。

启用 IPSec 的 OVN-Kubernetes 配置示例

```
defaultNetwork:
  type: OVNKubernetes
  ovnKubernetesConfig:
    mtu: 1400
    genevePort: 6081
      ipsecConfig:
        mode: Full
```

重要

使用 OVNKubernetes 可能会导致 IBM Power® 上的堆栈耗尽问题。

kubeProxyConfig 对象配置（仅限 OpenShiftSDN 容器网络接口）
kubeProxyConfig 对象的值在下表中定义：

表 8.11. kubeProxyConfig object

字段	类型	描述
iptablesSyncPeriod	**字符串**	**iptables** 规则的刷新周期。默认值为 **30s**。有效的后缀包括 **s**、**m** 和 **h**，具体参见 Go 时间 包 文档。  **注意** 由于 OpenShift Container Platform 4.3 及更高版本中引进了性能改进，不再需要调整 **iptablesSyncPeriod** 参数。

字段	类型	描述
proxyArguments.iptables-min-sync-period	**array**	刷新 **iptables** 规则前的最短持续时间。此字段确保刷新的频率不会过于频繁。有效的后缀包括 **s**、**m** 和 **h**，具体参见 Go **time** 软件包。默认值为： kubeProxyConfig: proxyArguments: iptables-min-sync-period: - 0s

8.4.12. 使用 OVN-Kubernetes 配置混合网络

您可以将集群配置为使用 OVN-Kubernetes 网络插件的混合网络。这允许支持不同节点网络配置的混合集群。

先决条件

- 您在 **install-config.yaml** 文件中为 **networking.networkType** 参数定义了 **OVNKubernetes**。如需更多信息，请参阅有关在所选云供应商上配置 OpenShift Container Platform 网络自定义的安装文档。

流程

1. 进入包含安装程序的目录并创建清单：

```
$ ./openshift-install create manifests --dir <installation_directory>
```

其中：

<installation_directory>
指定包含集群的 **install-config.yaml** 文件的目录名称。

2. 在 **<installation_directory>/manifests/** 目录中为高级网络配置创建一个名为 **cluster-network-03-config.yml** 的 stub 清单文件：

```
$ cat <<EOF > <installation_directory>/manifests/cluster-network-03-config.yml
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
EOF
```

其中：

<installation_directory>
指定包含集群的 **manifests/** 目录的目录名称。

3. 在编辑器中打开 **cluster-network-03-config.yml** 文件，并使用混合网络配置 OVN-Kubernetes，如下例所示：

指定混合网络配置

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  defaultNetwork:
    ovnKubernetesConfig:
      hybridOverlayConfig:
        hybridClusterNetwork: (1)
        - cidr: 10.132.0.0/14
          hostPrefix: 23
```

(1) 指定用于额外覆盖网络上节点的 CIDR 配置。**hybridClusterNetwork** CIDR 无法与 **clusterNetwork** CIDR 重叠。

4. 保存 **cluster-network-03-config.yml** 文件，再退出文本编辑器。
5. 可选：备份 **manifests/cluster-network-03-config.yml** 文件。创建集群时，安装程序会删除 **manifests/** 目录。

8.4.13. 部署集群

您可以在兼容云平台上安装 OpenShift Container Platform。

重要

在初始安装过程中，您只能运行安装程序的 **create cluster** 命令一次。

先决条件

- 您已使用托管集群的云平台配置了帐户。
- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。
- 已确认主机上的云供应商帐户具有部署集群的正确权限。权限不正确的帐户会导致安装过程失败，并显示包括缺失权限的错误消息。

流程

- 进入包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ (1)
    --log-level=info (2)
```

(1) 对于 **<installation_directory>**，请指定自定义 **./install-config.yaml** 文件的位置。

(2) 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不是 **info**。

验证

当集群部署成功完成时：

- 终端会显示用于访问集群的说明，包括指向 Web 控制台和 **kubeadmin** 用户的凭证的链接。
- 凭证信息还会输出到 **<installation_directory>/.openshift_install.log**.

重要

不要删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

8.4.14. 安装 OpenShift CLI

您可以安装 OpenShift CLI(**oc**)来使用命令行界面与 OpenShift Container Platform 进行交互。您可以在 Linux、Windows 或 macOS 上安装 **oc**。

重要

如果安装了旧版本的 **oc**，则可能无法使用 OpenShift Container Platform 4.16 中的所有命令。下载并安装新版本的 **oc**。

在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **产品变体** 下拉列表中选择架构。
3. 从 **版本** 下拉列表中选择适当的版本。
4. 点 **OpenShift v4.16 Linux Client** 条目旁的 **Download Now** 来保存文件。

5. 解包存档：

```
$ tar xvf <file>
```

6. 将 **oc** 二进制文件放到 **PATH 中的目录中**。
要查看您的 **PATH**，请执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

在 Windows 上安装 OpenShift CLI
您可以按照以下流程在 Windows 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 Windows Client** 条目旁的 **Download Now** 来保存文件。
4. 使用 ZIP 程序解压存档。
5. 将 **oc** 二进制文件移到 **PATH 中的目录中**。
要查看您的 **PATH**，请打开命令提示并执行以下命令：

```
C:\> path
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

在 macOS 上安装 OpenShift CLI
您可以按照以下流程在 macOS 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 macOS Client** 条目旁的 **Download Now** 来保存文件。

注意

对于 macOS arm64，请选择 **OpenShift v4.16 macOS arm64 Client** 条目。

4. 解包和解压存档。

5. 将 **oc** 二进制文件移到 PATH 的目录中。
要查看您的 **PATH**，请打开终端并执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

8.4.15. 使用 CLI 登录集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

8.4.16. 使用 Web 控制台登录到集群

kubeadmin 用户默认在 OpenShift Container Platform 安装后存在。您可以使用 OpenShift Container Platform Web 控制台以 **kubeadmin** 用户身份登录集群。

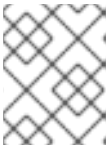
先决条件

- 有访问安装主机的访问权限。
- 您完成了集群安装，所有集群 Operator 都可用。

流程

1. 从安装主机上的 **kubeadmin -password** 文件中获取 kubeadmin 用户的密码：

```
$ cat <installation_directory>/auth/kubeadmin-password
```

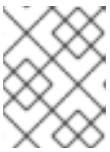


注意

另外，您还可以从安装主机上的 **<installation_directory>/.openshift_install.log** 日志文件获取 **kubeadmin** 密码。

2. 列出 OpenShift Container Platform Web 控制台路由：

```
$ oc get routes -n openshift-console | grep 'console-openshift'
```



注意

另外，您还可以从安装主机上的 **<installation_directory>/.openshift_install.log** **日志** 文件获取 OpenShift Container Platform 路由。

输出示例

```
console     console-openshift-console.apps.<cluster_name>.<base_domain>           console
https   reencrypt/Redirect   None
```

3. 在 Web 浏览器中导航到上一命令输出中包括的路由，以 **kubeadmin** 用户身份登录。

其他资源

- 访问 Web 控制台。

8.4.17. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 关于远程健康监控

8.4.18. 后续步骤

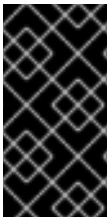
- 验证安装.
- 自定义集群。
- 如果需要，您可以选择 不使用远程健康报告。

- 如果需要，您可以删除云供应商凭证。

8.5. 使用 ARM 模板在 AZURE STACK HUB 上安装集群

在 OpenShift Container Platform 版本 4.16 中，您可以使用您提供的基础架构在 Microsoft Azure Stack Hub 上安装集群。

提供的几个 Azure Resource Manager（ARM） 模板可协助完成这些步骤，也可帮助您自行建模。



重要

进行用户置备的基础架构安装的步骤仅作为示例。使用您提供的基础架构安装集群需要了解云供应商和 OpenShift Container Platform 安装过程。提供的几个 ARM 模板可帮助完成这些步骤，或帮助您自行建模。您也可以自由选择通过其他方法创建所需的资源；模板仅作示例之用。

8.5.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读选择集群安装方法并为用户准备它的文档。
- 已将 Azure Stack Hub 帐户配置为 托管集群。
- 您下载了 Azure CLI 并安装到您的计算机上。请参阅 Azure 文档中的安装 Azure CLI。以下文档使用 Azure CLI 的版本 **2.28.0** 测试。Azure CLI 命令可能会根据您使用的版本的不同而不同。
- 如果使用防火墙并计划使用 Telemetry 服务，需要将防火墙配置为允许集群需要访问的站点。

注意

如果您要配置代理，请务必也要查看此站点列表。

8.5.2. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类型的基础架构上执行受限网络安装。在此过程中，您可以下载所需的内容，并使用它为镜像 registry 填充安装软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群之前，要更新 registry 镜像系统中的内容。

8.5.3. 配置 Azure Stack Hub 项目

在安装 OpenShift Container Platform 之前，您必须配置 Azure 项目来托管它。

重要

所有通过公共端点提供的 Azure Stack Hub 资源都受到资源名称的限制，您无法创建使用某些术语的资源。如需 Azure Stack Hub 限制词语列表，请参阅 Azure 文档中的解决保留资源名称错误。

8.5.3.1. Azure Stack Hub 帐户限制

OpenShift Container Platform 集群使用多个 Microsoft Azure Stack Hub 组件，Azure Stack Hub 中的默认配额类型会影响您安装 OpenShift Container Platform 集群的能力。

下表总结了 Azure Stack Hub 组件，它们的限值会影响您安装和运行 OpenShift Container Platform 集群的能力。

组件	默认所需的组件数	描述
vCPU	56	默认集群需要 56 个 vCPU，因此您必须提高帐户限值。 默认情况下，每个集群创建以下实例： • 一台 Bootstrap 机器，在安装后删除 • 三个 control plane 机器 • 三个计算（compute）机器 因为 bootstrap、control plane 和 worker 机器使用 **Standard_DS4_v2** 虚拟机（8 个 vCPU），所以默认集群需要 56 个 vCPU。bootstrap 节点虚拟机仅在安装过程中使用。 若要部署更多 worker 节点、启用自动扩展、部署大型工作负载或使用不同的实例类型，您必须进一步提高帐户的 vCPU 限值，以确保集群可以部署您需要的机器。
VNet	1	每个默认集群都需要一个虚拟网络 (VNet)，此网络包括两个子网。
网络接口	7	每个默认集群都需要 7 个网络接口。如果您要创建更多机器或者您部署的工作负载要创建负载均衡器，则集群会使用更多的网络接口。
网络安全组	2	每个集群为 VNet 中的每个子网创建网络安全组。默认集群为 control plane 和计算节点子网创建网络安全组： **controlplane**：允许从任何位置通过端口 6443 访问 control plane 机器 **node**：允许从互联网通过端口 80 和 443 访问 worker 节点

<table>
<tr><th>组件</th><th>默认所需的组件数</th><th>描述</th></tr>
<tr><td>网络负载均衡器</td><td>3</td><td>每个集群都会创建以下负载均衡器：<table><tr><td>default</td><td>用于在 worker 机器之间对端口 80 和 443 的请求进行负载均衡的公共 IP 地址</td></tr><tr><td>internal</td><td>用于在 control plane 机器之间对端口 6443 和 22623 的请求进行负载均衡的专用 IP 地址</td></tr><tr><td>external</td><td>用于在 control plane 机器之间对端口 6443 的请求进行负载均衡的公共 IP 地址</td></tr></table>如果您的应用程序创建了更多的 Kubernetes LoadBalancer 服务对象，您的集群会使用更多的负载均衡器。</td></tr>
<tr><td>公共 IP 地址</td><td>2</td><td>公共负载均衡器使用一个公共 IP 地址。bootstrap 机器也使用一个公共 IP 地址，以便您可以在安装期间通过 SSH 连接到该机器来进行故障排除。bootstrap 节点的 IP 地址仅在安装过程中使用。</td></tr>
<tr><td>专用 IP 地址</td><td>7</td><td>内部负载均衡器、三台 control plane 机器中的每一台以及三台 worker 机器中的每一台各自使用一个专用 IP 地址。</td></tr>
</table>

其他资源

- 优化存储.

8.5.3.2. 在 Azure Stack Hub 中配置 DNS 区域

要在 Azure Stack Hub 上成功安装 OpenShift Container Platform，您必须在 Azure Stack Hub DNS 区域中创建 DNS 记录。DNS 区域必须对域具有权威。要将注册商的 DNS 区域委派给 Azure Stack Hub，请参阅 Microsoft 有关 Azure Stack Hub 数据中心 DNS 集成 的文档。

您可以通过访问此创建 DNS 区域示例来查看 Azure 的 DNS 解决方案。

8.5.3.3. 证书签名请求管理

在使用您置备的基础架构时，集群只能有限地访问自动机器管理，因此您必须提供一种在安装后批准集群证书签名请求 (CSR) 的机制。**kube-controller-manager** 只能批准 kubelet 客户端 CSR。 **machine-approver** 无法保证使用 kubelet 凭证请求的提供证书的有效性，因为它不能确认是正确的机器发出了该请求。您必须决定并实施一种方法，以验证 kubelet 提供证书请求的有效性并进行批准。

8.5.3.4. 所需的 Azure Stack Hub 角色

Microsoft Azure Stack Hub 帐户必须具有您使用的订阅的以下角色：

- **所有者**

要在 Azure 门户上设置角色，请参阅 Microsoft 文档中的通过基于角色的访问控制管理对 Azure Stack Hub 中资源的管理访问权限。

8.5.3.5. 创建服务主体

由于 OpenShift Container Platform 及其安装程序使用 Azure Resource Manager 创建 Microsoft Azure 资源，因此您必须创建一个服务主体来代表它。

先决条件

- 安装或更新 Azure CLI。
- 您的 Azure 帐户具有您所用订阅所需的角色。

流程

1. 注册您的环境：

```
$ az cloud register -n AzureStackCloud --endpoint-resource-manager <endpoint> 1
```

1 指定 Azure Resource Manager 端点 'https://management.<region>.<fqdn>/'。

   详情请查看 Microsoft 文档。

2. 设置活跃环境：

```
$ az cloud set -n AzureStackCloud
```

3. 更新您的环境配置，以使用 Azure Stack Hub 的特定 API 版本：

```
$ az cloud update --profile 2019-03-01-hybrid
```

4. 登录 Azure CLI：

```
$ az login
```

   如果您在多租户环境中，还必须提供租户 ID。

5. 如果您的 Azure 帐户使用订阅，请确定您使用正确的订阅：

   a. 查看可用帐户列表并记录您要用于集群的订阅的 **tenantId** 值：

```
$ az account list --refresh
```

   输出示例

```
[
  {
    "cloudName": AzureStackCloud",
    "id": "9bab1460-96d5-40b3-a78e-17b15e978a80",
    "isDefault": true,
    "name": "Subscription Name",
    "state": "Enabled",
    "tenantId": "6057c7e9-b3ae-489d-a54e-de3f6bf6a8ee",
    "user": {
      "name": "you@example.com",
```

```
    "type": "user"
   }
  }
]
```

b. 查看您的活跃帐户详情，确认 **tenantId** 值与您要使用的订阅匹配：

```
$ az account show
```

输出示例

```
{
  "environmentName": AzureStackCloud",
  "id": "9bab1460-96d5-40b3-a78e-17b15e978a80",
  "isDefault": true,
  "name": "Subscription Name",
  "state": "Enabled",
  "tenantId": "6057c7e9-b3ae-489d-a54e-de3f6bf6a8ee", (1)
  "user": {
    "name": "you@example.com",
    "type": "user"
  }
}
```

(1) 确保 **tenantId** 参数的值是正确的订阅 ID。

c. 如果您使用的订阅不正确，请更改活跃的订阅：

```
$ az account set -s <subscription_id> (1)
```

(1) 指定订阅 ID。

d. 验证订阅 ID 更新：

```
$ az account show
```

输出示例

```
{
  "environmentName": AzureStackCloud",
  "id": "33212d16-bdf6-45cb-b038-f6565b61edda",
  "isDefault": true,
  "name": "Subscription Name",
  "state": "Enabled",
  "tenantId": "8049c7e9-c3de-762d-a54e-dc3f6be6a7ee",
  "user": {
    "name": "you@example.com",
    "type": "user"
  }
}
```

6. 记录输出中的 **tenantId** 和 **id** 参数值。OpenShift Container Platform 安装过程中需要这些值。

7. 为您的帐户创建服务主体：

```
$ az ad sp create-for-rbac --role Contributor --name <service_principal> \ 1
  --scopes /subscriptions/<subscription_id> 2
  --years <years> 3
```

1. 指定服务主体名称。
2. 指定订阅 ID。
3. 指定年数。默认情况下，服务主体在一年后过期。通过使用 **--years** 选项，您可以扩展服务主体的有效性。

输出示例

```
Creating 'Contributor' role assignment under scope '/subscriptions/<subscription_id>'
The output includes credentials that you must protect. Be sure that you do not
include these credentials in your code or check the credentials into your source
control. For more information, see https://aka.ms/azadsp-cli
{
  "appId": "ac461d78-bf4b-4387-ad16-7e32e328aec6",
  "displayName": <service_principal>",
  "password": "00000000-0000-0000-0000-000000000000",
  "tenantId": "8049c7e9-c3de-762d-a54e-dc3f6be6a7ee"
}
```

8. 记录前面输出中 **appId** 和 **password** 参数的值。OpenShift Container Platform 安装过程中需要这些值。

其他资源

- 如需有关 CCO 模式的更多信息，请参阅 关于 Cloud Credential Operator。

8.5.4. 获取安装程序

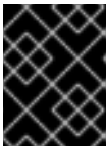
在安装 OpenShift Container Platform 前，将安装文件下载到您用于安装的主机上。

先决条件

- 您有一台运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB。

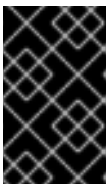
流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐户，请使用您的凭证登录。如果没有，请创建一个帐户。
2. 选择 **Azure** 作为云供应商。
3. 进入到安装类型的页面，下载与您的主机操作系统和架构对应的安装程序，并将该文件放在您要存储安装配置文件的目录中。



重要

安装程序会在用来安装集群的计算机上创建几个文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。



重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，请为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。此 pull secret 允许您与所含授权机构提供的服务进行身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

8.5.5. 为集群节点 SSH 访问生成密钥对

在 OpenShift Container Platform 安装过程中，您可以为安装程序提供 SSH 公钥。密钥通过它们的 Ignition 配置文件传递给 Red Hat Enterprise Linux CoreOS(RHCOS)节点，用于验证对节点的 SSH 访问。密钥添加到每个节点上 **core** 用户的 **~/.ssh/authorized_keys** 列表中，这将启用免密码身份验证。

将密钥传递给节点后，您可以使用密钥对作为用户 **核心** 通过 SSH 连接到 RHCOS 节点。若要通过 SSH 访问节点，必须由 SSH 为您的本地用户管理私钥身份。

如果要通过 SSH 连接到集群节点来执行安装调试或灾难恢复，则必须在安装过程中提供 SSH 公钥。**./openshift-install gather** 命令还需要在集群节点上设置 SSH 公钥。

重要

不要在生产环境中跳过这个过程，在生产环境中需要灾难恢复和调试。

注意

您必须使用本地密钥，而不是使用特定平台方法配置 的密钥，如 AWS 密钥对。

流程

1. 如果您在本地计算机上没有可用于在集群节点上进行身份验证的现有 SSH 密钥对，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_ed25519**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

注意

如果您计划在 **x86_64**、**ppc64le** 和 **s390x** 架构上安装使用 RHEL 加密库（这些加密库已提交给 NIST 用于 FIPS 140-2/140-3 验证）的 OpenShift Container Platform 集群，则不要创建使用 **ed25519** 算法的密钥。相反，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 查看公共 SSH 密钥：

```
$ cat <path>/<file_name>.pub
```

例如，运行以下命令来查看 **~/.ssh/id_ed25519.pub** 公钥：

```
$ cat ~/.ssh/id_ed25519.pub
```

3. 将 SSH 私钥身份添加到本地用户的 SSH 代理（如果尚未添加）。在集群节点上，或者要使用 **./openshift-install gather** 命令，需要对该密钥进行 SSH 代理管理，才能在集群节点上进行免密码 SSH 身份验证。

注意

在某些发行版中，自动管理默认 SSH 私钥身份，如 **~/.ssh/id_rsa** 和 **~/.ssh/id_dsa**。

a. 如果 **ssh-agent** 进程尚未为您的本地用户运行，请将其作为后台任务启动：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果集群处于 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

4. 将 SSH 私钥添加到 **ssh-agent** ：

```
$ ssh-add <path>/<file_name> (1)
```

(1) 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_ed25519.pub**

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

8.5.6. 为 Azure Stack Hub 创建安装文件

要使用用户置备的基础架构在 Microsoft Azure Stack Hub 上安装 OpenShift Container Platform，您必须生成安装程序部署集群所需的文件，并进行修改，以便集群只创建要使用的机器。您可以手动创建 **install-config.yaml** 文件，然后生成并自定义 Kubernetes 清单和 Ignition 配置文件。您也可以选择在安装准备阶段首先设置独立的 **var** 分区。

8.5.6.1. 手动创建安装配置文件

安装集群要求您手动创建安装配置文件。

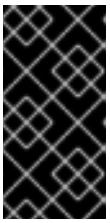
先决条件

- 您在本地机器上有一个 SSH 公钥来提供给安装程序。该密钥将用于在集群节点上进行 SSH 身份验证，以进行调试和灾难恢复。
- 已获取 OpenShift Container Platform 安装程序和集群的 pull secret。

流程

1. 创建一个安装目录来存储所需的安装资产：

```
$ mkdir <installation_directory>
```



重要

您必须创建一个目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

2. 自定义提供的 **install-config.yaml** 文件模板示例，并将其保存在 **<installation_directory>** 中。

注意

此配置文件必须命名为 **install-config.yaml**。

对 Azure Stack Hub 进行以下修改：

a. 将 **compute** 池的 **replicas** 参数设置为 **0** ：

```
compute:
- hyperthreading: Enabled
  name: worker
  platform: {}
  replicas: 0 1
```

1 设置为 **0**。

计算机器将在以后手动调配。

b. 更新 **install-config.yaml** 文件的 **platform.azure** 部分，以配置 Azure Stack Hub 配置：

```
platform:
  azure:
    armEndpoint: <azurestack_arm_endpoint> 1
    baseDomainResourceGroupName: <resource_group> 2
    cloudName: AzureStackCloud 3
    region: <azurestack_region> 4
```

1. 指定 Azure Stack Hub 环境的 Azure Resource Manager 端点，如 **https://management.local.azurestack.external**。
2. 指定包含基域的 DNS 区的资源组的名称。
3. 指定 Azure Stack Hub 环境，用于使用适当的 Azure API 端点配置 Azure SDK。
4. 指定 Azure Stack Hub 区域的名称。

3. 备份 **install-config.yaml** 文件，以便用于安装多个集群。

重要

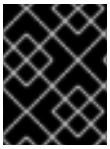
install-config.yaml 文件会在安装过程的下一步中使用。现在必须备份它。

其他资源

- Azure Stack Hub 的安装配置参数

8.5.6.2. Azure Stack Hub 的自定义 install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多详情，或修改所需参数的值。



重要

此示例 YAML 文件仅供参考。使用它作为资源，在您手动创建的安装配置文件中输入参数值。

```
apiVersion: v1
baseDomain: example.com
controlPlane: 1
  name: master
  platform:
    azure:
      osDisk:
        diskSizeGB: 1024 2
        diskType: premium_LRS
  replicas: 3
compute: 3
- name: worker
  platform:
    azure:
      osDisk:
        diskSizeGB: 512 4
```

```
      diskType: premium_LRS
  replicas: 0
metadata:
  name: test-cluster 5
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 6
  serviceNetwork:
  - 172.30.0.0/16
platform:
  azure:
    armEndpoint: azurestack_arm_endpoint 7
    baseDomainResourceGroupName: resource_group 8
    region: azure_stack_local_region 9
    resourceGroupName: existing_resource_group 10
    outboundType: Loadbalancer
    cloudName: AzureStackCloud 11
pullSecret: '{"auths": ...}' 12
fips: false 13
additionalTrustBundle: | 14
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
sshKey: ssh-ed25519 AAAA... 15
```

1 3 **controlPlane** 部分是一个单个映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，controlPlane **部分** 的第一行则不以连字符开头。仅使用一个 control plane 池。

2 4 您可以指定要使用的磁盘大小（以 GB 为单位）。control plane 节点的最低推荐值为 1024 GB。

5 指定集群的名称。

6 要安装的集群网络插件。默认值 **OVNKubernetes** 是唯一支持的值。

7 指定 Azure Stack Hub operator 提供的 Azure Resource Manager 端点。

8 指定包含基域的 DNS 区的资源组的名称。

9 指定 Azure Stack Hub 本地区域的名称。

10 指定要安装集群的现有资源组的名称。如果未定义，则会为集群创建新的资源组。

11 将 Azure Stack Hub 环境指定为目标平台。

12 指定验证集群所需的 pull secret。

13 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

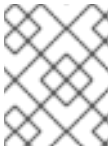
重要

要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 Red Hat Enterprise Linux (RHEL) 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅在 FIPS 模式中安装该系统。

当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS)时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。

14 如果您的 Azure Stack Hub 环境使用内部证书颁发机构(CA)，以 **.pem** 格式添加所需的证书捆绑包。

15 您可以选择提供您用来访问集群中机器的 **sshKey** 值。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

8.5.6.3. 在安装过程中配置集群范围的代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 **Proxy** 对象的 **spec.noProxy** 字段中添加站点来绕过代理。

注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，**Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

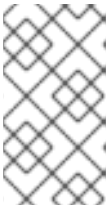
```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
```

```
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 **.** 以仅匹配子域。例如，**.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代理。
4. 如果提供，安装程序会在 **openshift-config** 命名空间中生成名为 **user-ca-bundle** 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 **trusted-ca-bundle** 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， **Proxy** 对象的 **trustedCA** 字段中也会引用此配置映射。**additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。
5. 可选：决定 **Proxy** 对象的配置以引用 **trustedCA** 字段中 **user-ca-bundle** 配置映射的策略。允许的值是 **Proxyonly** 和 **Always**。仅在配置了 **http/https** 代理时，使用 **Proxyonly** 引用 **user-ca-bundle** 配置映射。使用 **Always** 始终引用 **user-ca-bundle** 配置映射。默认值为 **Proxyonly**。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。



注意

如果安装程序超时，重启并使用安装程序的 **wait-for** 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. 保存该文件并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 cluster 的集群范围代理，该代理 **使用** 提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

8.5.6.4. 为 ARM 模板导出常用变量

您必须导出与提供的 Azure Resource Manager (ARM) 模板搭配使用的一组常用变量，它们有助于在 Microsoft Azure Stack Hub 上完成用户提供的基础架构安装。

注意

特定的 ARM 模板可能还需要其他导出变量，这些变量在相关的程序中详细介绍。

先决条件

- 获取 OpenShift Container Platform 安装程序和集群的 pull secret。

流程

1. 导出 **install-config.yaml** 中由提供的 ARM 模板使用的通用变量：

```
$ export CLUSTER_NAME=<cluster_name> (1)
$ export AZURE_REGION=<azure_region> (2)
$ export SSH_KEY=<ssh_key> (3)
$ export BASE_DOMAIN=<base_domain> (4)
$ export BASE_DOMAIN_RESOURCE_GROUP=<base_domain_resource_group> (5)
```

(1) **install-config.yaml** 文件中的**.metadata.name** 属性的值。

(2) 集群要部署到的区域。这是来自 **install-config.yaml** 文件中的 **.platform.azure.region** 属性的值。

(3) 作为字符串的 SSH RSA 公钥文件。您必须使用引号包括 SSH 密钥，因为它包含空格。这是来自 **install-config.yaml** 文件中的 **.sshKey** 属性的值。

(4) 集群要部署到的基域。基域与您为集群创建的 DNS 区域对应。这是来自 **install-config.yaml** 文件中的 **.baseDomain** 属性的值。

(5) 存在 DNS 区的资源组。这是来自 **install-config.yaml** 文件中的 **.platform.azure.baseDomainResourceGroupName** 属性的值。

例如：

```
$ export CLUSTER_NAME=test-cluster
$ export AZURE_REGION=centralus
$ export SSH_KEY="ssh-rsa xxx/xxx/xxx= user@email.com"
$ export BASE_DOMAIN=example.com
$ export BASE_DOMAIN_RESOURCE_GROUP=ocp-cluster
```

2. 导出 kubeadmin 凭证：

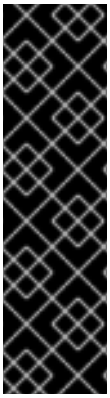
```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig (1)
```

(1) 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

8.5.6.5. 创建 Kubernetes 清单和 Ignition 配置文件

由于您必须修改一些集群定义文件并手动启动集群机器，因此您必须生成 Kubernetes 清单和 Ignition 配置文件来配置机器。

安装配置文件转换为 Kubernetes 清单。清单嵌套到 Ignition 配置文件中，稍后用于配置集群机器。



重要

- OpenShift Container Platform 安装程序生成的 Ignition 配置文件包含 24 小时后过期的证书，然后在该时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

先决条件

- 已获得 OpenShift Container Platform 安装程序。
- 已创建 **install-config.yaml** 安装配置文件。

流程

1. 进入包含 OpenShift Container Platform 安装程序的目录，并为集群生成 Kubernetes 清单：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定包含您创建的 **install-config.yaml** 文件的安装目录。

2. 删除定义 control plane 机器的 Kubernetes 清单文件：

```
$ rm -f <installation_directory>/openshift/99_openshift-cluster-api_master-machines-*.yaml
```

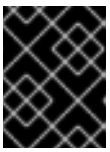
通过删除这些文件，您可以防止集群自动生成 control plane 机器。

3. 删除定义 control plane 机器集的 Kubernetes 清单文件：

```
$ rm -f <installation_directory>/openshift/99_openshift-machine-api_master-control-plane-machine-set.yaml
```

4. 删除定义 worker 机器的 Kubernetes 清单文件：

```
$ rm -f <installation_directory>/openshift/99_openshift-cluster-api_worker-machineset-*.yaml
```



重要

如果在用户置备的基础架构上安装集群时禁用了 **MachineAPI** 功能，则必须删除定义 worker 机器的 Kubernetes 清单文件。否则，集群将无法安装。

由于您要自行创建和管理 worker 机器，因此不需要初始化这些机器。

5. 检查 **<installation_directory>/manifests/cluster-scheduler-02-config.yml Kubernetes 清单文件中的 mastersSchedulable** 参数是否已设置为 **false**。此设置可防止在 control plane 机器上调度 pod：

a. 打开 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** 文件。

b. 找到 **mastersSchedulable** 参数，并确保它被设置为 **false**。

c. 保存并退出 文件。

6. 可选：如果您不希望 Ingress Operator 代表您创建 DNS 记录，请删除 **<installation_directory>/manifests/cluster-dns-02-config.yml** DNS 配置文件中的 **privateZone 和 publicZone** 部分：

```
apiVersion: config.openshift.io/v1
kind: DNS
metadata:
  creationTimestamp: null
  name: cluster
spec:
  baseDomain: example.openshift.com
  privateZone: 1
    id: mycluster-100419-private-zone
  publicZone: 2
    id: example.openshift.com
status: {}
```

1 2 完全删除此部分。

如果您这样做，后续步骤中必须手动添加入口 DNS 记录。

7. 可选： 如果您的 Azure Stack Hub 环境使用内部证书颁发机构(CA)，您必须更新 **<installation_directory>/manifests/cluster-proxy-01-config.yaml** 文件中的 **.spec.trustedCA.name** 字段来使用 **user-ca-bundle**：

```
...
spec:
  trustedCA:
    name: user-ca-bundle
...
```

之后，您必须更新 bootstrap ignition 使其包含 CA。

8. 在用户置备的基础架构上配置 Azure 时，您必须导出清单文件中定义的一些常见变量，以备稍后在 Azure Resource Manager（ARM）模板中使用：

a. 使用以下命令导出基础架构 ID:

```
$ export INFRA_ID=<infra_id> 1
```

1 OpenShift Container Platform 集群被分配了一个标识符（**INFRA_ID**），其格式为 **<cluster_name>-<random_string>**。这将作为使用提供的 ARM 模板创建的大部分资源的基本名称。这是来自 **manifests/cluster-infrastructure-02-config.yml** 文件中的 **.status.infrastructureName** 属性的值。

b. 使用以下命令导出资源组：

```
$ export RESOURCE_GROUP=<resource_group> 1
```

1 此 Azure 部署中创建的所有资源都作为资源组的一部分。资源组名称还基于 **INFRA_ID**，格式为 **<cluster_name>-<random_string>-rg**。这是来自 **manifests/cluster-infrastructure-02-config.yml** 文件中的 **.status.platformStatus.azure.resourceGroupName** 属性的值。

9. 手动创建云凭证。

a. 从包含安装程序的目录中，获取 **openshift-install** 二进制文件要使用的 OpenShift Container Platform 发行镜像详情：

```
$ openshift-install version
```

输出示例

```
release image quay.io/openshift-release-dev/ocp-release:4.y.z-x86_64
```

b. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

c. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 自定义资源 (CR) 列表：

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
  --included \ 1
  --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \
2
  --to=<path_to_directory_for_credentials_requests> 3
```

1 **--included** 参数仅包含特定集群配置所需的清单。

2 指定 **install-config.yaml** 文件的位置。

3 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

此命令为每个 **CredentialsRequest** 对象创建一个 YAML 文件。

CredentialsRequest 对象示例

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  labels:
    controller-tools.k8s.io: "1.0"
  name: openshift-image-registry-azure
  namespace: openshift-cloud-credential-operator
spec:
  secretRef:
```

```
    name: installer-cloud-credentials
    namespace: openshift-image-registry
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: AzureProviderSpec
    roleBindings:
    - role: Contributor
```

d. 在之前生成的 **openshift-install** 清单目录中为 secret 创建 YAML 文件。secret 必须使用在 **spec.secretRef** 中为每个 **CredentialsRequest** 定义的命名空间和 secret 名称存储。secret 数据的格式因云供应商而异。

secrets.yaml 文件示例：

```
apiVersion: v1
kind: Secret
metadata:
    name: ${secret_name}
    namespace: ${secret_namespace}
stringData:
  azure_subscription_id: ${subscription_id}
  azure_client_id: ${app_id}
  azure_client_secret: ${client_secret}
  azure_tenant_id: ${tenant_id}
  azure_resource_prefix: ${cluster_name}
  azure_resourcegroup: ${resource_group}
  azure_region: ${azure_region}
```

e. 在 manifests 目录中创建一个 **cco-configmap.yaml** 文件，并禁用 Cloud Credential Operator (CCO)：

ConfigMap 对象示例

```
apiVersion: v1
kind: ConfigMap
metadata:
name: cloud-credential-operator-config
namespace: openshift-cloud-credential-operator
  annotations:
    release.openshift.io/create-only: "true"
data:
  disabled: "true"
```

10. 要创建 Ignition 配置文件，从包含安装程序的目录运行以下命令：

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定相同的安装目录。

为安装目录中的 bootstrap、control plane 和计算节点创建 Ignition 配置文件。**kubeadmin-password** 和 **kubeconfig** 文件在 **./<installation_directory>/auth** 目录中创建：

```
.
```

```
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

其他资源

- 手动管理云凭证

8.5.6.6. 可选：创建独立 /var 分区

建议安装程序将 OpenShift Container Platform 的磁盘分区保留给安装程序。然而，在有些情况下您可能需要在文件系统的一部分中创建独立分区。

OpenShift Container Platform 支持添加单个分区来将存储附加到 **/var** 分区或 **/var** 的子目录中。例如：

- **/var/lib/containers** ：保存随着系统中添加更多镜像和容器而增长的容器相关内容。
- **/var/lib/etcd** ：保存您可能希望独立保留的数据，比如 etcd 存储的性能优化。
- **/var** ：保存您可能希望独立保留的数据，以满足审计等目的。

通过单独存储 **/var** 目录的内容，可以更轻松地根据需要为区域扩展存储，并在以后重新安装 OpenShift Container Platform，并保持该数据的完整性。使用这个方法，您不必再次拉取所有容器，在更新系统时也不必复制大量日志文件。

因为 **/var** 在进行一个全新的 Red Hat Enterprise Linux CoreOS（RHCOS）安装前必需存在，所以这个流程会在 OpenShift Container Platform 安装过程的 **openshift-install** 准备阶段插入一个创建的机器配置清单的机器配置来设置独立的 **/var** 分区。

重要

如果按照以下步骤在此流程中创建独立 **/var** 分区，则不需要再次创建 Kubernetes 清单和 Ignition 配置文件，如本节所述。

流程

1. 创建存放 OpenShift Container Platform 安装文件的目录：

```
$ mkdir $HOME/clusterconfig
```

2. 运行 **openshift-install**，以在 **manifest** 和 **openshift** 子目录中创建一组文件。在系统提示时回答系统问题：

```
$ openshift-install create manifests --dir $HOME/clusterconfig
```

输出示例

```
? SSH Public Key ...
INFO Credentials loaded from the "myprofile" profile in file "/home/myuser/.aws/credentials"
INFO Consuming Install Config from target directory
```

```
INFO Manifests created in: $HOME/clusterconfig/manifests and
$HOME/clusterconfig/openshift
```

3. 可选：确认安装程序在 **clusterconfig/openshift** 目录中创建了清单：

```
$ ls $HOME/clusterconfig/openshift/
```

输出示例

```
99_kubeadmin-password-secret.yaml
99_openshift-cluster-api_master-machines-0.yaml
99_openshift-cluster-api_master-machines-1.yaml
99_openshift-cluster-api_master-machines-2.yaml
...
```

4. 创建用于配置额外分区的 Butane 配置。例如，将文件命名为 **$HOME/clusterconfig/98-var-partition.bu**，将磁盘设备名称改为 **worker** 系统上存储设备的名称，并根据情况设置存储大小。这个示例将 **/var** 目录放在一个单独的分区中：

```
variant: openshift
version: 4.16.0
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
storage:
  disks:
  - device: /dev/disk/by-id/<device_name> 1
    partitions:
    - label: var
      start_mib: <partition_start_offset> 2
      size_mib: <partition_size> 3
      number: 5
  filesystems:
    - device: /dev/disk/by-partlabel/var
      path: /var
      format: xfs
      mount_options: [defaults, prjquota] 4
      with_mount_unit: true
```

1. 要分区的磁盘的存储设备名称。
2. 在引导磁盘中添加数据分区时，推荐最少使用 25000 MiB(Mebibytes)。root 文件系统会自动调整大小以填充所有可用空间（最多到指定的偏移值）。如果没有指定值，或者指定的值小于推荐的最小值，则生成的 root 文件系统会太小，而在以后进行的 RHCOS 重新安装可能会覆盖数据分区的开始部分。
3. 以兆字节为单位的数据分区大小。
4. 对于用于容器存储的文件系统，必须启用 **prjquota** 挂载选项。

注意

当创建单独的 **/var** 分区时，如果不同的实例类型没有相同的设备名称，则无法为 worker 节点使用不同的实例类型。

5. 从 Butane 配置创建一个清单，并将它保存到 **clusterconfig/openshift** 目录中。例如，运行以下命令：

```
$ butane $HOME/clusterconfig/98-var-partition.bu -o $HOME/clusterconfig/openshift/98-var-partition.yaml
```

6. 再次运行 **openshift-install**，从 **manifest** 和 **openshift** 子目录中的一组文件创建 Ignition 配置：

```
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
auth  bootstrap.ign  master.ign  metadata.json  worker.ign
```

现在，您可以使用 Ignition 配置文件作为安装程序的输入来安装 Red Hat Enterprise Linux CoreOS（RHCOS）系统。

8.5.7. 创建 Azure 资源组

您必须创建一个 Microsoft Azure 资源组。这用于在 Azure Stack Hub 上安装 OpenShift Container Platform 集群。

先决条件

- 配置 Azure 帐户。
- 为集群生成 Ignition 配置文件。

流程

- 在受支持的 Azure 区域中创建资源组：

```
$ az group create --name ${RESOURCE_GROUP} --location ${AZURE_REGION}
```

8.5.8. 上传 RHCOS 集群镜像和 bootstrap Ignition 配置文件

Azure 客户端不支持基于本地现有文件进行部署。您必须复制 RHCOS 虚拟硬盘(VHD)集群镜像，并将 bootstrap Ignition 配置文件存储在存储容器中，以便在部署过程中访问它们。

先决条件

- 配置 Azure 帐户。
- 为集群生成 Ignition 配置文件。

流程

1. 创建 Azure 存储帐户以存储 VHD 集群镜像：

```
$ az storage account create -g ${RESOURCE_GROUP} --location ${AZURE_REGION} --
name ${CLUSTER_NAME}sa --kind Storage --sku Standard_LRS
```

警告

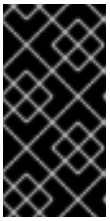
Azure 存储帐户名称的长度必须在 3 到 24 个字符之，且只使用数字和小写字母。如果您的 **CLUSTER_NAME** 变量没有遵循这些限制，您必须手动定义 Azure 存储帐户名称。如需有关 Azure 存储帐户名称限制的更多信息，请参阅 Azure 文档中的解决存储帐户名称的错误。

2. 将存储帐户密钥导出为环境变量：

```
$ export ACCOUNT_KEY=`az storage account keys list -g ${RESOURCE_GROUP} --
account-name ${CLUSTER_NAME}sa --query "[0].value" -o tsv`
```

3. 将 RHCOS VHD 的 URL 导出为环境变量：

```
$ export COMPRESSED_VHD_URL=$(openshift-install coreos print-stream-json | jq -r
'.architectures.x86_64.artifacts.azurestack.formats."vhd.gz".disk.location')
```



重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每个发行版本而改变。您必须指定一个最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。如果可用，请使用与 OpenShift Container Platform 版本匹配的镜像版本。

4. 为 VHD 创建存储容器：

```
$ az storage container create --name vhd --account-name ${CLUSTER_NAME}sa --account-
key ${ACCOUNT_KEY}
```

5. 在本地下载压缩的 RHCOS VHD 文件：

```
$ curl -O -L ${COMPRESSED_VHD_URL}
```

6. 解压缩 VHD 文件。

注意

解压缩的 VHD 文件大约为 16 GB，因此请确保您的主机系统有 16 GB 的可用空间。您可以在上传 VHD 文件后删除该文件。

7. 将本地 VHD 复制为一个 blob:

```
$ az storage blob upload --account-name ${CLUSTER_NAME}sa --account-key
${ACCOUNT_KEY} -c vhd -n "rhcos.vhd" -f rhcos-<rhcos_version>-azurestack.x86_64.vhd
```

8. 创建 blob 存储容器并上传生成的 **bootstrap.ign** 文件：

```
$ az storage container create --name files --account-name ${CLUSTER_NAME}sa --
account-key ${ACCOUNT_KEY}
```

```
$ az storage blob upload --account-name ${CLUSTER_NAME}sa --account-key
${ACCOUNT_KEY} -c "files" -f "<installation_directory>/bootstrap.ign" -n "bootstrap.ign"
```

8.5.9. 创建 DNS 区示例

使用用户置备的基础架构的集群需要 DNS 记录。您应该选择适合您的场景的 DNS 策略。

本例中使用了 Azure Stack Hub 的数据中心 DNS 集成，因此您将创建一个 DNS 区。

注意

DNS 区域不需要与集群部署位于同一个资源组中，且可能已在您的机构中为所需基域存在。如果情况如此，您可以跳过创建 DNS 区域；请确定您之前生成的安装配置反映了这种情况。

先决条件

- 配置 Azure 帐户。
- 为集群生成 Ignition 配置文件。

流程

- 在 **BASE_DOMAIN_RESOURCE_GROUP** 环境变量中导出的资源组中创建新的 DNS 区域：

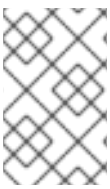
```
$ az network dns zone create -g ${BASE_DOMAIN_RESOURCE_GROUP} -n
${CLUSTER_NAME}.${BASE_DOMAIN}
```

如果您使用的是已存在的 DNS 区域，您可以跳过这一步。

您可以访问该部分来了解更多有关 在 Azure Stack Hub 中配置 DNS 区 的信息。

8.5.10. 在 Azure Stack Hub 中创建 VNet

您必须在 Microsoft Azure Stack Hub 中创建虚拟网络 (VNet)，供您的 OpenShift Container Platform 集群使用。您可以对 VNet 进行定制来满足您的要求。创建 VNet 的一种方法是修改提供的 Azure Resource Manager（ARM）模板。



注意

如果不使用提供的 ARM 模板来创建 Azure Stack Hub 基础架构，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 Azure 帐户。

- 为集群生成 Ignition 配置文件。

流程

1. 复制 **VNet 的 ARM 模板** 一节中的模板，并将它以 **01_vnet.json** 保存到集群的安装目录中。此模板描述了集群所需的 VNet。
2. 使用 **az** CLI 创建部署：

```
$ az deployment group create -g ${RESOURCE_GROUP} \
  --template-file "<installation_directory>/01_vnet.json" \
  --parameters baseName="${INFRA_ID}" 1
```

1 资源名称使用的基本名称 ; 这通常是集群的基础架构 ID。

8.5.10.1. VNet 的 ARM 模板

您可以使用以下 Azure Resource Manager（ARM）模板来部署 OpenShift Container Platform 集群所需的 VPC：

例 8.1. 01_vnet.json ARM 模板

link:https://raw.githubusercontent.com/openshift/installer/release-4.16/upi/azurestack/01_vnet.json[]

8.5.11. 为 Azure Stack Hub 基础架构部署 RHCOS 集群镜像

您必须对 OpenShift Container Platform 节点的 Microsoft Azure Stack Hub 使用有效的 Red Hat Enterprise Linux CoreOS (RHCOS) 镜像。

先决条件

- 配置 Azure 帐户。
- 为集群生成 Ignition 配置文件。
- 将 RHCOS 虚拟硬盘（VHD）集群镜像存储在 Azure 存储容器中。
- 在 Azure 存储容器中存储 bootstrap Ignition 配置文件。

流程

1. 复制**镜像存储的 ARM 模板** 部分中的模板，并将它以 **02_storage.json** 保存到集群的安装目录中。此模板描述了集群所需的镜像存储。
2. 以一个变量的形式将 RHCOS VHD blob URL 导出：

```
$ export VHD_BLOB_URL=`az storage blob url --account-name ${CLUSTER_NAME}sa --
account-key ${ACCOUNT_KEY} -c vhd -n "rhcos.vhd" -o tsv`
```

3. 部署集群镜像

```
$ az deployment group create -g ${RESOURCE_GROUP} \
  --template-file "<installation_directory>/02_storage.json" \
  --parameters vhdBlobURL="${VHD_BLOB_URL}" \ 1
  --parameters baseName="${INFRA_ID}" \ 2
  --parameters storageAccount="${CLUSTER_NAME}sa" \ 3
  --parameters architecture="<architecture>" 4
```

1. 用于创建 master 和 worker 机器的 RHCOS VHD 的 blob URL。
2. 资源名称使用的基本名称；这通常是集群的基础架构 ID。
3. Azure 存储帐户的名称。
4. 指定系统架构。有效值为 **x64** （默认）或 **Arm64**。

8.5.11.1. 镜像存储的 ARM 模板

您可以使用以下 Azure Resource Manager（ARM）模板来部署 OpenShift Container Platform 集群所需的存储的 Red Hat Enterprise Linux CoreOS（RHCOS）镜像：

例 8.2. 02_storage.json ARM 模板

link:https://raw.githubusercontent.com/openshift/installer/release-4.16/upi/azurestack/02_storage.json[]

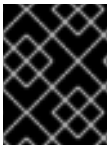
8.5.12. 用户置备的基础架构对网络的要求

所有 Red Hat Enterprise Linux CoreOS（RHCOS）机器需要在启动过程中在 **initramfs** 中配置网络，以获取其 Ignition 配置文件。

8.5.12.1. 网络连接要求

您必须配置机器之间的网络连接，以允许 OpenShift Container Platform 集群组件进行通信。每台机器都必须能够解析集群中所有其他机器的主机名。

本节详细介绍了所需的端口。



重要

在连接的 OpenShift Container Platform 环境中，所有节点都需要访问互联网才能为平台容器拉取镜像，并向红帽提供遥测数据。

表 8.12. 用于全机器到所有机器通信的端口

协议	port	描述
ICMP	N/A	网络可访问性测试
TCP	**1936**	指标

协议	port	描述
	9000-9999	主机级别的服务，包括端口 9**100-9101 上的节点导出器**，以及端口 **9099** 上的 Cluster Version Operator。
	10250-10259	Kubernetes 保留的默认端口
UDP	**4789**	VXLAN
	6081	Geneve
	9000-9999	主机级别的服务，包括端口 **9100**到**9101** 上的节点导出器。
	500	IPsec IKE 数据包
	4500	IPsec NAT-T 数据包
	123	UDP 端口 **123** 上的网络时间协议 (NTP) 如果配置了外部 NTP 时间服务器，需要打开 UDP 端口 **123**。
TCP/UDP	**30000-32767**	Kubernetes 节点端口
ESP	N/A	IPsec Encapsulating Security Payload(ESP)

表 8.13. 用于所有机器控制平面通信的端口

协议	port	描述
TCP	**6443**	Kubernetes API

表 8.14. control plane 机器用于 control plane 机器通信的端口

协议	port	描述
TCP	**2379-2380**	etcd 服务器和对等端口

8.5.13. 在 Azure Stack Hub 中创建网络和负载均衡组件

您必须在 Microsoft Azure Stack Hub 中配置网络和负载均衡，供您的 OpenShift Container Platform 集群使用。创建这些组件的一种方法是修改提供的 Azure Resource Manager（ARM）模板。

负载平衡需要以下 DNS 记录：

- DNS 区域中的 API 公共负载均衡器的 **api** DNS 记录。

- DNS 区域中 API 内部负载均衡器的 **api-int** DNS 记录。

注意

如果不使用提供的 ARM 模板来创建 Azure Stack Hub 基础架构，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 Azure 帐户。
- 为集群生成 Ignition 配置文件。
- 在 Azure Stack Hub 中创建和配置 VNet 及相关子网。

流程

1. 复制 **网络和负载均衡器的 ARM 模板** 一节中的模板，并将它以 **03_infra.json** 保存到集群的安装目录中。此模板描述了集群所需的网络和负载均衡对象。
2. 使用 **az** CLI 创建部署：

```
$ az deployment group create -g ${RESOURCE_GROUP} \
  --template-file "<installation_directory>/03_infra.json" \
  --parameters baseName="${INFRA_ID}" 1
```

1 资源名称使用的基本名称 ; 这通常是集群的基础架构 ID。

3. 创建 **api** DNS 记录和 **api-int** DNS 记录。在创建 API DNS 记录时，**${BASE_DOMAIN_RESOURCE_GROUP}** 变量必须指向 DNS 区域存在的资源组。
   a. 导出以下变量：

```
$ export PUBLIC_IP=`az network public-ip list -g ${RESOURCE_GROUP} --query "[?
name=='${INFRA_ID}-master-pip'] | [0].ipAddress" -o tsv`
```

   b. 导出以下变量：

```
$ export PRIVATE_IP=`az network lb frontend-ip show -g "$RESOURCE_GROUP" --lb-
name "${INFRA_ID}-internal" -n internal-lb-ip --query "privateIpAddress" -o tsv`
```

   c. 在新的 DNS 区域中创建 **api** DNS 记录：

```
$ az network dns record-set a add-record -g ${BASE_DOMAIN_RESOURCE_GROUP} -
z ${CLUSTER_NAME}.${BASE_DOMAIN} -n api -a ${PUBLIC_IP} --ttl 60
```

   如果要将集群添加到现有 DNS 区域中，您可以在其中创建 **api** DNS 记录：

```
$ az network dns record-set a add-record -g ${BASE_DOMAIN_RESOURCE_GROUP} -
z ${BASE_DOMAIN} -n api.${CLUSTER_NAME} -a ${PUBLIC_IP} --ttl 60
```

   d. 在新的 DNS 区域中创建 **api-int** DNS 记录：

```
$ az network dns record-set a add-record -g ${BASE_DOMAIN_RESOURCE_GROUP} -
z "${CLUSTER_NAME}.${BASE_DOMAIN}" -n api-int -a ${PRIVATE_IP} --ttl 60
```

如果要将集群添加到现有 DNS 区域中，您可以在其中创建 **api-int** DNS 记录：

```
$ az network dns record-set a add-record -g ${BASE_DOMAIN_RESOURCE_GROUP} -
z ${BASE_DOMAIN} -n api-int.${CLUSTER_NAME} -a ${PRIVATE_IP} --ttl 60
```

8.5.13.1. 网络和负载均衡器的 ARM 模板

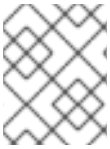
您可以使用以下 Azure Resource Manager(ARM)模板来部署 OpenShift Container Platform 集群所需的网络对象和负载均衡器：

例 8.3. 03_infra.json ARM template

link:https://raw.githubusercontent.com/openshift/installer/release-4.16/upi/azurestack/03_infra.json[]

8.5.14. 在 Azure Stack Hub 中创建 bootstrap 机器

您必须在 Microsoft Azure Stack Hub 中创建 bootstrap 机器，以便在 OpenShift Container Platform 集群初始化过程中使用。创建此机器的一种方法是修改提供的 Azure Resource Manager(ARM)模板。



注意

如果不使用提供的 ARM 模板来创建 bootstrap 机器，您必须检查提供的信息并手动创建基础架构。如果集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 Azure 帐户。
- 为集群生成 Ignition 配置文件。
- 在 Azure Stack Hub 中创建和配置 VNet 及相关子网。
- 在 Azure Stack Hub 中创建和配置联网与负载均衡器。
- 创建 control plane 和计算角色。

流程

1. 复制 **bootstrap 机器的 ARM 模板一节中的模板**，并将它以 **04_bootstrap.json** 保存到集群的安装目录中。此模板描述了集群所需的 bootstrap 机器。
2. 导出 bootstrap URL 变量：

```
$ bootstrap_url_expiry=`date -u -d "10 hours" '+%Y-%m-%dT%H:%MZ'`
```

```
$ export BOOTSTRAP_URL=`az storage blob generate-sas -c 'files' -n 'bootstrap.ign' --https-only --full-uri --permissions r --expiry $bootstrap_url_expiry --account-name ${CLUSTER_NAME}sa --account-key ${ACCOUNT_KEY} -o tsv`
```

3. 导出 bootstrap ignition 变量：

   a. 如果您的环境使用公共证书颁发机构(CA)，请运行以下命令：

   ```
   $ export BOOTSTRAP_IGNITION=`jq -rcnM --arg v "3.2.0" --arg url ${BOOTSTRAP_URL} '{ignition:{version:$v,config:{replace:{source:$url}}}}' | base64 | tr -d '\n'`
   ```

   b. 如果您的环境使用内部 CA，您必须将 PEM 编码捆绑包添加到 bootstrap ignition stub 中，以便 bootstrap 虚拟机可以从存储帐户中提取 bootstrap ignition。运行以下命令，这假设您的 CA 位于名为 **CA.pem** 的文件中：

   ```
   $ export CA="data:text/plain;charset=utf-8;base64,$(cat CA.pem |base64 |tr -d '\n')"
   ```

   ```
   $ export BOOTSTRAP_IGNITION=`jq -rcnM --arg v "3.2.0" --arg url "$BOOTSTRAP_URL" --arg cert "$CA" '{ignition:{version:$v,security:{tls:{certificateAuthorities:[{source:$cert}]}},config:{replace:{source:$url}}}}' | base64 | tr -d '\n'`
   ```

4. 使用 **az** CLI 创建部署：

```
$ az deployment group create --verbose -g ${RESOURCE_GROUP} \
  --template-file "<installation_directory>/04_bootstrap.json" \
  --parameters bootstrapIgnition="${BOOTSTRAP_IGNITION}" \ 1
  --parameters baseName="${INFRA_ID}" \ 2
  --parameters diagnosticsStorageAccountName="${CLUSTER_NAME}sa" 3
```

1. bootstrap 集群的 bootstrap Ignition 内容。
2. 资源名称使用的基本名称 ; 这通常是集群的基础架构 ID。
3. 集群的存储帐户的名称。

8.5.14.1. bootstrap 机器的 ARM 模板

您可以使用以下 Azure Resource Manager(ARM)模板来部署 OpenShift Container Platform 集群所需的 bootstrap 机器：

例 8.4. 04_bootstrap.json ARM template

link:https://raw.githubusercontent.com/openshift/installer/release-4.16/upi/azurestack/04_bootstrap.json[]

8.5.15. 在 Azure Stack Hub 中创建 control plane 机器

您必须在 Microsoft Azure Stack Hub 中创建 control plane 机器，供您的集群使用。创建这些机器的一种方法是修改提供的 Azure Resource Manager(ARM)模板。

如果不使用提供的 ARM 模板来创建 control plane 机器，您必须检查提供的信息并手动创建基础架构。如果您的集群没有正确初始化，请考虑与安装日志联系红帽支持。

先决条件

- 配置 Azure 帐户。
- 为集群生成 Ignition 配置文件。
- 在 Azure Stack Hub 中创建和配置 VNet 及相关子网。
- 在 Azure Stack Hub 中创建和配置联网与负载均衡器。
- 创建 control plane 和计算角色。
- 创建 bootstrap 机器。

流程

1. 复制 **control plane 机器的 ARM 模板**一节中的模板，并将它以 **05_masters.json** 保存到集群的安装目录中。此模板描述了集群所需的 control plane 机器。
2. 导出 control plane 机器部署所需的以下变量：

```
$ export MASTER_IGNITION=`cat <installation_directory>/master.ign | base64 | tr -d '\n'`
```

3. 使用 **az** CLI 创建部署：

```
$ az deployment group create -g ${RESOURCE_GROUP} \
  --template-file "<installation_directory>/05_masters.json" \
  --parameters masterIgnition="${MASTER_IGNITION}" \ 1
  --parameters baseName="${INFRA_ID}" \ 2
  --parameters diagnosticsStorageAccountName="${CLUSTER_NAME}sa" 3
```

1. control plane 节点的 Ignition 内容（也称为 master 节点）。
2. 资源名称使用的基本名称 ; 这通常是集群的基础架构 ID。
3. 集群的存储帐户的名称。

8.5.15.1. control plane 机器的 ARM 模板

您可以使用以下 Azure Resource Manager(ARM)模板来部署 OpenShift Container Platform 集群所需的 control plane 机器：

例 8.5. 05_masters.json ARM template

link:https://raw.githubusercontent.com/openshift/installer/release-4.16/upi/azurestack/05_masters.json[]

8.5.16. 等待 bootstrap 完成并删除 Azure Stack Hub 中的 bootstrap 资源

在 Microsoft Azure Stack Hub 中创建所有所需的基础架构后，请等待您通过安装程序生成的 Ignition 配置文件所置备的机器上完成 bootstrap 过程。

先决条件

- 配置 Azure 帐户。
- 为集群生成 Ignition 配置文件。
- 在 Azure Stack Hub 中创建和配置 VNet 及相关子网。
- 在 Azure Stack Hub 中创建和配置联网与负载均衡器。
- 创建 control plane 和计算角色。
- 创建 bootstrap 机器。
- 创建 control plane 机器。

流程

1. 进入包含安装程序的目录并运行以下命令：

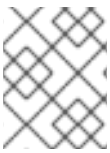
```
$ ./openshift-install wait-for bootstrap-complete --dir <installation_directory> \ 1
    --log-level info 2
```

1. 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。
2. 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不是 **info**。

如果命令退出时没有 **FATAL** 警告，则您的生产环境 control plane 已被初始化。

2. 删除 bootstrap 资源：

```
$ az network nsg rule delete -g ${RESOURCE_GROUP} --nsg-name ${INFRA_ID}-nsg --
name bootstrap_ssh_in
$ az vm stop -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap
$ az vm deallocate -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap
$ az vm delete -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap --yes
$ az disk delete -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap_OSDisk --no-
wait --yes
$ az network nic delete -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap-nic --no-
wait
$ az storage blob delete --account-key ${ACCOUNT_KEY} --account-name
${CLUSTER_NAME}sa --container-name files --name bootstrap.ign
$ az network public-ip delete -g ${RESOURCE_GROUP} --name ${INFRA_ID}-bootstrap-
ssh-pip
```



注意

如果没有删除 bootstrap 服务器，因为 API 流量会路由到 bootstrap 服务器，所以安装可能无法成功。

8.5.17. 在 Azure Stack Hub 中创建额外的 worker 机器

您可以通过分散启动各个实例或利用集群外自动化流程（如自动缩放组），在 Microsoft Azure Stack Hub 中为您的集群创建 worker 机器。您还可以利用 OpenShift Container Platform 中的内置集群扩展机制和机器 API。

在本例中，您要使用 Azure Resource Manager(ARM)模板手动启动一个实例。通过在 文件中包括类型为 **06_workers.json** 的其他资源，即可启动其他实例。

如果不使用提供的 ARM 模板来创建 control plane 机器，您必须检查提供的信息并手动创建基础架构。如果您的集群没有正确初始化，请考虑与安装日志联系红帽支持。

先决条件

- 配置 Azure 帐户。
- 为集群生成 Ignition 配置文件。
- 在 Azure Stack Hub 中创建和配置 VNet 及相关子网。
- 在 Azure Stack Hub 中创建和配置联网与负载均衡器。
- 创建 control plane 和计算角色。
- 创建 bootstrap 机器。
- 创建 control plane 机器。

流程

1. 复制 **worker 机器的 ARM 模板一节中的模板**，并将它以 **06_workers.json** 保存到集群的安装目录中。此模板描述了集群所需的 worker 机器。
2. 导出 worker 机器部署所需的以下变量：

```
$ export WORKER_IGNITION=`cat <installation_directory>/worker.ign | base64 | tr -d '\n'`
```

3. 使用 **az** CLI 创建部署：

```
$ az deployment group create -g ${RESOURCE_GROUP} \
  --template-file "<installation_directory>/06_workers.json" \
  --parameters workerIgnition="${WORKER_IGNITION}" \ 1
  --parameters baseName="${INFRA_ID}" 2
  --parameters diagnosticsStorageAccountName="${CLUSTER_NAME}sa" 3
```

1. worker 节点的 Ignition 内容。
2. 资源名称使用的基本名称；这通常是集群的基础架构 ID。
3. 集群的存储帐户的名称。

8.5.17.1. worker 机器的 ARM 模板

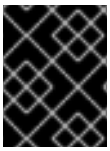
您可以使用以下 Azure Resource Manager(ARM)模板来部署 OpenShift Container Platform 集群所需的 worker 机器：

例 8.6. 06_workers.json ARM template

link:https://raw.githubusercontent.com/openshift/installer/release-4.16/upi/azurestack/06_workers.json[]

8.5.18. 安装 OpenShift CLI

您可以安装 OpenShift CLI(**oc**)来使用命令行界面与 OpenShift Container Platform 进行交互。您可以在 Linux、Windows 或 macOS 上安装 **oc**。



重要

如果安装了旧版本的 **oc**，则可能无法使用 OpenShift Container Platform 4.16 中的所有命令。下载并安装新版本的 **oc**。

在 Linux 上安装 OpenShift CLI
您可以按照以下流程在 Linux 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **产品变体** 下拉列表中选择架构。
3. 从 **版本** 下拉列表中选择适当的版本。
4. 点 **OpenShift v4.16 Linux Client** 条目旁的 **Download Now** 来保存文件。
5. 解包存档：

```
$ tar xvf <file>
```

6. 将 **oc** 二进制文件放到 **PATH 中的目录中**。
要查看您的 **PATH**，请执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

在 Windows 上安装 OpenShift CLI
您可以按照以下流程在 Windows 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 Windows Client** 条目旁的 **Download Now** 来保存文件。
4. 使用 ZIP 程序解压存档。
5. 将 **oc** 二进制文件移到 **PATH 中的目录中**。
   要查看您的 **PATH**，请打开命令提示并执行以下命令：

```
C:\> path
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

在 macOS 上安装 OpenShift CLI
您可以按照以下流程在 macOS 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 macOS Client** 条目旁的 **Download Now** 来保存文件。

注意

对于 macOS arm64，请选择 **OpenShift v4.16 macOS arm64 Client** 条目。

4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
   要查看您的 **PATH**，请打开终端并执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

8.5.19. 使用 CLI 登录集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

8.5.20. 批准机器的证书签名请求

当您将机器添加到集群时，会为您添加的每台机器生成两个待处理证书签名请求(CSR)。您必须确认这些 CSR 已获得批准，或根据需要自行批准。必须首先批准客户端请求，然后批准服务器请求。

先决条件

- 您已将机器添加到集群中。

流程

1. 确认集群可以识别这些机器：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS    ROLES   AGE  VERSION
master-0 Ready     master  63m  v1.29.4
master-1 Ready     master  63m  v1.29.4
master-2 Ready     master  64m  v1.29.4
```

输出中列出了您创建的所有机器。

注意

在有些 CSR 被批准前，前面的输出可能不包括计算节点（也称为 worker 节点）。

2. 检查待处理的 CSR，并确保添加到集群中的每台机器都有 **Pending** 或 **Approved** 状态的客户端请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE    REQUESTOR                                                   CONDITION
csr-8b2br   15m    system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
csr-8vnps   15m    system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
...
```

在本例中，两台机器加入集群。您可能会在列表中看到更多已批准的 CSR。

3. 如果 CSR 没有获得批准，在您添加的机器的所有待处理 CSR 都处于 **Pending 状态** 后，请批准集群机器的 CSR：

注意

由于 CSR 会自动轮转，因此请在将机器添加到集群后一小时内批准您的 CSR。如果没有在一小时内批准它们，证书将会轮转，每个节点会存在多个证书。您必须批准所有这些证书。批准客户端 CSR 后，Kubelet 为服务证书创建一个二级 CSR，这需要手动批准。然后，如果 Kubelet 请求具有相同参数的新证书，则后续提供证书续订请求由 **machine-approver** 自动批准。

注意

对于在未启用机器 API 的平台上运行的集群，如裸机和其他用户置备的基础架构，您必须实施一种方法来自动批准 kubelet 提供证书请求(CSR)。如果没有批准请求，则 **oc exec**、ocrsh 和 **oc logs** 命令将无法成功，因为 API 服务器连接到 kubelet 时需要服务证书。与 Kubelet 端点联系的任何操作都需要此证书批准。该方法必须监视新的 CSR，确认 CSR 由 system: **node** 或 **system:admin** 组中的 **node-bootstrapper** 服务帐户提交，并确认节点的身份。

- 要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```

注意

在有些 CSR 被批准前，一些 Operator 可能无法使用。

4. 现在，您的客户端请求已被批准，您必须查看添加到集群中的每台机器的服务器请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE     REQUESTOR                                                   CONDITION
csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal        Pending
csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal        Pending
...
```

5. 如果剩余的 CSR 没有被批准，且处于 **Pending** 状态，请批准集群机器的 CSR：

- 要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs oc adm certificate approve
```

6. 批准所有客户端和服务器 CSR 后，机器将 处于 **Ready 状态**。运行以下命令验证：

```
$ oc get nodes
```

输出示例

```
NAME      STATUS    ROLES   AGE  VERSION
master-0  Ready     master  73m  v1.29.4
master-1  Ready     master  73m  v1.29.4
master-2  Ready     master  74m  v1.29.4
worker-0  Ready     worker  11m  v1.29.4
worker-1  Ready     worker  11m  v1.29.4
```

注意

批准服务器 CSR 后可能需要几分钟时间让机器过渡到 **Ready 状态**。

其他信息

- 如需有关 CSR 的更多信息，请参阅 证书签名请求。

8.5.21. 添加 Ingress DNS 记录

如果在创建 Kubernetes 清单并生成 Ignition 配置时删除了 DNS 区配置，您必须手动创建指向 Ingress 负载均衡器的 DNS 记录。您可以创建一个通配符 ***.apps.{baseDomain}.** 或特定的记录。您可以根据要求使用 A、CNAME 和其他记录。

先决条件

- 已使用您置备的基础架构在 Microsoft Azure Stack Hub 上部署了 OpenShift Container Platform 集群。
- 安装 OpenShift CLI(**oc**)。
- 安装或更新 Azure CLI。

流程

1. 确认 Ingress 路由器已创建了负载均衡器并填充 **EXTERNAL-IP** 字段：

```
$ oc -n openshift-ingress get service router-default
```

输出示例

```
NAME            TYPE           CLUSTER-IP     EXTERNAL-IP     PORT(S)                      AGE
router-default  LoadBalancer  172.30.20.10  35.130.120.110
80:32288/TCP,443:31215/TCP  20
```

2. 将 Ingress 路由器 IP 导出为变量：

```
$ export PUBLIC_IP_ROUTER=`oc -n openshift-ingress get service router-default --no-
headers | awk '{print $4}'`
```

3. 在 DNS 区域中添加 ***.apps** 记录。

   a. 如果您要将此集群添加到新的 DNS 区，请运行：

```
$ az network dns record-set a add-record -g ${BASE_DOMAIN_RESOURCE_GROUP} -
z ${CLUSTER_NAME}.${BASE_DOMAIN} -n *.apps -a ${PUBLIC_IP_ROUTER} --ttl 300
```

   b. 如果您要将此集群添加到已存在的 DNS 区中，请运行：

```
$ az network dns record-set a add-record -g ${BASE_DOMAIN_RESOURCE_GROUP} -
z ${BASE_DOMAIN} -n *.apps.${CLUSTER_NAME} -a ${PUBLIC_IP_ROUTER} --ttl 300
```

如果您更喜欢添加特定域而不是使用通配符，可以为集群的每个当前路由创建条目：

```
$ oc get --all-namespaces -o jsonpath='{range .items[*]}{range .status.ingress[*]}{.host}{"\n"}{end}
{end}' routes
```

输出示例

```
oauth-openshift.apps.cluster.basedomain.com
console-openshift-console.apps.cluster.basedomain.com
downloads-openshift-console.apps.cluster.basedomain.com
alertmanager-main-openshift-monitoring.apps.cluster.basedomain.com
prometheus-k8s-openshift-monitoring.apps.cluster.basedomain.com
```

8.5.22. 在用户置备的基础架构上完成 Azure Stack Hub 安装

在 Microsoft Azure Stack Hub 用户置备的基础架构上启动 OpenShift Container Platform 安装后，您可以监控集群事件，直到集群就绪。

先决条件

- 在用户置备的 Azure Stack Hub 基础架构上为 OpenShift Container Platform 集群部署 bootstrap 机器。
- 安装 **oc** CLI 并登录。

流程

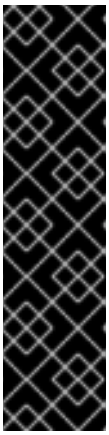
- 完成集群安装：

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

输出示例

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。



重要

- 安装程序生成的 Ignition 配置文件包含 24 小时后过期的证书，然后在该时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

8.6. AZURE STACK HUB 的安装配置参数

在 Azure Stack Hub 上部署 OpenShift Container Platform 集群前，您可以提供一个自定义的 **install-config.yaml** 安装配置文件，该文件描述了您的环境详情。

8.6.1. Azure Stack Hub 可用的安装配置参数

下表指定可设置为安装过程的一部分所需的、可选和 Azure Stack Hub 安装配置参数。

注意

安装后，您无法在 **install-config.yaml** 文件中修改这些参数。

8.6.1.1. 所需的配置参数

下表描述了所需的安装配置参数：

表 8.15. 所需的参数

参数	描述	值
apiVersion:	**install-config.yaml** 内容的 API 版本。当前版本为 **v1**。安装程序可能还支持旧的 API 版本。	字符串
baseDomain:	云供应商的基域。基域用于创建到 OpenShift Container Platform 集群组件的路由。集群的完整 DNS 名称是 **baseDomain** 和 **metadata.name** 参数值的组合，其格式为 **<metadata.name>.<baseDomain>**。	完全限定域名或子域名，如 **example.com**。
metadata:	Kubernetes 资源 **ObjectMeta**，其中只消耗 **name** 参数。	对象
metadata: name:	集群的名称。集群的 DNS 记录是 **{{.metadata.name}}.{{.baseDomain}}** 的子域。	小写字母、连字符(**-**)和句点(**.**)字符串，如 **dev**。
platform:	对于特定平台的配置取决于执行安装的环境： **aws**, **baremetal**, **azure**, **gcp**, **ibmcloud**, **nutanix**, **openstack**, **powervs**, **vsphere**, 或 **{}**.有关 **platform.<platform>** 参数的更多信息，请参考下表中您的特定平台。	对象

参数	描述	值
pullSecret:	从 Red Hat OpenShift Cluster Manager 获取 pull secret，验证从 Quay.io 等服务中下载 OpenShift Container Platform 组件的容器镜像。	{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }

8.6.1.2. 网络配置参数

您可以根据现有网络基础架构的要求自定义安装配置。例如，您可以扩展集群网络的 IP 地址块，或者提供不同于默认值的不同 IP 地址块。

仅支持 IPv4 地址。

注意

Red Hat OpenShift Data Foundation 灾难恢复解决方案不支持 Globalnet。对于区域灾难恢复场景，请确保为每个集群中的集群和服务网络使用非重叠的专用 IP 地址。

表 8.16. 网络参数

参数	描述	值
networking:	集群网络的配置。	对象  **注意** 您无法在安装后修改 **网络** 对象指定的参数。
networking: networkType:	要安装的 Red Hat OpenShift Networking 网络插件。	**OVNKubernetes**。**OVNKubernetes** 是 Linux 网络和包含 Linux 和 Windows 服务器的混合网络的 CNI 插件。默认值为 **OVNKubernetes**。

参数	描述	值
networking: clusterNetwork:	pod 的 IP 地址块。 默认值为 **10.128.0.0/14**，主机前缀为 **/23**。 如果您指定了多个 IP 地址块，块不得重叠。	对象数组。例如： networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23
networking: clusterNetwork: cidr:	使用 **networking.clusterNetwork** 时需要此项。IP 地址块。 IPv4 网络。	无类别域间路由(CIDR)表示法中的 IP 地址块。IPv4 块的前缀长度介于 **0 到 32** 之间 **。**
networking: clusterNetwork: hostPrefix:	分配给每个节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从 given **cidr** 中分配 a **/23** 子网。**hostPrefix** 值 **23** 提供 510（2^(32 - 23)- 2）pod IP 地址。	子网前缀。 默认值为 **23**。
networking: serviceNetwork:	服务的 IP 地址块。默认值为 **172.30.0.0/16**。 OVN-Kubernetes 网络插件只支持服务网络的一个 IP 地址块。	CIDR 格式具有 IP 地址块的数组。例如： networking: serviceNetwork: - 172.30.0.0/16
networking: machineNetwork:	机器的 IP 地址块。 如果您指定了多个 IP 地址块，块不得重叠。	对象数组。例如： networking: machineNetwork: - cidr: 10.0.0.0/16
networking: machineNetwork: cidr:	使用 **networking.machineNetwork** 时需要此项。IP 地址块。对于 libvirt 和 IBM Power® Virtual Server 以外的所有平台，默认值为 **10.0.0.0/16**。对于 libvirt，默认值 为 **192.168.126.0/24**。对于 IBM Power® Virtual Server，默认值为 **192.168.0.0/24**。	CIDR 表示法中的 IP 网络块。 例如： **10.0.0.0/16**。 **注意** 将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

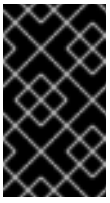
8.6.1.3. 可选的配置参数

下表描述了可选的安装配置参数：

表 8.17. 可选参数

参数	描述	值
additionalTrustBundle:	添加到节点可信证书存储中的 PEM 编码 X.509 证书捆绑包。配置了代理时，也可以使用此信任捆绑包。	字符串
capabilities:	控制可选核心组件的安装。您可以通过禁用可选组件来减少 OpenShift Container Platform 集群的空间。如需更多信息，请参阅*安装*中的"集群功能"页面。	字符串数组
capabilities: baselineCapabilitySet:	选择要启用的一组初始可选功能。有效值为 **None**、**v4.11**、**v4.12** 和 **vCurrent**。默认值为 **vCurrent**。	字符串
capabilities: additionalEnabledCapabilities:	将可选功能集合扩展到您在 **baselineCapabilitySet** 中指定的范围。您可以在此参数中指定多个功能。	字符串数组
cpuPartitioningMode:	启用工作负载分区，它会隔离 OpenShift Container Platform 服务、集群管理工作负载和基础架构 pod，以便在保留的一组 CPU 上运行。工作负载分区只能在安装过程中启用，且在安装后无法禁用。虽然此字段启用工作负载分区，但它不会将工作负载配置为使用特定的 CPU。如需更多信息，请参阅 *Scalability and Performance* 部分中的 *Workload partitioning* 页面。	**None** 或 **AllNodes**.**None** 是默认值。
compute:	组成计算节点的机器的配置。	**MachinePool** 对象的数组。
compute: architecture:	决定池中机器的指令集合架构。目前，不支持具有不同架构的集群。所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串

参数	描述	值
compute: hyperthreading:	是否在计算机器上启用或禁用并发多**线程或超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。	**enabled** 或 **Disabled**
compute: name:	使用 **compute** 时需要此项。机器池的名称。	**worker**
compute: platform:	使用 **compute** 时需要此项。使用此参数指定托管 worker 机器的云供应商。此参数值必须与 **controlPlane.platform** 参数值匹配。	**aws**, **azure**, **gcp**, **ibmcloud**, **nutanix**, **openstack**, **powervs**, **vsphere**, 或 **{}**
compute: replicas:	要置备的计算机器数量，也称为 worker 机器。	大于或等于 **2** 的正整数。默认值为 **3**。
featureSet:	为功能集启用集群。功能集是 OpenShift Container Platform 功能的集合，默认情况下不启用。有关在安装过程中启用功能集的更多信息，请参阅"使用功能门启用功能"。	字符串.要启用的功能集的名称，如 **TechPreviewNoUpgrade**。
controlPlane:	组成 control plane 的机器的配置。	**MachinePool** 对象的数组。
controlPlane: architecture:	决定池中机器的指令集合架构。目前，不支持具有不同架构的集群。所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串

参数	描述	值
controlPlane: hyperthreading:	是否在 control plane 机器上启用或禁用并发多**线程或超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。	**enabled** 或 **Disabled**
controlPlane: name:	使用 **controlPlane** 时需要此项。机器池的名称。	**master**
controlPlane: platform:	使用 **controlPlane** 时需要此项。使用此参数指定托管 control plane 机器的云供应商。此参数值必须与 **compute.platform** 参数值匹配。	**aws**, **azure**, **gcp**, **ibmcloud**, **nutanix**, **openstack**, **powervs**, **vsphere**, 或 **{}**
controlPlane: replicas:	要置备的 control plane 机器数量。	部署单节点 OpenShift 时支持的值为 **3** 或 **1**。
credentialsMode:	Cloud Credential Operator(CCO)模式。如果没有指定模式，CCO 会动态尝试决定提供的凭证的功能，在支持多个模式的平台上首选 mint 模式。	**Mint**、**Passthrough**、**Manual** 或空字符串(**""**)。[1]

参数	描述	值
fips:	启用或禁用 FIPS 模式。默认值为 **false**（禁用）。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。  **重要** 要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 Red Hat Enterprise Linux (RHEL) 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅在 FIPS 模式中安装该系统。 当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS) 时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。  **注意** 如果使用 Azure File 存储，则无法启用 FIPS 模式。	**false** 或 **true**
imageContentSources:	release-image 内容的源和存储库。	对象数组。包括一个 **source** 以及可选的 **mirrors**，如本表的以下行所述。
imageContentSources: source:	使用 **imageContentSources** 时需要此项。指定用户在镜像拉取规格中引用的存储库。	字符串

参数	描述	值
imageContentSources: mirrors:	指定可能还包含同一镜像的一个或多个仓库。	字符串数组
publish:	如何发布或公开集群的面向用户的端点，如 Kubernetes API、OpenShift 路由。	**内部或外部 .**默认值为 **External**。 在非云平台上不支持将此字段设置为 **Internal**。 **重要** 如果将字段的值设为 **Internal**，集群将无法运行。如需更多信息，请参阅 BZ#1953035。
sshKey:	用于验证对集群机器的访问的 SSH 密钥。 **注意** 对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。	例如，**sshKey: ssh-ed25519 AAAA..**。

1. 不是所有 CCO 模式都支持所有云供应商。有关 CCO 模式的更多信息，请参阅*身份验证和授权*内容中的"管理云供应商凭证"条目。

8.6.1.4. 其他 Azure Stack Hub 配置参数

下表描述了其他 Azure 配置参数：

表 8.18. 其他 Azure Stack Hub 参数

参数	描述	值
compute: platform: azure: osDisk: diskSizeGB:	虚拟机的 Azure 磁盘大小。	以 GB 为单位表示磁盘大小的整数。默认值为 **128**。

参数	描述	值
compute: platform: azure: osDisk: diskType:	定义磁盘的类型。	**standard_LRS** 或 **premium_LRS**. 默认值为 **Premium_LRS**。
compute: platform: azure: type:	定义计算机器的 azure 实例类型。	字符串
controlPlane: platform: azure: osDisk: diskSizeGB:	虚拟机的 Azure 磁盘大小。	以 GB 为单位表示磁盘大小的整数。默认值为 **1024**。
controlPlane: platform: azure: osDisk: diskType:	定义磁盘的类型。	**premium_LRS**。
controlPlane: platform: azure: type:	定义 control plane 机器的 azure 实例类型。	字符串
platform: azure: defaultMachinePlatform: osDisk: diskSizeGB:	虚拟机的 Azure 磁盘大小。	以 GB 为单位表示磁盘大小的整数。默认值为 **128**。

参数	描述	值

platform:
  azure:
 	
defaultMachinePlatform:
  osDisk:
  diskType:
	定义磁盘的类型。	**standard_LRS** 或 **premium_LRS**. 默认值为 **Premium_LRS**。

platform:
  azure:
 	
defaultMachinePlatform:
  type:
	control plane 和计算机器的 Azure 实例类型。	Azure 实例类型。

platform:
  azure:
  armEndpoint:
	Azure Stack Hub operator 提供的 Azure Resource Manager 端点的 URL。	字符串

platform: azure: baseDomainResourceGroupName:	包含基域的 DNS 区的资源组的名称。	字符串，如 **production_cluster**。

platform:
  azure:
  region:
	Azure Stack Hub 本地区域的名称。	字符串

platform: azure: resourceGroupName:	集群要安装到的已有资源组的名称。此资源组必须为空，并且仅用于此特定群集；群集组件假定资源组中所有资源的所有权。如果您将安装程序的服务主体范围限制到这个资源组，您必须确保您的环境中安装程序使用的所有其他资源都有必要的权限，如公共 DNS 区和虚拟网络。使用安装程序销毁集群会删除此资源组。	字符串，如 **existing_resource_group**。

参数	描述	值
platform: azure: outboundType:	用于将集群连接到互联网的出站路由策略。如果您使用用户定义的路由，则必须在安装集群前配置出站路由。安装程序不负责配置用户定义的路由。	**LoadBalancer** 或 **UserDefinedRouting**。默认值为 **LoadBalancer**。
platform: azure: cloudName:	用于使用适当的 Azure API 端点配置 Azure SDK 的 Azure 云环境名称。	**AzureStackCloud**
clusterOSImage:	Azure Stack 环境中包含 RHCOS VHD 存储 blob 的 URL。	字符串，例如 https://vhdsa.blob.example.example.com/vhd/rhcos-410.84.202112040202-0-azurestack.x86_64.vhd

8.7. 在 AZURE STACK HUB 上卸载集群

您可以删除部署到 Azure Stack Hub 的集群。

8.7.1. 删除使用安装程序置备的基础架构的集群

您可以从云中删除使用安装程序置备的基础架构的集群。

注意

卸载后，检查云供应商是否有未正确删除的资源，特别是在用户置备基础架构(UPI)集群中。可能存在安装程序未创建或安装程序无法访问的资源。

先决条件

- 有用于部署集群的安装程序副本。
- 有创建集群时安装程序生成的文件。

流程

1. 在用来安装集群的计算机中包含安装程序的目录中，运行以下命令：

```
$ ./openshift-install destroy cluster \
--dir <installation_directory> --log-level info 1 2
```

1. 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。
2. 要查看不同的详情，请指定 **warn**、**debug** 或 **error**，而不是 **info**。

注意

您必须为集群指定包含集群定义文件的目录。安装程序需要此目录中的 **metadata.json** 文件来删除集群。

2. 可选：删除 **<installation_directory>** 目录和 OpenShift Container Platform 安装程序。

第 9 章 在 GCP 上安装

9.1. 准备在 GCP 上安装

9.1.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读有关 选择集群安装方法的文档，并为用户准备它。

9.1.2. 在 GCP 上安装 OpenShift Container Platform 的要求

在 Google Cloud Platform(GCP)上安装 OpenShift Container Platform 之前，您必须创建一个服务帐户并配置 GCP 项目。如需有关创建项目 、启用 API 服务、配置 DNS、GCP 帐户限值和支持的 GCP 区域的详情，请参阅 配置 GCP 项目。

如果环境中无法访问云身份和访问管理 (IAM) API，或者不想将管理员级别的凭证 secret 存储在 **kube-system** 命名空间中，请参阅为 GCP 手动创建长期凭证。

9.1.3. 选择在 GCP 上安装 OpenShift Container Platform 的方法

您可以在安装程序置备或用户置备的基础架构上安装 OpenShift Container Platform。默认安装类型使用安装程序置备的基础架构，安装程序会在其中为集群置备底层基础架构。您还可以在您置备的基础架构上安装 OpenShift Container Platform。如果不使用安装程序置备的基础架构，您必须自己管理和维护集群资源。

如需有关安装程序置备和用户置备的安装过程的更多信息，请参阅 安装过程。

9.1.3.1. 在安装程序置备的基础架构上安装集群

您可以使用以下方法之一在 OpenShift Container Platform 安装程序置备的 GCP 基础架构上安装集群：

- **在 GCP 上快速安装集群** ：您可以在 OpenShift Container Platform 安装程序置备的 GCP 基础架构上安装 OpenShift Container Platform。您可以使用默认配置选项快速安装集群。
- 在 **GCP 上安装自定义集群** ：您可以在安装程序置备的 GCP 基础架构上安装自定义集群。安装程序允许在安装阶段应用一些自定义。其它自定义选项可 在安装后使用。
- **使用自定义网络在 GCP 上安装集群** ：您可以在安装过程中自定义 OpenShift Container Platform 网络配置，以便集群可以与现有 IP 地址分配共存并遵循您的网络要求。
- **在受限网络中的 GCP 上安装集群** ：您可以使用安装发行内容的内部镜像在 GCP 上安装 OpenShift Container Platform。您可以使用此方法安装不需要活跃互联网连接的集群来获取软件组件。虽然您可以使用镜像内容安装 OpenShift Container Platform，但您的集群仍需要访问互联网才能使用 GCP API。
- **将集群安装到现有的 Virtual Private Cloud**中：您可以在现有的 GCP Virtual Private Cloud(VPC)上安装 OpenShift Container Platform。如果您按照公司的指南设置了限制，可以使用这个安装方法，例如创建新帐户或基础架构的限制。
- **在现有 VPC 上安装私有集群：您可以在现有的** GCP VPC 上安装私有集群。您可以使用此方法在互联网不可见的内部网络中部署 OpenShift Container Platform。

9.1.3.2. 在用户置备的基础架构上安装集群

您可以使用以下方法之一在您置备的 GCP 基础架构上安装集群：

- **使用用户置备的基础架构在 GCP 上安装集群：** 您可以在您提供的 GCP 基础架构上安装 OpenShift Container Platform。您可以使用提供的 Deployment Manager 模板来协助安装。
- **在 GCP 中的用户置备的基础架构上安装带有共享 VPC 的集群**：您可以使用提供的 Deployment Manager 模板在共享 VPC 基础架构中创建 GCP 资源。
- **在带有用户置备的受限网络中的 GCP 上安装集群：您可以使用用户置备的基础架构** 在 GCP 上安装 OpenShift Container Platform。通过创建安装发行内容的内部镜像，您可以安装不需要活跃互联网连接的集群来获取软件组件。您还可以使用此安装方法来确保集群只使用满足您组织对外部内容控制的容器镜像。

9.1.4. 后续步骤

- 配置 GCP 项目

9.2. 配置 GCP 项目

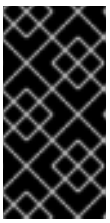
在安装 OpenShift Container Platform 之前，您必须配置 Google Cloud Platform(GCP)项目来托管它。

9.2.1. 创建 GCP 项目

要安装 OpenShift Container Platform，您必须在 Google Cloud Platform(GCP)帐户中创建项目来托管集群。

流程

- 创建一个项目来托管 OpenShift Container Platform 集群。请参阅 GCP 文档中的创建和管理 项目。



重要

如果您使用安装程序置备的基础架构，您的 GCP 项目必须使用 Premium Network Service Tier。使用安装程序安装的集群不支持 Standard Network Service Tier。安装程序为 **api-int.<cluster_name>.<base_domain>** URL 配置内部负载均衡；内部负载均衡需要 Premium Tier。

9.2.2. 在 GCP 中启用 API 服务

Google Cloud Platform(GCP)项目需要访问多个 API 服务来完成 OpenShift Container Platform 安装。

先决条件

- 已创建一个项目来托管集群。

流程

- 在托管集群的项目中启用以下所需的 API 服务。您还可以启用安装不需要的可选 API 服务。请参阅 GCP 文档中的 启用服务。

表 9.1. 所需的 API 服务

API 服务	控制台服务名称
Compute Engine API	**compute.googleapis.com**
Cloud Resource Manager API	**cloudresourcemanager.googleapis.com**
Google DNS API	**dns.googleapis.com**
IAM Service Account Credentials API	**iamcredentials.googleapis.com**
Identity and Access Management(IAM)API	**iam.googleapis.com**
Service Usage API	**serviceusage.googleapis.com**

表 9.2. 可选 API 服务

API 服务	控制台服务名称
Google Cloud API	**cloudapis.googleapis.com**
服务管理 API	**servicemanagement.googleapis.com**
Google Cloud Storage JSON API	**storage-api.googleapis.com**
Cloud Storage	**storage-component.googleapis.com**

9.2.3. 为 GCP 配置 DNS

要安装 OpenShift Container Platform，您使用的 Google Cloud Platform(GCP)帐户必须在托管 OpenShift Container Platform 集群的同一项目中有一个专用的公共托管区。此区域必须对域具有权威。DNS 服务为集群外部连接提供集群 DNS 解析和名称查询。

流程

1. 确定您的域或子域，以及注册商。您可以转移现有的域和注册商，或通过 GCP 或其他来源获取新的域和注册商。

注意

如果您购买了新的域，则需要时间来传播相关的 DNS 更改。有关通过 Google 购买域的更多信息，请参阅 Google Domains。

2. 在 GCP 项目中为您的域或子域创建一个公共托管区。请参阅 GCP 文档中的 创建公共区。使用适当的根域，如 **openshiftcorp.com 或** 子域，如 cluster **.openshiftcorp.com**。

3. 从托管区域记录中提取新的权威名称服务器。请参阅 GCP 文档中的 查找您的云 DNS 名称服务器。

您通常有四个名称服务器。

4. 更新域所用名称服务器的注册商记录。例如，如果您将域注册到 Google Domains，请参阅 Google Domains 帮助中的以下主题： 如何切换到自定义名称服务器。

5. 如果您将根域迁移到 Google Cloud DNS，请迁移您的 DNS 记录。请参阅 GCP 文档中的 Migrating to Cloud DNS。

6. 如果您使用子域，请按照贵公司的步骤将其委派记录添加到父域。这个过程可能包括对您公司的 IT 部门或控制您公司的根域和 DNS 服务的部门发出的请求。

9.2.4. GCP 帐户限值

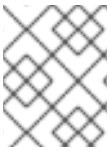
OpenShift Container Platform 集群使用许多 Google Cloud Platform(GCP)组件，但默认的 配额 不会影响您安装默认 OpenShift Container Platform 集群的能力。

默认集群包含三台计算和三台 control plane 机器，它使用以下资源。请注意，有些资源只在 bootstrap 过程中需要，并在集群部署后删除。

表 9.3. 默认集群中使用的 GCP 资源

service	组件	位置	所需的资源总数	bootstrap 后删除的资源
服务帐户	IAM	全局	6	1
防火墙规则	Compute	全局	11	1
转发规则	Compute	全局	2	0
使用的全局 IP 地址	Compute	全局	4	1
健康检查	Compute	全局	3	0
镜像	Compute	全局	1	0
网络	Compute	全局	2	0
静态 IP 地址	Compute	区域	4	1
路由器	Compute	全局	1	0
Routes	Compute	全局	2	0
子网	Compute	全局	2	0
目标池	Compute	全局	3	0
CPU	Compute	区域	28	4

service	组件	位置	所需的资源总数	bootstrap 后删除的资源
永久磁盘 SSD(GB)	Compute	区域	896	128



注意

如果在安装过程中任何配额不足，安装程序会显示一个错误信息，包括超过哪个配额，以及显示区域。

请考虑您的集群的实际大小、预定的集群增长，以及来自与您的帐户关联的其他集群的使用情况。CPU、静态 IP 地址和持久磁盘 SSD（存储）配额是最可能不足的。

如果您计划在以下区域之一部署集群，您将超过最大存储配额，并可能会超过 CPU 配额限制：

- **asia-east2**
- **asia-northeast2**
- **asia-south1**
- **australia-southeast1**
- **europe-north1**
- **europe-west2**
- **europe-west3**
- **europe-west6**
- **northamerica-northeast1**
- **southamerica-east1**
- **us-west2**

您可以从 GCP 控制台 增加资源配额，但可能需要提交一个支持问题单。务必提前规划集群大小，以便在安装 OpenShift Container Platform 集群前有足够的时间来等待支持问题单被处理。

9.2.5. 在 GCP 中创建服务帐户

OpenShift Container Platform 需要一个 Google Cloud Platform(GCP)服务帐户，它提供访问 Google API 中数据的验证和授权。如果您没有包含项目中所需角色的现有 IAM 服务帐户，您必须创建一个。

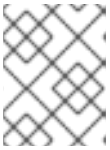
先决条件

- 已创建一个项目来托管集群。

流程

1. 在用于托管 OpenShift Container Platform 集群的项目中创建一个服务帐户。请参阅 GCP 文档中的 创建服务帐户。

2. 为服务帐户授予适当的权限。您可以逐一授予权限，也可以为其分配 **Owner** 角色。请参阅将角色转换到特定资源的服务帐户。



注意

将服务帐户设置为项目的所有者是获取所需权限的最简单方法，这意味着该服务帐户对项目有完全的控制权。您必须确定提供这种能力所带来的风险是否可以接受。

3. 您可以使用 JSON 格式创建服务帐户密钥，或将服务帐户附加到 GCP 虚拟机。请参阅 GCP 文档中的创建服务帐户密钥以及为实例创建并启用服务帐户。
您必须有一个服务帐户密钥或带有附加服务帐户的虚拟机来创建集群。

注意

如果您使用附加服务帐户的虚拟机来创建集群，则必须在安装前在 **install-config.yaml** 文件中设置 **credentialsMode: Manual**。

9.2.5.1. 所需的 GCP 角色

将 **Owner** 角色附加到您创建的服务帐户时，您可以为该服务帐户授予所有权限，包括安装 OpenShift Container Platform 所需的权限。如果机构的安全策略需要更严格的权限集，您可以创建具有以下权限的服务帐户：如果您将集群部署到现有的虚拟私有云 (VPC) 中，则服务帐户不需要某些网络权限，如以下列表中记录：

安装程序所需的角色

- Compute Admin
- Role Administrator
- Security Admin
- Service Account Admin
- Service Account Key Admin
- Service Account User
- Storage Admin

安装过程中创建网络资源所需的角色

- DNS Administrator

在 passthrough 模式中使用 Cloud Credential Operator 所需的角色

- Compute Load Balancer Admin

以下角色应用到 control plane 和计算机器使用的服务帐户：

表 9.4. GCP 服务帐户角色

帐户	角色
Control Plane	**roles/compute.instanceAdmin**
	roles/compute.networkAdmin
	roles/compute.securityAdmin
	roles/storage.admin
	roles/iam.serviceAccountUser
Compute	**roles/compute.viewer**
	roles/storage.admin

9.2.5.2. 安装程序置备的基础架构所需的 GCP 权限

将 **Owner** 角色附加到您创建的服务帐户时，您可以为该服务帐户授予所有权限，包括安装 OpenShift Container Platform 所需的权限。

如果机构的安全策略需要更严格的权限集，您可以创建具有所需权限的自定义角色。安装程序置备的基础架构需要以下权限来创建和删除 OpenShift Container Platform 集群。

例 9.1. 创建网络资源所需的权限

- **compute.addresses.create**
- **compute.addresses.createInternal**
- **compute.addresses.delete**
- **compute.addresses.get**
- **compute.addresses.list**
- **compute.addresses.use**
- **compute.addresses.useInternal**
- **compute.firewalls.create**
- **compute.firewalls.delete**
- **compute.firewalls.get**
- **compute.firewalls.list**
- **compute.forwardingRules.create**
- **compute.forwardingRules.get**

- **compute.forwardingRules.list**
- **compute.forwardingRules.setLabels**
- **compute.networks.create**
- **compute.networks.get**
- **compute.networks.list**
- **compute.networks.updatePolicy**
- **compute.routers.create**
- **compute.routers.get**
- **compute.routers.list**
- **compute.routers.update**
- **compute.routes.list**
- **compute.subnetworks.create**
- **compute.subnetworks.get**
- **compute.subnetworks.list**
- **compute.subnetworks.use**
- **compute.subnetworks.useExternalIp**

例 9.2. 创建负载均衡器资源所需的权限

- **compute.regionBackendServices.create**
- **compute.regionBackendServices.get**
- **compute.regionBackendServices.list**
- **compute.regionBackendServices.update**
- **compute.regionBackendServices.use**
- **compute.targetPools.addInstance**
- **compute.targetPools.create**
- **compute.targetPools.get**
- **compute.targetPools.list**
- **compute.targetPools.removeInstance**
- **compute.targetPools.use**

例 9.3. 创建 DNS 资源所需的权限

- **dns.changes.create**
- **dns.changes.get**
- **dns.managedZones.create**
- **dns.managedZones.get**
- **dns.managedZones.list**
- **dns.networks.bindPrivateDNSZone**
- **dns.resourceRecordSets.create**
- **dns.resourceRecordSets.list**

例 9.4. 创建服务帐户资源所需的权限

- **iam.serviceAccountKeys.create**
- **iam.serviceAccountKeys.delete**
- **iam.serviceAccountKeys.get**
- **iam.serviceAccountKeys.list**
- **iam.serviceAccounts.actAs**
- **iam.serviceAccounts.create**
- **iam.serviceAccounts.delete**
- **iam.serviceAccounts.get**
- **iam.serviceAccounts.list**
- **resourcemanager.projects.get**
- **resourcemanager.projects.getIamPolicy**
- **resourcemanager.projects.setIamPolicy**

例 9.5. 创建计算资源所需的权限

- **compute.disks.create**
- **compute.disks.get**
- **compute.disks.list**
- **compute.disks.setLabels**
- **compute.instanceGroups.create**

- **compute.instanceGroups.delete**
- **compute.instanceGroups.get**
- **compute.instanceGroups.list**
- **compute.instanceGroups.update**
- **compute.instanceGroups.use**
- **compute.instances.create**
- **compute.instances.delete**
- **compute.instances.get**
- **compute.instances.list**
- **compute.instances.setLabels**
- **compute.instances.setMetadata**
- **compute.instances.setServiceAccount**
- **compute.instances.setTags**
- **compute.instances.use**
- **compute.machineTypes.get**
- **compute.machineTypes.list**

例 9.6. 创建存储资源需要

- **storage.buckets.create**
- **storage.buckets.delete**
- **storage.buckets.get**
- **storage.buckets.list**
- **storage.objects.create**
- **storage.objects.delete**
- **storage.objects.get**
- **storage.objects.list**

例 9.7. 创建健康检查资源所需的权限

- **compute.healthChecks.create**
- **compute.healthChecks.get**

- compute.healthChecks.list
- compute.healthChecks.useReadOnly
- compute.httpHealthChecks.create
- compute.httpHealthChecks.get
- compute.httpHealthChecks.list
- compute.httpHealthChecks.useReadOnly

例 9.8. 获取 GCP 区域和区域相关信息所需的权限

- compute.globalOperations.get
- compute.regionOperations.get
- compute.regions.list
- compute.zoneOperations.get
- compute.zones.get
- compute.zones.list

例 9.9. 检查服务和配额所需的权限

- monitoring.timeSeries.list
- serviceusage.quotas.get
- serviceusage.services.list

例 9.10. 安装所需的 IAM 权限

- iam.roles.get

例 9.11. 安装的可选镜像权限

- compute.images.list

例 9.12. 运行收集 bootstrap 的可选权限

- compute.instances.getSerialPortOutput

例 9.13. 删除网络资源所需的权限

- compute.addresses.delete

- compute.addresses.deleteInternal
- compute.addresses.list
- compute.firewalls.delete
- compute.firewalls.list
- compute.forwardingRules.delete
- compute.forwardingRules.list
- compute.networks.delete
- compute.networks.list
- compute.networks.updatePolicy
- compute.routers.delete
- compute.routers.list
- compute.routes.list
- compute.subnetworks.delete
- compute.subnetworks.list

例 9.14. 删除负载均衡器资源所需的权限

- compute.regionBackendServices.delete
- compute.regionBackendServices.list
- compute.targetPools.delete
- compute.targetPools.list

例 9.15. 删除 DNS 资源所需的权限

- dns.changes.create
- dns.managedZones.delete
- dns.managedZones.get
- dns.managedZones.list
- dns.resourceRecordSets.delete
- dns.resourceRecordSets.list

例 9.16. 删除服务帐户资源所需的权限

- iam.serviceAccounts.delete
- iam.serviceAccounts.get
- iam.serviceAccounts.list
- resourcemanager.projects.getIamPolicy
- resourcemanager.projects.setIamPolicy

例 9.17. 删除计算资源所需的权限

- compute.disks.delete
- compute.disks.list
- compute.instanceGroups.delete
- compute.instanceGroups.list
- compute.instances.delete
- compute.instances.list
- compute.instances.stop
- compute.machineTypes.list

例 9.18. 删除存储资源需要

- storage.buckets.delete
- storage.buckets.getIamPolicy
- storage.buckets.list
- storage.objects.delete
- storage.objects.list

例 9.19. 删除健康检查资源所需的权限

- compute.healthChecks.delete
- compute.healthChecks.list
- compute.httpHealthChecks.delete
- compute.httpHealthChecks.list

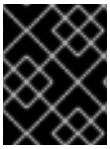
例 9.20. 删除所需的镜像权限

- **compute.images.list**

9.2.5.3. 共享 VPC 安装所需的 GCP 权限

当将集群安装到 共享 VPC 时，您必须为主机项目和服务项目配置服务帐户。如果您没有安装到共享 VPC 中，您可以跳过本节。

您必须将上面列出的标准安装所需的最小角色应用到服务项目。



重要

对于以手动或 mint 凭证模式运行的 Cloud Credential Operator，您可以使用粒度权限。您不能在 passthrough 凭证模式中使用粒度权限。

确保主机项目将以下配置之一应用到服务帐户：

例 9.21. 在主机项目中创建防火墙所需的权限

- **projects/<host-project>/roles/dns.networks.bindPrivateDNSZone**
- **roles/compute.networkAdmin**
- **roles/compute.securityAdmin**

例 9.22. 所需的最小权限

- **projects/<host-project>/roles/dns.networks.bindPrivateDNSZone**
- **roles/compute.networkUser**

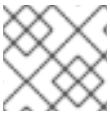
9.2.6. 支持的 GCP 区域

您可以将 OpenShift Container Platform 集群部署到以下 Google Cloud Platform(GCP)区域：

- **africa-south1** (Johannesburg, South Africa)
- **asia-east1** (Changhua County, Taiwan)
- **asia-east2** (Hong Kong)
- **asia-northeast1** (Tokyo, Japan)
- **asia-northeast2** (Osaka, Japan)
- **asia-northeast3** (Seoul, South Korea)
- **asia-south1** (Mumbai, India)
- **asia-south2** (Delhi, India)
- **asia-southeast1** (Jurong West, Singapore)

- **asia-southeast2** (Jakarta, Indonesia)
- **aia-southeast1(** Sydney, Australia)
- **australia-southeast2** (Melbourne, Australia)
- **europe-central2** (Warsaw, USA)
- **europe-north1** (Hamina, Finland)
- **europe-southwest1** (Madrid, Spain)
- **europe-west1** (St. Ghislain, Belgium)
- **europe-west2** (London, England, UK)
- **europe-west3** (Frankfurt, Germany)
- **europe-west4** (Eemshaven, Netherlands)
- **europe-west6** (Zürich, Switzerland)
- **europe-west8** (Milan, Italy)
- **europe-west9** (Paris, France)
- **europe-west12** (Turin, Italy)
- **me-central1** (Doha, Qatar, Middle East)
- **me-central2** (Dammam, Saudi Arabia, Middle East)
- **me-west1** (Tel Aviv, Israel)
- **northamerica-northeast1** (Montréal, Québec, Canada)
- **northamerica-northeast2** (Toronto, Ontario, Canada)
- **southamerica-east1** (São Paulo, Brazil)
- **southamerica-west1** (Santiago, Chile)
- **us-central1** (Council Bluffs, Iowa, USA)
- **us-east1** (Moncks Corner, South Carolina, USA)
- **us-east4** (Ashburn, North Virginia, USA)
- **us-east5** (Columbus, Ohio)
- **us-south1** (Dallas, Texas)
- **us-west1** (The Dalles, Oregon, USA)
- **us-west2** (Los Angeles, California, USA)
- **us-west3** (Salt Lake City, Utah, USA)

- **us-west4** (Las Vegas, Nevada, USA)



注意

要确定地区和区域中可以使用哪些机器类型实例，请参阅 Google 文档。

9.2.7. 后续步骤

- 在 GCP 上安装 OpenShift Container Platform 集群。您可以安装自定义集群，或使用默认选项快速安装集群。

9.3. 在 GCP 上快速安装集群

在 OpenShift Container Platform 版本 4.16 中，您可以使用默认配置选项在 Google Cloud Platform (GCP) 上安装集群。

9.3.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读有关 选择集群安装方法的文档，并为用户准备它。
- 已将 GCP 项目配置为 托管集群。
- 如果使用防火墙，将其配置为允许集群需要访问的站点。

9.3.2. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类型的基础架构上执行受限网络安装。在此过程中，您可以下载所需的内容，并使用它为镜像 registry 填充安装软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群前，您要更新镜像 registry 的内容。

9.3.3. 为集群节点 SSH 访问生成密钥对

在 OpenShift Container Platform 安装过程中，您可以为安装程序提供 SSH 公钥。密钥通过它们的 Ignition 配置文件传递给 Red Hat Enterprise Linux CoreOS(RHCOS)节点，用于验证对节点的 SSH 访问。密钥添加到每个节点上 **core** 用户的 **~/.ssh/authorized_keys** 列表中，这将启用免密码身份验证。

将密钥传递给节点后，您可以使用密钥对作为用户 **核心** 通过 SSH 连接到 RHCOS 节点。若要通过 SSH 访问节点，必须由 SSH 为您的本地用户管理私钥身份。

如果要通过 SSH 连接到集群节点来执行安装调试或灾难恢复，则必须在安装过程中提供 SSH 公钥。**./openshift-install gather** 命令还需要在集群节点上设置 SSH 公钥。

重要

不要在生产环境中跳过这个过程，在生产环境中需要灾难恢复和调试。

注意

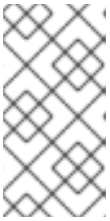
您必须使用本地密钥，而不是使用特定平台方法配置 的密钥，如 AWS 密钥对。

流程

1. 如果您在本地计算机上没有可用于在集群节点上进行身份验证的现有 SSH 密钥对，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 ~/**.ssh/id_ed25519**。如果您已有密钥对，请确保您的公钥位于 ~/**.ssh** 目录中。



注意

如果您计划在 **x86_64**、**ppc64le** 和 **s390x** 架构上安装使用 RHEL 加密库（这些加密库已提交给 NIST 用于 FIPS 140-2/140-3 验证）的 OpenShift Container Platform 集群，则不要创建使用 **ed25519** 算法的密钥。相反，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 查看公共 SSH 密钥：

```
$ cat <path>/<file_name>.pub
```

例如，运行以下命令来查看 ~/**.ssh/id_ed25519.pub** 公钥：

```
$ cat ~/.ssh/id_ed25519.pub
```

3. 将 SSH 私钥身份添加到本地用户的 SSH 代理（如果尚未添加）。在集群节点上，或者要使用 **./openshift-install gather** 命令，需要对该密钥进行 SSH 代理管理，才能在集群节点上进行免密码 SSH 身份验证。

注意

在某些发行版中，自动管理默认 SSH 私钥身份，如 ~/**.ssh/id_rsa** 和 ~/**.ssh/id_dsa**。

a. 如果 **ssh-agent** 进程尚未为您的本地用户运行，请将其作为后台任务启动：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果集群处于 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

4. 将 SSH 私钥添加到 **ssh-agent** ：

```
$ ssh-add <path>/<file_name> 1
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_ed25519.pub**

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

后续步骤

- 安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

9.3.4. 获取安装程序

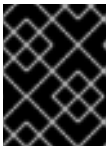
在安装 OpenShift Container Platform 前，将安装文件下载到您用于安装的主机上。

先决条件

- 您有一台运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB。

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐户，请使用您的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入到安装类型的页面，下载与您的主机操作系统和架构对应的安装程序，并将该文件放在您要存储安装配置文件的目录中。



重要

安装程序会在用来安装集群的计算机上创建几个文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。

重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，请为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。此 pull secret 允许您与所含授权机构提供的服务进行身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

9.3.5. 部署集群

您可以在兼容云平台上安装 OpenShift Container Platform。

重要

在初始安装过程中，您只能运行安装程序的 **create cluster** 命令一次。

先决条件

- 您已使用托管集群的云平台配置了帐户。
- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。
- 已确认主机上的云供应商帐户具有部署集群的正确权限。权限不正确的帐户会导致安装过程失败，并显示包括缺失权限的错误消息。

流程

1. 删除所有不使用您为集群配置的 GCP 帐户的服务帐户密钥的现有 GCP 凭证，这些凭证存储在以下位置：
   - **GOOGLE_CREDENTIALS**、**GOOGLE_CLOUD_KEYFILE_JSON** 或 **GCLOUD_KEYFILE_JSON** 环境变量
   - The ~/**.gcp/osServiceAccount.json** file
   - **gcloud cli** 默认凭证
2. 进入包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定要存储安装程序创建的文件的目录名称。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不是 **info**。

在指定目录时：

- 验证该目录是否具有**执行**权限。在安装目录中运行 Terraform 二进制文件需要这个权限。
- 使用空目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

3. 在提示符处提供值：

a. 可选：选择用于访问集群机器的 SSH 密钥。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

b. 选择 **gcp** 作为目标平台。

c. 如果您没有为主机上的 GCP 帐户配置服务帐户密钥，则必须从 GCP 获取它，并粘贴文件的内容或输入文件的绝对路径。

d. 选择要在其中置备集群的项目 ID。默认值由您配置的服务帐户指定。

e. 选择要将集群部署到的区域。

f. 选择集群要部署到的基域。基域与您为集群创建的公共 DNS 区对应。

g. 为集群输入描述性名称。如果您提供的名称超过 6 个字符，则只有前 6 个字符用于从集群名称生成的基础架构 ID 中。

h. 粘贴 Red Hat OpenShift Cluster Manager 中的 pull secret 。

4. 可选：您可以减少用来安装集群的服务帐户的权限数量。

- 如果将 **Owner** 角色分配给您的服务帐户，您可以删除该角色并将其替换为 **Viewer** 角色。
- 如果包含 **Service Account Key Admin** 角色，您可以将其删除。

验证

当集群部署成功完成时：

- 终端会显示用于访问集群的说明，包括指向 Web 控制台和 **kubeadmin** 用户的凭证的链接。
- 凭证信息还会输出到 **<installation_directory>/.openshift_install.log**.

重要

不要删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

输出示例

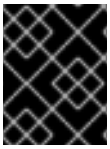
```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

9.3.6. 安装 OpenShift CLI

您可以安装 OpenShift CLI(**oc**)来使用命令行界面与 OpenShift Container Platform 进行交互。您可以在 Linux、Windows 或 macOS 上安装 **oc**。



重要

如果安装了旧版本的 **oc**，则可能无法使用 OpenShift Container Platform 4.16 中的所有命令。下载并安装新版本的 **oc**。

在 Linux 上安装 OpenShift CLI
您可以按照以下流程在 Linux 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **产品变体** 下拉列表中选择架构。
3. 从 **版本** 下拉列表中选择适当的版本。
4. 点 **OpenShift v4.16 Linux Client** 条目旁的 **Download Now** 来保存文件。
5. 解包存档：

```
$ tar xvf <file>
```

6. 将 **oc** 二进制文件放到 **PATH 中的目录中**。
   要查看您的 **PATH**，请执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

在 Windows 上安装 OpenShift CLI
您可以按照以下流程在 Windows 上安装 OpenShift CLI(**oc**)二进制文件。

流程

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 Windows Client** 条目旁的 **Download Now** 来保存文件。
4. 使用 ZIP 程序解压存档。
5. 将 **oc** 二进制文件移到 **PATH 中的目录中**。
   要查看您的 **PATH**，请打开命令提示并执行以下命令：

```
C:\> path
```

验证

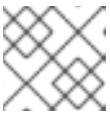
- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

在 macOS 上安装 OpenShift CLI
您可以按照以下流程在 macOS 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 macOS Client** 条目旁的 **Download Now** 来保存文件。



注意

对于 macOS arm64，请选择 **OpenShift v4.16 macOS arm64 Client** 条目。

4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
   要查看您的 **PATH**，请打开终端并执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

9.3.7. 使用 CLI 登录集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

其他资源

- 如需有关 访问和了解 OpenShift Container Platform Web 控制台的更多详情，请参阅 访问 Web 控制台。

9.3.8. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于 远程健康监控

9.3.9. 后续步骤

- 自定义集群。
- 如果需要，您可以选择 不使用远程健康报告。

9.4. 使用自定义在 GCP 上安装集群

在 OpenShift Container Platform 版本 4.16 中，您可以在安装程序在 Google Cloud Platform(GCP)上置备的基础架构上安装自定义的集群。要自定义安装，请在安装集群前修改 **install-config.yaml** 文件中的参数。

9.4.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读有关 选择集群安装方法的文档，并为用户准备它。
- 已将 GCP 项目配置为 托管集群。
- 如果使用防火墙，将其配置为允许集群需要访问的站点。

9.4.2. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

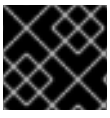
如果您的集群无法直接访问互联网，则可以在置备的某些类型的基础架构上执行受限网络安装。在此过程中，您可以下载所需的内容，并使用它为镜像 registry 填充安装软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群前，您要更新镜像 registry 的内容。

9.4.3. 为集群节点 SSH 访问生成密钥对

在 OpenShift Container Platform 安装过程中，您可以为安装程序提供 SSH 公钥。密钥通过它们的 Ignition 配置文件传递给 Red Hat Enterprise Linux CoreOS(RHCOS)节点，用于验证对节点的 SSH 访问。密钥添加到每个节点上 **core** 用户的 **~/.ssh/authorized_keys** 列表中，这将启用免密码身份验证。

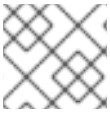
将密钥传递给节点后，您可以使用密钥对作为用户 **核心** 通过 SSH 连接到 RHCOS 节点。若要通过 SSH 访问节点，必须由 SSH 为您的本地用户管理私钥身份。

如果要通过 SSH 连接到集群节点来执行安装调试或灾难恢复，则必须在安装过程中提供 SSH 公钥。**./openshift-install gather** 命令还需要在集群节点上设置 SSH 公钥。



重要

不要在生产环境中跳过这个过程，在生产环境中需要灾难恢复和调试。



注意

您必须使用本地密钥，而不是使用特定平台方法配置 的密钥，如 AWS 密钥对。

流程

1. 如果您在本地计算机上没有可用于在集群节点上进行身份验证的现有 SSH 密钥对，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 ~/**.ssh/id_ed25519**。如果您已有密钥对，请确保您的公钥位于 ~/**.ssh** 目录中。

注意

如果您计划在 **x86_64**、**ppc64le** 和 **s390x** 架构上安装使用 RHEL 加密库（这些加密库已提交给 NIST 用于 FIPS 140-2/140-3 验证）的 OpenShift Container Platform 集群，则不要创建使用 **ed25519** 算法的密钥。相反，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

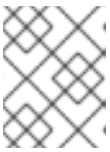
2. 查看公共 SSH 密钥：

```
$ cat <path>/<file_name>.pub
```

例如，运行以下命令来查看 ~/**.ssh/id_ed25519.pub** 公钥：

```
$ cat ~/.ssh/id_ed25519.pub
```

3. 将 SSH 私钥身份添加到本地用户的 SSH 代理（如果尚未添加）。在集群节点上，或者要使用 **./openshift-install gather** 命令，需要对该密钥进行 SSH 代理管理，才能在集群节点上进行免密码 SSH 身份验证。



注意

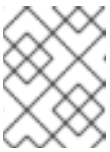
在某些发行版中，自动管理默认 SSH 私钥身份，如 ~/**.ssh/id_rsa** 和 ~/**.ssh/id_dsa**。

a. 如果 **ssh-agent** 进程尚未为您的本地用户运行，请将其作为后台任务启动：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```



注意

如果集群处于 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

4. 将 SSH 私钥添加到 **ssh-agent** ：

```
$ ssh-add <path>/<file_name> 1
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_ed25519.pub**

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

后续步骤

- 安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

9.4.4. 获取安装程序

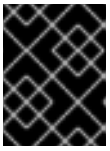
在安装 OpenShift Container Platform 前，将安装文件下载到您用于安装的主机上。

先决条件

- 您有一台运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB。

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐户，请使用您的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入到安装类型的页面，下载与您的主机操作系统和架构对应的安装程序，并将该文件放在您要存储安装配置文件的目录中。



重要

安装程序会在用来安装集群的计算机上创建几个文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。

重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，请为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。此 pull secret 允许您与所含授权机构提供的服务进行身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

9.4.5. 创建安装配置文件

您可以自定义在 Google Cloud Platform(GCP)上安装的 OpenShift Container Platform 集群。

先决条件

- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。

流程

1. 创建 **install-config.yaml** 文件。
   a. 进入包含安装程序的目录并运行以下命令：

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

   1 对于 **<installation_directory>**，请指定要存储安装程序创建的文件的目录名称。

   在指定目录时：

   - 验证该目录是否具有**执行**权限。在安装目录中运行 Terraform 二进制文件需要这个权限。
   - 使用空目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

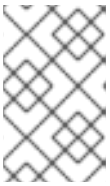
   b. 在提示符处，提供云的配置详情：
      i. 可选： 选择用于访问集群机器的 SSH 密钥。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

      ii. 选择 **gcp** 作为目标平台。
      iii. 如果您没有为计算机上的 GCP 帐户配置服务帐户密钥，则必须从 GCP 获取它，并粘贴文件的内容或输入文件的绝对路径。
      iv. 选择要在其中置备集群的项目 ID。默认值由您配置的服务帐户指定。
      v. 选择要将集群部署到的区域。
      vi. 选择集群要部署到的基域。基域与您为集群创建的公共 DNS 区对应。
      vii. 为集群输入描述性名称。

2. 修改 **install-config.yaml** 文件。您可以在"安装配置参数"部分找到有关可用参数的更多信息。



注意

如果要安装三节点集群，请确保将 **compute.replicas** 参数设置为 **0**。这样可确保集群的 control plane 可以调度。如需更多信息，请参阅"在 GCP 上安装三节点集群"。

3. 备份 **install-config.yaml** 文件，以便您可以使用它安装多个集群。

重要

install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用此文件，必须现在备份。

其他资源

- GCP 的安装配置参数

9.4.5.1. 集群安装的最低资源要求

每台集群机器都必须满足以下最低要求：

表 9.5. 最低资源要求

机器	操作系统	vCPU [1]	虚拟内存	Storage	每秒输入/输出 (IOPS) [2]
bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane（控制平面）	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS、RHEL 8.6 及更新版本 [3]	2	8 GB	100 GB	300

1. 当未启用并发多线程(SMT)或超线程时，一个 vCPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比例：（每个内核数的线程）× sockets = vCPU。

2. OpenShift Container Platform 和 Kubernetes 对磁盘性能非常敏感，建议使用更快的存储速度，特别是 control plane 节点上需要 10 ms p99 fsync 持续时间的 etcd。请注意，在许多云平台上，存储大小和 IOPS 可一起扩展，因此您可能需要过度分配存储卷来获取足够的性能。

3. 与所有用户置备的安装一样，如果您选择在集群中使用 RHEL 计算机器，则负责所有操作系统生命周期管理和维护，包括执行系统更新、应用补丁和完成所有其他必要的任务。RHEL 7 计算机器的使用已弃用，并已在 OpenShift Container Platform 4.10 及更新的版本中删除。

注意

从 OpenShift Container Platform 版本 4.13 开始，RHCOS 基于 RHEL 版本 9.2，它更新了微架构要求。以下列表包含每个架构需要的最小指令集架构 (ISA)：

- x86-64 体系结构需要 x86-64-v2 ISA
- ARM64 架构需要 ARMv8.0-A ISA
- IBM Power 架构需要 Power 9 ISA
- s390x 架构需要 z14 ISA

如需更多信息，请参阅 RHEL 架构。

如果平台的实例类型满足集群机器的最低要求，则 OpenShift Container Platform 支持使用它。

其他资源

- 优化存储

9.4.5.2. 为 GCP 测试的实例类型

以下 Google Cloud Platform 实例类型已使用 OpenShift Container Platform 测试。

例 9.23. 机器系列

- **A2**
- **A3**
- **C2**
- **C2D**
- **C3**
- **C3D**
- **E2**
- **M1**
- **N1**
- **N2**
- **N2D**
- **Tau T2D**

9.4.5.3. 在 64 位 ARM 基础架构上为 GCP 测试的实例类型

以下 Google Cloud Platform (GCP) 64 位 ARM 实例类型已使用 OpenShift Container Platform 测试。

例 9.24. 64 位 ARM 机器的机器系列

- **Tau T2A**

9.4.5.4. 使用自定义机器类型

支持使用自定义机器类型来安装 OpenShift Container Platform 集群。

使用自定义机器类型时请考虑以下几点：

- 与预定义的实例类型类似，自定义机器类型必须满足 control plane 和计算机器的最低资源要求。如需更多信息，请参阅"集群安装的资源要求"。
- 自定义机器类型的名称必须遵循以下语法：
  custom-<number_of_cpus>-<amount_of_memory_in_mb>

  例如，**custom-6-20480**。

作为安装过程的一部分，您可以在 **install-config.yaml** 文件中指定自定义机器类型。

带有自定义机器类型的 install-config.yaml 文件示例

```
compute:
- architecture: amd64
  hyperthreading: Enabled
  name: worker
  platform:
    gcp:
      type: custom-6-20480
  replicas: 2
controlPlane:
  architecture: amd64
  hyperthreading: Enabled
  name: master
  platform:
    gcp:
      type: custom-6-20480
  replicas: 3
```

9.4.5.5. 启用屏蔽虚拟机

您可在安装集群时使用 Shielded 虚拟机。Shielded 虚拟机具有额外的安全功能，包括安全引导、固件和完整性监控和 rootkit 检测。如需更多信息，请参阅 Google 文档中有关 Shielded 虚拟机 的文档。

注意

目前，在具有 64 位 ARM 基础架构的集群中不支持 Shielded 虚拟机。

先决条件

- 您已创建了 **install-config.yaml** 文件。

流程

- 在部署集群前，使用文本编辑器编辑 **install-config.yaml** 文件并添加以下部分之一：

  a. 仅将屏蔽的虚拟机用于 control plane 机器：

```
controlPlane:
  platform:
    gcp:
      secureBoot: Enabled
```

  b. 仅将屏蔽的虚拟机用于计算机器：

```
compute:
- platform:
    gcp:
      secureBoot: Enabled
```

  c. 将屏蔽的虚拟机用于所有机器：

```
platform:
  gcp:
    defaultMachinePlatform:
      secureBoot: Enabled
```

9.4.5.6. 启用机密虚拟机

您可在安装集群时使用机密虚拟机。机密虚拟机在处理数据时加密数据。如需更多信息，请参阅 Google 文档中有关 机密计算的内容。您可以同时启用机密虚拟机和 Shielded 虚拟机，虽然它们不相互依赖。

注意

64 位 ARM 架构目前不支持机密虚拟机。

先决条件

- 您已创建了 **install-config.yaml** 文件。

流程

- 在部署集群前，使用文本编辑器编辑 **install-config.yaml** 文件并添加以下部分之一：

  a. 仅将机密虚拟机用于 control plane 机器：

```
controlPlane:
  platform:
    gcp:
      confidentialCompute: Enabled 1
      type: n2d-standard-8 2
      onHostMaintenance: Terminate 3
```

  1 启用机密虚拟机。

  2 指定支持机密虚拟机的机器类型。机密虚拟机需要 N2D 或 C2D 系列机器类型。有关支持的机器类型的更多信息，请参阅支持的操作系统和机器类型。

3 指定主机维护事件期间虚拟机的行为，如硬件或软件更新。对于使用机密虚拟机的机器，此值必须设置为 **Terminate**，这会停止虚拟机。机密虚拟机不支持实时迁移。

b. 仅将机密虚拟机用于计算机器：

```
compute:
- platform:
    gcp:
      confidentialCompute: Enabled
      type: n2d-standard-8
      onHostMaintenance: Terminate
```

c. 将机密虚拟机用于所有机器：

```
platform:
  gcp:
    defaultMachinePlatform:
      confidentialCompute: Enabled
      type: n2d-standard-8
      onHostMaintenance: Terminate
```

9.4.5.7. GCP 的自定义 install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多详情，或修改所需参数的值。

重要

此示例 YAML 文件仅供参考。您必须使用安装程序来获取 **install-config.yaml** 文件，并进行修改。

```
apiVersion: v1
baseDomain: example.com 1
credentialsMode: Mint 2
controlPlane: 3 4
  hyperthreading: Enabled 5
  name: master
  platform:
    gcp:
      type: n2-standard-4
      zones:
      - us-central1-a
      - us-central1-c
      osDisk:
        diskType: pd-ssd
        diskSizeGB: 1024
        encryptionKey: 6
          kmsKey:
            name: worker-key
            keyRing: test-machine-keys
            location: global
            projectID: project-id
```

```
      tags: 7
      - control-plane-tag1
      - control-plane-tag2
      osImage: 8
        project: example-project-name
        name: example-image-name
  replicas: 3
compute: 9 10
- hyperthreading: Enabled 11
  name: worker
  platform:
    gcp:
      type: n2-standard-4
      zones:
      - us-central1-a
      - us-central1-c
      osDisk:
        diskType: pd-standard
        diskSizeGB: 128
        encryptionKey: 12
          kmsKey:
            name: worker-key
            keyRing: test-machine-keys
            location: global
            projectID: project-id
        tags: 13
        - compute-tag1
        - compute-tag2
        osImage: 14
          project: example-project-name
          name: example-image-name
  replicas: 3
metadata:
  name: test-cluster 15
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 16
  serviceNetwork:
  - 172.30.0.0/16
platform:
  gcp:
    projectID: openshift-production 17
    region: us-central1 18
    defaultMachinePlatform:
      tags: 19
      - global-tag1
      - global-tag2
      osImage: 20
        project: example-project-name
        name: example-image-name
```

```
pullSecret: '{"auths": ...}' 21
fips: false 22
sshKey: ssh-ed25519 AAAA... 23
```

1 15 17 18 21 必需。安装程序会提示您输入这个值。

2 可选：添加此参数来强制 Cloud Credential Operator (CCO) 使用指定的模式。默认情况下，CCO 使用 **kube-system** 命名空间中的 root 凭证来动态尝试决定凭证的功能。有关 CCO 模式的详情，请参阅*身份验证和授权*指南中的"About the Cloud Credential Operator"部分。

3 9 如果没有提供这些参数和值，安装程序会提供默认值。

4 10 **controlPlane** 部分是一个单个映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，controlPlane **部分** 的第一行则不以连字符开头。仅使用一个 control plane 池。

5 11 是否要启用或禁用并发多线程或 **超线程**。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将 参数值设置为 **Disabled** 来禁用它。如果在某些集群机器中禁用并发多线程，则必须在所有集群机器中禁用它。

重要

如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。如果您禁用并发多线程，请为您的机器使用较大的类型，如 **n1-standard-8**。

6 12 可选：自定义加密密钥部分来加密虚拟机和持久性卷。您的默认计算服务帐户必须具有相应的权限才能使用您的 KMS 密钥并分配了正确的 IAM 角色。默认服务帐户名称遵循 **service-<project_number>@compute-system.iam.gserviceaccount.com** 模式。有关为您的服务帐户授予正确权限的更多信息，请参阅 "Machine management" → "Creating compute machine set" → "Creating a compute machine set on GCP"。

7 13 19 可选：要应用到 control plane 或计算机器集的一组网络标签。**platform.gcp.defaultMachinePlatform.tags** 参数将应用到 control plane 和计算机器。如果设置了 **compute.platform.gcp.tags** 或 **controlPlane.platform.gcp.tags** 参数，它们会覆盖 **platform.gcp.defaultMachinePlatform.tags** 参数。

8 14 20 可选：应该用来引导 control plane 和计算机器的自定义 Red Hat Enterprise Linux CoreOS (RHCOS)。**platform.gcp.defaultMachinePlatform.osImage** 下的 **project** 和 **name** 参数应用到 control plane 和计算机器。如果设置了 **controlPlane.platform.gcp.osImage** 或 **compute.platform.gcp.osImage** 下的 **project** 和 **name** 参数，它们会覆盖 **platform.gcp.defaultMachinePlatform.osImage** 参数。

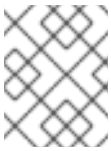
16 要安装的集群网络插件。默认值 **OVNKubernetes** 是唯一支持的值。

22 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS)时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。

23 您可选择提供用于访问集群中机器的 **sshKey** 值。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

其他资源

- 为计算机器设置启用客户管理的加密密钥

9.4.5.8. 在安装过程中配置集群范围的代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 **Proxy** 对象的 **spec.noProxy** 字段中添加站点来绕过代理。

注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，**Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 **http**。

2 用于创建集群外 HTTPS 连接的代理 URL。

3 要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 . 以仅匹配子域。例如，**.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代理。

4 如果提供，安装程序会在 **openshift-config** 命名空间中生成名为 **user-ca-bundle** 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 **trusted-ca-bundle** 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， **Proxy** 对象的 **trustedCA** 字段中也会引用此配置映射。**additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

5 可选：决定 **Proxy** 对象的配置以引用 **trustedCA** 字段中 **user-ca-bundle** 配置映射的策略。允许的值是 **Proxyonly** 和 **Always**。仅在配置了 **http/https** 代理时，使用 **Proxyonly** 引用 **user-ca-bundle** 配置映射。使用 **Always** 始终引用 **user-ca-bundle** 配置映射。默认值为 **Proxyonly**。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。

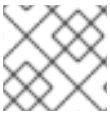
注意

如果安装程序超时，重启并使用安装程序的 **wait-for** 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. 保存该文件并在安装 OpenShift Container Platform 时引用。

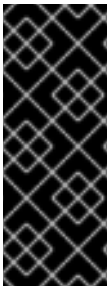
安装程序会创建一个名为 cluster 的集群范围代理，该代理 **使用** 提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。



注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

9.4.6. 为 GCP 管理用户定义的标记（label）和标签（tag）



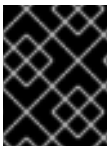
重要

对 GCP 的用户定义的标记和标签的支持只是一个技术预览功能。技术预览功能不受红帽产品服务等级协议（SLA）支持，且功能可能并不完整。红帽不推荐在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。

有关红帽技术预览功能支持范围的更多信息，请参阅技术预览功能支持范围。

Google Cloud Platform (GCP) 提供标记（label）和标签（tag），以帮助识别和组织为特定 OpenShift Container Platform 集群创建的资源，从而使它们更易于管理。

您只能在 OpenShift Container Platform 集群安装过程中为每个 GCP 资源定义标签和标记。



重要

升级到 OpenShift Container Platform 4.16 版本的 OpenShift Container Platform 集群不支持用户定义的标签和标记。

注意

您无法更新已添加的标签。另外，如果删除了配置的标签键或标签值，新的标签支持的资源创建会失败。

用户定义的标记(label)

用户定义的标记和 OpenShift Container Platform 特定的标记只适用于 OpenShift Container Platform 安装程序创建的资源及其核心组件，例如：

- GCP filestore CSI Driver Operator
- GCP PD CSI Driver Operator
- Image Registry Operator
- GCP 的机器 API 供应商

用户定义的标签附加到 OpenShift Container Platform 安装程序、Image Registry Operator 和 Machine API Operator 创建的资源。用户定义的标签不会附加到任何其他 Operator 或 Kubernetes in-tree 组件创建的资源中。

以下 GCP 资源提供了用户定义的标记和 OpenShift Container Platform 标记：

- Compute 磁盘
- Compute 实例
- Compute 镜像
- Compute 转发规则
- DNS 受管区
- FileStore 实例
- 存储桶

用户定义的标记的限制

- GCP beta 版本支持 **ComputeAddress** 的标记。OpenShift Container Platform 不向资源添加标记。

用户定义的标签（tag）

用户定义的标签附加到 OpenShift Container Platform Image Registry Operator 创建的资源中，而不是由任何其他 Operator 或 Kubernetes in-tree 组件创建的资源。

用户定义的标签在以下 GCP 资源中可用：

- 存储存储桶

- 计算实例
- Compute 磁盘

用户定义的标签的限制

- 标记不会附加到以下项目：
  - GCP filestore CSI 驱动程序 Operator 创建的 FileStore 实例资源
  - GCP PD CSI 驱动程序 Operator 创建的计算磁盘和计算镜像资源
- 标签不能仅限于特定的服务帐户，因为 Operator 会创建和使用最少的角色的服务帐户。
- OpenShift Container Platform 不创建标签的任何键和值资源。
- OpenShift Container Platform 特定的标签不会添加到任何资源。

其他资源

- 有关识别 **OrganizationID** 的更多信息，请参阅： OrganizationID
- 有关识别 **ProjectID** 的更多信息，请参阅： ProjectID
- 有关标记的更多信息，请参阅 标记概述。
- 有关标签的更多信息，请参阅 标签概述。

9.4.6.1. 为 GCP 配置用户定义的标记和标签

先决条件

- 安装程序要求服务帐户包含 **TagUser** 角色，以便安装程序可以在机构和项目级别创建带有定义标签的 OpenShift Container Platform 集群。

流程

- 更新 **install-config.yaml** 文件，以定义所需标记和标签列表。

注意

标记和标签在 **install-config.yaml** 创建阶段定义，且无法在集群创建后使用新标记和标签修改或更新。

install-config.yaml 文件示例

```
apiVersion: v1
featureSet: TechPreviewNoUpgrade
platform:
 gcp:
   userLabels: 1
   - key: <label_key> 2
     value: <label_value> 3
   userTags: 4
```

```
    - parentID: <OrganizationID/ProjectID> 5
      key: <tag_key_short_name>
      value: <tag_value_short_name>
```

1 将键和值作为标记添加到 GCP 上创建的资源。

2 定义标记名称。

3 定义标记内容。

4 将键和值作为标签添加到 GCP 上创建的资源。

5 在组织或项目级别上定义标签的层次结构资源的 ID。

以下是用户定义的标记的要求：

- 标记的键和值必须至少为 1 个字符，最多可有 63 个字符。
- 标记的键和值只能包含小写字母、数字字符、下划线(_)和短划线(-)。
- 标记的键必须以小写开头。
- 您可以为每个资源配置最多 32 个标记。每个资源最多可具有 64 个标记，并且为 OpenShift Container Platform 内部使用保留 32 个标记。

以下是用户定义的标签的要求：

- 标签键和标签值必须已经存在。OpenShift Container Platform 不创建键和值。
- 标签 **parentID** 可以是 **OrganizationID** 或 **ProjectID** ：
  - **OrganizationID** 必需是没有前导零的十进制数字。
  - **ProjectID** 的长度为 6 到 30 个字符，其中仅包含小写字母、数字和连字符。
  - **ProjectID** 必须以字母开头，且不能以连字符结尾。
- 标签键只能包含大写和小写字母数字字符、连字符(-)、下划线(_)和句点(.)。
- 标签值必须仅包含大写和小写字母字符、连字符 (-), 下划线 (_), 句点 (.), at 符号 (@), 百分比符号 (%), 等号 (=), 加号 (+), colon (:),逗号 (,), 星号 (*), pound sign ($), ampersand (&), parentheses (()), square braces ([]), curly braces ({}), 和空格。
- 标签键和值必须以字母数字字符开头和结尾。
- 标签值必须是键的预定义值之一。
- 您可以配置最多 50 个标签。
- 不应存在使用与任何已存在的标签键的值（从父资源中继承）相同的定义的标签键。

9.4.6.2. 为 GCP 查询用户定义的标记和标签

创建 OpenShift Container Platform 集群后，您可以访问在 **infrastructures.config.openshift.io/cluster** 对象中的 GCP 资源定义的标记和标签列表，如以下示例 **infrastructure.yaml** 文件所示。

infrastructure.yaml 文件示例

```
apiVersion: config.openshift.io/v1
kind: Infrastructure
metadata:
 name: cluster
spec:
 platformSpec:
   type: GCP
status:
 infrastructureName: <cluster_id> (1)
 platform: GCP
 platformStatus:
   gcp:
     resourceLabels:
     - key: <label_key>
       value: <label_value>
     resourceTags:
     - key: <tag_key_short_name>
       parentID: <OrganizationID/ProjectID>
       value: <tag_value_short_name>
   type: GCP
```

(1) 集群安装过程中生成的集群 ID。

除了用户定义的标记外，资源还具有 OpenShift Container Platform 定义的标记。OpenShift Container Platform 标记的格式是 **kubernetes-io-cluster-<cluster_id>:owned**。

9.4.7. 安装 OpenShift CLI

您可以安装 OpenShift CLI(**oc**)来使用命令行界面与 OpenShift Container Platform 进行交互。您可以在 Linux、Windows 或 macOS 上安装 **oc**。

重要

如果安装了旧版本的 **oc**，则可能无法使用 OpenShift Container Platform 4.16 中的所有命令。下载并安装新版本的 **oc**。

在 Linux 上安装 OpenShift CLI
您可以按照以下流程在 Linux 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **产品变体** 下拉列表中选择架构。
3. 从 **版本** 下拉列表中选择适当的版本。
4. 点 **OpenShift v4.16 Linux Client** 条目旁的 **Download Now** 来保存文件。
5. 解包存档：

```
$ tar xvf <file>
```

6. 将 **oc** 二进制文件放到 **PATH 中的目录中**。
要查看您的 **PATH**，请执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

在 Windows 上安装 OpenShift CLI
您可以按照以下流程在 Windows 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 Windows Client** 条目旁的 **Download Now** 来保存文件。
4. 使用 ZIP 程序解压存档。
5. 将 **oc** 二进制文件移到 **PATH 中的目录中**。
要查看您的 **PATH**，请打开命令提示并执行以下命令：

```
C:\> path
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

在 macOS 上安装 OpenShift CLI
您可以按照以下流程在 macOS 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 macOS Client** 条目旁的 **Download Now** 来保存文件。

注意

对于 macOS arm64，请选择 **OpenShift v4.16 macOS arm64 Client** 条目。

4. 解包和解压存档。

5. 将 **oc** 二进制文件移到 PATH 的目录中。
要查看您的 **PATH**，请打开终端并执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

9.4.8. 在 kube-system 项目中存储管理员级别的 secret 的替代方案

默认情况下，管理员 secret 存储在 **kube-system** 项目中。如果您在 **install-config.yaml** 文件中将 **credentialsMode** 参数配置为 **Manual**，则必须使用以下替代方案之一：

- 要手动管理长期云凭证，请按照手动创建长期凭证中的步骤操作。
- 要实现在集群外为各个组件管理的短期凭证，请按照配置 GCP 集群以使用短期凭证中的步骤操作。

9.4.8.1. 手动创建长期凭证

在无法访问云身份和访问管理(IAM)API 的环境中，或者管理员更不希望将管理员级别的凭证 secret 存储在集群 **kube-system** 命名空间中时，可以在安装前将 Cloud Credential Operator(CCO)放入手动模式。

流程

1. 在安装程序使用的 GCP 帐户中添加以下粒度权限：

例 9.25. 所需的 GCP 权限

- compute.machineTypes.list
- compute.regions.list
- compute.zones.list
- dns.changes.create
- dns.changes.get
- dns.managedZones.create
- dns.managedZones.delete
- dns.managedZones.get
- dns.managedZones.list
- dns.networks.bindPrivateDNSZone
- dns.resourceRecordSets.create

- dns.resourceRecordSets.delete
- dns.resourceRecordSets.list

2. 如果您没有将 **install-config.yaml** 配置文件中的 **credentialsMode** 参数设置为 **Manual**，请修改值，如下所示：

配置文件片段示例

```
apiVersion: v1
baseDomain: example.com
credentialsMode: Manual
# ...
```

3. 如果您之前还没有创建安装清单文件，请运行以下命令：

```
$ openshift-install create manifests --dir <installation_directory>
```

其中 **<installation_directory>** 是安装程序在其中创建文件的目录。

4. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

5. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 自定义资源 (CR) 列表：

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
  --included \ 1
  --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
  --to=<path_to_directory_for_credentials_requests> 3
```

1. **--included** 参数仅包含特定集群配置所需的清单。
2. 指定 **install-config.yaml** 文件的位置。
3. 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

此命令为每个 **CredentialsRequest** 对象创建一个 YAML 文件。

CredentialsRequest 对象示例

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component_credentials_request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
```

```
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: GCPProviderSpec
    predefinedRoles:
    - roles/storage.admin
    - roles/iam.serviceAccountUser
    skipServiceCheck: true
  ...
```

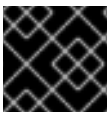
6. 在之前生成的 **openshift-install** 清单目录中为 secret 创建 YAML 文件。secret 必须使用在 **spec.secretRef** 中为每个 **CredentialsRequest** 定义的命名空间和 secret 名称存储。

带有 secret 的 CredentialsRequest 对象示例

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component_credentials_request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
      ...
  secretRef:
    name: <component_secret>
    namespace: <component_namespace>
  ...
```

Secret 对象示例

```
apiVersion: v1
kind: Secret
metadata:
  name: <component_secret>
  namespace: <component_namespace>
data:
  service_account.json: <base64_encoded_gcp_service_account_file>
```



重要

在升级使用手动维护凭证的集群前，您必须确保 CCO 处于可升级状态。

9.4.8.2. 将 GCP 集群配置为使用短期凭证

要安装配置为使用 GCP Workload Identity 的集群，您必须配置 CCO 实用程序并为集群创建所需的 GCP 资源。

9.4.8.2.1. 配置 Cloud Credential Operator 工具

当 Cloud Credential Operator(CCO)以手动模式运行时，要从集群外部创建和管理云凭证，提取并准备 CCO 实用程序(**ccoctl**)二进制文件。

注意

ccoctl 工具是在 Linux 环境中运行的 Linux 二进制文件。

先决条件

- 您可以访问具有集群管理员权限的 OpenShift Container Platform 帐户。
- 已安装 OpenShift CLI(**oc**)。
- 您已在安装程序使用的 GCP 帐户中添加了以下身份验证选项之一：
  - **IAM Workload Identity Pool Admin**角色。
  - 以下粒度权限：

    例 9.26. 所需的 GCP 权限

    - compute.projects.get
    - iam.googleapis.com/workloadIdentityPoolProviders.create
    - iam.googleapis.com/workloadIdentityPoolProviders.get
    - iam.googleapis.com/workloadIdentityPools.create
    - iam.googleapis.com/workloadIdentityPools.delete
    - iam.googleapis.com/workloadIdentityPools.get
    - iam.googleapis.com/workloadIdentityPools.undelete
    - iam.roles.create
    - iam.roles.delete
    - iam.roles.list
    - iam.roles.undelete
    - iam.roles.update
    - iam.serviceAccounts.create
    - iam.serviceAccounts.delete
    - iam.serviceAccounts.getIamPolicy
    - iam.serviceAccounts.list
    - iam.serviceAccounts.setIamPolicy
    - iam.workloadIdentityPoolProviders.get
    - iam.workloadIdentityPools.delete
    - resourcemanager.projects.get
    - resourcemanager.projects.getIamPolicy

- resourcemanager.projects.setIamPolicy
- storage.buckets.create
- storage.buckets.delete
- storage.buckets.get
- storage.buckets.getIamPolicy
- storage.buckets.setIamPolicy
- storage.objects.create
- storage.objects.delete
- storage.objects.list

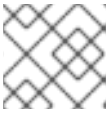
流程

1. 运行以下命令，为 OpenShift Container Platform 发行镜像设置变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. 运行以下命令，从 OpenShift Container Platform 发行镜像获取 CCO 容器镜像：

```
$ CCO_IMAGE=$(oc adm release info --image-for='cloud-credential-operator'
$RELEASE_IMAGE -a ~/.pull-secret)
```



注意

确保 **$RELEASE_IMAGE** 的架构与将使用 **ccoctl** 工具的环境架构相匹配。

3. 运行以下命令，将 CCO 容器镜像中的 **ccoctl** 二进制文件提取到 OpenShift Container Platform 发行镜像中：

```
$ oc image extract $CCO_IMAGE \
  --file="/usr/bin/ccoctl.<rhel_version>" \ 1
  -a ~/.pull-secret
```

1 对于 **<rhel_version>**，请指定与主机使用的 Red Hat Enterprise Linux (RHEL) 版本对应的值。如果没有指定值，则默认使用 **ccoctl.rhel8**。以下值有效：

- **rhel8**: 为使用 RHEL 8 的主机指定这个值。
- **rhel9** ：为使用 RHEL 9 的主机指定这个值。

4. 运行以下命令更改权限以使 **ccoctl** 可执行：

```
$ chmod 775 ccoctl.<rhel_version>
```

验证

- 要验证 **ccoctl** 是否可使用，请运行以下命令来显示帮助文件：

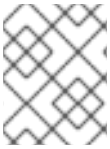
```
$ ccoctl --help
```

ccoctl --help 的输出

```
OpenShift credentials provisioning tool

Usage:
  ccoctl [command]

Available Commands:
  aws         Manage credentials objects for AWS cloud
  azure       Manage credentials objects for Azure
  gcp         Manage credentials objects for Google cloud
  help        Help about any command
  ibmcloud    Manage credentials objects for IBM Cloud
  nutanix     Manage credentials objects for Nutanix

Flags:
  -h, --help   help for ccoctl

Use "ccoctl [command] --help" for more information about a command.
```

9.4.8.2.2. 使用 Cloud Credential Operator 实用程序创建 GCP 资源

您可以使用 **ccoctl gcp create-all** 命令自动创建 GCP 资源。



注意

默认情况下，**ccoctl** 在运行命令的目录中创建对象。要在其他目录中创建对象，请使用 **--output-dir** 标志。此流程使用 **<path_to_ccoctl_output_dir>** 来引用这个目录。

先决条件

您必须：

- 提取并准备好 **ccoctl** 二进制文件。

流程

1. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 对象列表：

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
```

```
    --included \ 1
    --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
    --to=<path_to_directory_for_credentials_requests> 3
```

1. **--included** 参数仅包含特定集群配置所需的清单。
2. 指定 **install-config.yaml** 文件的位置。
3. 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

注意

此命令可能需要一些时间才能运行。

3. 运行以下命令，使用 **ccoctl** 工具处理所有 **CredentialsRequest** 对象：

```
$ ccoctl gcp create-all \
  --name=<name> \ 1
  --region=<gcp_region> \ 2
  --project=<gcp_project_id> \ 3
  --credentials-requests-dir=<path_to_credentials_requests_directory> 4
```

1. 为用于跟踪的所有创建 GCP 资源指定用户定义的名称。
2. 指定在其中创建云资源的 GCP 区域。
3. 指定在其中创建云资源的 GCP 项目 ID。
4. 指定包含 **CredentialsRequest** 清单文件的目录，以创建 GCP 服务帐户。

注意

如果您的集群使用 **TechPreviewNoUpgrade** 功能集启用的技术预览功能，则必须包含 **--enable-tech-preview** 参数。

验证

- 要验证 OpenShift Container Platform secret 是否已创建，列出 **<path_to_ccoctl_output_dir>/manifests** 目录中的文件：

```
$ ls <path_to_ccoctl_output_dir>/manifests
```

输出示例

```
cluster-authentication-02-config.yaml
openshift-cloud-controller-manager-gcp-ccm-cloud-credentials-credentials.yaml
openshift-cloud-credential-operator-cloud-credential-operator-gcp-ro-creds-credentials.yaml
openshift-cloud-network-config-controller-cloud-credentials-credentials.yaml
openshift-cluster-api-capg-manager-bootstrap-credentials-credentials.yaml
openshift-cluster-csi-drivers-gcp-pd-cloud-credentials-credentials.yaml
```

```
openshift-image-registry-installer-cloud-credentials-credentials.yaml
openshift-ingress-operator-cloud-credentials-credentials.yaml
openshift-machine-api-gcp-cloud-credentials-credentials.yaml
```

您可以通过查询 GCP 来验证是否已创建 IAM 服务帐户。如需更多信息，请参阅有关列出 IAM 服务帐户的 GCP 文档。

9.4.8.2.3. 整合 Cloud Credential Operator 实用程序清单

要为单个组件在集群外实现短期安全凭证，您必须将创建 Cloud Credential Operator 实用程序 (**ccoctl**) 的清单文件移到安装程序的正确目录中。

先决条件

- 您已使用托管集群的云平台配置了帐户。
- 您已配置了 Cloud Credential Operator 实用程序 (**ccoctl**)。
- 已使用 **ccoctl** 工具创建了集群所需的云供应商资源。

流程

1. 在安装程序使用的 GCP 帐户中添加以下粒度权限：

   例 9.27. 所需的 GCP 权限

   - compute.machineTypes.list
   - compute.regions.list
   - compute.zones.list
   - dns.changes.create
   - dns.changes.get
   - dns.managedZones.create
   - dns.managedZones.delete
   - dns.managedZones.get
   - dns.managedZones.list
   - dns.networks.bindPrivateDNSZone
   - dns.resourceRecordSets.create
   - dns.resourceRecordSets.delete
   - dns.resourceRecordSets.list

2. 如果您没有将 **install-config.yaml** 配置文件中的 **credentialsMode** 参数设置为 **Manual**，请修改值，如下所示：

   配置文件片段示例

```
apiVersion: v1
baseDomain: example.com
credentialsMode: Manual
# ...
```

3. 如果您之前还没有创建安装清单文件，请运行以下命令：

```
$ openshift-install create manifests --dir <installation_directory>
```

其中 **<installation_directory>** 是安装程序在其中创建文件的目录。

4. 运行以下命令，将 **ccoctl** 工具生成的清单复制到安装程序创建的 **manifests** 目录中：

```
$ cp /<path_to_ccoctl_output_dir>/manifests/* ./manifests/
```

5. 运行以下命令，将 **ccoctl** 工具在 **tls** 目录中生成的私钥复制到安装目录中：

```
$ cp -a /<path_to_ccoctl_output_dir>/tls .
```

9.4.9. 使用 GCP Marketplace 产品

使用 GCP Marketplace 产品可让您部署 OpenShift Container Platform 集群，该集群按照使用付费（按小时、每个内核）进行计费，同时仍由红帽直接支持。

默认情况下，安装程序会下载并安装用于部署计算机器的 Red Hat Enterprise Linux CoreOS (RHCOS) 镜像。要使用 GCP Marketplace 中的 RHCOS 镜像部署 OpenShift Container Platform 集群，修改 **install-config.yaml** 文件以引用 GCP Marketplace 提供的位置来覆盖默认行为。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。

流程

1. 编辑 **compute.platform.gcp.osImage** 参数，以指定 GCP Marketplace 镜像的位置：
   - 将 **project** 参数设置为 **redhat-marketplace-public**
   - 将 **name** 参数设置为以下提供之一：

     OpenShift Container Platform
     redhat-coreos-ocp-413-x86-64-202305021736
     OpenShift Platform Plus
     redhat-coreos-opp-413-x86-64-202305021736
     OpenShift Kubernetes Engine
     redhat-coreos-oke-413-x86-64-202305021736
2. 保存文件并在部署集群时引用。

为计算机器指定 GCP Marketplace 镜像的 install-config.yaml 文件示例

```
apiVersion: v1
```

```
baseDomain: example.com
controlPlane:
# ...
compute:
  platform:
    gcp:
      osImage:
        project: redhat-marketplace-public
        name: redhat-coreos-ocp-413-x86-64-202305021736
# ...
```

9.4.10. 部署集群

您可以在兼容云平台上安装 OpenShift Container Platform。

重要

在初始安装过程中，您只能运行安装程序的 **create cluster** 命令一次。

先决条件

- 您已使用托管集群的云平台配置了帐户。
- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。
- 已确认主机上的云供应商帐户具有部署集群的正确权限。权限不正确的帐户会导致安装过程失败，并显示包括缺失权限的错误消息。

流程

1. 删除所有不使用您为集群配置的 GCP 帐户的服务帐户密钥的现有 GCP 凭证，这些凭证存储在以下位置：
   - **GOOGLE_CREDENTIALS**、**GOOGLE_CLOUD_KEYFILE_JSON** 或 **GCLOUD_KEYFILE_JSON** 环境变量
   - The ~/**.gcp/osServiceAccount.json** file
   - **gcloud cli** 默认凭证
2. 进入包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1. 对于 **<installation_directory>**，请指定自定义 **./install-config.yaml** 文件的位置。
2. 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不是 **info**。

3. 可选：您可以减少用来安装集群的服务帐户的权限数量。
   - 如果将 **Owner** 角色分配给您的服务帐户，您可以删除该角色并将其替换为 **Viewer** 角色。
   - 如果包含 **Service Account Key Admin** 角色，您可以将其删除。

验证

当集群部署成功完成时：

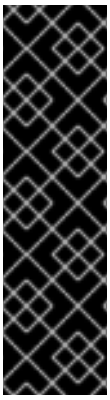
- 终端会显示用于访问集群的说明，包括指向 Web 控制台和 **kubeadmin** 用户的凭证的链接。
- 凭证信息还会输出到 **<installation_directory>/.openshift_install.log**.

重要

不要删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

9.4.11. 使用 CLI 登录集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

其他资源

- 如需有关 访问和了解 OpenShift Container Platform Web 控制台的更多详情，请参阅 访问 Web 控制台。

9.4.12. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于 远程健康监控

9.4.13. 后续步骤

- 自定义集群。
- 如果需要，您可以选择 不使用远程健康报告。

9.5. 使用自定义网络在 GCP 上安装集群

在 OpenShift Container Platform 版本 4.16 中，您可以使用自定义的网络配置在 Google Cloud Platform(GCP)上的由安装程序置备的基础架构上安装集群。通过自定义网络配置，您的集群可以与环境中现有的 IP 地址分配共存，并与现有的 MTU 和 VXLAN 配置集成。要自定义安装，请在安装集群前修改 **install-config.yaml** 文件中的参数。

您必须在安装过程中设置大多数网络配置参数，且您只能在正在运行的集群中修改 **kubeProxy** 配置参数。

9.5.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读有关 选择集群安装方法的文档，并为用户准备它。
- 已将 GCP 项目配置为 托管集群。

- 如果使用防火墙，将其配置为允许集群需要访问的站点。

9.5.2. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

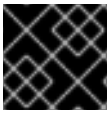
如果您的集群无法直接访问互联网，则可以在置备的某些类型的基础架构上执行受限网络安装。在此过程中，您可以下载所需的内容，并使用它为镜像 registry 填充安装软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群前，您要更新镜像 registry 的内容。

9.5.3. 为集群节点 SSH 访问生成密钥对

在 OpenShift Container Platform 安装过程中，您可以为安装程序提供 SSH 公钥。密钥通过它们的 Ignition 配置文件传递给 Red Hat Enterprise Linux CoreOS(RHCOS)节点，用于验证对节点的 SSH 访问。密钥添加到每个节点上 **core** 用户的 **~/.ssh/authorized_keys** 列表中，这将启用免密码身份验证。

将密钥传递给节点后，您可以使用密钥对作为用户 **核心** 通过 SSH 连接到 RHCOS 节点。若要通过 SSH 访问节点，必须由 SSH 为您的本地用户管理私钥身份。

如果要通过 SSH 连接到集群节点来执行安装调试或灾难恢复，则必须在安装过程中提供 SSH 公钥。**./openshift-install gather** 命令还需要在集群节点上设置 SSH 公钥。



重要

不要在生产环境中跳过这个过程，在生产环境中需要灾难恢复和调试。

注意

您必须使用本地密钥，而不是使用特定平台方法配置 的密钥，如 AWS 密钥对。

流程

1. 如果您在本地计算机上没有可用于在集群节点上进行身份验证的现有 SSH 密钥对，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_ed25519**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

注意

如果您计划在 **x86_64**、**ppc64le** 和 **s390x** 架构上安装使用 RHEL 加密库（这些加密库已提交给 NIST 用于 FIPS 140-2/140-3 验证）的 OpenShift Container Platform 集群，则不要创建使用 **ed25519** 算法的密钥。相反，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

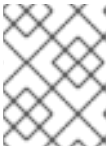
2. 查看公共 SSH 密钥：

```
$ cat <path>/<file_name>.pub
```

例如，运行以下命令来查看 **~/.ssh/id_ed25519.pub** 公钥：

```
$ cat ~/.ssh/id_ed25519.pub
```

3. 将 SSH 私钥身份添加到本地用户的 SSH 代理（如果尚未添加）。在集群节点上，或者要使用 **./openshift-install gather** 命令，需要对该密钥进行 SSH 代理管理，才能在集群节点上进行免密码 SSH 身份验证。



注意

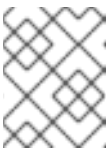
在某些发行版中，自动管理默认 SSH 私钥身份，如 **~/.ssh/id_rsa** 和 **~/.ssh/id_dsa**。

a. 如果 **ssh-agent** 进程尚未为您的本地用户运行，请将其作为后台任务启动：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```



注意

如果集群处于 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

4. 将 SSH 私钥添加到 **ssh-agent** ：

```
$ ssh-add <path>/<file_name> (1)
```

(1) 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_ed25519.pub**

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

后续步骤

- 安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

9.5.4. 获取安装程序

在安装 OpenShift Container Platform 前，将安装文件下载到您用于安装的主机上。

先决条件

- 您有一台运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB。

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐户，请使用您的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入到安装类型的页面，下载与您的主机操作系统和架构对应的安装程序，并将该文件放在您要存储安装配置文件的目录中。

重要

安装程序会在用来安装集群的计算机上创建几个文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。

重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，请为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。此 pull secret 允许您与所含授权机构提供的服务进行身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

9.5.5. 创建安装配置文件

您可以自定义在 Google Cloud Platform(GCP)上安装的 OpenShift Container Platform 集群。

先决条件

- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。

流程

1. 创建 **install-config.yaml** 文件。
   a. 进入包含安装程序的目录并运行以下命令：

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定要存储安装程序创建的文件的目录名称。

在指定目录时：

- 验证该目录是否具有**执行**权限。在安装目录中运行 Terraform 二进制文件需要这个权限。
- 使用空目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

b. 在提示符处，提供云的配置详情：

i. 可选：选择用于访问集群机器的 SSH 密钥。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

ii. 选择 **gcp** 作为目标平台。

iii. 如果您没有为计算机上的 GCP 帐户配置服务帐户密钥，则必须从 GCP 获取它，并粘贴文件的内容或输入文件的绝对路径。

iv. 选择要在其中置备集群的项目 ID。默认值由您配置的服务帐户指定。

v. 选择要将集群部署到的区域。

vi. 选择集群要部署到的基域。基域与您为集群创建的公共 DNS 区对应。

vii. 为集群输入描述性名称。

2. 修改 **install-config.yaml** 文件。您可以在"安装配置参数"部分找到有关可用参数的更多信息。

3. 备份 **install-config.yaml** 文件，以便您可以使用它安装多个集群。

重要

install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用此文件，必须现在备份。

其他资源

- GCP 的安装配置参数

9.5.5.1. 集群安装的最低资源要求

每台集群机器都必须满足以下最低要求：

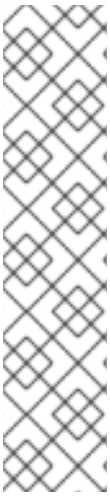
表 9.6. 最低资源要求

机器	操作系统	vCPU [1]	虚拟内存	Storage	每秒输入/输出 (IOPS) [2]
bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane（控制平面）	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS、RHEL 8.6 及更新版本 [3]	2	8 GB	100 GB	300

1. 当未启用并发多线程(SMT)或超线程时，一个 vCPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比例：（每个内核数的线程）× sockets = vCPU。

2. OpenShift Container Platform 和 Kubernetes 对磁盘性能非常敏感，建议使用更快的存储速度，特别是 control plane 节点上需要 10 ms p99 fsync 持续时间的 etcd。请注意，在许多云平台上，存储大小和 IOPS 可一起扩展，因此您可能需要过度分配存储卷来获取足够的性能。

3. 与所有用户置备的安装一样，如果您选择在集群中使用 RHEL 计算机器，则负责所有操作系统生命周期管理和维护，包括执行系统更新、应用补丁和完成所有其他必要的任务。RHEL 7 计算机器的使用已弃用，并已在 OpenShift Container Platform 4.10 及更新的版本中删除。



注意

从 OpenShift Container Platform 版本 4.13 开始，RHCOS 基于 RHEL 版本 9.2，它更新了微架构要求。以下列表包含每个架构需要的最小指令集架构 (ISA)：

- x86-64 体系结构需要 x86-64-v2 ISA
- ARM64 架构需要 ARMv8.0-A ISA
- IBM Power 架构需要 Power 9 ISA
- s390x 架构需要 z14 ISA

如需更多信息，请参阅 RHEL 架构。

如果平台的实例类型满足集群机器的最低要求，则 OpenShift Container Platform 支持使用它。

其他资源

- 优化存储

9.5.5.2. 为 GCP 测试的实例类型

以下 Google Cloud Platform 实例类型已使用 OpenShift Container Platform 测试。

例 9.28. 机器系列

- **A2**

- **A3**
- **C2**
- **C2D**
- **C3**
- **C3D**
- **E2**
- **M1**
- **N1**
- **N2**
- **N2D**
- **Tau T2D**

9.5.5.3. 在 64 位 ARM 基础架构上为 GCP 测试的实例类型

以下 Google Cloud Platform (GCP) 64 位 ARM 实例类型已使用 OpenShift Container Platform 测试。

例 9.29. 64 位 ARM 机器的机器系列

- **Tau T2A**

9.5.5.4. 使用自定义机器类型

支持使用自定义机器类型来安装 OpenShift Container Platform 集群。

使用自定义机器类型时请考虑以下几点：

- 与预定义的实例类型类似，自定义机器类型必须满足 control plane 和计算机器的最低资源要求。如需更多信息，请参阅"集群安装的资源要求"。
- 自定义机器类型的名称必须遵循以下语法：
  custom-<number_of_cpus>-<amount_of_memory_in_mb>

  例如，**custom-6-20480**。

作为安装过程的一部分，您可以在 **install-config.yaml** 文件中指定自定义机器类型。

带有自定义机器类型的 install-config.yaml 文件示例

```
compute:
- architecture: amd64
  hyperthreading: Enabled
  name: worker
  platform:
    gcp:
```

```
      type: custom-6-20480
  replicas: 2
controlPlane:
  architecture: amd64
  hyperthreading: Enabled
  name: master
  platform:
    gcp:
      type: custom-6-20480
  replicas: 3
```

9.5.5.5. 启用屏蔽虚拟机

您可在安装集群时使用 Shielded 虚拟机。Shielded 虚拟机具有额外的安全功能，包括安全引导、固件和完整性监控和 rootkit 检测。如需更多信息，请参阅 Google 文档中有关 Shielded 虚拟机 的文档。

注意

目前，在具有 64 位 ARM 基础架构的集群中不支持 Shielded 虚拟机。

先决条件

- 您已创建了 **install-config.yaml** 文件。

流程

- 在部署集群前，使用文本编辑器编辑 **install-config.yaml** 文件并添加以下部分之一：

  a. 仅将屏蔽的虚拟机用于 control plane 机器：

  ```
  controlPlane:
    platform:
      gcp:
        secureBoot: Enabled
  ```

  b. 仅将屏蔽的虚拟机用于计算机器：

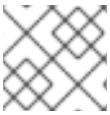
  ```
  compute:
  - platform:
      gcp:
        secureBoot: Enabled
  ```

  c. 将屏蔽的虚拟机用于所有机器：

  ```
  platform:
    gcp:
      defaultMachinePlatform:
        secureBoot: Enabled
  ```

9.5.5.6. 启用机密虚拟机

您可在安装集群时使用机密虚拟机。机密虚拟机在处理数据时加密数据。如需更多信息，请参阅 Google 文档中有关 机密计算的内容。您可以同时启用机密虚拟机和 Shielded 虚拟机，虽然它们不相互依赖。



注意

64 位 ARM 架构目前不支持机密虚拟机。

先决条件

- 您已创建了 **install-config.yaml** 文件。

流程

- 在部署集群前，使用文本编辑器编辑 **install-config.yaml** 文件并添加以下部分之一：

  a. 仅将机密虚拟机用于 control plane 机器：

```
controlPlane:
  platform:
    gcp:
      confidentialCompute: Enabled 1
      type: n2d-standard-8 2
      onHostMaintenance: Terminate 3
```

  1 启用机密虚拟机。

  2 指定支持机密虚拟机的机器类型。机密虚拟机需要 N2D 或 C2D 系列机器类型。有关支持的机器类型的更多信息，请参阅支持的操作系统和机器类型。

  3 指定主机维护事件期间虚拟机的行为，如硬件或软件更新。对于使用机密虚拟机的机器，此值必须设置为 **Terminate**，这会停止虚拟机。机密虚拟机不支持实时迁移。

  b. 仅将机密虚拟机用于计算机器：

```
compute:
- platform:
    gcp:
      confidentialCompute: Enabled
      type: n2d-standard-8
      onHostMaintenance: Terminate
```

  c. 将机密虚拟机用于所有机器：

```
platform:
  gcp:
    defaultMachinePlatform:
      confidentialCompute: Enabled
      type: n2d-standard-8
      onHostMaintenance: Terminate
```

9.5.5.7. GCP 的自定义 install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多详情，或修改所需参数的值。

重要

此示例 YAML 文件仅供参考。您必须使用安装程序来获取 **install-config.yaml** 文件，并进行修改。

```
apiVersion: v1
baseDomain: example.com (1)
credentialsMode: Mint (2)
controlPlane: (3) (4)
  hyperthreading: Enabled (5)
  name: master
  platform:
    gcp:
      type: n2-standard-4
      zones:
      - us-central1-a
      - us-central1-c
      osDisk:
        diskType: pd-ssd
        diskSizeGB: 1024
        encryptionKey: (6)
          kmsKey:
            name: worker-key
            keyRing: test-machine-keys
            location: global
            projectID: project-id
      tags: (7)
      - control-plane-tag1
      - control-plane-tag2
      osImage: (8)
        project: example-project-name
        name: example-image-name
  replicas: 3
compute: (9) (10)
- hyperthreading: Enabled (11)
  name: worker
  platform:
    gcp:
      type: n2-standard-4
      zones:
      - us-central1-a
      - us-central1-c
      osDisk:
        diskType: pd-standard
        diskSizeGB: 128
        encryptionKey: (12)
          kmsKey:
            name: worker-key
            keyRing: test-machine-keys
            location: global
            projectID: project-id
        tags: (13)
        - compute-tag1
        - compute-tag2
```

```
        osImage: 14
          project: example-project-name
          name: example-image-name
  replicas: 3
metadata:
  name: test-cluster 15
networking: 16
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 17
  serviceNetwork:
  - 172.30.0.0/16
platform:
  gcp:
    projectID: openshift-production 18
    region: us-central1 19
    defaultMachinePlatform:
      tags: 20
      - global-tag1
      - global-tag2
      osImage: 21
        project: example-project-name
        name: example-image-name
pullSecret: '{"auths": ...}' 22
fips: false 23
sshKey: ssh-ed25519 AAAA... 24
```

1 15 18 19 22 必需。安装程序会提示您输入这个值。

2 可选：添加此参数来强制 Cloud Credential Operator (CCO) 使用指定的模式。默认情况下，CCO 使用 **kube-system** 命名空间中的 root 凭证来动态尝试决定凭证的功能。有关 CCO 模式的详情，请参阅*身份验证和授权*指南中的"About the Cloud Credential Operator"部分。

3 9 16 如果没有提供这些参数和值，安装程序会提供默认值。

4 10 **controlPlane** 部分是一个单个映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，controlPlane **部分** 的第一行则不以连字符开头。仅使用一个 control plane 池。

5 11 是否要启用或禁用并发多线程或 **超线程**。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将 参数值设置为 **Disabled** 来禁用它。如果在某些集群机器中禁用并发多线程，则必须在所有集群机器中禁用它。

重要

如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。如果您禁用并发多线程，请为您的机器使用较大的类型，如 **n1-standard-8**。

6 12 可选：自定义加密密钥部分来加密虚拟机和持久性卷。您的默认计算服务帐户必须具有相应的权限才能使用您的 KMS 密钥并分配了正确的 IAM 角色。默认服务帐户名称遵循 **service-<project_number>@compute-system.iam.gserviceaccount.com** 模式。有关为您的服务帐户授

予正确权限的更多信息，请参阅 "Machine management" → "Creating compute machine set" → "Creating a compute machine set on GCP"。

7 13 20 可选：要应用到 control plane 或计算机器集的一组网络标签。**platform.gcp.defaultMachinePlatform.tags** 参数将应用到 control plane 和计算机器。如果设置了 **compute.platform.gcp.tags** 或 **controlPlane.platform.gcp.tags** 参数，它们会覆盖 **platform.gcp.defaultMachinePlatform.tags** 参数。

8 14 21 可选：应该用来引导 control plane 和计算机器的自定义 Red Hat Enterprise Linux CoreOS (RHCOS)。**platform.gcp.defaultMachinePlatform.osImage** 下的 **project** 和 **name** 参数应用到 control plane 和计算机器。如果设置了 **controlPlane.platform.gcp.osImage** 或 **compute.platform.gcp.osImage** 下的 **project** 和 **name** 参数，它们会覆盖 **platform.gcp.defaultMachinePlatform.osImage** 参数。

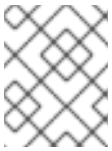
17 要安装的集群网络插件。默认值 **OVNKubernetes** 是唯一支持的值。

23 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS)时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。

24 您可选择提供用于访问集群中机器的 **sshKey** 值。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

其他资源

- 为计算机器设置启用客户管理的加密密钥

9.5.5.8. 在安装过程中配置集群范围的代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 **Proxy** 对象的 **spec.noProxy** 字段中添加站点来绕过代理。

注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，**Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 **.** 以仅匹配子域。例如，**.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代理。
4. 如果提供，安装程序会在 **openshift-config** 命名空间中生成名为 **user-ca-bundle** 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 **trusted-ca-bundle** 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， **Proxy** 对象的 **trustedCA** 字段中也会引用此配置映射。**additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。
5. 可选：决定 **Proxy** 对象的配置以引用 **trustedCA** 字段中 **user-ca-bundle** 配置映射的策略。允许的值是 **Proxyonly** 和 **Always**。仅在配置了 **http/https** 代理时，使用 **Proxyonly** 引用 **user-ca-bundle** 配置映射。使用 **Always** 始终引用 **user-ca-bundle** 配置映射。默认值为 **Proxyonly**。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。

注意

如果安装程序超时，重启并使用安装程序的 **wait-for** 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. 保存该文件并在安装 OpenShift Container Platform 时引用。

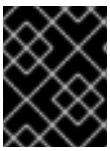
安装程序会创建一个名为 cluster 的集群范围代理，该代理 **使用** 提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

9.5.6. 安装 OpenShift CLI

您可以安装 OpenShift CLI(**oc**)来使用命令行界面与 OpenShift Container Platform 进行交互。您可以在 Linux、Windows 或 macOS 上安装 **oc**。



重要

如果安装了旧版本的 **oc**，则可能无法使用 OpenShift Container Platform 4.16 中的所有命令。下载并安装新版本的 **oc**。

在 Linux 上安装 OpenShift CLI
您可以按照以下流程在 Linux 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **产品变体** 下拉列表中选择架构。
3. 从 **版本** 下拉列表中选择适当的版本。
4. 点 **OpenShift v4.16 Linux Client** 条目旁的 **Download Now** 来保存文件。
5. 解包存档：

```
$ tar xvf <file>
```

6. 将 **oc** 二进制文件放到 **PATH** 中的目录中。
要查看您的 **PATH**，请执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 Windows Client** 条目旁的 **Download Now** 来保存文件。
4. 使用 ZIP 程序解压存档。
5. 将 **oc** 二进制文件移到 **PATH 中的目录中**。
   要查看您的 **PATH**，请打开命令提示并执行以下命令：

```
C:\> path
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 macOS Client** 条目旁的 **Download Now** 来保存文件。

注意

对于 macOS arm64，请选择 **OpenShift v4.16 macOS arm64 Client** 条目。

4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
   要查看您的 **PATH**，请打开终端并执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

9.5.7. 在 kube-system 项目中存储管理员级别的 secret 的替代方案

默认情况下，管理员 secret 存储在 **kube-system** 项目中。如果您在 **install-config.yaml** 文件中将 **credentialsMode** 参数配置为 **Manual**，则必须使用以下替代方案之一：

- 要手动管理长期云凭证，请按照手动创建长期凭证中的步骤操作。
- 要实现在集群外为各个组件管理的短期凭证，请按照配置 GCP 集群以使用短期凭证中的步骤操作。

9.5.7.1. 手动创建长期凭证

在无法访问云身份和访问管理(IAM)API 的环境中，或者管理员更不希望将管理员级别的凭证 secret 存储在集群 **kube-system** 命名空间中时，可以在安装前将 Cloud Credential Operator(CCO)放入手动模式。

流程

1. 在安装程序使用的 GCP 帐户中添加以下粒度权限：

   例 9.30. 所需的 GCP 权限

   - compute.machineTypes.list
   - compute.regions.list
   - compute.zones.list
   - dns.changes.create
   - dns.changes.get
   - dns.managedZones.create
   - dns.managedZones.delete
   - dns.managedZones.get
   - dns.managedZones.list
   - dns.networks.bindPrivateDNSZone
   - dns.resourceRecordSets.create
   - dns.resourceRecordSets.delete
   - dns.resourceRecordSets.list

2. 如果您没有将 **install-config.yaml** 配置文件中的 **credentialsMode** 参数设置为 **Manual**，请修改值，如下所示：

   配置文件片段示例

   ```
   apiVersion: v1
   baseDomain: example.com
   credentialsMode: Manual
   # ...
   ```

3. 如果您之前还没有创建安装清单文件，请运行以下命令：

```
$ openshift-install create manifests --dir <installation_directory>
```

其中 **<installation_directory>** 是安装程序在其中创建文件的目录。

4. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

5. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 自定义资源 (CR) 列表：

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
  --included \ 1
  --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
  --to=<path_to_directory_for_credentials_requests> 3
```

1. **--included** 参数仅包含特定集群配置所需的清单。
2. 指定 **install-config.yaml** 文件的位置。
3. 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

此命令为每个 **CredentialsRequest** 对象创建一个 YAML 文件。

CredentialsRequest 对象示例

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component_credentials_request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: GCPProviderSpec
    predefinedRoles:
    - roles/storage.admin
    - roles/iam.serviceAccountUser
    skipServiceCheck: true
  ...
```

6. 在之前生成的 **openshift-install** 清单目录中为 secret 创建 YAML 文件。secret 必须使用在 **spec.secretRef** 中为每个 **CredentialsRequest** 定义的命名空间和 secret 名称存储。

带有 secret 的 CredentialsRequest 对象示例

```
apiVersion: cloudcredential.openshift.io/v1
```

```
kind: CredentialsRequest
metadata:
  name: <component_credentials_request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
      ...
  secretRef:
    name: <component_secret>
    namespace: <component_namespace>
  ...
```

Secret 对象示例

```
apiVersion: v1
kind: Secret
metadata:
  name: <component_secret>
  namespace: <component_namespace>
data:
  service_account.json: <base64_encoded_gcp_service_account_file>
```

重要

在升级使用手动维护凭证的集群前，您必须确保 CCO 处于可升级状态。

9.5.7.2. 将 GCP 集群配置为使用短期凭证

要安装配置为使用 GCP Workload Identity 的集群，您必须配置 CCO 实用程序并为集群创建所需的 GCP 资源。

9.5.7.2.1. 配置 Cloud Credential Operator 工具

当 Cloud Credential Operator(CCO)以手动模式运行时，要从集群外部创建和管理云凭证，提取并准备 CCO 实用程序(**ccoctl**)二进制文件。

注意

ccoctl 工具是在 Linux 环境中运行的 Linux 二进制文件。

先决条件

- 您可以访问具有集群管理员权限的 OpenShift Container Platform 帐户。
- 已安装 OpenShift CLI(**oc**)。
- 您已在安装程序使用的 GCP 帐户中添加了以下身份验证选项之一：
  - **IAM Workload Identity Pool Admin**角色。
  - 以下粒度权限：

    例 9.31. 所需的 GCP 权限

- compute.projects.get
- iam.googleapis.com/workloadIdentityPoolProviders.create
- iam.googleapis.com/workloadIdentityPoolProviders.get
- iam.googleapis.com/workloadIdentityPools.create
- iam.googleapis.com/workloadIdentityPools.delete
- iam.googleapis.com/workloadIdentityPools.get
- iam.googleapis.com/workloadIdentityPools.undelete
- iam.roles.create
- iam.roles.delete
- iam.roles.list
- iam.roles.undelete
- iam.roles.update
- iam.serviceAccounts.create
- iam.serviceAccounts.delete
- iam.serviceAccounts.getIamPolicy
- iam.serviceAccounts.list
- iam.serviceAccounts.setIamPolicy
- iam.workloadIdentityPoolProviders.get
- iam.workloadIdentityPools.delete
- resourcemanager.projects.get
- resourcemanager.projects.getIamPolicy
- resourcemanager.projects.setIamPolicy
- storage.buckets.create
- storage.buckets.delete
- storage.buckets.get
- storage.buckets.getIamPolicy
- storage.buckets.setIamPolicy
- storage.objects.create
- storage.objects.delete

```
- storage.objects.list
```

流程

1. 运行以下命令，为 OpenShift Container Platform 发行镜像设置变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. 运行以下命令，从 OpenShift Container Platform 发行镜像获取 CCO 容器镜像：

```
$ CCO_IMAGE=$(oc adm release info --image-for='cloud-credential-operator'
$RELEASE_IMAGE -a ~/.pull-secret)
```

注意

确保 **$RELEASE_IMAGE** 的架构与将使用 **ccoctl** 工具的环境架构相匹配。

3. 运行以下命令，将 CCO 容器镜像中的 **ccoctl** 二进制文件提取到 OpenShift Container Platform 发行镜像中：

```
$ oc image extract $CCO_IMAGE \
  --file="/usr/bin/ccoctl.<rhel_version>" \ 1
  -a ~/.pull-secret
```

1 对于 **<rhel_version>**，请指定与主机使用的 Red Hat Enterprise Linux (RHEL) 版本对应的值。如果没有指定值，则默认使用 **ccoctl.rhel8**。以下值有效：

- **rhel8**: 为使用 RHEL 8 的主机指定这个值。
- **rhel9** ：为使用 RHEL 9 的主机指定这个值。

4. 运行以下命令更改权限以使 **ccoctl** 可执行：

```
$ chmod 775 ccoctl.<rhel_version>
```

验证

- 要验证 **ccoctl** 是否可使用，请运行以下命令来显示帮助文件：

```
$ ccoctl --help
```

ccoctl --help 的输出

```
OpenShift credentials provisioning tool

Usage:
  ccoctl [command]

Available Commands:
  aws         Manage credentials objects for AWS cloud
```

```
  azure       Manage credentials objects for Azure
  gcp         Manage credentials objects for Google cloud
  help        Help about any command
  ibmcloud    Manage credentials objects for IBM Cloud
  nutanix     Manage credentials objects for Nutanix

Flags:
  -h, --help   help for ccoctl

Use "ccoctl [command] --help" for more information about a command.
```

9.5.7.2.2. 使用 Cloud Credential Operator 实用程序创建 GCP 资源

您可以使用 **ccoctl gcp create-all** 命令自动创建 GCP 资源。

注意

默认情况下，**ccoctl** 在运行命令的目录中创建对象。要在其他目录中创建对象，请使用 **--output-dir** 标志。此流程使用 **<path_to_ccoctl_output_dir>** 来引用这个目录。

先决条件

您必须：

- 提取并准备好 **ccoctl** 二进制文件。

流程

1. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 对象列表：

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
  --included \ 1
  --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
  --to=<path_to_directory_for_credentials_requests> 3
```

1. **--included** 参数仅包含特定集群配置所需的清单。
2. 指定 **install-config.yaml** 文件的位置。
3. 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

注意

此命令可能需要一些时间才能运行。

3. 运行以下命令，使用 **ccoctl** 工具处理所有 **CredentialsRequest** 对象：

```
$ ccoctl gcp create-all \
  --name=<name> \1
  --region=<gcp_region> \2
  --project=<gcp_project_id> \3
  --credentials-requests-dir=<path_to_credentials_requests_directory> 4
```

1. 为用于跟踪的所有创建 GCP 资源指定用户定义的名称。
2. 指定在其中创建云资源的 GCP 区域。
3. 指定在其中创建云资源的 GCP 项目 ID。
4. 指定包含 **CredentialsRequest** 清单文件的目录，以创建 GCP 服务帐户。

注意

如果您的集群使用 **TechPreviewNoUpgrade** 功能集启用的技术预览功能，则必须包含 **--enable-tech-preview** 参数。

验证

- 要验证 OpenShift Container Platform secret 是否已创建，列出 **<path_to_ccoctl_output_dir>/manifests** 目录中的文件：

```
$ ls <path_to_ccoctl_output_dir>/manifests
```

输出示例

```
cluster-authentication-02-config.yaml
openshift-cloud-controller-manager-gcp-ccm-cloud-credentials-credentials.yaml
openshift-cloud-credential-operator-cloud-credential-operator-gcp-ro-creds-credentials.yaml
openshift-cloud-network-config-controller-cloud-credentials-credentials.yaml
openshift-cluster-api-capg-manager-bootstrap-credentials-credentials.yaml
openshift-cluster-csi-drivers-gcp-pd-cloud-credentials-credentials.yaml
openshift-image-registry-installer-cloud-credentials-credentials.yaml
openshift-ingress-operator-cloud-credentials-credentials.yaml
openshift-machine-api-gcp-cloud-credentials-credentials.yaml
```

您可以通过查询 GCP 来验证是否已创建 IAM 服务帐户。如需更多信息，请参阅有关列出 IAM 服务帐户的 GCP 文档。

9.5.7.2.3. 整合 Cloud Credential Operator 实用程序清单

要为单个组件在集群外实现短期安全凭证，您必须将创建 Cloud Credential Operator 实用程序 (**ccoctl**) 的清单文件移到安装程序的正确目录中。

先决条件

- 您已使用托管集群的云平台配置了帐户。
- 您已配置了 Cloud Credential Operator 实用程序 (**ccoctl**)。

- 已使用 **ccoctl** 工具创建了集群所需的云供应商资源。

流程

1. 在安装程序使用的 GCP 帐户中添加以下粒度权限：

   例 9.32. 所需的 GCP 权限

   - compute.machineTypes.list
   - compute.regions.list
   - compute.zones.list
   - dns.changes.create
   - dns.changes.get
   - dns.managedZones.create
   - dns.managedZones.delete
   - dns.managedZones.get
   - dns.managedZones.list
   - dns.networks.bindPrivateDNSZone
   - dns.resourceRecordSets.create
   - dns.resourceRecordSets.delete
   - dns.resourceRecordSets.list

2. 如果您没有将 **install-config.yaml** 配置文件中的 **credentialsMode** 参数设置为 **Manual**，请修改值，如下所示：

   配置文件片段示例

   ```
   apiVersion: v1
   baseDomain: example.com
   credentialsMode: Manual
   # ...
   ```

3. 如果您之前还没有创建安装清单文件，请运行以下命令：

   ```
   $ openshift-install create manifests --dir <installation_directory>
   ```

   其中 **<installation_directory>** 是安装程序在其中创建文件的目录。

4. 运行以下命令，将 **ccoctl** 工具生成的清单复制到安装程序创建的 **manifests** 目录中：

   ```
   $ cp /<path_to_ccoctl_output_dir>/manifests/* ./manifests/
   ```

5. 运行以下命令，将 **ccoctl** 工具在 **tls** 目录中生成的私钥复制到安装目录中：

```
$ cp -a /<path_to_ccoctl_output_dir>/tls .
```

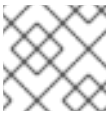
9.5.8. 网络配置阶段

OpenShift Container Platform 安装前有两个阶段，您可以在其中自定义网络配置。

第 1 阶段

在创建清单文件前，您可以自定义 **install-config.yaml** 文件中的以下与网络相关的字段：

- **networking.networkType**
- **networking.clusterNetwork**
- **networking.serviceNetwork**
- **networking.machineNetwork**
  有关这些字段的更多信息，请参阅 *安装配置参数*。



注意

将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

重要

CIDR 范围 **172.17.0.0/16** 由 libVirt 保留。对于集群中的任何网络，您无法使用此范围或与这个范围重叠的范围。

第 2 阶段

运行 **openshift-install create 清单创建** 清单文件后，您可以只使用您要修改的字段定义自定义 Cluster Network Operator 清单。您可以使用 清单指定高级网络配置。

您不能覆盖在 stage 2 阶段 1 中在 **install-config.yaml** 文件中指定的值。但是，您可以在第 2 阶段进一步自定义网络插件。

9.5.9. 指定高级网络配置

您可以使用网络插件的高级网络配置将集群集成到现有网络环境中。您只能在安装集群前指定高级网络配置。

重要

不支持通过修改安装程序创建的 OpenShift Container Platform 清单文件来自定义网络配置。支持应用您创建的清单文件，如以下流程中所示。

先决条件

- 您已创建 **install-config.yaml** 文件并完成对其所做的任何修改。

流程

1. 进入包含安装程序的目录并创建清单：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 **<installation_directory>** 指定包含集群的 **install-config.yaml** 文件的目录名称。

2. 在 **<installation_directory>/manifests/** 目录中 为**高级网络配置创建一个名为 cluster-network-03-config.yml** 的 stub 清单文件：

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
```

3. 在 **cluster-network-03-config.yml** 文件中指定集群的高级网络配置，如下例所示：

   为 OVN-Kubernetes 网络供应商启用 IPsec

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  defaultNetwork:
    ovnKubernetesConfig:
      ipsecConfig:
        mode: Full
```

4. 可选：备份 **manifests/cluster-network-03-config.yml** 文件。创建 Ignition 配置文件时，安装程序会使用 **manifests/** 目录。

9.5.10. Cluster Network Operator 配置

集群网络的配置作为 Cluster Network Operator(CNO)配置的一部分指定，并存储在名为 **cluster** 的自定义资源(CR)对象中。CR 指定 **operator.openshift.io** API 组中的 **Network** API 的字段。

CNO 配置在集群安装过程中从 **Network.config.openshift.io** API 组中的 **Network** API 继承以下字段：

clusterNetwork

从中分配 Pod IP 地址的 IP 地址池。

serviceNetwork

服务的 IP 地址池.

defaultNetwork.type

集群网络插件。**OVNKubernetes** 是安装期间唯一支持的插件。

您可以通过在名为 **cluster** 的 CNO 对象中设置 **defaultNetwork** 对象的字段来为集群指定集群网络插件配置。

9.5.10.1. Cluster Network Operator 配置对象

下表中描述了 Cluster Network Operator(CNO)的字段：

表 9.7. Cluster Network Operator 配置对象

字段	类型	描述
metadata.name	**字符串**	CNO 对象的名称。这个名称始终是 **集群**。
spec.clusterNetwork	**array**	用于指定从哪些 IP 地址块分配 Pod IP 地址以及集群中每个节点的子网前缀长度的列表。例如： spec: clusterNetwork: - cidr: 10.128.0.0/19 hostPrefix: 23 - cidr: 10.128.32.0/19 hostPrefix: 23
spec.serviceNetwork	**array**	服务的 IP 地址块。OpenShift SDN 和 OVN-Kubernetes 网络插件只支持服务网络的一个 IP 地址块。例如： spec: serviceNetwork: - 172.30.0.0/14 您只能在创建清单前在 **install-config.yaml** 文件中自定义此字段。该值在清单文件中是只读的。
spec.defaultNetwork	**object**	为集群网络配置网络插件。
spec.kubeProxyConfig	**object**	此对象的字段指定 kube-proxy 配置。如果使用 OVN-Kubernetes 集群网络供应商，则 kube-proxy 配置不会起作用。

defaultNetwork 对象配置
下表列出了 **defaultNetwork** 对象的值：

表 9.8. **defaultNetwork** 对象

字段	类型	描述
type	**字符串**	**OVNKubernetes**。Red Hat OpenShift Networking 网络插件在安装过程中被选择。此值在集群安装后无法更改。 **注意** OpenShift Container Platform 默认使用 OVN-Kubernetes 网络插件。OpenShift SDN 不再作为新集群的安装选择提供。
ovnKubernetesConfig	**object**	此对象仅对 OVN-Kubernetes 网络插件有效。

配置 OVN-Kubernetes 网络插件
下表描述了 OVN-Kubernetes 网络插件的配置字段：

表 9.9. **ovnKubernetesConfig** object

字段	类型	描述
mtu	**integer**	Geneve（通用网络虚拟化封装）覆盖网络的最大传输单元 (MTU)。这根据主网络接口的 MTU 自动探测。您通常不需要覆盖检测到的 MTU。 如果自动探测的值不是您期望的值，请确认节点上主网络接口上的 MTU 是否正确。您不能使用这个选项更改节点上主网络接口的 MTU 值。 如果集群中不同节点需要不同的 MTU 值，则必须将此值设置为 **比** 集群中的最低 MTU 值小 100。例如，如果集群中的某些节点的 MTU 为 **9001**，而某些节点的 MTU 为 **1500**，则必须将此值设置为 **1400**。
genevePort	**integer**	用于所有 Geneve 数据包的端口。默认值为 **6081**。此值在集群安装后无法更改。
ipsecConfig	**object**	指定用于自定义 IPsec 配置的配置对象。
ipv4	**object**	为 IPv4 设置指定配置对象。
ipv6	**object**	为 IPv6 设置指定配置对象。
policyAuditConfig	**object**	指定用于自定义网络策略审计日志的配置对象。如果未设置，则使用默认的审计日志设置。
gatewayConfig	**object**	可选：指定一个配置对象来自定义如何将出口流量发送到节点网关。  **注意** 在迁移出口流量时，工作负载和服务流量会受到一定影响，直到 Cluster Network Operator (CNO) 成功推出更改。

表 9.10. **ovnKubernetesConfig.ipv4** object

字段	类型	描述

字段	类型	描述
internalTransitSwitchSubnet	字符串	如果您的现有网络基础架构与 **100.88.0.0/16** IPv4 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。启用东西流量的分布式传输交换机的子网。此子网不能与 OVN-Kubernetes 或主机本身使用的任何其他子网重叠。必须足够大，以适应集群中的每个节点一个 IP 地址。 默认值为 **100.88.0.0/16**。
internalJoinSubnet	字符串	如果您的现有网络基础架构与 **100.64.0.0/16** IPv4 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。您必须确保 IP 地址范围没有与 OpenShift Container Platform 安装使用的任何其他子网重叠。IP 地址范围必须大于可添加到集群的最大节点数。例如，如果 **clusterNetwork.cidr** 值为 **10.128.0.0/14**，并且 **clusterNetwork.hostPrefix** 值为 **/23**，则最大节点数量为 **2^(23-14)=512**。 默认值为 **100.64.0.0/16**。

表 9.11. **ovnKubernetesConfig.ipv6** object

字段	类型	描述
internalTransitSwitchSubnet	字符串	如果您的现有网络基础架构与 **fd97::/64** IPv6 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。启用东西流量的分布式传输交换机的子网。此子网不能与 OVN-Kubernetes 或主机本身使用的任何其他子网重叠。必须足够大，以适应集群中的每个节点一个 IP 地址。 默认值为 **fd97::/64**。
internalJoinSubnet	字符串	如果您的现有网络基础架构与 **fd98::/64** IPv6 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。您必须确保 IP 地址范围没有与 OpenShift Container Platform 安装使用的任何其他子网重叠。IP 地址范围必须大于可添加到集群的最大节点数。 默认值为 **fd98::/64**。

表 9.12. **policyAuditConfig** object

字段	类型	描述
rateLimit	整数	每个节点每秒生成一次的消息数量上限。默认值为每秒 **20** 条消息。
maxFileSize	整数	审计日志的最大大小，以字节为单位。默认值为 **50000000** 或 50 MB。
maxLogFiles	整数	保留的日志文件的最大数量。

字段	类型	描述
目的地	字符串	以下附加审计日志目标之一： **libc** 主机上的 journald 进程的 libc **syslog（）** 函数。 **UDP:<host>:<port>** 一个 syslog 服务器。将 **<host>:<port>** 替换为 **syslog 服务器的主机** 和端口。 **Unix:<file>** 由 **<file>** 指定的 Unix 域套接字文件。 **null** 不要将审计日志发送到任何其他目标。
syslogFacility	字符串	syslog 工具，如 as **kern**，如 RFC5424 定义。默认值为 **local0。**

表 9.13. **gatewayConfig** object

字段	类型	描述
routingViaHost	**布尔值**	将此字段设置为 **true**，将来自 pod 的出口流量发送到主机网络堆栈。对于依赖于在内核路由表中手动配置路由的高级别安装和应用程序，您可能需要将出口流量路由到主机网络堆栈。默认情况下，出口流量在 OVN 中进行处理以退出集群，不受内核路由表中的特殊路由的影响。默认值为 **false**。 此字段与 Open vSwitch 硬件卸载功能有交互。如果将此字段设置为 **true**，则不会获得卸载的性能优势，因为主机网络堆栈会处理出口流量。
ipForwarding	**object**	您可以使用 **Network** 资源中的 **ipForwarding** 规格来控制 OVN-Kubernetes 管理接口上所有流量的 IP 转发。指定 **Restricted** 只允许 Kubernetes 相关流量的 IP 转发。指定 **Global** 以允许转发所有 IP 流量。对于新安装，默认值为 **Restricted**。对于到 OpenShift Container Platform 4.14 或更高版本的更新，默认值为 **Global**。
ipv4	**object**	可选：指定一个对象来为主机配置内部 OVN-Kubernetes 伪装地址，以服务 IPv4 地址的流量。
ipv6	**object**	可选：指定一个对象来为主机配置内部 OVN-Kubernetes 伪装地址，以服务 IPv6 地址的流量。

表 9.14. **gatewayConfig.ipv4** 对象

字段	类型	描述
internalMasqueradeSubnet	**字符串**	内部使用的伪装 IPv4 地址，以启用主机服务流量。主机配置了这些 IP 地址和共享网关网桥接口。默认值为 **169.254.169.0/29**。

表 9.15. **gatewayConfig.ipv6** 对象

字段	类型	描述
internalMasqueradeSubnet	**字符串**	内部使用的伪装 IPv6 地址，以启用主机服务流量。主机配置了这些 IP 地址和共享网关网桥接口。默认值为 **fd69::/125**。

表 9.16. **ipsecConfig** 对象

字段	类型	描述
模式	**字符串**	指定 IPsec 实现的行为。必须是以下值之一： • **Disabled**: 在集群节点上不启用 IPsec。 • **External** ：对于带有外部主机的网络流量，启用 IPsec。 • **Full**: IPsec 为带有外部主机的 pod 流量和网络流量启用 IPsec。

启用 IPSec 的 OVN-Kubernetes 配置示例

```
defaultNetwork:
  type: OVNKubernetes
  ovnKubernetesConfig:
    mtu: 1400
    genevePort: 6081
      ipsecConfig:
        mode: Full
```

重要

使用 OVNKubernetes 可能会导致 IBM Power® 上的堆栈耗尽问题。

kubeProxyConfig 对象配置（仅限 OpenShiftSDN 容器网络接口）
kubeProxyConfig 对象的值在下表中定义：

表 9.17. **kubeProxyConfig** object

字段	类型	描述

字段	类型	描述
iptablesSyncPeriod	**字符串**	**iptables** 规则的刷新周期。默认值为 **30s**。有效的后缀包括 **s**、**m** 和 **h**，具体参见 Go 时间 包 文档。  **注意** 由于 OpenShift Container Platform 4.3 及更高版本中引进了性能改进，不再需要调整 **iptablesSyncPeriod** 参数。
proxyArguments.iptables-min-sync-period	**array**	刷新 **iptables** 规则前的最短持续时间。此字段确保刷新的频率不会过于频繁。有效的后缀包括 **s**、**m** 和 **h**，具体参见 Go **time** 软件包。默认值为： kubeProxyConfig: proxyArguments: iptables-min-sync-period: - 0s

9.5.11. 部署集群

您可以在兼容云平台上安装 OpenShift Container Platform。

重要

在初始安装过程中，您只能运行安装程序的 **create cluster** 命令一次。

先决条件

- 您已使用托管集群的云平台配置了帐户。
- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。
- 已确认主机上的云供应商帐户具有部署集群的正确权限。权限不正确的帐户会导致安装过程失败，并显示包括缺失权限的错误消息。

流程

1. 删除所有不使用您为集群配置的 GCP 帐户的服务帐户密钥的现有 GCP 凭证，这些凭证存储在以下位置：
   - **GOOGLE_CREDENTIALS**、**GOOGLE_CLOUD_KEYFILE_JSON** 或 **GCLOUD_KEYFILE_JSON** 环境变量
   - The ~/**.gcp/osServiceAccount.json** file
   - **gcloud cli** 默认凭证
2. 进入包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定自定义 **./install-config.yaml** 文件的位置。

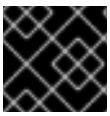
2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不是 **info**。

3. 可选：您可以减少用来安装集群的服务帐户的权限数量。
   - 如果将 **Owner** 角色分配给您的服务帐户，您可以删除该角色并将其替换为 **Viewer** 角色。
   - 如果包含 **Service Account Key Admin 角色**，您可以将其删除。

验证

当集群部署成功完成时：

- 终端会显示用于访问集群的说明，包括指向 Web 控制台和 **kubeadmin** 用户的凭证的链接。
- 凭证信息还会输出到 **<installation_directory>/.openshift_install.log**.

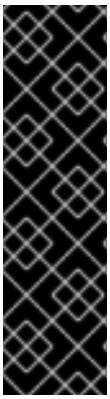


重要

不要删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

9.5.12. 使用 CLI 登录集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

其他资源

- 如需有关 访问和了解 OpenShift Container Platform Web 控制台的更多详情，请参阅 访问 Web 控制台。

9.5.13. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

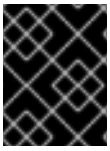
- 有关 Telemetry 服务的更多信息，请参阅关于 远程健康监控

9.5.14. 后续步骤

- 自定义集群。
- 如果需要，您可以选择 不使用远程健康报告。

9.6. 在受限网络中的 GCP 上安装集群

在 OpenShift Container Platform 4.16 中，您可以通过在现有 Google Virtual Private Cloud (VPC) 上创建安装发行内容的内部镜像在受限网络中的 Google Cloud Platform(GCP)上安装集群。

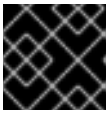


重要

您可以使用镜像安装发行内容安装 OpenShift Container Platform 集群，但集群将需要访问互联网才能使用 GCP API。

9.6.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读有关 选择集群安装方法的文档，并为用户准备它。
- 已将 GCP 项目配置为 托管集群。
- 您已 将断开连接的安装的镜像镜像 到 registry，并获取了 OpenShift Container Platform 版本的 **imageContentSources** 数据。

  重要

  由于安装介质位于镜像主机上，因此您可以使用该计算机完成所有安装步骤。

- 在 GCP 中有一个现有的 VPC。在使用安装程序置备的基础架构的受限网络中安装集群时，您无法使用安装程序置备的 VPC。您必须使用用户置备的 VPC 来满足以下要求之一：
  - 包含镜像 registry
  - 具有防火墙规则或对等连接，以访问在其他位置托管的镜像 registry
- 如果使用防火墙，则会 将其配置为允许集群需要访问的站点。虽然您可能需要授予更多站点的访问权限，但您必须授予对 ***.googleapis.com** 和 **accounts.google.com** 的访问权限。

9.6.2. 关于在受限网络中安装

在 OpenShift Container Platform 4.16 中，可以执行不需要有效的互联网连接来获取软件组件的安装。受限网络安装可以使用安装程序置备的基础架构或用户置备的基础架构完成，具体取决于您要安装集群的云平台。

如果您选择在云平台中执行受限网络安装，您仍需要访问其云 API。有些云功能，比如 Amazon Web Service 的 Route 53 DNS 和 IAM 服务，需要访问互联网。根据您的网络，在裸机硬件、Nutanix 或 VMware vSphere 上安装可能需要较少的互联网访问。

要完成受限网络安装，您必须创建一个 registry，以镜像 OpenShift 镜像 registry 的内容并包含安装介质。您可以在镜像主机上创建此 registry，该主机可同时访问互联网和您的封闭网络，也可以使用满足您的限制条件的其他方法。

9.6.2.1. 其他限制

受限网络中的集群有以下额外限制和限制：

- **ClusterVersion** 状态包含一个 **Unable to retrieve available updates** 错误。
- 默认情况下，您无法使用 Developer Catalog 的内容，因为您无法访问所需的镜像流标签。

9.6.3. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来获得用来安装集群的镜像。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

9.6.4. 为集群节点 SSH 访问生成密钥对

在 OpenShift Container Platform 安装过程中，您可以为安装程序提供 SSH 公钥。密钥通过它们的 Ignition 配置文件传递给 Red Hat Enterprise Linux CoreOS(RHCOS)节点，用于验证对节点的 SSH 访问。密钥添加到每个节点上 **core** 用户的 **~/.ssh/authorized_keys** 列表中，这将启用免密码身份验证。

将密钥传递给节点后，您可以使用密钥对作为用户 **核心** 通过 SSH 连接到 RHCOS 节点。若要通过 SSH 访问节点，必须由 SSH 为您的本地用户管理私钥身份。

如果要通过 SSH 连接到集群节点来执行安装调试或灾难恢复，则必须在安装过程中提供 SSH 公钥。**./openshift-install gather** 命令还需要在集群节点上设置 SSH 公钥。

重要

不要在生产环境中跳过这个过程，在生产环境中需要灾难恢复和调试。

注意

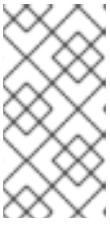
您必须使用本地密钥，而不是使用特定平台方法配置 的密钥，如 AWS 密钥对。

流程

1. 如果您在本地计算机上没有可用于在集群节点上进行身份验证的现有 SSH 密钥对，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_ed25519**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。



注意

如果您计划在 **x86_64**、**ppc64le** 和 **s390x** 架构上安装使用 RHEL 加密库（这些加密库已提交给 NIST 用于 FIPS 140-2/140-3 验证）的 OpenShift Container Platform 集群，则不要创建使用 **ed25519** 算法的密钥。相反，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

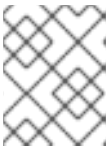
2. 查看公共 SSH 密钥：

```
$ cat <path>/<file_name>.pub
```

例如，运行以下命令来查看 **~/.ssh/id_ed25519.pub** 公钥：

```
$ cat ~/.ssh/id_ed25519.pub
```

3. 将 SSH 私钥身份添加到本地用户的 SSH 代理（如果尚未添加）。在集群节点上，或者要使用 **./openshift-install gather** 命令，需要对该密钥进行 SSH 代理管理，才能在集群节点上进行免密码 SSH 身份验证。



注意

在某些发行版中，自动管理默认 SSH 私钥身份，如 **~/.ssh/id_rsa** 和 **~/.ssh/id_dsa**。

a. 如果 **ssh-agent** 进程尚未为您的本地用户运行，请将其作为后台任务启动：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果集群处于 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

4. 将 SSH 私钥添加到 **ssh-agent** ：

```
$ ssh-add <path>/<file_name> 1
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_ed25519.pub**

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

后续步骤

- 安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

9.6.5. 创建安装配置文件

您可以自定义在 Google Cloud Platform(GCP)上安装的 OpenShift Container Platform 集群。

先决条件

- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。对于受限网络安装，这些文件位于您的镜像主机上。
- 您有创建镜像 registry 期间生成的 **imageContentSources** 值。
- 您已获取了镜像 registry 的证书内容。

流程

1. 创建 **install-config.yaml** 文件。

   a. 进入包含安装程序的目录并运行以下命令：

   ```
   $ ./openshift-install create install-config --dir <installation_directory> 1
   ```

   1 对于 **<installation_directory>**，请指定要存储安装程序创建的文件的目录名称。

   在指定目录时：

   - 验证该目录是否具有**执行**权限。在安装目录中运行 Terraform 二进制文件需要这个权限。
   - 使用空目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

   b. 在提示符处，提供云的配置详情：

      i. 可选：选择用于访问集群机器的 SSH 密钥。

      注意

      对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

      ii. 选择 **gcp** 作为目标平台。

      iii. 如果您没有为计算机上的 GCP 帐户配置服务帐户密钥，则必须从 GCP 获取它，并粘贴文件的内容或输入文件的绝对路径。

      iv. 选择要在其中置备集群的项目 ID。默认值由您配置的服务帐户指定。

      v. 选择要将集群部署到的区域。

      vi. 选择集群要部署到的基域。基域与您为集群创建的公共 DNS 区对应。

      vii. 为集群输入描述性名称。

2. 编辑 **install-config.yaml** 文件，以提供在受限网络中安装所需的额外信息。

   a. 更新 **pullSecret** 值，使其包含 registry 的身份验证信息：

   ```
   pullSecret: '{"auths":{"<mirror_host_name>:5000": {"auth": "<credentials>","email":
   "you@example.com"}}}'
   ```

   对于 **<mirror_host_name>**，请指定 您在镜像 registry 证书中指定的 registry 域名 ；对于 **<credentials>**， 请指定您的镜像 registry 的 base64 编码用户名和密码。

   b. 添加 **additionalTrustBundle** 参数和值。

   ```
   additionalTrustBundle: |
   ```

```
  -----BEGIN CERTIFICATE-----

ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
  -----END CERTIFICATE-----
```

该值必须是您用于镜像 registry 的证书文件内容。证书文件可以是现有的可信证书颁发机构，也可以是您为镜像 registry 生成的自签名证书。

c. 定义 VPC 的网络和子网，以便在父 **platform.gcp** 字段下安装集群：

```
network: <existing_vpc>
controlPlaneSubnet: <control_plane_subnet>
computeSubnet: <compute_subnet>
```

对于 **platform.gcp.network**，请指定现有 Google VPC 的名称。对于 **platform.gcp.controlPlaneSubnet** 和 **platform.gcp.computeSubnet**，请分别指定部署 control plane 机器和计算机器的现有子网。

d. 添加镜像内容资源，类似于以下 YAML 摘录：

```
imageContentSources:
- mirrors:
  - <mirror_host_name>:5000/<repo_name>/release
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <mirror_host_name>:5000/<repo_name>/release
  source: registry.redhat.io/ocp/release
```

对于这些值，请使用您在创建镜像 registry 时记录的 **imageContentSources**。

e. 可选：将发布策略设置为 **Internal**：

```
publish: Internal
```

通过设置这个选项，您可以创建一个内部 Ingress Controller 和一个私有负载均衡器。

3. 对您需要的 **install-config.yaml** 文件进行任何其他修改。
有关参数的更多信息，请参阅"安装配置参数"。

4. 备份 **install-config.yaml** 文件，以便您可以使用它安装多个集群。

重要

install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用此文件，必须现在备份。

其他资源

- GCP 的安装配置参数

9.6.5.1. 集群安装的最低资源要求

每台集群机器都必须满足以下最低要求：

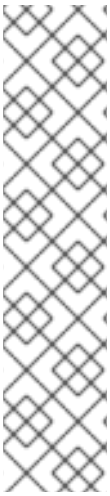
表 9.18. 最低资源要求

机器	操作系统	vCPU [1]	虚拟内存	Storage	每秒输入/输出 (IOPS) [2]
bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane（控制平面）	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS、RHEL 8.6 及更新版本 [3]	2	8 GB	100 GB	300

1. 当未启用并发多线程(SMT)或超线程时，一个 vCPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比例：（每个内核数的线程）× sockets = vCPU。

2. OpenShift Container Platform 和 Kubernetes 对磁盘性能非常敏感，建议使用更快的存储速度，特别是 control plane 节点上需要 10 ms p99 fsync 持续时间的 etcd。请注意，在许多云平台上，存储大小和 IOPS 可一起扩展，因此您可能需要过度分配存储卷来获取足够的性能。

3. 与所有用户置备的安装一样，如果您选择在集群中使用 RHEL 计算机器，则负责所有操作系统生命周期管理和维护，包括执行系统更新、应用补丁和完成所有其他必要的任务。RHEL 7 计算机器的使用已弃用，并已在 OpenShift Container Platform 4.10 及更新的版本中删除。



注意

从 OpenShift Container Platform 版本 4.13 开始，RHCOS 基于 RHEL 版本 9.2，它更新了微架构要求。以下列表包含每个架构需要的最小指令集架构 (ISA)：

- x86-64 体系结构需要 x86-64-v2 ISA
- ARM64 架构需要 ARMv8.0-A ISA
- IBM Power 架构需要 Power 9 ISA
- s390x 架构需要 z14 ISA

如需更多信息，请参阅 RHEL 架构。

如果平台的实例类型满足集群机器的最低要求，则 OpenShift Container Platform 支持使用它。

其他资源

- 优化存储

9.6.5.2. 为 GCP 测试的实例类型

以下 Google Cloud Platform 实例类型已使用 OpenShift Container Platform 测试。

> **例 9.33. 机器系列**
>
> - **A2**

- **A3**
- **C2**
- **C2D**
- **C3**
- **C3D**
- **E2**
- **M1**
- **N1**
- **N2**
- **N2D**
- **Tau T2D**

9.6.5.3. 在 64 位 ARM 基础架构上为 GCP 测试的实例类型

以下 Google Cloud Platform (GCP) 64 位 ARM 实例类型已使用 OpenShift Container Platform 测试。

例 9.34. 64 位 ARM 机器的机器系列

- **Tau T2A**

9.6.5.4. 使用自定义机器类型

支持使用自定义机器类型来安装 OpenShift Container Platform 集群。

使用自定义机器类型时请考虑以下几点：

- 与预定义的实例类型类似，自定义机器类型必须满足 control plane 和计算机器的最低资源要求。如需更多信息，请参阅"集群安装的资源要求"。
- 自定义机器类型的名称必须遵循以下语法：
  custom-<number_of_cpus>-<amount_of_memory_in_mb>

  例如，**custom-6-20480**。

作为安装过程的一部分，您可以在 **install-config.yaml** 文件中指定自定义机器类型。

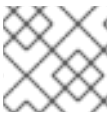
带有自定义机器类型的 install-config.yaml 文件示例

```
compute:
- architecture: amd64
  hyperthreading: Enabled
  name: worker
  platform:
    gcp:
```

```
      type: custom-6-20480
  replicas: 2
controlPlane:
  architecture: amd64
  hyperthreading: Enabled
  name: master
  platform:
    gcp:
      type: custom-6-20480
  replicas: 3
```

9.6.5.5. 启用屏蔽虚拟机

您可在安装集群时使用 Shielded 虚拟机。Shielded 虚拟机具有额外的安全功能，包括安全引导、固件和完整性监控和 rootkit 检测。如需更多信息，请参阅 Google 文档中有关 Shielded 虚拟机 的文档。



注意

目前，在具有 64 位 ARM 基础架构的集群中不支持 Shielded 虚拟机。

先决条件

- 您已创建了 **install-config.yaml** 文件。

流程

- 在部署集群前，使用文本编辑器编辑 **install-config.yaml** 文件并添加以下部分之一：

  a. 仅将屏蔽的虚拟机用于 control plane 机器：

  ```
  controlPlane:
    platform:
      gcp:
        secureBoot: Enabled
  ```

  b. 仅将屏蔽的虚拟机用于计算机器：

  ```
  compute:
  - platform:
      gcp:
        secureBoot: Enabled
  ```

  c. 将屏蔽的虚拟机用于所有机器：

  ```
  platform:
    gcp:
      defaultMachinePlatform:
        secureBoot: Enabled
  ```

9.6.5.6. 启用机密虚拟机

您可在安装集群时使用机密虚拟机。机密虚拟机在处理数据时加密数据。如需更多信息，请参阅 Google 文档中有关 机密计算的内容。您可以同时启用机密虚拟机和 Shielded 虚拟机，虽然它们不相互依赖。

注意

64 位 ARM 架构目前不支持机密虚拟机。

先决条件

- 您已创建了 **install-config.yaml** 文件。

流程

- 在部署集群前，使用文本编辑器编辑 **install-config.yaml** 文件并添加以下部分之一：

  a. 仅将机密虚拟机用于 control plane 机器：

```
controlPlane:
  platform:
    gcp:
      confidentialCompute: Enabled 1
      type: n2d-standard-8 2
      onHostMaintenance: Terminate 3
```

  1 启用机密虚拟机。

  2 指定支持机密虚拟机的机器类型。机密虚拟机需要 N2D 或 C2D 系列机器类型。有关支持的机器类型的更多信息，请参阅支持的操作系统和机器类型。

  3 指定主机维护事件期间虚拟机的行为，如硬件或软件更新。对于使用机密虚拟机的机器，此值必须设置为 **Terminate**，这会停止虚拟机。机密虚拟机不支持实时迁移。

  b. 仅将机密虚拟机用于计算机器：

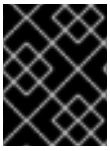
```
compute:
- platform:
    gcp:
      confidentialCompute: Enabled
      type: n2d-standard-8
      onHostMaintenance: Terminate
```

  c. 将机密虚拟机用于所有机器：

```
platform:
  gcp:
    defaultMachinePlatform:
      confidentialCompute: Enabled
      type: n2d-standard-8
      onHostMaintenance: Terminate
```

9.6.5.7. GCP 的自定义 install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多详情，或修改所需参数的值。



重要

此示例 YAML 文件仅供参考。您必须使用安装程序来获取 **install-config.yaml** 文件，并进行修改。

```
apiVersion: v1
baseDomain: example.com (1)
credentialsMode: Mint (2)
controlPlane: (3) (4)
  hyperthreading: Enabled (5)
  name: master
  platform:
    gcp:
      type: n2-standard-4
      zones:
      - us-central1-a
      - us-central1-c
      osDisk:
        diskType: pd-ssd
        diskSizeGB: 1024
        encryptionKey: (6)
          kmsKey:
            name: worker-key
            keyRing: test-machine-keys
            location: global
            projectID: project-id
      tags: (7)
      - control-plane-tag1
      - control-plane-tag2
      osImage: (8)
        project: example-project-name
        name: example-image-name
  replicas: 3
compute: (9) (10)
- hyperthreading: Enabled (11)
  name: worker
  platform:
    gcp:
      type: n2-standard-4
      zones:
      - us-central1-a
      - us-central1-c
      osDisk:
        diskType: pd-standard
        diskSizeGB: 128
        encryptionKey: (12)
          kmsKey:
            name: worker-key
            keyRing: test-machine-keys
            location: global
            projectID: project-id
        tags: (13)
        - compute-tag1
        - compute-tag2
```

```
        osImage: 14
          project: example-project-name
          name: example-image-name
  replicas: 3
metadata:
  name: test-cluster 15
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 16
  serviceNetwork:
  - 172.30.0.0/16
platform:
  gcp:
    projectID: openshift-production 17
    region: us-central1 18
    defaultMachinePlatform:
      tags: 19
      - global-tag1
      - global-tag2
      osImage: 20
        project: example-project-name
        name: example-image-name
    network: existing_vpc 21
    controlPlaneSubnet: control_plane_subnet 22
    computeSubnet: compute_subnet 23
pullSecret: '{"auths":{"<local_registry>": {"auth": "<credentials>","email": "you@example.com"}}}' 24
fips: false 25
sshKey: ssh-ed25519 AAAA... 26
additionalTrustBundle: | 27
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
imageContentSources: 28
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-v4.0-art-dev
```

1 15 17 18 必需。安装程序会提示您输入这个值。

2 可选：添加此参数来强制 Cloud Credential Operator (CCO) 使用指定的模式。默认情况下，CCO 使用 **kube-system** 命名空间中的 root 凭证来动态尝试决定凭证的功能。有关 CCO 模式的详情，请参阅 *身份验证和授权* 指南中的"About the Cloud Credential Operator"部分。

3 9 如果没有提供这些参数和值，安装程序会提供默认值。

4 10 **controlPlane** 部分是一个单个映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要

5 11 是否要启用或禁用并发多线程或 **超线程**。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将 参数值设置为 **Disabled** 来禁用它。如果在某些集群机器中禁用并发多线程，则必须在

重要

如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。如果您禁用并发多线程，请为您的机器使用较大的类型，如 **n1-standard-8**。

6 12 可选：自定义加密密钥部分来加密虚拟机和持久性卷。您的默认计算服务帐户必须具有相应的权限才能使用您的 KMS 密钥并分配了正确的 IAM 角色。默认服务帐户名称遵循 **service-<project_number>@compute-system.iam.gserviceaccount.com** 模式。有关为您的服务帐户授予正确权限的更多信息，请参阅 "Machine management" → "Creating compute machine set" → "Creating a compute machine set on GCP"。

7 13 19 可选：要应用到 control plane 或计算机器集的一组网络标签。**platform.gcp.defaultMachinePlatform.tags** 参数将应用到 control plane 和计算机器。如果设置了 **compute.platform.gcp.tags** 或 **controlPlane.platform.gcp.tags** 参数，它们会覆盖 **platform.gcp.defaultMachinePlatform.tags** 参数。

8 14 20 可选：应该用来引导 control plane 和计算机器的自定义 Red Hat Enterprise Linux CoreOS (RHCOS)。**platform.gcp.defaultMachinePlatform.osImage** 下的 **project** 和 **name** 参数应用到 control plane 和计算机器。如果设置了 **controlPlane.platform.gcp.osImage** 或 **compute.platform.gcp.osImage** 下的 **project** 和 **name** 参数，它们会覆盖 **platform.gcp.defaultMachinePlatform.osImage** 参数。

16 要安装的集群网络插件。默认值 **OVNKubernetes** 是唯一支持的值。

21 指定现有 VPC 的名称。

22 指定要将 control plane 机器部署到的现有子网的名称。该子网必须属于您指定的 VPC。

23 指定要将计算机器部署到的现有子网的名称。该子网必须属于您指定的 VPC。

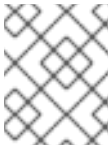
24 对于 **<local_registry>**，请指定 registry 域名，以及您的镜像 registry 用来提供内容的可选端口。例如： **registry.example.com 或 registry.example.com:5000**。对于 **<credentials>**，请为您的镜像 registry 指定 base64 编码的用户名和密码。

25 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS)时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。

26 您可以选择提供您用来访问集群中机器的 **sshKey** 值。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

27 提供用于镜像 registry 的证书文件内容。

28 提供命令输出中的 **imageContentSources 部分来** 镜像存储库。

9.6.5.8. 在 GCP 上创建具有全局访问权限的 Ingress Controller

您可以创建一个对 Google Cloud Platform(GCP)集群全局访问权限的 Ingress Controller。只有使用内部负载均衡器的 Ingress Controller 才可使用全局访问。

先决条件

- 已创建 **install-config.yaml**，并完成对其所做的任何修改。

流程

在新 GCP 集群上创建具有全局访问权限的 Ingress Controller。

1. 进入包含安装程序的目录并创建清单文件：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定包含集群的 **install-config.yaml** 文件的目录名称。

2. 在 **<installation_directory>/manifests/** 目录中创建一个名为 **cluster-ingress-default-ingresscontroller.yaml** 的文件：

```
$ touch <installation_directory>/manifests/cluster-ingress-default-ingresscontroller.yaml 1
```

1 对于 **<installation_directory>**，请指定包含集群的 **manifests/** 目录的目录名称。

创建该文件后，几个网络配置文件位于 **manifests/** 目录中，如下所示：

```
$ ls <installation_directory>/manifests/cluster-ingress-default-ingresscontroller.yaml
```

输出示例

```
cluster-ingress-default-ingresscontroller.yaml
```

3. 在编辑器中打开 **cluster-ingress-default-ingresscontroller.yaml** 文件，并输入描述您想要的 Operator 配置的自定义资源(CR)：

到 全局的 clientAccess 配置示例

```
apiVersion: operator.openshift.io/v1
kind: IngressController
metadata:
  name: default
  namespace: openshift-ingress-operator
spec:
  endpointPublishingStrategy:
    loadBalancer:
```

```
    providerParameters:
      gcp:
        clientAccess: Global 1
      type: GCP
    scope: Internal 2
  type: LoadBalancerService
```

1 将 **gcp.clientAccess** 设置为 **Global**。

2 只有使用内部负载均衡器的 Ingress Controller 才可使用全局访问。

9.6.5.9. 在安装过程中配置集群范围的代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 **Proxy** 对象的 **spec.noProxy** 字段中添加站点来绕过代理。

注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，**Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

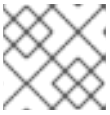
1 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 **http**。

2 用于创建集群外 HTTPS 连接的代理 URL。

3 要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 **.** 以仅匹配子域。例如，**.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代

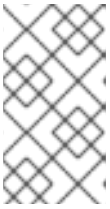
4 如果提供，安装程序会在 **openshift-config** 命名空间中生成名为 **user-ca-bundle** 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 **trusted-ca-bundle** 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， **Proxy** 对象的 **trustedCA** 字段中也会引用此配置映射。**additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

5 可选：决定 **Proxy** 对象的配置以引用 **trustedCA** 字段中 **user-ca-bundle** 配置映射的策略。允许的值是 **Proxyonly** 和 **Always**。仅在配置了 **http/https** 代理时，使用 **Proxyonly** 引用 **user-ca-bundle** 配置映射。使用 **Always** 始终引用 **user-ca-bundle** 配置映射。默认值为 **Proxyonly**。



注意

安装程序不支持代理的 **readinessEndpoints** 字段。



注意

如果安装程序超时，重启并使用安装程序的 **wait-for** 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. 保存该文件并在安装 OpenShift Container Platform 时引用。

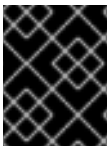
安装程序会创建一个名为 cluster 的集群范围代理，该代理 **使用** 提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

9.6.6. 安装 OpenShift CLI

您可以安装 OpenShift CLI(**oc**)来使用命令行界面与 OpenShift Container Platform 进行交互。您可以在 Linux、Windows 或 macOS 上安装 **oc**。



重要

如果安装了旧版本的 **oc**，则可能无法使用 OpenShift Container Platform 4.16 中的所有命令。下载并安装新版本的 **oc**。

在 Linux 上安装 OpenShift CLI
您可以按照以下流程在 Linux 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **产品变体** 下拉列表中选择架构。

3. 从 **版本** 下拉列表中选择适当的版本。
4. 点 **OpenShift v4.16 Linux Client** 条目旁的 **Download Now** 来保存文件。
5. 解包存档：

```
$ tar xvf <file>
```

6. 将 **oc** 二进制文件放到 **PATH 中的目录中**。
要查看您的 **PATH**，请执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

在 Windows 上安装 OpenShift CLI
您可以按照以下流程在 Windows 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 Windows Client** 条目旁的 **Download Now** 来保存文件。
4. 使用 ZIP 程序解压存档。
5. 将 **oc** 二进制文件移到 **PATH 中的目录中**。
要查看您的 **PATH**，请打开命令提示并执行以下命令：

```
C:\> path
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

在 macOS 上安装 OpenShift CLI
您可以按照以下流程在 macOS 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 macOS Client** 条目旁的 **Download Now** 来保存文件。

注意

对于 macOS arm64，请选择 **OpenShift v4.16 macOS arm64 Client** 条目。

4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
   要查看您的 **PATH**，请打开终端并执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

9.6.7. 在 kube-system 项目中存储管理员级别的 secret 的替代方案

默认情况下，管理员 secret 存储在 **kube-system** 项目中。如果您在 **install-config.yaml** 文件中将 **credentialsMode** 参数配置为 **Manual**，则必须使用以下替代方案之一：

- 要手动管理长期云凭证，请按照手动创建长期凭证中的步骤操作。
- 要实现在集群外为各个组件管理的短期凭证，请按照配置 GCP 集群以使用短期凭证中的步骤操作。

9.6.7.1. 手动创建长期凭证

在无法访问云身份和访问管理(IAM)API 的环境中，或者管理员更不希望将管理员级别的凭证 secret 存储在集群 **kube-system** 命名空间中时，可以在安装前将 Cloud Credential Operator(CCO)放入手动模式。

流程

1. 在安装程序使用的 GCP 帐户中添加以下粒度权限：

例 9.35. 所需的 GCP 权限

- compute.machineTypes.list
- compute.regions.list
- compute.zones.list
- dns.changes.create
- dns.changes.get
- dns.managedZones.create
- dns.managedZones.delete
- dns.managedZones.get
- dns.managedZones.list

- dns.networks.bindPrivateDNSZone
- dns.resourceRecordSets.create
- dns.resourceRecordSets.delete
- dns.resourceRecordSets.list

2. 如果您没有将 **install-config.yaml** 配置文件中的 **credentialsMode** 参数设置为 **Manual**，请修改值，如下所示：

配置文件片段示例

```
apiVersion: v1
baseDomain: example.com
credentialsMode: Manual
# ...
```

3. 如果您之前还没有创建安装清单文件，请运行以下命令：

```
$ openshift-install create manifests --dir <installation_directory>
```

其中 **<installation_directory>** 是安装程序在其中创建文件的目录。

4. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

5. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 自定义资源 (CR) 列表：

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
  --included \ 1
  --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
  --to=<path_to_directory_for_credentials_requests> 3
```

1. **--included** 参数仅包含特定集群配置所需的清单。
2. 指定 **install-config.yaml** 文件的位置。
3. 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

此命令为每个 **CredentialsRequest** 对象创建一个 YAML 文件。

CredentialsRequest 对象示例

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
```

```
  name: <component_credentials_request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: GCPProviderSpec
    predefinedRoles:
    - roles/storage.admin
    - roles/iam.serviceAccountUser
    skipServiceCheck: true
  ...
```

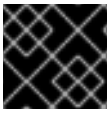
6. 在之前生成的 **openshift-install** 清单目录中为 secret 创建 YAML 文件。secret 必须使用在 **spec.secretRef** 中为每个 **CredentialsRequest** 定义的命名空间和 secret 名称存储。

带有 secret 的 CredentialsRequest 对象示例

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component_credentials_request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
      ...
  secretRef:
    name: <component_secret>
    namespace: <component_namespace>
  ...
```

Secret 对象示例

```
apiVersion: v1
kind: Secret
metadata:
  name: <component_secret>
  namespace: <component_namespace>
data:
  service_account.json: <base64_encoded_gcp_service_account_file>
```



重要

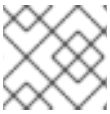
在升级使用手动维护凭证的集群前，您必须确保 CCO 处于可升级状态。

9.6.7.2. 将 GCP 集群配置为使用短期凭证

要安装配置为使用 GCP Workload Identity 的集群，您必须配置 CCO 实用程序并为集群创建所需的 GCP 资源。

9.6.7.2.1. 配置 Cloud Credential Operator 工具

当 Cloud Credential Operator(CCO)以手动模式运行时，要从集群外部创建和管理云凭证，提取并准备 CCO 实用程序(**ccoctl**)二进制文件。



注意

ccoctl 工具是在 Linux 环境中运行的 Linux 二进制文件。

先决条件

- 您可以访问具有集群管理员权限的 OpenShift Container Platform 帐户。
- 已安装 OpenShift CLI(**oc**)。
- 您已在安装程序使用的 GCP 帐户中添加了以下身份验证选项之一：
  - **IAM Workload Identity Pool Admin**角色。
  - 以下粒度权限：

例 9.36. 所需的 GCP 权限

- compute.projects.get
- iam.googleapis.com/workloadIdentityPoolProviders.create
- iam.googleapis.com/workloadIdentityPoolProviders.get
- iam.googleapis.com/workloadIdentityPools.create
- iam.googleapis.com/workloadIdentityPools.delete
- iam.googleapis.com/workloadIdentityPools.get
- iam.googleapis.com/workloadIdentityPools.undelete
- iam.roles.create
- iam.roles.delete
- iam.roles.list
- iam.roles.undelete
- iam.roles.update
- iam.serviceAccounts.create
- iam.serviceAccounts.delete
- iam.serviceAccounts.getIamPolicy
- iam.serviceAccounts.list
- iam.serviceAccounts.setIamPolicy
- iam.workloadIdentityPoolProviders.get
- iam.workloadIdentityPools.delete

- resourcemanager.projects.get
- resourcemanager.projects.getIamPolicy
- resourcemanager.projects.setIamPolicy
- storage.buckets.create
- storage.buckets.delete
- storage.buckets.get
- storage.buckets.getIamPolicy
- storage.buckets.setIamPolicy
- storage.objects.create
- storage.objects.delete
- storage.objects.list

流程

1. 运行以下命令，为 OpenShift Container Platform 发行镜像设置变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. 运行以下命令，从 OpenShift Container Platform 发行镜像获取 CCO 容器镜像：

```
$ CCO_IMAGE=$(oc adm release info --image-for='cloud-credential-operator'
$RELEASE_IMAGE -a ~/.pull-secret)
```

注意

确保 **$RELEASE_IMAGE** 的架构与将使用 **ccoctl** 工具的环境架构相匹配。

3. 运行以下命令，将 CCO 容器镜像中的 **ccoctl** 二进制文件提取到 OpenShift Container Platform 发行镜像中：

```
$ oc image extract $CCO_IMAGE \
  --file="/usr/bin/ccoctl.<rhel_version>" \ 1
  -a ~/.pull-secret
```

1 对于 **<rhel_version>**，请指定与主机使用的 Red Hat Enterprise Linux (RHEL) 版本对应的值。如果没有指定值，则默认使用 **ccoctl.rhel8**。以下值有效：

- **rhel8**: 为使用 RHEL 8 的主机指定这个值。
- **rhel9** ：为使用 RHEL 9 的主机指定这个值。

4. 运行以下命令更改权限以使 **ccoctl** 可执行：

```
$ chmod 775 ccoctl.<rhel_version>
```

验证

- 要验证 **ccoctl** 是否可使用，请运行以下命令来显示帮助文件：

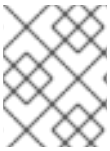
```
$ ccoctl --help
```

ccoctl --help 的输出

```
OpenShift credentials provisioning tool

Usage:
  ccoctl [command]

Available Commands:
  aws         Manage credentials objects for AWS cloud
  azure       Manage credentials objects for Azure
  gcp         Manage credentials objects for Google cloud
  help        Help about any command
  ibmcloud    Manage credentials objects for IBM Cloud
  nutanix     Manage credentials objects for Nutanix

Flags:
  -h, --help   help for ccoctl

Use "ccoctl [command] --help" for more information about a command.
```

9.6.7.2.2. 使用 Cloud Credential Operator 实用程序创建 GCP 资源

您可以使用 **ccoctl gcp create-all** 命令自动创建 GCP 资源。



注意

默认情况下，**ccoctl** 在运行命令的目录中创建对象。要在其他目录中创建对象，请使用 **--output-dir** 标志。此流程使用 **<path_to_ccoctl_output_dir>** 来引用这个目录。

先决条件

您必须：

- 提取并准备好 **ccoctl** 二进制文件。

流程

1. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 对象列表：

```
$ oc adm release extract \
```

```
--from=$RELEASE_IMAGE \
--credentials-requests \
--included \ 1
--install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
--to=<path_to_directory_for_credentials_requests> 3
```

1. **--included** 参数仅包含特定集群配置所需的清单。
2. 指定 **install-config.yaml** 文件的位置。
3. 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

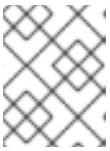
注意

此命令可能需要一些时间才能运行。

3. 运行以下命令，使用 **ccoctl** 工具处理所有 **CredentialsRequest** 对象：

```
$ ccoctl gcp create-all \
  --name=<name> \ 1
  --region=<gcp_region> \ 2
  --project=<gcp_project_id> \ 3
  --credentials-requests-dir=<path_to_credentials_requests_directory> 4
```

1. 为用于跟踪的所有创建 GCP 资源指定用户定义的名称。
2. 指定在其中创建云资源的 GCP 区域。
3. 指定在其中创建云资源的 GCP 项目 ID。
4. 指定包含 **CredentialsRequest** 清单文件的目录，以创建 GCP 服务帐户。



注意

如果您的集群使用 **TechPreviewNoUpgrade** 功能集启用的技术预览功能，则必须包含 **--enable-tech-preview** 参数。

验证

- 要验证 OpenShift Container Platform secret 是否已创建，列出 **<path_to_ccoctl_output_dir>/manifests** 目录中的文件：

```
$ ls <path_to_ccoctl_output_dir>/manifests
```

输出示例

```
cluster-authentication-02-config.yaml
openshift-cloud-controller-manager-gcp-ccm-cloud-credentials-credentials.yaml
openshift-cloud-credential-operator-cloud-credential-operator-gcp-ro-creds-credentials.yaml
openshift-cloud-network-config-controller-cloud-credentials-credentials.yaml
```

```
openshift-cluster-api-capg-manager-bootstrap-credentials-credentials.yaml
openshift-cluster-csi-drivers-gcp-pd-cloud-credentials-credentials.yaml
openshift-image-registry-installer-cloud-credentials-credentials.yaml
openshift-ingress-operator-cloud-credentials-credentials.yaml
openshift-machine-api-gcp-cloud-credentials-credentials.yaml
```

您可以通过查询 GCP 来验证是否已创建 IAM 服务帐户。如需更多信息，请参阅有关列出 IAM 服务帐户的 GCP 文档。

9.6.7.2.3. 整合 Cloud Credential Operator 实用程序清单

要为单个组件在集群外实现短期安全凭证，您必须将创建 Cloud Credential Operator 实用程序 (**ccoctl**) 的清单文件移到安装程序的正确目录中。

先决条件

- 您已使用托管集群的云平台配置了帐户。
- 您已配置了 Cloud Credential Operator 实用程序 (**ccoctl**)。
- 已使用 **ccoctl** 工具创建了集群所需的云供应商资源。

流程

1. 在安装程序使用的 GCP 帐户中添加以下粒度权限：

例 9.37. 所需的 GCP 权限

- compute.machineTypes.list
- compute.regions.list
- compute.zones.list
- dns.changes.create
- dns.changes.get
- dns.managedZones.create
- dns.managedZones.delete
- dns.managedZones.get
- dns.managedZones.list
- dns.networks.bindPrivateDNSZone
- dns.resourceRecordSets.create
- dns.resourceRecordSets.delete
- dns.resourceRecordSets.list

2. 如果您没有将 **install-config.yaml** 配置文件中的 **credentialsMode** 参数设置为 **Manual**，请修改值，如下所示：

配置文件片段示例

```
apiVersion: v1
baseDomain: example.com
credentialsMode: Manual
# ...
```

3. 如果您之前还没有创建安装清单文件，请运行以下命令：

```
$ openshift-install create manifests --dir <installation_directory>
```

其中 **<installation_directory>** 是安装程序在其中创建文件的目录。

4. 运行以下命令，将 **ccoctl** 工具生成的清单复制到安装程序创建的 **manifests** 目录中：

```
$ cp /<path_to_ccoctl_output_dir>/manifests/* ./manifests/
```

5. 运行以下命令，将 **ccoctl** 工具在 **tls** 目录中生成的私钥复制到安装目录中：

```
$ cp -a /<path_to_ccoctl_output_dir>/tls .
```

9.6.8. 部署集群

您可以在兼容云平台上安装 OpenShift Container Platform。

重要

在初始安装过程中，您只能运行安装程序的 **create cluster** 命令一次。

先决条件

- 您已使用托管集群的云平台配置了帐户。
- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。
- 已确认主机上的云供应商帐户具有部署集群的正确权限。权限不正确的帐户会导致安装过程失败，并显示包括缺失权限的错误消息。

流程

1. 删除所有不使用您为集群配置的 GCP 帐户的服务帐户密钥的现有 GCP 凭证，这些凭证存储在以下位置：
   - **GOOGLE_CREDENTIALS**、**GOOGLE_CLOUD_KEYFILE_JSON** 或 **GCLOUD_KEYFILE_JSON** 环境变量
   - The ~/**.gcp/osServiceAccount.json** file
   - **gcloud cli** 默认凭证
2. 进入包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定自定义 **./install-config.yaml** 文件的位置。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不是 **info**。

3. 可选：您可以减少用来安装集群的服务帐户的权限数量。
   - 如果将 **Owner** 角色分配给您的服务帐户，您可以删除该角色并将其替换为 **Viewer** 角色。
   - 如果包含 **Service Account Key Admin** 角色，您可以将其删除。

验证

当集群部署成功完成时：

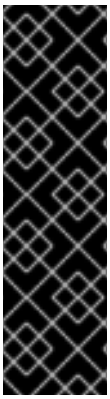
- 终端会显示用于访问集群的说明，包括指向 Web 控制台和 **kubeadmin** 用户的凭证的链接。
- 凭证信息还会输出到 **<installation_directory>/.openshift_install.log**.

重要

不要删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

9.6.9. 使用 CLI 登录集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

9.6.10. 禁用默认的 OperatorHub 目录源

在 OpenShift Container Platform 安装过程中，默认为 OperatorHub 配置由红帽和社区项目提供的源内容的 operator 目录。在受限网络环境中，必须以集群管理员身份禁用默认目录。

流程

- 通过在 **OperatorHub** 对象中添加 **disableAllDefaultSources: true 来** 禁用默认目录的源：

```
$ oc patch OperatorHub cluster --type json \
    -p '[{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]'
```

提示

或者，您可以使用 Web 控制台管理目录源。在 **Administration → Cluster Settings → Configuration → OperatorHub** 页面中，点 **Sources** 选项卡，您可以在其中创建、更新、删除、禁用和启用单独的源。

9.6.11. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于 远程健康监控

9.6.12. 后续步骤

- 验证安装。
- 自定义集群。
- 为 Cluster Samples Operator 和 **must-gather** 工具 配置镜像流。
- 了解如何在 受限网络中使用 Operator Lifecycle Manager(OLM)。
- 如果您用来安装集群的镜像 registry 具有可信任的 CA，请通过 配置额外的信任存储将其添加到集群中。
- 如果需要，您可以选择 不使用远程健康报告。
- 如果需要，请参阅 注册断开连接的集群

9.7. 将 GCP 上的集群安装到现有的 VPC 中

在 OpenShift Container Platform 版本 4.16 中，您可以在 Google Cloud Platform(GCP)上将集群安装到现有的 Virtual Private Cloud(VPC)中。安装程序会置备所需基础架构的其余部分，您可以进一步自定义这些基础架构。要自定义安装，请在安装集群前修改 **install-config.yaml** 文件中的参数。

9.7.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读有关 选择集群安装方法的文档，并为用户准备它。
- 已将 GCP 项目配置为 托管集群。
- 如果使用防火墙，则会 将其配置为允许集群需要访问的站点。

9.7.2. 关于使用自定义 VPC

在 OpenShift Container Platform 4.16 中，您可以在 Google Cloud Platform (GCP) 的现有 Virtual Private Cloud (VPC) 中将集群部署到现有子网中。通过将 OpenShift Container Platform 部署到现有的 GCP VPC 中，您可以避免新帐户中的限制，或者更容易地利用公司所设置的操作限制。如果您无法获得您自己创建 VPC 所需的基础架构创建权限，请使用这个安装选项。您必须为子网配置网络。

9.7.2.1. 使用 VPC 的要求

VPC CIDR 块的 union，机器网络 CIDR 不能为空。子网必须位于机器网络中。

安装程序不会创建以下组件：

- NAT 网关
- 子网
- 路由表
- VPC 网络

注意

安装程序要求您使用由云提供的 DNS 服务器。不支持使用自定义 DNS 服务器，并导致安装失败。

9.7.2.2. VPC 验证

要确保您提供的子网适合您的环境，安装程序会确认以下信息：

- 您指定的所有子网都存在。
- 您可以为 control-plane 机器提供一个子网，并为计算机器提供一个子网。
- 子网的 CIDR 属于您指定的机器 CIDR。

9.7.2.3. 权限划分

有些个人可以在您的云中创建不同于其他个人的资源。例如，您可以创建针对于特定应用程序的对象，如实例、存储桶和负载均衡器，但不能创建与网络相关的组件，如 VPC 、子网或入站规则。

9.7.2.4. 集群间隔离

如果您将 OpenShift Container Platform 部署到现有网络中，集群服务的隔离将在以下方面减少：

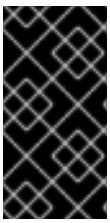
- 您可以在同一 VPC 中安装多个 OpenShift Container Platform 集群。
- 整个网络允许 ICMP 入站流量。
- 整个网络都允许 TCP 22 入站流量 (SSH)。
- 整个网络都允许 control plane TCP 6443 入站流量 (Kubernetes API)。
- 整个网络都允许 control plane TCP 22623 入站流量 (MCS) 。

9.7.3. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。



重要

如果您的集群无法直接访问互联网，则可以在置备的某些类型的基础架构上执行受限网络安装。在此过程中，您可以下载所需的内容，并使用它为镜像 registry 填充安装软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群前，您要更新镜像 registry 的内容。

9.7.4. 为集群节点 SSH 访问生成密钥对

在 OpenShift Container Platform 安装过程中，您可以为安装程序提供 SSH 公钥。密钥通过它们的 Ignition 配置文件传递给 Red Hat Enterprise Linux CoreOS(RHCOS)节点，用于验证对节点的 SSH 访问。密钥添加到每个节点上 **core** 用户的 **~/.ssh/authorized_keys** 列表中，这将启用免密码身份验证。

将密钥传递给节点后，您可以使用密钥对作为用户 **核心** 通过 SSH 连接到 RHCOS 节点。若要通过 SSH 访问节点，必须由 SSH 为您的本地用户管理私钥身份。

如果要通过 SSH 连接到集群节点来执行安装调试或灾难恢复，则必须在安装过程中提供 SSH 公钥。**./openshift-install gather** 命令还需要在集群节点上设置 SSH 公钥。

重要

不要在生产环境中跳过这个过程，在生产环境中需要灾难恢复和调试。

注意

您必须使用本地密钥，而不是使用特定平台方法配置 的密钥，如 AWS 密钥对。

流程

1. 如果您在本地计算机上没有可用于在集群节点上进行身份验证的现有 SSH 密钥对，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_ed25519**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

注意

如果您计划在 **x86_64**、**ppc64le** 和 **s390x** 架构上安装使用 RHEL 加密库（这些加密库已提交给 NIST 用于 FIPS 140-2/140-3 验证）的 OpenShift Container Platform 集群，则不要创建使用 **ed25519** 算法的密钥。相反，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 查看公共 SSH 密钥：

```
$ cat <path>/<file_name>.pub
```

例如，运行以下命令来查看 **~/.ssh/id_ed25519.pub** 公钥：

```
$ cat ~/.ssh/id_ed25519.pub
```

3. 将 SSH 私钥身份添加到本地用户的 SSH 代理（如果尚未添加）。在集群节点上，或者要使用 **./openshift-install gather** 命令，需要对该密钥进行 SSH 代理管理，才能在集群节点上进行免密码 SSH 身份验证。

注意

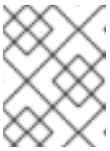
在某些发行版中，自动管理默认 SSH 私钥身份，如 **~/.ssh/id_rsa** 和 **~/.ssh/id_dsa**。

a. 如果 **ssh-agent** 进程尚未为您的本地用户运行，请将其作为后台任务启动：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```



注意

如果集群处于 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

4. 将 SSH 私钥添加到 **ssh-agent** ：

```
$ ssh-add <path>/<file_name> 1
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_ed25519.pub**

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

后续步骤

- 安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

9.7.5. 获取安装程序

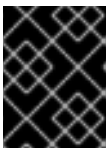
在安装 OpenShift Container Platform 前，将安装文件下载到您用于安装的主机上。

先决条件

- 您有一台运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB。

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐户，请使用您的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入到安装类型的页面，下载与您的主机操作系统和架构对应的安装程序，并将该文件放在您要存储安装配置文件的目录中。



重要

安装程序会在用来安装集群的计算机上创建几个文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。

重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，请为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。此 pull secret 允许您与所含授权机构提供的服务进行身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

9.7.6. 创建安装配置文件

您可以自定义在 Google Cloud Platform(GCP)上安装的 OpenShift Container Platform 集群。

先决条件

- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。

流程

1. 创建 **install-config.yaml** 文件。
   a. 进入包含安装程序的目录并运行以下命令：

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

   1 对于 **<installation_directory>**，请指定要存储安装程序创建的文件的目录名称。

   在指定目录时：

   - 验证该目录是否具有**执行**权限。在安装目录中运行 Terraform 二进制文件需要这个权限。
   - 使用空目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

   b. 在提示符处，提供云的配置详情：
      i. 可选： 选择用于访问集群机器的 SSH 密钥。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

      ii. 选择 **gcp** 作为目标平台。

iii. 如果您没有为计算机上的 GCP 帐户配置服务帐户密钥，则必须从 GCP 获取它，并粘贴文件的内容或输入文件的绝对路径。

iv. 选择要在其中置备集群的项目 ID。默认值由您配置的服务帐户指定。

v. 选择要将集群部署到的区域。

vi. 选择集群要部署到的基域。基域与您为集群创建的公共 DNS 区对应。

vii. 为集群输入描述性名称。

2. 修改 **install-config.yaml** 文件。您可以在"安装配置参数"部分找到有关可用参数的更多信息。

3. 备份 **install-config.yaml** 文件，以便您可以使用它安装多个集群。

重要

install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用此文件，必须现在备份。

其他资源

- GCP 的安装配置参数

9.7.6.1. 集群安装的最低资源要求

每台集群机器都必须满足以下最低要求：

表 9.19. 最低资源要求

机器	操作系统	vCPU [1]	虚拟内存	Storage	每秒输入/输出 (IOPS) [2]
bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane（控制平面）	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS、RHEL 8.6 及更新版本 [3]	2	8 GB	100 GB	300

1. 当未启用并发多线程(SMT)或超线程时，一个 vCPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比例：（每个内核数的线程）× sockets = vCPU。

2. OpenShift Container Platform 和 Kubernetes 对磁盘性能非常敏感，建议使用更快的存储速度，特别是 control plane 节点上需要 10 ms p99 fsync 持续时间的 etcd。请注意，在许多云平台上，存储大小和 IOPS 可一起扩展，因此您可能需要过度分配存储卷来获取足够的性能。

3. 与所有用户置备的安装一样，如果您选择在集群中使用 RHEL 计算机器，则负责所有操作系统生命周期管理和维护，包括执行系统更新、应用补丁和完成所有其他必要的任务。RHEL 7 计算机器的使用已弃用，并已在 OpenShift Container Platform 4.10 及更新的版本中删除。

注意

从 OpenShift Container Platform 版本 4.13 开始，RHCOS 基于 RHEL 版本 9.2，它更新了微架构要求。以下列表包含每个架构需要的最小指令集架构 (ISA)：

- x86-64 体系结构需要 x86-64-v2 ISA
- ARM64 架构需要 ARMv8.0-A ISA
- IBM Power 架构需要 Power 9 ISA
- s390x 架构需要 z14 ISA

如需更多信息，请参阅 RHEL 架构。

如果平台的实例类型满足集群机器的最低要求，则 OpenShift Container Platform 支持使用它。

其他资源

- 优化存储

9.7.6.2. 为 GCP 测试的实例类型

以下 Google Cloud Platform 实例类型已使用 OpenShift Container Platform 测试。

例 9.38. 机器系列

- **A2**
- **A3**
- **C2**
- **C2D**
- **C3**
- **C3D**
- **E2**
- **M1**
- **N1**
- **N2**
- **N2D**
- **Tau T2D**

9.7.6.3. 在 64 位 ARM 基础架构上为 GCP 测试的实例类型

以下 Google Cloud Platform (GCP) 64 位 ARM 实例类型已使用 OpenShift Container Platform 测试。

例 9.39. 64 位 ARM 机器的机器系列

- **Tau T2A**

9.7.6.4. 使用自定义机器类型

支持使用自定义机器类型来安装 OpenShift Container Platform 集群。

使用自定义机器类型时请考虑以下几点：

- 与预定义的实例类型类似，自定义机器类型必须满足 control plane 和计算机器的最低资源要求。如需更多信息，请参阅"集群安装的资源要求"。
- 自定义机器类型的名称必须遵循以下语法：
  custom-<number_of_cpus>-<amount_of_memory_in_mb>

  例如，**custom-6-20480**。

作为安装过程的一部分，您可以在 **install-config.yaml** 文件中指定自定义机器类型。

带有自定义机器类型的 install-config.yaml 文件示例

```
compute:
- architecture: amd64
  hyperthreading: Enabled
  name: worker
  platform:
    gcp:
      type: custom-6-20480
  replicas: 2
controlPlane:
  architecture: amd64
  hyperthreading: Enabled
  name: master
  platform:
    gcp:
      type: custom-6-20480
  replicas: 3
```

9.7.6.5. 启用屏蔽虚拟机

您可在安装集群时使用 Shielded 虚拟机。Shielded 虚拟机具有额外的安全功能，包括安全引导、固件和完整性监控和 rootkit 检测。如需更多信息，请参阅 Google 文档中有关 Shielded 虚拟机 的文档。

注意

目前，在具有 64 位 ARM 基础架构的集群中不支持 Shielded 虚拟机。

先决条件

- 您已创建了 **install-config.yaml** 文件。

流程

- 在部署集群前，使用文本编辑器编辑 **install-config.yaml** 文件并添加以下部分之一：

  a. 仅将屏蔽的虚拟机用于 control plane 机器：

```
controlPlane:
  platform:
    gcp:
      secureBoot: Enabled
```

  b. 仅将屏蔽的虚拟机用于计算机器：

```
compute:
- platform:
    gcp:
      secureBoot: Enabled
```

  c. 将屏蔽的虚拟机用于所有机器：

```
platform:
  gcp:
    defaultMachinePlatform:
      secureBoot: Enabled
```

9.7.6.6. 启用机密虚拟机

您可在安装集群时使用机密虚拟机。机密虚拟机在处理数据时加密数据。如需更多信息，请参阅 Google 文档中有关 机密计算的内容。您可以同时启用机密虚拟机和 Shielded 虚拟机，虽然它们不相互依赖。

注意

64 位 ARM 架构目前不支持机密虚拟机。

先决条件

- 您已创建了 **install-config.yaml** 文件。

流程

- 在部署集群前，使用文本编辑器编辑 **install-config.yaml** 文件并添加以下部分之一：

  a. 仅将机密虚拟机用于 control plane 机器：

```
controlPlane:
  platform:
    gcp:
      confidentialCompute: Enabled (1)
      type: n2d-standard-8 (2)
      onHostMaintenance: Terminate (3)
```

  1. 启用机密虚拟机。
  2. 指定支持机密虚拟机的机器类型。机密虚拟机需要 N2D 或 C2D 系列机器类型。有关支持的机器类型的更多信息，请参阅支持的操作系统和机器类型。

3 指定主机维护事件期间虚拟机的行为，如硬件或软件更新。对于使用机密虚拟机的机器，此值必须设置为 **Terminate**，这会停止虚拟机。机密虚拟机不支持实时迁移。

b. 仅将机密虚拟机用于计算机器：

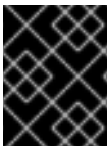
```
compute:
- platform:
    gcp:
      confidentialCompute: Enabled
      type: n2d-standard-8
      onHostMaintenance: Terminate
```

c. 将机密虚拟机用于所有机器：

```
platform:
  gcp:
    defaultMachinePlatform:
      confidentialCompute: Enabled
      type: n2d-standard-8
      onHostMaintenance: Terminate
```

9.7.6.7. GCP 的自定义 install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多详情，或修改所需参数的值。



重要

此示例 YAML 文件仅供参考。您必须使用安装程序来获取 **install-config.yaml** 文件，并进行修改。

```
apiVersion: v1
baseDomain: example.com 1
credentialsMode: Mint 2
controlPlane: 3 4
  hyperthreading: Enabled 5
  name: master
  platform:
    gcp:
      type: n2-standard-4
      zones:
      - us-central1-a
      - us-central1-c
      osDisk:
        diskType: pd-ssd
        diskSizeGB: 1024
        encryptionKey: 6
          kmsKey:
            name: worker-key
            keyRing: test-machine-keys
            location: global
            projectID: project-id
```

```
      tags: 7
      - control-plane-tag1
      - control-plane-tag2
      osImage: 8
        project: example-project-name
        name: example-image-name
  replicas: 3
compute: 9 10
- hyperthreading: Enabled 11
  name: worker
  platform:
    gcp:
      type: n2-standard-4
      zones:
      - us-central1-a
      - us-central1-c
      osDisk:
        diskType: pd-standard
        diskSizeGB: 128
        encryptionKey: 12
          kmsKey:
            name: worker-key
            keyRing: test-machine-keys
            location: global
            projectID: project-id
        tags: 13
        - compute-tag1
        - compute-tag2
        osImage: 14
          project: example-project-name
          name: example-image-name
  replicas: 3
metadata:
  name: test-cluster 15
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 16
  serviceNetwork:
  - 172.30.0.0/16
platform:
  gcp:
    projectID: openshift-production 17
    region: us-central1 18
    defaultMachinePlatform:
      tags: 19
      - global-tag1
      - global-tag2
      osImage: 20
        project: example-project-name
        name: example-image-name
```

```
    network: existing_vpc 21
    controlPlaneSubnet: control_plane_subnet 22
    computeSubnet: compute_subnet 23
pullSecret: '{"auths": ...}' 24
fips: false 25
sshKey: ssh-ed25519 AAAA... 26
```

1 15 17 18 24 必需。安装程序会提示您输入这个值。

2 可选：添加此参数来强制 Cloud Credential Operator (CCO) 使用指定的模式。默认情况下，CCO 使用 **kube-system** 命名空间中的 root 凭证来动态尝试决定凭证的功能。有关 CCO 模式的详情，请参阅*身份验证和授权*指南中的"About the Cloud Credential Operator"部分。

3 9 如果没有提供这些参数和值，安装程序会提供默认值。

4 10 **controlPlane** 部分是一个单个映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，controlPlane **部分** 的第一行则不以连字符开头。仅使用一个 control plane 池。

5 11 是否要启用或禁用并发多线程或 **超线程**。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将 参数值设置为 **Disabled** 来禁用它。如果在某些集群机器中禁用并发多线程，则必须在所有集群机器中禁用它。

重要

如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。如果您禁用并发多线程，请为您的机器使用较大的类型，如 **n1-standard-8**。

6 12 可选：自定义加密密钥部分来加密虚拟机和持久性卷。您的默认计算服务帐户必须具有相应的权限才能使用您的 KMS 密钥并分配了正确的 IAM 角色。默认服务帐户名称遵循 **service-<project_number>@compute-system.iam.gserviceaccount.com** 模式。有关为您的服务帐户授予正确权限的更多信息，请参阅 "Machine management" → "Creating compute machine set" → "Creating a compute machine set on GCP"。

7 13 19 可选：要应用到 control plane 或计算机器集的一组网络标签。**platform.gcp.defaultMachinePlatform.tags** 参数将应用到 control plane 和计算机器。如果设置了 **compute.platform.gcp.tags** 或 **controlPlane.platform.gcp.tags** 参数，它们会覆盖 **platform.gcp.defaultMachinePlatform.tags** 参数。

8 14 20 可选：应该用来引导 control plane 和计算机器的自定义 Red Hat Enterprise Linux CoreOS (RHCOS)。**platform.gcp.defaultMachinePlatform.osImage** 下的 **project** 和 **name** 参数应用到 control plane 和计算机器。如果设置了 **controlPlane.platform.gcp.osImage** 或 **compute.platform.gcp.osImage** 下的 **project** 和 **name** 参数，它们会覆盖 **platform.gcp.defaultMachinePlatform.osImage** 参数。

16 要安装的集群网络插件。默认值 **OVNKubernetes** 是唯一支持的值。

21 指定现有 VPC 的名称。

22 指定要将 control plane 机器部署到的现有子网的名称。该子网必须属于您指定的 VPC。

23 指定要将计算机器部署到的现有子网的名称。该子网必须属于您指定的 VPC。

25

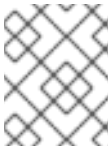
是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes

重要

要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 Red Hat Enterprise Linux (RHEL) 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅在 FIPS 模式中安装该系统。

当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS)时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。

26 您可以选择提供您用来访问集群中机器的 **sshKey** 值。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

其他资源

- 为计算机器设置启用客户管理的加密密钥

9.7.6.8. 在 GCP 上创建具有全局访问权限的 Ingress Controller

您可以创建一个对 Google Cloud Platform(GCP)集群全局访问权限的 Ingress Controller。只有使用内部负载均衡器的 Ingress Controller 才可使用全局访问。

先决条件

- 已创建 **install-config.yaml**，并完成对其所做的任何修改。

流程

在新 GCP 集群上创建具有全局访问权限的 Ingress Controller。

1. 进入包含安装程序的目录并创建清单文件：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定包含集群的 **install-config.yaml** 文件的目录名称。

2. 在 **<installation_directory>/manifests/** 目录中创建一个名为 **cluster-ingress-default-ingresscontroller.yaml** 的文件：

```
$ touch <installation_directory>/manifests/cluster-ingress-default-ingresscontroller.yaml 1
```

1 对于 **<installation_directory>**，请指定包含集群的 **manifests/** 目录的目录名称。

创建该文件后，几个网络配置文件位于 **manifests/** 目录中，如下所示：

```
$ ls <installation_directory>/manifests/cluster-ingress-default-ingresscontroller.yaml
```

输出示例

```
cluster-ingress-default-ingresscontroller.yaml
```

3. 在编辑器中打开 **cluster-ingress-default-ingresscontroller.yaml** 文件，并输入描述您想要的 Operator 配置的自定义资源(CR)：

到 全局的 clientAccess 配置示例

```
apiVersion: operator.openshift.io/v1
kind: IngressController
metadata:
  name: default
  namespace: openshift-ingress-operator
spec:
  endpointPublishingStrategy:
    loadBalancer:
      providerParameters:
        gcp:
          clientAccess: Global 1
        type: GCP
      scope: Internal 2
    type: LoadBalancerService
```

1 将 **gcp.clientAccess** 设置为 **Global**。

2 只有使用内部负载均衡器的 Ingress Controller 才可使用全局访问。

9.7.6.9. 在安装过程中配置集群范围的代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 **Proxy** 对象的 **spec.noProxy** 字段中添加站点来绕过代理。

注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

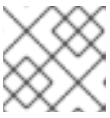
对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，**Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

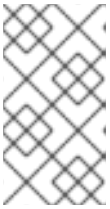
```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 **.** 以仅匹配子域。例如，**.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代理。
4. 如果提供，安装程序会在 **openshift-config** 命名空间中生成名为 **user-ca-bundle** 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 **trusted-ca-bundle** 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， **Proxy** 对象的 **trustedCA** 字段中也会引用此配置映射。**additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。
5. 可选：决定 **Proxy** 对象的配置以引用 **trustedCA** 字段中 **user-ca-bundle** 配置映射的策略。允许的值是 **Proxyonly** 和 **Always**。仅在配置了 **http/https** 代理时，使用 **Proxyonly** 引用 **user-ca-bundle** 配置映射。使用 **Always** 始终引用 **user-ca-bundle** 配置映射。默认值为 **Proxyonly**。



注意

安装程序不支持代理的 **readinessEndpoints** 字段。



注意

如果安装程序超时，重启并使用安装程序的 **wait-for** 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. 保存该文件并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 cluster 的集群范围代理，该代理 **使用** 提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

9.7.7. 安装 OpenShift CLI

您可以安装 OpenShift CLI(**oc**)来使用命令行界面与 OpenShift Container Platform 进行交互。您可以在 Linux、Windows 或 macOS 上安装 **oc**。

重要

如果安装了旧版本的 **oc**，则可能无法使用 OpenShift Container Platform 4.16 中的所有命令。下载并安装新版本的 **oc**。

在 Linux 上安装 OpenShift CLI
您可以按照以下流程在 Linux 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **产品变体** 下拉列表中选择架构。
3. 从 **版本** 下拉列表中选择适当的版本。
4. 点 **OpenShift v4.16 Linux Client** 条目旁的 **Download Now** 来保存文件。
5. 解包存档：

```
$ tar xvf <file>
```

6. 将 **oc** 二进制文件放到 **PATH 中的目录中**。
   要查看您的 **PATH**，请执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

在 Windows 上安装 OpenShift CLI
您可以按照以下流程在 Windows 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 Windows Client** 条目旁的 **Download Now** 来保存文件。

4. 使用 ZIP 程序解压存档。
5. 将 **oc** 二进制文件移到 **PATH 中的目录中**。
   要查看您的 **PATH**，请打开命令提示并执行以下命令：

```
C:\> path
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

在 macOS 上安装 OpenShift CLI
您可以按照以下流程在 macOS 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 macOS Client** 条目旁的 **Download Now** 来保存文件。

注意

对于 macOS arm64，请选择 **OpenShift v4.16 macOS arm64 Client** 条目。

4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
   要查看您的 **PATH**，请打开终端并执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

9.7.8. 在 kube-system 项目中存储管理员级别的 secret 的替代方案

默认情况下，管理员 secret 存储在 **kube-system** 项目中。如果您在 **install-config.yaml** 文件中将 **credentialsMode** 参数配置为 **Manual**，则必须使用以下替代方案之一：

- 要手动管理长期云凭证，请按照手动创建长期凭证中的步骤操作。
- 要实现在集群外为各个组件管理的短期凭证，请按照配置 GCP 集群以使用短期凭证中的步骤操作。

9.7.8.1. 手动创建长期凭证

在无法访问云身份和访问管理(IAM)API 的环境中，或者管理员更不希望将管理员级别的凭证 secret 存储在集群 **kube-system** 命名空间中时，可以在安装前将 Cloud Credential Operator(CCO)放入手动模式。

流程

1. 在安装程序使用的 GCP 帐户中添加以下粒度权限：

   例 9.40. 所需的 GCP 权限

   - compute.machineTypes.list
   - compute.regions.list
   - compute.zones.list
   - dns.changes.create
   - dns.changes.get
   - dns.managedZones.create
   - dns.managedZones.delete
   - dns.managedZones.get
   - dns.managedZones.list
   - dns.networks.bindPrivateDNSZone
   - dns.resourceRecordSets.create
   - dns.resourceRecordSets.delete
   - dns.resourceRecordSets.list

2. 如果您没有将 **install-config.yaml** 配置文件中的 **credentialsMode** 参数设置为 **Manual**，请修改值，如下所示：

   配置文件片段示例

   ```
   apiVersion: v1
   baseDomain: example.com
   credentialsMode: Manual
   # ...
   ```

3. 如果您之前还没有创建安装清单文件，请运行以下命令：

   ```
   $ openshift-install create manifests --dir <installation_directory>
   ```

   其中 **<installation_directory>** 是安装程序在其中创建文件的目录。

4. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

   ```
   $ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
   ```

5. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 自定义资源 (CR) 列表：

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
  --included \ 1
  --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
  --to=<path_to_directory_for_credentials_requests> 3
```

1. **--included** 参数仅包含特定集群配置所需的清单。
2. 指定 **install-config.yaml** 文件的位置。
3. 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

此命令为每个 **CredentialsRequest** 对象创建一个 YAML 文件。

CredentialsRequest 对象示例

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component_credentials_request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: GCPProviderSpec
    predefinedRoles:
    - roles/storage.admin
    - roles/iam.serviceAccountUser
    skipServiceCheck: true
  ...
```

6. 在之前生成的 **openshift-install** 清单目录中为 secret 创建 YAML 文件。secret 必须使用在 **spec.secretRef** 中为每个 **CredentialsRequest** 定义的命名空间和 secret 名称存储。

带有 secret 的 CredentialsRequest 对象示例

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component_credentials_request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
      ...
  secretRef:
```

```
    name: <component_secret>
    namespace: <component_namespace>
  ...
```

Secret 对象示例

```
apiVersion: v1
kind: Secret
metadata:
  name: <component_secret>
  namespace: <component_namespace>
data:
  service_account.json: <base64_encoded_gcp_service_account_file>
```

重要

在升级使用手动维护凭证的集群前，您必须确保 CCO 处于可升级状态。

9.7.8.2. 将 GCP 集群配置为使用短期凭证

要安装配置为使用 GCP Workload Identity 的集群，您必须配置 CCO 实用程序并为集群创建所需的 GCP 资源。

9.7.8.2.1. 配置 Cloud Credential Operator 工具

当 Cloud Credential Operator(CCO)以手动模式运行时，要从集群外部创建和管理云凭证，提取并准备 CCO 实用程序(**ccoctl**)二进制文件。

注意

ccoctl 工具是在 Linux 环境中运行的 Linux 二进制文件。

先决条件

- 您可以访问具有集群管理员权限的 OpenShift Container Platform 帐户。
- 已安装 OpenShift CLI(**oc**)。
- 您已在安装程序使用的 GCP 帐户中添加了以下身份验证选项之一：
  - **IAM Workload Identity Pool Admin**角色。
  - 以下粒度权限：

    例 9.41. 所需的 GCP 权限

    - compute.projects.get
    - iam.googleapis.com/workloadIdentityPoolProviders.create
    - iam.googleapis.com/workloadIdentityPoolProviders.get
    - iam.googleapis.com/workloadIdentityPools.create
    - iam.googleapis.com/workloadIdentityPools.delete

- iam.googleapis.com/workloadIdentityPools.get
- iam.googleapis.com/workloadIdentityPools.undelete
- iam.roles.create
- iam.roles.delete
- iam.roles.list
- iam.roles.undelete
- iam.roles.update
- iam.serviceAccounts.create
- iam.serviceAccounts.delete
- iam.serviceAccounts.getIamPolicy
- iam.serviceAccounts.list
- iam.serviceAccounts.setIamPolicy
- iam.workloadIdentityPoolProviders.get
- iam.workloadIdentityPools.delete
- resourcemanager.projects.get
- resourcemanager.projects.getIamPolicy
- resourcemanager.projects.setIamPolicy
- storage.buckets.create
- storage.buckets.delete
- storage.buckets.get
- storage.buckets.getIamPolicy
- storage.buckets.setIamPolicy
- storage.objects.create
- storage.objects.delete
- storage.objects.list

流程

1. 运行以下命令，为 OpenShift Container Platform 发行镜像设置变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. 运行以下命令，从 OpenShift Container Platform 发行镜像获取 CCO 容器镜像：

```
$ CCO_IMAGE=$(oc adm release info --image-for='cloud-credential-operator'
$RELEASE_IMAGE -a ~/.pull-secret)
```

注意

确保 **$RELEASE_IMAGE** 的架构与将使用 **ccoctl** 工具的环境架构相匹配。

3. 运行以下命令，将 CCO 容器镜像中的 **ccoctl** 二进制文件提取到 OpenShift Container Platform 发行镜像中：

```
$ oc image extract $CCO_IMAGE \
  --file="/usr/bin/ccoctl.<rhel_version>" \ 1
  -a ~/.pull-secret
```

1 对于 **<rhel_version>**，请指定与主机使用的 Red Hat Enterprise Linux (RHEL) 版本对应的值。如果没有指定值，则默认使用 **ccoctl.rhel8**。以下值有效：

- **rhel8**: 为使用 RHEL 8 的主机指定这个值。
- **rhel9** ：为使用 RHEL 9 的主机指定这个值。

4. 运行以下命令更改权限以使 **ccoctl** 可执行：

```
$ chmod 775 ccoctl.<rhel_version>
```

验证

- 要验证 **ccoctl** 是否可使用，请运行以下命令来显示帮助文件：

```
$ ccoctl --help
```

ccoctl --help 的输出

```
OpenShift credentials provisioning tool

Usage:
  ccoctl [command]

Available Commands:
  aws         Manage credentials objects for AWS cloud
  azure       Manage credentials objects for Azure
  gcp         Manage credentials objects for Google cloud
  help        Help about any command
  ibmcloud    Manage credentials objects for IBM Cloud
  nutanix     Manage credentials objects for Nutanix

Flags:
  -h, --help   help for ccoctl

Use "ccoctl [command] --help" for more information about a command.
```

9.7.8.2.2. 使用 Cloud Credential Operator 实用程序创建 GCP 资源

您可以使用 **ccoctl gcp create-all** 命令自动创建 GCP 资源。

注意

默认情况下，**ccoctl** 在运行命令的目录中创建对象。要在其他目录中创建对象，请使用 **--output-dir** 标志。此流程使用 **<path_to_ccoctl_output_dir>** 来引用这个目录。

先决条件

您必须：

- 提取并准备好 **ccoctl** 二进制文件。

流程

1. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 对象列表：

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
  --included \ 1
  --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
  --to=<path_to_directory_for_credentials_requests> 3
```

1. **--included** 参数仅包含特定集群配置所需的清单。
2. 指定 **install-config.yaml** 文件的位置。
3. 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

注意

此命令可能需要一些时间才能运行。

3. 运行以下命令，使用 **ccoctl** 工具处理所有 **CredentialsRequest** 对象：

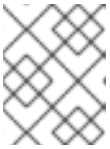
```
$ ccoctl gcp create-all \
  --name=<name> \ 1
  --region=<gcp_region> \ 2
  --project=<gcp_project_id> \ 3
  --credentials-requests-dir=<path_to_credentials_requests_directory> 4
```

1. 为用于跟踪的所有创建 GCP 资源指定用户定义的名称。

2 指定在其中创建云资源的 GCP 区域。

3 指定在其中创建云资源的 GCP 项目 ID。

4 指定包含 **CredentialsRequest** 清单文件的目录，以创建 GCP 服务帐户。



注意

如果您的集群使用 **TechPreviewNoUpgrade** 功能集启用的技术预览功能，则必须包含 **--enable-tech-preview** 参数。

验证

- 要验证 OpenShift Container Platform secret 是否已创建，列出 **<path_to_ccoctl_output_dir>/manifests** 目录中的文件：

```
$ ls <path_to_ccoctl_output_dir>/manifests
```

输出示例

```
cluster-authentication-02-config.yaml
openshift-cloud-controller-manager-gcp-ccm-cloud-credentials-credentials.yaml
openshift-cloud-credential-operator-cloud-credential-operator-gcp-ro-creds-credentials.yaml
openshift-cloud-network-config-controller-cloud-credentials-credentials.yaml
openshift-cluster-api-capg-manager-bootstrap-credentials-credentials.yaml
openshift-cluster-csi-drivers-gcp-pd-cloud-credentials-credentials.yaml
openshift-image-registry-installer-cloud-credentials-credentials.yaml
openshift-ingress-operator-cloud-credentials-credentials.yaml
openshift-machine-api-gcp-cloud-credentials-credentials.yaml
```

您可以通过查询 GCP 来验证是否已创建 IAM 服务帐户。如需更多信息，请参阅有关列出 IAM 服务帐户的 GCP 文档。

9.7.8.2.3. 整合 Cloud Credential Operator 实用程序清单

要为单个组件在集群外实现短期安全凭证，您必须将创建 Cloud Credential Operator 实用程序 (**ccoctl**) 的清单文件移到安装程序的正确目录中。

先决条件

- 您已使用托管集群的云平台配置了帐户。
- 您已配置了 Cloud Credential Operator 实用程序 (**ccoctl**)。
- 已使用 **ccoctl** 工具创建了集群所需的云供应商资源。

流程

1. 在安装程序使用的 GCP 帐户中添加以下粒度权限：

例 9.42. 所需的 GCP 权限

- compute.machineTypes.list

- compute.regions.list
- compute.zones.list
- dns.changes.create
- dns.changes.get
- dns.managedZones.create
- dns.managedZones.delete
- dns.managedZones.get
- dns.managedZones.list
- dns.networks.bindPrivateDNSZone
- dns.resourceRecordSets.create
- dns.resourceRecordSets.delete
- dns.resourceRecordSets.list

2. 如果您没有将 **install-config.yaml** 配置文件中的 **credentialsMode** 参数设置为 **Manual**，请修改值，如下所示：

   配置文件片段示例

   ```
   apiVersion: v1
   baseDomain: example.com
   credentialsMode: Manual
   # ...
   ```

3. 如果您之前还没有创建安装清单文件，请运行以下命令：

   ```
   $ openshift-install create manifests --dir <installation_directory>
   ```

   其中 **<installation_directory>** 是安装程序在其中创建文件的目录。

4. 运行以下命令，将 **ccoctl** 工具生成的清单复制到安装程序创建的 **manifests** 目录中：

   ```
   $ cp /<path_to_ccoctl_output_dir>/manifests/* ./manifests/
   ```

5. 运行以下命令，将 **ccoctl** 工具在 **tls** 目录中生成的私钥复制到安装目录中：

   ```
   $ cp -a /<path_to_ccoctl_output_dir>/tls .
   ```

9.7.9. 部署集群

您可以在兼容云平台上安装 OpenShift Container Platform。

重要

在初始安装过程中，您只能运行安装程序的 **create cluster** 命令一次。

先决条件

- 您已使用托管集群的云平台配置了帐户。
- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。
- 已确认主机上的云供应商帐户具有部署集群的正确权限。权限不正确的帐户会导致安装过程失败，并显示包括缺失权限的错误消息。

流程

1. 删除所有不使用您为集群配置的 GCP 帐户的服务帐户密钥的现有 GCP 凭证，这些凭证存储在以下位置：
   - **GOOGLE_CREDENTIALS**、**GOOGLE_CLOUD_KEYFILE_JSON** 或 **GCLOUD_KEYFILE_JSON** 环境变量
   - The ~/**.gcp/osServiceAccount.json** file
   - **gcloud cli** 默认凭证
2. 进入包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1. 对于 **<installation_directory>**，请指定自定义 **./install-config.yaml** 文件的位置。
2. 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不是 **info**。

3. 可选：您可以减少用来安装集群的服务帐户的权限数量。
   - 如果将 **Owner** 角色分配给您的服务帐户，您可以删除该角色并将其替换为 **Viewer** 角色。
   - 如果包含 **Service Account Key Admin 角色**，您可以将其删除。

验证

当集群部署成功完成时：

- 终端会显示用于访问集群的说明，包括指向 Web 控制台和 **kubeadmin** 用户的凭证的链接。
- 凭证信息还会输出到 **<installation_directory>/.openshift_install.log**.

重要

不要删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

输出示例

```
...
```

```
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

9.7.10. 使用 CLI 登录集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

其他资源

- 如需有关 访问和了解 OpenShift Container Platform Web 控制台的更多详情，请参阅 访问 Web 控制台。

9.7.11. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于 远程健康监控

9.7.12. 后续步骤

- 自定义集群。
- 如果需要，您可以选择 不使用远程健康报告。

9.8. 在 GCP 上将集群安装到共享 VPC 中

在 OpenShift Container Platform 版本 4.16 中，您可以在 Google Cloud Platform (GCP) 上将集群安装到共享 VPC 中。在这个安装方法中，集群被配置为使用来自不同 GCP 项目的 VPC。共享 VPC 可让组织将资源从多个项目连接到一个通用 VPC 网络。您可以使用来自该网络的内部 IP 地址，在组织内安全、高效地通信。有关共享 VPC 的更多信息，请参阅 GCP 文档中的共享 VPC 概述。

安装程序会置备所需基础架构的其余部分，您可以进一步自定义这些基础架构。要自定义安装，请在安装集群前修改 **install-config.yaml** 文件中的参数。

9.8.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读有关 选择集群安装方法的文档，并为用户准备它。
- 如果使用防火墙，将其配置为允许集群需要访问的站点。
- 您有一个包含共享 VPC 网络的 GCP 主机项目。
- 已将 GCP 项目配置为 托管集群。此项目（称为服务项目）必须附加到主机项目。如需更多信息，请参阅 GCP 文档中的附加服务项目。
- 您有一个 GCP 服务帐户，它在主机和服务项目中都有所需的 GCP 权限。

9.8.2. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。

- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类型的基础架构上执行受限网络安装。在此过程中，您可以下载所需的内容，并使用它为镜像 registry 填充安装软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群前，您要更新镜像 registry 的内容。

9.8.3. 为集群节点 SSH 访问生成密钥对

在 OpenShift Container Platform 安装过程中，您可以为安装程序提供 SSH 公钥。密钥通过它们的 Ignition 配置文件传递给 Red Hat Enterprise Linux CoreOS(RHCOS)节点，用于验证对节点的 SSH 访问。密钥添加到每个节点上 **core** 用户的 **~/.ssh/authorized_keys** 列表中，这将启用免密码身份验证。

将密钥传递给节点后，您可以使用密钥对作为用户 **核心** 通过 SSH 连接到 RHCOS 节点。若要通过 SSH 访问节点，必须由 SSH 为您的本地用户管理私钥身份。

如果要通过 SSH 连接到集群节点来执行安装调试或灾难恢复，则必须在安装过程中提供 SSH 公钥。**./openshift-install gather** 命令还需要在集群节点上设置 SSH 公钥。

重要

不要在生产环境中跳过这个过程，在生产环境中需要灾难恢复和调试。

注意

您必须使用本地密钥，而不是使用特定平台方法配置 的密钥，如 AWS 密钥对。

流程

1. 如果您在本地计算机上没有可用于在集群节点上进行身份验证的现有 SSH 密钥对，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_ed25519**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

注意

如果您计划在 **x86_64**、**ppc64le** 和 **s390x** 架构上安装使用 RHEL 加密库（这些加密库已提交给 NIST 用于 FIPS 140-2/140-3 验证）的 OpenShift Container Platform 集群，则不要创建使用 **ed25519** 算法的密钥。相反，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 查看公共 SSH 密钥：

```
$ cat <path>/<file_name>.pub
```

例如，运行以下命令来查看 **~/.ssh/id_ed25519.pub** 公钥：

```
$ cat ~/.ssh/id_ed25519.pub
```

3. 将 SSH 私钥身份添加到本地用户的 SSH 代理（如果尚未添加）。在集群节点上，或者要使用 **./openshift-install gather** 命令，需要对该密钥进行 SSH 代理管理，才能在集群节点上进行免密码 SSH 身份验证。

注意

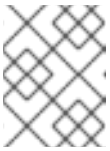
在某些发行版中，自动管理默认 SSH 私钥身份，如 **~/.ssh/id_rsa** 和 **~/.ssh/id_dsa**。

a. 如果 **ssh-agent** 进程尚未为您的本地用户运行，请将其作为后台任务启动：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```



注意

如果集群处于 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

4. 将 SSH 私钥添加到 **ssh-agent** ：

```
$ ssh-add <path>/<file_name> 1
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_ed25519.pub**

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

后续步骤

- 安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

9.8.4. 获取安装程序

在安装 OpenShift Container Platform 前，将安装文件下载到您用于安装的主机上。

先决条件

- 您有一台运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB。

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐户，请使用您的凭证登录。如果没有，请创建一个帐户。

2. 选择您的基础架构供应商。

3. 进入到安装类型的页面，下载与您的主机操作系统和架构对应的安装程序，并将该文件放在您要存储安装配置文件的目录中。

重要

安装程序会在用来安装集群的计算机上创建几个文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。

重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，请为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。此 pull secret 允许您与所含授权机构提供的服务进行身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

9.8.5. 为 GCP 创建安装文件

要在 Google Cloud Platform (GCP) 上安装 OpenShift Container Platform 到共享 VPC 中，您必须生成 **install-config.yaml** 文件并进行修改，以便集群使用正确的 VPC 网络、DNS 区域和项目名称。

9.8.5.1. 手动创建安装配置文件

安装集群要求您手动创建安装配置文件。

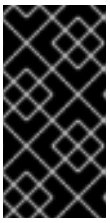
先决条件

- 您在本地机器上有一个 SSH 公钥来提供给安装程序。该密钥将用于在集群节点上进行 SSH 身份验证，以进行调试和灾难恢复。
- 已获取 OpenShift Container Platform 安装程序和集群的 pull secret。

流程

1. 创建一个安装目录来存储所需的安装资产：

```
$ mkdir <installation_directory>
```



重要

您必须创建一个目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

2. 自定义提供的 **install-config.yaml** 文件模板示例，并将其保存在 **<installation_directory>** 中。

注意

此配置文件必须命名为 **install-config.yaml**。

3. 备份 **install-config.yaml** 文件，以便您可以使用它安装多个集群。

重要

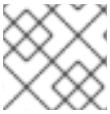
install-config.yaml 文件会在安装过程的下一步中使用。现在必须备份它。

其他资源

- GCP 的安装配置参数

9.8.5.2. 启用屏蔽虚拟机

您可在安装集群时使用 Shielded 虚拟机。Shielded 虚拟机具有额外的安全功能，包括安全引导、固件和完整性监控和 rootkit 检测。如需更多信息，请参阅 Google 文档中有关 Shielded 虚拟机 的文档。



注意

目前，在具有 64 位 ARM 基础架构的集群中不支持 Shielded 虚拟机。

先决条件

- 您已创建了 **install-config.yaml** 文件。

流程

- 在部署集群前，使用文本编辑器编辑 **install-config.yaml** 文件并添加以下部分之一：

a. 仅将屏蔽的虚拟机用于 control plane 机器：

```
controlPlane:
  platform:
    gcp:
      secureBoot: Enabled
```

b. 仅将屏蔽的虚拟机用于计算机器：

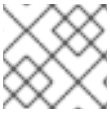
```
compute:
- platform:
    gcp:
      secureBoot: Enabled
```

c. 将屏蔽的虚拟机用于所有机器：

```
platform:
  gcp:
    defaultMachinePlatform:
      secureBoot: Enabled
```

9.8.5.3. 启用机密虚拟机

您可在安装集群时使用机密虚拟机。机密虚拟机在处理数据时加密数据。如需更多信息，请参阅 Google 文档中有关 机密计算的内容。您可以同时启用机密虚拟机和 Shielded 虚拟机，虽然它们不相互依赖。



注意

64 位 ARM 架构目前不支持机密虚拟机。

先决条件

- 您已创建了 **install-config.yaml** 文件。

流程

- 在部署集群前，使用文本编辑器编辑 **install-config.yaml** 文件并添加以下部分之一：

a. 仅将机密虚拟机用于 control plane 机器：

```
controlPlane:
  platform:
    gcp:
      confidentialCompute: Enabled 1
      type: n2d-standard-8 2
      onHostMaintenance: Terminate 3
```

1 启用机密虚拟机。

2 指定支持机密虚拟机的机器类型。机密虚拟机需要 N2D 或 C2D 系列机器类型。有关支持的机器类型的更多信息，请参阅支持的操作系统和机器类型。

3 指定主机维护事件期间虚拟机的行为，如硬件或软件更新。对于使用机密虚拟机的机器，此值必须设置为 **Terminate**，这会停止虚拟机。机密虚拟机不支持实时迁移。

b. 仅将机密虚拟机用于计算机器：

```
compute:
- platform:
    gcp:
      confidentialCompute: Enabled
      type: n2d-standard-8
      onHostMaintenance: Terminate
```

c. 将机密虚拟机用于所有机器：

```
platform:
  gcp:
    defaultMachinePlatform:
      confidentialCompute: Enabled
      type: n2d-standard-8
      onHostMaintenance: Terminate
```

9.8.5.4. 共享 VPC 安装的自定义 install-config.yaml 文件示例

要使用共享 VPC 在 GCP 上安装 OpenShift Container Platform 时需要一些配置参数。以下是显示这些字段的 **install-config.yaml** 文件示例。

重要

此示例 YAML 文件仅供参考。您必须使用正确的环境和集群值修改此文件。

```
apiVersion: v1
baseDomain: example.com
credentialsMode: Passthrough 1
metadata:
  name: cluster_name
platform:
  gcp:
    computeSubnet: shared-vpc-subnet-1 2
    controlPlaneSubnet: shared-vpc-subnet-2 3
    network: shared-vpc 4
    networkProjectID: host-project-name 5
    projectID: service-project-name 6
    region: us-east1
    defaultMachinePlatform:
      tags: 7
      - global-tag1
controlPlane:
  name: master
  platform:
    gcp:
      tags: 8
      - control-plane-tag1
      type: n2-standard-4
      zones:
      - us-central1-a
      - us-central1-c
  replicas: 3
compute:
- name: worker
  platform:
    gcp:
      tags: 9
      - compute-tag1
      type: n2-standard-4
      zones:
      - us-central1-a
      - us-central1-c
  replicas: 3
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
pullSecret: '{"auths": ...}'
sshKey: ssh-ed25519 AAAA... 10
```

1. **credentialsMode** 必须设置为 **Passthrough** 或 **Manual**。如需服务帐户必须具有所需的 GCP 权限，请参阅"先决条件"部分。
2. 共享 VPC 中要使用的计算机器的子网名称。
3. 共享 VPC 中要使用的 control plane 机器的子网名称。
4. 共享 VPC 的名称。
5. 共享 VPC 所在的主机项目的名称。
6. 要安装集群的 GCP 项目的名称。
7. 8. 9. 可选。一个或多个网络标签，应用到计算机器、control plane 机器或所有机器。
10. 您可以选择提供您用来访问集群中机器的 **sshKey** 值。

9.8.5.5. 在安装过程中配置集群范围的代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 **Proxy** 对象的 **spec.noProxy** 字段中添加站点来绕过代理。

注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，**Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
```

```
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle>
```
(5)

(1) 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 **http**。

(2) 用于创建集群外 HTTPS 连接的代理 URL。

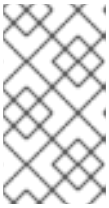
(3) 要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 **.** 以仅匹配子域。例如，**.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代理。

(4) 如果提供，安装程序会在 **openshift-config** 命名空间中生成名为 **user-ca-bundle** 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 **trusted-ca-bundle** 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， **Proxy** 对象的 **trustedCA** 字段中也会引用此配置映射。**additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

(5) 可选：决定 **Proxy** 对象的配置以引用 **trustedCA** 字段中 **user-ca-bundle** 配置映射的策略。允许的值是 **Proxyonly** 和 **Always**。仅在配置了 **http/https** 代理时，使用 **Proxyonly** 引用 **user-ca-bundle** 配置映射。使用 **Always** 始终引用 **user-ca-bundle** 配置映射。默认值为 **Proxyonly**。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。



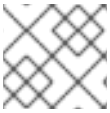
注意

如果安装程序超时，重启并使用安装程序的 **wait-for** 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. 保存该文件并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 cluster 的集群范围代理，该代理 **使用** 提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。



注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

9.8.6. 安装 OpenShift CLI

您可以安装 OpenShift CLI(**oc**)来使用命令行界面与 OpenShift Container Platform 进行交互。您可以在 Linux、Windows 或 macOS 上安装 **oc**。

重要

如果安装了旧版本的 **oc**，则可能无法使用 OpenShift Container Platform 4.16 中的所有命令。下载并安装新版本的 **oc**。

在 Linux 上安装 OpenShift CLI
您可以按照以下流程在 Linux 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **产品变体** 下拉列表中选择架构。
3. 从 **版本** 下拉列表中选择适当的版本。
4. 点 **OpenShift v4.16 Linux Client** 条目旁的 **Download Now** 来保存文件。
5. 解包存档：

```
$ tar xvf <file>
```

6. 将 **oc** 二进制文件放到 **PATH 中的目录中**。
要查看您的 **PATH**，请执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

在 Windows 上安装 OpenShift CLI
您可以按照以下流程在 Windows 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 Windows Client** 条目旁的 **Download Now** 来保存文件。
4. 使用 ZIP 程序解压存档。
5. 将 **oc** 二进制文件移到 **PATH 中的目录中**。
要查看您的 **PATH**，请打开命令提示并执行以下命令：

```
C:\> path
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 macOS Client** 条目旁的 **Download Now** 来保存文件。

注意

对于 macOS arm64，请选择 **OpenShift v4.16 macOS arm64 Client** 条目。

4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
要查看您的 **PATH**，请打开终端并执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

9.8.7. 在 kube-system 项目中存储管理员级别的 secret 的替代方案

默认情况下，管理员 secret 存储在 **kube-system** 项目中。如果您在 **install-config.yaml** 文件中将 **credentialsMode** 参数配置为 **Manual**，则必须使用以下替代方案之一：

- 要手动管理长期云凭证，请按照手动创建长期凭证中的步骤操作。
- 要实现在集群外为各个组件管理的短期凭证，请按照配置 GCP 集群以使用短期凭证中的步骤操作。

9.8.7.1. 手动创建长期凭证

在无法访问云身份和访问管理(IAM)API 的环境中，或者管理员更不希望将管理员级别的凭证 secret 存储在集群 **kube-system** 命名空间中时，可以在安装前将 Cloud Credential Operator(CCO)放入手动模式。

流程

1. 在安装程序使用的 GCP 帐户中添加以下粒度权限：

例 9.43. 所需的 GCP 权限

- compute.machineTypes.list
- compute.regions.list
- compute.zones.list

- dns.changes.create
- dns.changes.get
- dns.managedZones.create
- dns.managedZones.delete
- dns.managedZones.get
- dns.managedZones.list
- dns.networks.bindPrivateDNSZone
- dns.resourceRecordSets.create
- dns.resourceRecordSets.delete
- dns.resourceRecordSets.list

2. 如果您没有将 **install-config.yaml** 配置文件中的 **credentialsMode** 参数设置为 **Manual**，请修改值，如下所示：

配置文件片段示例

```
apiVersion: v1
baseDomain: example.com
credentialsMode: Manual
# ...
```

3. 如果您之前还没有创建安装清单文件，请运行以下命令：

```
$ openshift-install create manifests --dir <installation_directory>
```

其中 **<installation_directory>** 是安装程序在其中创建文件的目录。

4. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

5. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 自定义资源 (CR) 列表：

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
  --included \ 1
  --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
  --to=<path_to_directory_for_credentials_requests> 3
```

1. **--included** 参数仅包含特定集群配置所需的清单。
2. 指定 **install-config.yaml** 文件的位置。

3 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

此命令为每个 **CredentialsRequest** 对象创建一个 YAML 文件。

CredentialsRequest 对象示例

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component_credentials_request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: GCPProviderSpec
    predefinedRoles:
    - roles/storage.admin
    - roles/iam.serviceAccountUser
    skipServiceCheck: true
  ...
```

6. 在之前生成的 **openshift-install** 清单目录中为 secret 创建 YAML 文件。secret 必须使用在 **spec.secretRef** 中为每个 **CredentialsRequest** 定义的命名空间和 secret 名称存储。

带有 secret 的 CredentialsRequest 对象示例

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component_credentials_request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
      ...
  secretRef:
    name: <component_secret>
    namespace: <component_namespace>
  ...
```

Secret 对象示例

```
apiVersion: v1
kind: Secret
metadata:
  name: <component_secret>
  namespace: <component_namespace>
data:
  service_account.json: <base64_encoded_gcp_service_account_file>
```

重要

在升级使用手动维护凭证的集群前，您必须确保 CCO 处于可升级状态。

9.8.7.2. 将 GCP 集群配置为使用短期凭证

要安装配置为使用 GCP Workload Identity 的集群，您必须配置 CCO 实用程序并为集群创建所需的 GCP 资源。

9.8.7.2.1. 配置 Cloud Credential Operator 工具

当 Cloud Credential Operator(CCO)以手动模式运行时，要从集群外部创建和管理云凭证，提取并准备 CCO 实用程序(**ccoctl**)二进制文件。

注意

ccoctl 工具是在 Linux 环境中运行的 Linux 二进制文件。

先决条件

- 您可以访问具有集群管理员权限的 OpenShift Container Platform 帐户。
- 已安装 OpenShift CLI(**oc**)。
- 您已在安装程序使用的 GCP 帐户中添加了以下身份验证选项之一：
  - **IAM Workload Identity Pool Admin**角色。
  - 以下粒度权限：

    例 9.44. 所需的 GCP 权限

    - compute.projects.get
    - iam.googleapis.com/workloadIdentityPoolProviders.create
    - iam.googleapis.com/workloadIdentityPoolProviders.get
    - iam.googleapis.com/workloadIdentityPools.create
    - iam.googleapis.com/workloadIdentityPools.delete
    - iam.googleapis.com/workloadIdentityPools.get
    - iam.googleapis.com/workloadIdentityPools.undelete
    - iam.roles.create
    - iam.roles.delete
    - iam.roles.list
    - iam.roles.undelete
    - iam.roles.update
    - iam.serviceAccounts.create

- iam.serviceAccounts.delete
- iam.serviceAccounts.getIamPolicy
- iam.serviceAccounts.list
- iam.serviceAccounts.setIamPolicy
- iam.workloadIdentityPoolProviders.get
- iam.workloadIdentityPools.delete
- resourcemanager.projects.get
- resourcemanager.projects.getIamPolicy
- resourcemanager.projects.setIamPolicy
- storage.buckets.create
- storage.buckets.delete
- storage.buckets.get
- storage.buckets.getIamPolicy
- storage.buckets.setIamPolicy
- storage.objects.create
- storage.objects.delete
- storage.objects.list

流程

1. 运行以下命令，为 OpenShift Container Platform 发行镜像设置变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. 运行以下命令，从 OpenShift Container Platform 发行镜像获取 CCO 容器镜像：

```
$ CCO_IMAGE=$(oc adm release info --image-for='cloud-credential-operator'
$RELEASE_IMAGE -a ~/.pull-secret)
```

注意

确保 **$RELEASE_IMAGE** 的架构与将使用 **ccoctl** 工具的环境架构相匹配。

3. 运行以下命令，将 CCO 容器镜像中的 **ccoctl** 二进制文件提取到 OpenShift Container Platform 发行镜像中：

```
$ oc image extract $CCO_IMAGE \
  --file="/usr/bin/ccoctl.<rhel_version>" \ 1
  -a ~/.pull-secret
```

1 对于 **<rhel_version>**，请指定与主机使用的 Red Hat Enterprise Linux (RHEL) 版本对应的值。如果没有指定值，则默认使用 **ccoctl.rhel8**。以下值有效：

- **rhel8**: 为使用 RHEL 8 的主机指定这个值。
- **rhel9** ：为使用 RHEL 9 的主机指定这个值。

4. 运行以下命令更改权限以使 **ccoctl** 可执行：

```
$ chmod 775 ccoctl.<rhel_version>
```

验证

- 要验证 **ccoctl** 是否可使用，请运行以下命令来显示帮助文件：

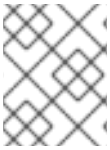
```
$ ccoctl --help
```

ccoctl --help 的输出

```
OpenShift credentials provisioning tool

Usage:
  ccoctl [command]

Available Commands:
  aws         Manage credentials objects for AWS cloud
  azure       Manage credentials objects for Azure
  gcp         Manage credentials objects for Google cloud
  help        Help about any command
  ibmcloud    Manage credentials objects for IBM Cloud
  nutanix     Manage credentials objects for Nutanix

Flags:
  -h, --help   help for ccoctl

Use "ccoctl [command] --help" for more information about a command.
```

9.8.7.2.2. 使用 Cloud Credential Operator 实用程序创建 GCP 资源

您可以使用 **ccoctl gcp create-all** 命令自动创建 GCP 资源。



注意

默认情况下，**ccoctl** 在运行命令的目录中创建对象。要在其他目录中创建对象，请使用 **--output-dir** 标志。此流程使用 **<path_to_ccoctl_output_dir>** 来引用这个目录。

先决条件

您必须：

- 提取并准备好 **ccoctl** 二进制文件。

流程

1. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 对象列表：

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
  --included \(1)
  --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \(2)
  --to=<path_to_directory_for_credentials_requests> (3)
```

(1) **--included** 参数仅包含特定集群配置所需的清单。

(2) 指定 **install-config.yaml** 文件的位置。

(3) 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

注意

此命令可能需要一些时间才能运行。

3. 运行以下命令，使用 **ccoctl** 工具处理所有 **CredentialsRequest** 对象：

```
$ ccoctl gcp create-all \
  --name=<name> \(1)
  --region=<gcp_region> \(2)
  --project=<gcp_project_id> \(3)
  --credentials-requests-dir=<path_to_credentials_requests_directory> (4)
```

(1) 为用于跟踪的所有创建 GCP 资源指定用户定义的名称。

(2) 指定在其中创建云资源的 GCP 区域。

(3) 指定在其中创建云资源的 GCP 项目 ID。

(4) 指定包含 **CredentialsRequest** 清单文件的目录，以创建 GCP 服务帐户。

注意

如果您的集群使用 **TechPreviewNoUpgrade** 功能集启用的技术预览功能，则必须包含 **--enable-tech-preview** 参数。

验证

- 要验证 OpenShift Container Platform secret 是否已创建，列出 **<path_to_ccoctl_output_dir>/manifests** 目录中的文件：

```
$ ls <path_to_ccoctl_output_dir>/manifests
```

输出示例

```
cluster-authentication-02-config.yaml
openshift-cloud-controller-manager-gcp-ccm-cloud-credentials-credentials.yaml
openshift-cloud-credential-operator-cloud-credential-operator-gcp-ro-creds-credentials.yaml
openshift-cloud-network-config-controller-cloud-credentials-credentials.yaml
openshift-cluster-api-capg-manager-bootstrap-credentials-credentials.yaml
openshift-cluster-csi-drivers-gcp-pd-cloud-credentials-credentials.yaml
openshift-image-registry-installer-cloud-credentials-credentials.yaml
openshift-ingress-operator-cloud-credentials-credentials.yaml
openshift-machine-api-gcp-cloud-credentials-credentials.yaml
```

您可以通过查询 GCP 来验证是否已创建 IAM 服务帐户。如需更多信息，请参阅有关列出 IAM 服务帐户的 GCP 文档。

9.8.7.2.3. 整合 Cloud Credential Operator 实用程序清单

要为单个组件在集群外实现短期安全凭证，您必须将创建 Cloud Credential Operator 实用程序 (**ccoctl**) 的清单文件移到安装程序的正确目录中。

先决条件

- 您已使用托管集群的云平台配置了帐户。
- 您已配置了 Cloud Credential Operator 实用程序 (**ccoctl**)。
- 已使用 **ccoctl** 工具创建了集群所需的云供应商资源。

流程

1. 在安装程序使用的 GCP 帐户中添加以下粒度权限：

例 9.45. 所需的 GCP 权限

- compute.machineTypes.list
- compute.regions.list
- compute.zones.list
- dns.changes.create
- dns.changes.get
- dns.managedZones.create
- dns.managedZones.delete
- dns.managedZones.get

- dns.managedZones.list
- dns.networks.bindPrivateDNSZone
- dns.resourceRecordSets.create
- dns.resourceRecordSets.delete
- dns.resourceRecordSets.list

2. 如果您没有将 **install-config.yaml** 配置文件中的 **credentialsMode** 参数设置为 **Manual**，请修改值，如下所示：

   配置文件片段示例

   ```
   apiVersion: v1
   baseDomain: example.com
   credentialsMode: Manual
   # ...
   ```

3. 如果您之前还没有创建安装清单文件，请运行以下命令：

   ```
   $ openshift-install create manifests --dir <installation_directory>
   ```

   其中 **<installation_directory>** 是安装程序在其中创建文件的目录。

4. 运行以下命令，将 **ccoctl** 工具生成的清单复制到安装程序创建的 **manifests** 目录中：

   ```
   $ cp /<path_to_ccoctl_output_dir>/manifests/* ./manifests/
   ```

5. 运行以下命令，将 **ccoctl** 工具在 **tls** 目录中生成的私钥复制到安装目录中：

   ```
   $ cp -a /<path_to_ccoctl_output_dir>/tls .
   ```

9.8.8. 部署集群

您可以在兼容云平台上安装 OpenShift Container Platform。

重要

在初始安装过程中，您只能运行安装程序的 **create cluster** 命令一次。

先决条件

- 您已使用托管集群的云平台配置了帐户。
- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。
- 已确认主机上的云供应商帐户具有部署集群的正确权限。权限不正确的帐户会导致安装过程失败，并显示包括缺失权限的错误消息。

流程

1. 删除所有不使用您为集群配置的 GCP 帐户的服务帐户密钥的现有 GCP 凭证，这些凭证存储在以下位置：

   - **GOOGLE_CREDENTIALS**、**GOOGLE_CLOUD_KEYFILE_JSON** 或 **GCLOUD_KEYFILE_JSON** 环境变量
   - The ~/**.gcp/osServiceAccount.json** file
   - **gcloud cli** 默认凭证

2. 进入包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定自定义 **./install-config.yaml** 文件的位置。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不是 **info**。

3. 可选：您可以减少用来安装集群的服务帐户的权限数量。

   - 如果将 **Owner** 角色分配给您的服务帐户，您可以删除该角色并将其替换为 **Viewer** 角色。
   - 如果包含 **Service Account Key Admin 角色**，您可以将其删除。

验证

当集群部署成功完成时：

- 终端会显示用于访问集群的说明，包括指向 Web 控制台和 **kubeadmin** 用户的凭证的链接。
- 凭证信息还会输出到 **<installation_directory>/.openshift_install.log**.

重要

不要删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

9.8.9. 使用 CLI 登录集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

其他资源

- 如需有关 访问和了解 OpenShift Container Platform Web 控制台的更多详情，请参阅 访问 Web 控制台。

9.8.10. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于 远程健康监控

9.8.11. 后续步骤

- 自定义集群。
- 如果需要，您可以选择 不使用远程健康报告。

9.9. 在 GCP 上安装私有集群

在 OpenShift Container Platform 版本 4.16 中，您可以在 Google Cloud Platform(GCP)上将私有集群安装到现有的 VPC 中。安装程序会置备所需基础架构的其余部分，您可以进一步自定义这些基础架构。要自定义安装，请在安装集群前修改 **install-config.yaml** 文件中的参数。

9.9.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读有关 选择集群安装方法的文档，并为用户准备它。
- 已将 GCP 项目配置为 托管集群。
- 如果使用防火墙，将其配置为允许集群需要访问的站点。

9.9.2. 私有集群

您可以部署不公开外部端点的私有 OpenShift Container Platform 集群。私有集群只能从内部网络访问，且无法在互联网中看到。

默认情况下，OpenShift Container Platform 被置备为使用可公开访问的 DNS 和端点。在部署集群时，私有集群会将 DNS、Ingress Controller 和 API 服务器设置为私有。这意味着集群资源只能从您的内部网络访问，且不能在互联网中看到。

重要

如果集群有任何公共子网，管理员创建的负载均衡器服务可能会公开访问。为确保集群安全性，请验证这些服务是否已明确标注为私有。

要部署私有集群，您必须：

- 使用满足您的要求的现有网络。集群资源可能会在网络上的其他集群间共享。
- 从有权访问的机器中部署：
  - 您置备的云的 API 服务。
  - 您调配的网络上的主机。

- 用于获取安装介质的互联网。

您可以使用符合这些访问要求的机器，并按照您的公司规定进行操作。例如，此机器可以是云网络上的堡垒主机，也可以是可通过 VPN 访问网络的机器。

9.9.2.1. GCP 中的私有集群

要在 Google Cloud Platform(GCP)上创建私有集群，您必须提供一个现有的私有 VPC 和子网来托管集群。安装程序还必须能够解析集群所需的 DNS 记录。安装程序只为内部流量配置 Ingress Operator 和 API 服务器。

集群仍然需要访问互联网来访问 GCP API。

安装私有集群时不需要或创建以下项目：

- 公共子网
- 支持公共入口的公共网络负载均衡器
- 与集群的 **baseDomain** 匹配的公共 DNS 区域

安装程序会使用您指定的 **baseDomain** 来创建私有 DNS 区域以及集群所需的记录。集群被配置，以便 Operator 不会为集群创建公共记录，并将所有集群机器放置在您指定的私有子网中。

由于无法根据源标签限制对外部负载均衡器的访问，私有集群只使用内部负载均衡器来允许对内部实例的访问。

内部负载均衡器依赖于实例组而不是网络负载均衡器使用的目标池。安装程序为每个区创建实例组，即使该组中没有实例。

- 集群 IP 地址仅为内部地址。
- 一个转发规则管理 Kubernetes API 和机器配置服务器端口。
- 后端服务由每个区实例组以及 bootstrap 实例组组成。
- 防火墙使用一条仅基于内部源范围的规则。

9.9.2.1.1. 限制

因为负载均衡器功能不同，机器配置服务器 **/healthz** 没有运行健康检查。两个内部负载均衡器无法共享一个 IP 地址，但两个网络负载均衡器可以共享一个外部 IP 地址。相反，实例的健康状况完全由端口 6443 上的 **/readyz** 检查决定。

9.9.3. 关于使用自定义 VPC

在 OpenShift Container Platform 4.16 中，您可以在 Google Cloud Platform(GCP)上将集群部署到现有的 VPC 中。如果您这样做，还必须使用 VPC 中的现有子网和路由规则。

通过将 OpenShift Container Platform 部署到现有的 GCP VPC 中，您可以避免新帐户中的限制，或者更容易地利用公司所设置的操作限制。如果您无法获得创建 VPC 所需的基础架构创建权限，则可以使用这个选项。

9.9.3.1. 使用 VPC 的要求

安装程序将不再创建以下组件：

- VPC
- 子网
- 云路由器
- Cloud NAT
- NAT IP 地址

如果使用自定义 VPC，您必须为安装程序和集群正确配置它及其子网。安装程序不能为集群分配要使用的网络范围，为子网设置路由表，或者设置类似 DHCP 的 VPC 选项，因此您必须在安装集群前这样做。

您的 VPC 和子网必须满足以下特征：

- VPC 必须位于您将 OpenShift Container Platform 集群部署到的同一 GCP 项目中。
- 要允许 control plane 和计算机器访问互联网，您必须在子网上配置云 NAT 以允许出站数据。这些机器没有公共地址。即使不需要访问互联网，您也必须允许到 VPC 网络的出口，以获取安装程序和镜像。因为不能在共享子网中配置多个云 NAT，所以安装程序无法配置它。

为确保您提供的子网适合，安装程序会确认以下数据：

- 您指定的所有子网都存在，并属于您指定的 VPC。
- 子网 CIDR 属于机器 CIDR。
- 您必须提供一个子网来部署集群 control plane 和计算机器。您可以将相同的子网用于这两种机器类型。

如果您销毁了使用现有 VPC 的集群，则 VPC 不会被删除。

9.9.3.2. 权限划分

从 OpenShift Container Platform 4.3 开始，您不需要安装程序置备的基础架构集群部署所需的所有权限。这与您所在机构可能拥有的权限划分类似：一些个人可以在您的云中创建不同的资源。例如，您可以创建特定于应用程序的对象，如实例、存储桶和负载均衡器，但不能创建与网络相关的组件，如 VPC、子网或 Ingress 规则。

您在创建集群时使用的 GCP 凭证不需要 VPC 和 VPC 中的核心网络组件（如子网、路由表、互联网网关、NAT 和 VPN）所需的网络权限。您仍然需要获取集群中的机器需要的应用程序资源的权限，如负载均衡器、安全组、存储和节点。

9.9.3.3. 集群间隔离

如果您将 OpenShift Container Platform 部署到现有网络中，则集群服务的隔离由防火墙规则保留，该规则根据集群的基础架构 ID 来引用集群中的机器。仅允许集群中的流量。

如果您将多个集群部署到同一个 VPC 中，则以下组件可能会在集群间共享访问权限：

- API，可通过外部发布策略全局可用，或通过内部发布策略在整个网络中可用
- 调试工具，如对机器 CIDR 开放的用于 SSH 和 ICMP 访问的虚拟机实例上的端口

9.9.4. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类型的基础架构上执行受限网络安装。在此过程中，您可以下载所需的内容，并使用它为镜像 registry 填充安装软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群前，您要更新镜像 registry 的内容。

9.9.5. 为集群节点 SSH 访问生成密钥对

在 OpenShift Container Platform 安装过程中，您可以为安装程序提供 SSH 公钥。密钥通过它们的 Ignition 配置文件传递给 Red Hat Enterprise Linux CoreOS(RHCOS)节点，用于验证对节点的 SSH 访问。密钥添加到每个节点上 **core** 用户的 **~/.ssh/authorized_keys** 列表中，这将启用免密码身份验证。

将密钥传递给节点后，您可以使用密钥对作为用户 **核心** 通过 SSH 连接到 RHCOS 节点。若要通过 SSH 访问节点，必须由 SSH 为您的本地用户管理私钥身份。

如果要通过 SSH 连接到集群节点来执行安装调试或灾难恢复，则必须在安装过程中提供 SSH 公钥。**./openshift-install gather** 命令还需要在集群节点上设置 SSH 公钥。

重要

不要在生产环境中跳过这个过程，在生产环境中需要灾难恢复和调试。

注意

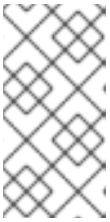
您必须使用本地密钥，而不是使用特定平台方法配置 的密钥，如 AWS 密钥对。

流程

1. 如果您在本地计算机上没有可用于在集群节点上进行身份验证的现有 SSH 密钥对，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_ed25519**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。



注意

如果您计划在 **x86_64**、**ppc64le** 和 **s390x** 架构上安装使用 RHEL 加密库（这些加密库已提交给 NIST 用于 FIPS 140-2/140-3 验证）的 OpenShift Container Platform 集群，则不要创建使用 **ed25519** 算法的密钥。相反，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

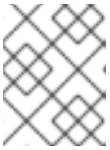
2. 查看公共 SSH 密钥：

```
$ cat <path>/<file_name>.pub
```

例如，运行以下命令来查看 **~/.ssh/id_ed25519.pub** 公钥：

```
$ cat ~/.ssh/id_ed25519.pub
```

3. 将 SSH 私钥身份添加到本地用户的 SSH 代理（如果尚未添加）。在集群节点上，或者要使用 **./openshift-install gather** 命令，需要对该密钥进行 SSH 代理管理，才能在集群节点上进行免密码 SSH 身份验证。



注意

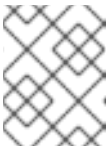
在某些发行版中，自动管理默认 SSH 私钥身份，如 **~/.ssh/id_rsa** 和 **~/.ssh/id_dsa**。

a. 如果 **ssh-agent** 进程尚未为您的本地用户运行，请将其作为后台任务启动：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```



注意

如果集群处于 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

4. 将 SSH 私钥添加到 **ssh-agent** ：

```
$ ssh-add <path>/<file_name> (1)
```

(1) 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_ed25519.pub**

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

后续步骤

- 安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

9.9.6. 获取安装程序

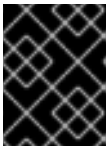
在安装 OpenShift Container Platform 前，将安装文件下载到您用于安装的主机上。

先决条件

- 您有一台运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB。

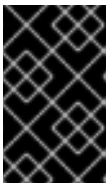
流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐户，请使用您的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入到安装类型的页面，下载与您的主机操作系统和架构对应的安装程序，并将该文件放在您要存储安装配置文件的目录中。



重要

安装程序会在用来安装集群的计算机上创建几个文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。



重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，请为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。此 pull secret 允许您与所含授权机构提供的服务进行身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

9.9.7. 手动创建安装配置文件

安装集群要求您手动创建安装配置文件。

先决条件

- 您在本地机器上有一个 SSH 公钥来提供给安装程序。该密钥将用于在集群节点上进行 SSH 身份验证，以进行调试和灾难恢复。
- 已获取 OpenShift Container Platform 安装程序和集群的 pull secret。

流程

1. 创建一个安装目录来存储所需的安装资产：

```
$ mkdir <installation_directory>
```

重要

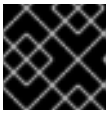
您必须创建一个目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

2. 自定义提供的 **install-config.yaml** 文件模板示例，并将其保存在 **<installation_directory>** 中。

注意

此配置文件必须命名为 **install-config.yaml**。

3. 备份 **install-config.yaml** 文件，以便您可以使用它安装多个集群。



重要

install-config.yaml 文件会在安装过程的下一步中使用。现在必须备份它。

其他资源

- GCP 的安装配置参数

9.9.7.1. 集群安装的最低资源要求

每台集群机器都必须满足以下最低要求：

表 9.20. 最低资源要求

机器	操作系统	vCPU [1]	虚拟内存	Storage	每秒输入/输出 (IOPS) [2]
bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane（控制平面）	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS、RHEL 8.6 及更新版本 [3]	2	8 GB	100 GB	300

1. 当未启用并发多线程(SMT)或超线程时，一个 vCPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比例：（每个内核数的线程）× sockets = vCPU。

2. OpenShift Container Platform 和 Kubernetes 对磁盘性能非常敏感，建议使用更快的存储速度，特别是 control plane 节点上需要 10 ms p99 fsync 持续时间的 etcd。请注意，在许多云平台上，存储大小和 IOPS 可一起扩展，因此您可能需要过度分配存储卷来获取足够的性能。

3. 与所有用户置备的安装一样，如果您选择在集群中使用 RHEL 计算机器，则负责所有操作系统生命周期管理和维护，包括执行系统更新、应用补丁和完成所有其他必要的任务。RHEL 7 计算机器的使用已弃用，并已在 OpenShift Container Platform 4.10 及更新的版本中删除。

注意

从 OpenShift Container Platform 版本 4.13 开始，RHCOS 基于 RHEL 版本 9.2，它更新了微架构要求。以下列表包含每个架构需要的最小指令集架构 (ISA)：

- x86-64 体系结构需要 x86-64-v2 ISA
- ARM64 架构需要 ARMv8.0-A ISA
- IBM Power 架构需要 Power 9 ISA
- s390x 架构需要 z14 ISA

如需更多信息，请参阅 RHEL 架构。

如果平台的实例类型满足集群机器的最低要求，则 OpenShift Container Platform 支持使用它。

其他资源

- 优化存储

9.9.7.2. 为 GCP 测试的实例类型

以下 Google Cloud Platform 实例类型已使用 OpenShift Container Platform 测试。

例 9.46. 机器系列

- **A2**
- **A3**
- **C2**
- **C2D**
- **C3**
- **C3D**
- **E2**
- **M1**
- **N1**
- **N2**
- **N2D**
- **Tau T2D**

9.9.7.3. 在 64 位 ARM 基础架构上为 GCP 测试的实例类型

以下 Google Cloud Platform (GCP) 64 位 ARM 实例类型已使用 OpenShift Container Platform 测试。

例 9.47. 64 位 ARM 机器的机器系列

- **Tau T2A**

9.9.7.4. 使用自定义机器类型

支持使用自定义机器类型来安装 OpenShift Container Platform 集群。

使用自定义机器类型时请考虑以下几点：

- 与预定义的实例类型类似，自定义机器类型必须满足 control plane 和计算机器的最低资源要求。如需更多信息，请参阅"集群安装的资源要求"。
- 自定义机器类型的名称必须遵循以下语法：
  custom-<number_of_cpus>-<amount_of_memory_in_mb>

  例如，**custom-6-20480**。

作为安装过程的一部分，您可以在 **install-config.yaml** 文件中指定自定义机器类型。

带有自定义机器类型的 install-config.yaml 文件示例

```
compute:
- architecture: amd64
  hyperthreading: Enabled
  name: worker
  platform:
    gcp:
      type: custom-6-20480
  replicas: 2
controlPlane:
  architecture: amd64
  hyperthreading: Enabled
  name: master
  platform:
    gcp:
      type: custom-6-20480
  replicas: 3
```

9.9.7.5. 启用屏蔽虚拟机

您可在安装集群时使用 Shielded 虚拟机。Shielded 虚拟机具有额外的安全功能，包括安全引导、固件和完整性监控和 rootkit 检测。如需更多信息，请参阅 Google 文档中有关 Shielded 虚拟机 的文档。

注意

目前，在具有 64 位 ARM 基础架构的集群中不支持 Shielded 虚拟机。

先决条件

- 您已创建了 **install-config.yaml** 文件。

流程

- 在部署集群前，使用文本编辑器编辑 **install-config.yaml** 文件并添加以下部分之一：

  a. 仅将屏蔽的虚拟机用于 control plane 机器：

```
controlPlane:
  platform:
    gcp:
      secureBoot: Enabled
```

  b. 仅将屏蔽的虚拟机用于计算机器：

```
compute:
- platform:
    gcp:
      secureBoot: Enabled
```

  c. 将屏蔽的虚拟机用于所有机器：

```
platform:
  gcp:
    defaultMachinePlatform:
      secureBoot: Enabled
```

9.9.7.6. 启用机密虚拟机

您可在安装集群时使用机密虚拟机。机密虚拟机在处理数据时加密数据。如需更多信息，请参阅 Google 文档中有关 机密计算的内容。您可以同时启用机密虚拟机和 Shielded 虚拟机，虽然它们不相互依赖。

注意

64 位 ARM 架构目前不支持机密虚拟机。

先决条件

- 您已创建了 **install-config.yaml** 文件。

流程

- 在部署集群前，使用文本编辑器编辑 **install-config.yaml** 文件并添加以下部分之一：

  a. 仅将机密虚拟机用于 control plane 机器：

```
controlPlane:
  platform:
    gcp:
      confidentialCompute: Enabled (1)
      type: n2d-standard-8 (2)
      onHostMaintenance: Terminate (3)
```

  1. 启用机密虚拟机。
  2. 指定支持机密虚拟机的机器类型。机密虚拟机需要 N2D 或 C2D 系列机器类型。有关支持的机器类型的更多信息，请参阅支持的操作系统和机器类型。

3 指定主机维护事件期间虚拟机的行为，如硬件或软件更新。对于使用机密虚拟机的机器，此值必须设置为 **Terminate**，这会停止虚拟机。机密虚拟机不支持实时迁移。

b. 仅将机密虚拟机用于计算机器：

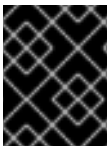
```
compute:
- platform:
    gcp:
      confidentialCompute: Enabled
      type: n2d-standard-8
      onHostMaintenance: Terminate
```

c. 将机密虚拟机用于所有机器：

```
platform:
  gcp:
    defaultMachinePlatform:
      confidentialCompute: Enabled
      type: n2d-standard-8
      onHostMaintenance: Terminate
```

9.9.7.7. GCP 的自定义 install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多详情，或修改所需参数的值。



重要

此示例 YAML 文件仅供参考。您必须使用安装程序来获取 **install-config.yaml** 文件，并进行修改。

```
apiVersion: v1
baseDomain: example.com 1
credentialsMode: Mint 2
controlPlane: 3 4
  hyperthreading: Enabled 5
  name: master
  platform:
    gcp:
      type: n2-standard-4
      zones:
      - us-central1-a
      - us-central1-c
      osDisk:
        diskType: pd-ssd
        diskSizeGB: 1024
        encryptionKey: 6
          kmsKey:
            name: worker-key
            keyRing: test-machine-keys
            location: global
            projectID: project-id
```

```
      tags: 7
      - control-plane-tag1
      - control-plane-tag2
      osImage: 8
        project: example-project-name
        name: example-image-name
  replicas: 3
compute: 9 10
- hyperthreading: Enabled 11
  name: worker
  platform:
    gcp:
      type: n2-standard-4
      zones:
      - us-central1-a
      - us-central1-c
      osDisk:
        diskType: pd-standard
        diskSizeGB: 128
        encryptionKey: 12
          kmsKey:
            name: worker-key
            keyRing: test-machine-keys
            location: global
            projectID: project-id
        tags: 13
        - compute-tag1
        - compute-tag2
        osImage: 14
          project: example-project-name
          name: example-image-name
  replicas: 3
metadata:
  name: test-cluster 15
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 16
  serviceNetwork:
  - 172.30.0.0/16
platform:
  gcp:
    projectID: openshift-production 17
    region: us-central1 18
    defaultMachinePlatform:
      tags: 19
      - global-tag1
      - global-tag2
      osImage: 20
        project: example-project-name
        name: example-image-name
```

```
    network: existing_vpc 21
    controlPlaneSubnet: control_plane_subnet 22
    computeSubnet: compute_subnet 23
pullSecret: '{"auths": ...}' 24
fips: false 25
sshKey: ssh-ed25519 AAAA... 26
publish: Internal 27
```

1 15 17 18 24 必需。安装程序会提示您输入这个值。

2 可选：添加此参数来强制 Cloud Credential Operator (CCO) 使用指定的模式。默认情况下，CCO 使用 **kube-system** 命名空间中的 root 凭证来动态尝试决定凭证的功能。有关 CCO 模式的详情，请参阅*身份验证和授权*指南中的"About the Cloud Credential Operator"部分。

3 9 如果没有提供这些参数和值，安装程序会提供默认值。

4 10 **controlPlane** 部分是一个单个映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，controlPlane **部分** 的第一行则不以连字符开头。仅使用一个 control plane 池。

5 11 是否要启用或禁用并发多线程或 **超线程**。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将 参数值设置为 **Disabled** 来禁用它。如果在某些集群机器中禁用并发多线程，则必须在所有集群机器中禁用它。

重要

如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。如果您禁用并发多线程，请为您的机器使用较大的类型，如 **n1-standard-8**。

6 12 可选：自定义加密密钥部分来加密虚拟机和持久性卷。您的默认计算服务帐户必须具有相应的权限才能使用您的 KMS 密钥并分配了正确的 IAM 角色。默认服务帐户名称遵循 **service-<project_number>@compute-system.iam.gserviceaccount.com** 模式。有关为您的服务帐户授予正确权限的更多信息，请参阅 "Machine management" → "Creating compute machine set" → "Creating a compute machine set on GCP"。

7 13 19 可选：要应用到 control plane 或计算机器集的一组网络标签。**platform.gcp.defaultMachinePlatform.tags** 参数将应用到 control plane 和计算机器。如果设置了 **compute.platform.gcp.tags** 或 **controlPlane.platform.gcp.tags** 参数，它们会覆盖 **platform.gcp.defaultMachinePlatform.tags** 参数。

8 14 20 可选：应该用来引导 control plane 和计算机器的自定义 Red Hat Enterprise Linux CoreOS (RHCOS)。**platform.gcp.defaultMachinePlatform.osImage** 下的 **project** 和 **name** 参数应用到 control plane 和计算机器。如果设置了 **controlPlane.platform.gcp.osImage** 或 **compute.platform.gcp.osImage** 下的 **project** 和 **name** 参数，它们会覆盖 **platform.gcp.defaultMachinePlatform.osImage** 参数。

16 要安装的集群网络插件。默认值 **OVNKubernetes** 是唯一支持的值。

21 指定现有 VPC 的名称。

22 指定要将 control plane 机器部署到的现有子网的名称。该子网必须属于您指定的 VPC。

23 指定要将计算机器部署到的现有子网的名称。该子网必须属于您指定的 VPC。

25

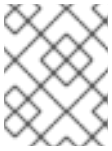
是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes

重要

要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 Red Hat Enterprise Linux (RHEL) 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅在 FIPS 模式中安装该系统。

当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS)时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。

26 您可以选择提供您用来访问集群中机器的 **sshKey** 值。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

27 如何发布集群的面向用户的端点。将 **publish** 设置为 **Internal** 以部署一个私有集群，它不能被互联网访问。默认值为 **External**。

其他资源

- 为计算机器设置启用客户管理的加密密钥

9.9.7.8. 在 GCP 上创建具有全局访问权限的 Ingress Controller

您可以创建一个对 Google Cloud Platform(GCP)集群全局访问权限的 Ingress Controller。只有使用内部负载均衡器的 Ingress Controller 才可使用全局访问。

先决条件

- 已创建 **install-config.yaml**，并完成对其所做的任何修改。

流程

在新 GCP 集群上创建具有全局访问权限的 Ingress Controller。

1. 进入包含安装程序的目录并创建清单文件：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定包含集群的 **install-config.yaml** 文件的目录名称。

2. 在 **<installation_directory>/manifests/** 目录中创建一个名为 **cluster-ingress-default-ingresscontroller.yaml** 的文件：

```
$ touch <installation_directory>/manifests/cluster-ingress-default-ingresscontroller.yaml 1
```

1 对于 **<installation_directory>**，请指定包含集群的 **manifests/** 目录的目录名称。

创建该文件后，几个网络配置文件位于 **manifests/** 目录中，如下所示：

```
$ ls <installation_directory>/manifests/cluster-ingress-default-ingresscontroller.yaml
```

输出示例

```
cluster-ingress-default-ingresscontroller.yaml
```

3. 在编辑器中打开 **cluster-ingress-default-ingresscontroller.yaml** 文件，并输入描述您想要的 Operator 配置的自定义资源(CR)：

到 全局的 clientAccess 配置示例

```
apiVersion: operator.openshift.io/v1
kind: IngressController
metadata:
  name: default
  namespace: openshift-ingress-operator
spec:
  endpointPublishingStrategy:
    loadBalancer:
      providerParameters:
        gcp:
          clientAccess: Global 1
        type: GCP
      scope: Internal 2
    type: LoadBalancerService
```

1. 将 **gcp.clientAccess** 设置为 **Global**。
2. 只有使用内部负载均衡器的 Ingress Controller 才可使用全局访问。

9.9.7.9. 在安装过程中配置集群范围的代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 **Proxy** 对象的 **spec.noProxy** 字段中添加站点来绕过代理。

注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，**Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

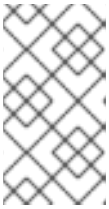
1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 **.** 以仅匹配子域。例如，**.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代理。
4. 如果提供，安装程序会在 **openshift-config** 命名空间中生成名为 **user-ca-bundle** 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 **trusted-ca-bundle** 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， **Proxy** 对象的 **trustedCA** 字段中也会引用此配置映射。**additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。
5. 可选：决定 **Proxy** 对象的配置以引用 **trustedCA** 字段中 **user-ca-bundle** 配置映射的策略。允许的值是 **Proxyonly** 和 **Always**。仅在配置了 **http/https** 代理时，使用 **Proxyonly** 引用 **user-ca-bundle** 配置映射。使用 **Always** 始终引用 **user-ca-bundle** 配置映射。默认值为 **Proxyonly**。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。



注意

如果安装程序超时，重启并使用安装程序的 **wait-for** 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. 保存该文件并在安装 OpenShift Container Platform 时引用。

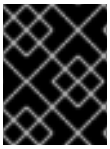
安装程序会创建一个名为 cluster 的集群范围代理，该代理 **使用** 提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

9.9.8. 安装 OpenShift CLI

您可以安装 OpenShift CLI(**oc**)来使用命令行界面与 OpenShift Container Platform 进行交互。您可以在 Linux、Windows 或 macOS 上安装 **oc**。



重要

如果安装了旧版本的 **oc**，则可能无法使用 OpenShift Container Platform 4.16 中的所有命令。下载并安装新版本的 **oc**。

在 Linux 上安装 OpenShift CLI
您可以按照以下流程在 Linux 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **产品变体** 下拉列表中选择架构。
3. 从 **版本** 下拉列表中选择适当的版本。
4. 点 **OpenShift v4.16 Linux Client** 条目旁的 **Download Now** 来保存文件。
5. 解包存档：

```
$ tar xvf <file>
```

6. 将 **oc** 二进制文件放到 **PATH** 中的目录中。
要查看您的 **PATH**，请执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 Windows Client** 条目旁的 **Download Now** 来保存文件。
4. 使用 ZIP 程序解压存档。
5. 将 **oc** 二进制文件移到 **PATH 中的目录中**。
   要查看您的 **PATH**，请打开命令提示并执行以下命令：

```
C:\> path
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 macOS Client** 条目旁的 **Download Now** 来保存文件。

注意

对于 macOS arm64，请选择 **OpenShift v4.16 macOS arm64 Client** 条目。

4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
   要查看您的 **PATH**，请打开终端并执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

9.9.9. 在 kube-system 项目中存储管理员级别的 secret 的替代方案

默认情况下，管理员 secret 存储在 **kube-system** 项目中。如果您在 **install-config.yaml** 文件中将 **credentialsMode** 参数配置为 **Manual**，则必须使用以下替代方案之一：

- 要手动管理长期云凭证，请按照手动创建长期凭证中的步骤操作。
- 要实现在集群外为各个组件管理的短期凭证，请按照配置 GCP 集群以使用短期凭证中的步骤操作。

9.9.9.1. 手动创建长期凭证

在无法访问云身份和访问管理(IAM)API 的环境中，或者管理员更不希望将管理员级别的凭证 secret 存储在集群 **kube-system** 命名空间中时，可以在安装前将 Cloud Credential Operator(CCO)放入手动模式。

流程

1. 在安装程序使用的 GCP 帐户中添加以下粒度权限：

   例 9.48. 所需的 GCP 权限

   - compute.machineTypes.list
   - compute.regions.list
   - compute.zones.list
   - dns.changes.create
   - dns.changes.get
   - dns.managedZones.create
   - dns.managedZones.delete
   - dns.managedZones.get
   - dns.managedZones.list
   - dns.networks.bindPrivateDNSZone
   - dns.resourceRecordSets.create
   - dns.resourceRecordSets.delete
   - dns.resourceRecordSets.list

2. 如果您没有将 **install-config.yaml** 配置文件中的 **credentialsMode** 参数设置为 **Manual**，请修改值，如下所示：

   配置文件片段示例

   ```
   apiVersion: v1
   baseDomain: example.com
   credentialsMode: Manual
   # ...
   ```

3. 如果您之前还没有创建安装清单文件，请运行以下命令：

```
$ openshift-install create manifests --dir <installation_directory>
```

其中 **<installation_directory>** 是安装程序在其中创建文件的目录。

4. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

5. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 自定义资源 (CR) 列表：

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
  --included \ 1
  --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
  --to=<path_to_directory_for_credentials_requests> 3
```

1. **--included** 参数仅包含特定集群配置所需的清单。
2. 指定 **install-config.yaml** 文件的位置。
3. 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

此命令为每个 **CredentialsRequest** 对象创建一个 YAML 文件。

CredentialsRequest 对象示例

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  name: <component_credentials_request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: GCPProviderSpec
    predefinedRoles:
    - roles/storage.admin
    - roles/iam.serviceAccountUser
    skipServiceCheck: true
  ...
```

6. 在之前生成的 **openshift-install** 清单目录中为 secret 创建 YAML 文件。secret 必须使用在 **spec.secretRef** 中为每个 **CredentialsRequest** 定义的命名空间和 secret 名称存储。

带有 secret 的 CredentialsRequest 对象示例

```
apiVersion: cloudcredential.openshift.io/v1
```

```
kind: CredentialsRequest
metadata:
  name: <component_credentials_request>
  namespace: openshift-cloud-credential-operator
  ...
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
      ...
  secretRef:
    name: <component_secret>
    namespace: <component_namespace>
  ...
```

Secret 对象示例

```
apiVersion: v1
kind: Secret
metadata:
  name: <component_secret>
  namespace: <component_namespace>
data:
  service_account.json: <base64_encoded_gcp_service_account_file>
```

重要

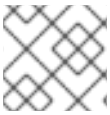
在升级使用手动维护凭证的集群前，您必须确保 CCO 处于可升级状态。

9.9.9.2. 将 GCP 集群配置为使用短期凭证

要安装配置为使用 GCP Workload Identity 的集群，您必须配置 CCO 实用程序并为集群创建所需的 GCP 资源。

9.9.9.2.1. 配置 Cloud Credential Operator 工具

当 Cloud Credential Operator(CCO)以手动模式运行时，要从集群外部创建和管理云凭证，提取并准备 CCO 实用程序(**ccoctl**)二进制文件。



注意

ccoctl 工具是在 Linux 环境中运行的 Linux 二进制文件。

先决条件

- 您可以访问具有集群管理员权限的 OpenShift Container Platform 帐户。
- 已安装 OpenShift CLI(**oc**)。
- 您已在安装程序使用的 GCP 帐户中添加了以下身份验证选项之一：
  - **IAM Workload Identity Pool Admin**角色。
  - 以下粒度权限：

    例 9.49. 所需的 GCP 权限

- compute.projects.get
- iam.googleapis.com/workloadIdentityPoolProviders.create
- iam.googleapis.com/workloadIdentityPoolProviders.get
- iam.googleapis.com/workloadIdentityPools.create
- iam.googleapis.com/workloadIdentityPools.delete
- iam.googleapis.com/workloadIdentityPools.get
- iam.googleapis.com/workloadIdentityPools.undelete
- iam.roles.create
- iam.roles.delete
- iam.roles.list
- iam.roles.undelete
- iam.roles.update
- iam.serviceAccounts.create
- iam.serviceAccounts.delete
- iam.serviceAccounts.getIamPolicy
- iam.serviceAccounts.list
- iam.serviceAccounts.setIamPolicy
- iam.workloadIdentityPoolProviders.get
- iam.workloadIdentityPools.delete
- resourcemanager.projects.get
- resourcemanager.projects.getIamPolicy
- resourcemanager.projects.setIamPolicy
- storage.buckets.create
- storage.buckets.delete
- storage.buckets.get
- storage.buckets.getIamPolicy
- storage.buckets.setIamPolicy
- storage.objects.create
- storage.objects.delete

- storage.objects.list

流程

1. 运行以下命令，为 OpenShift Container Platform 发行镜像设置变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. 运行以下命令，从 OpenShift Container Platform 发行镜像获取 CCO 容器镜像：

```
$ CCO_IMAGE=$(oc adm release info --image-for='cloud-credential-operator'
$RELEASE_IMAGE -a ~/.pull-secret)
```

注意

确保 **$RELEASE_IMAGE** 的架构与将使用 **ccoctl** 工具的环境架构相匹配。

3. 运行以下命令，将 CCO 容器镜像中的 **ccoctl** 二进制文件提取到 OpenShift Container Platform 发行镜像中：

```
$ oc image extract $CCO_IMAGE \
  --file="/usr/bin/ccoctl.<rhel_version>" \ 1
  -a ~/.pull-secret
```

1 对于 **<rhel_version>**，请指定与主机使用的 Red Hat Enterprise Linux (RHEL) 版本对应的值。如果没有指定值，则默认使用 **ccoctl.rhel8**。以下值有效：

- **rhel8**: 为使用 RHEL 8 的主机指定这个值。
- **rhel9** ：为使用 RHEL 9 的主机指定这个值。

4. 运行以下命令更改权限以使 **ccoctl** 可执行：

```
$ chmod 775 ccoctl.<rhel_version>
```

验证

- 要验证 **ccoctl** 是否可使用，请运行以下命令来显示帮助文件：

```
$ ccoctl --help
```

ccoctl --help 的输出

```
OpenShift credentials provisioning tool

Usage:
  ccoctl [command]

Available Commands:
  aws         Manage credentials objects for AWS cloud
```

```
  azure       Manage credentials objects for Azure
  gcp         Manage credentials objects for Google cloud
  help        Help about any command
  ibmcloud    Manage credentials objects for IBM Cloud
  nutanix     Manage credentials objects for Nutanix

Flags:
  -h, --help   help for ccoctl

Use "ccoctl [command] --help" for more information about a command.
```

9.9.9.2.2. 使用 Cloud Credential Operator 实用程序创建 GCP 资源

您可以使用 **ccoctl gcp create-all** 命令自动创建 GCP 资源。

注意

默认情况下，**ccoctl** 在运行命令的目录中创建对象。要在其他目录中创建对象，请使用 **--output-dir** 标志。此流程使用 **<path_to_ccoctl_output_dir>** 来引用这个目录。

先决条件

您必须：

- 提取并准备好 **ccoctl** 二进制文件。

流程

1. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 对象列表：

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
  --included \ 1
  --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
  --to=<path_to_directory_for_credentials_requests> 3
```

1. **--included** 参数仅包含特定集群配置所需的清单。
2. 指定 **install-config.yaml** 文件的位置。
3. 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

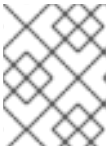
注意

此命令可能需要一些时间才能运行。

3. 运行以下命令，使用 **ccoctl** 工具处理所有 **CredentialsRequest** 对象：

```
$ ccoctl gcp create-all \
  --name=<name> \1
  --region=<gcp_region> \2
  --project=<gcp_project_id> \3
  --credentials-requests-dir=<path_to_credentials_requests_directory> 4
```

1. 为用于跟踪的所有创建 GCP 资源指定用户定义的名称。
2. 指定在其中创建云资源的 GCP 区域。
3. 指定在其中创建云资源的 GCP 项目 ID。
4. 指定包含 **CredentialsRequest** 清单文件的目录，以创建 GCP 服务帐户。



注意

如果您的集群使用 **TechPreviewNoUpgrade** 功能集启用的技术预览功能，则必须包含 **--enable-tech-preview** 参数。

验证

- 要验证 OpenShift Container Platform secret 是否已创建，列出 **<path_to_ccoctl_output_dir>/manifests** 目录中的文件：

```
$ ls <path_to_ccoctl_output_dir>/manifests
```

输出示例

```
cluster-authentication-02-config.yaml
openshift-cloud-controller-manager-gcp-ccm-cloud-credentials-credentials.yaml
openshift-cloud-credential-operator-cloud-credential-operator-gcp-ro-creds-credentials.yaml
openshift-cloud-network-config-controller-cloud-credentials-credentials.yaml
openshift-cluster-api-capg-manager-bootstrap-credentials-credentials.yaml
openshift-cluster-csi-drivers-gcp-pd-cloud-credentials-credentials.yaml
openshift-image-registry-installer-cloud-credentials-credentials.yaml
openshift-ingress-operator-cloud-credentials-credentials.yaml
openshift-machine-api-gcp-cloud-credentials-credentials.yaml
```

您可以通过查询 GCP 来验证是否已创建 IAM 服务帐户。如需更多信息，请参阅有关列出 IAM 服务帐户的 GCP 文档。

9.9.9.2.3. 整合 Cloud Credential Operator 实用程序清单

要为单个组件在集群外实现短期安全凭证，您必须将创建 Cloud Credential Operator 实用程序 (**ccoctl**) 的清单文件移到安装程序的正确目录中。

先决条件

- 您已使用托管集群的云平台配置了帐户。
- 您已配置了 Cloud Credential Operator 实用程序 (**ccoctl**)。

- 已使用 **ccoctl** 工具创建了集群所需的云供应商资源。

流程

1. 在安装程序使用的 GCP 帐户中添加以下粒度权限：

   例 9.50. 所需的 GCP 权限

   - compute.machineTypes.list
   - compute.regions.list
   - compute.zones.list
   - dns.changes.create
   - dns.changes.get
   - dns.managedZones.create
   - dns.managedZones.delete
   - dns.managedZones.get
   - dns.managedZones.list
   - dns.networks.bindPrivateDNSZone
   - dns.resourceRecordSets.create
   - dns.resourceRecordSets.delete
   - dns.resourceRecordSets.list

2. 如果您没有将 **install-config.yaml** 配置文件中的 **credentialsMode** 参数设置为 **Manual**，请修改值，如下所示：

   配置文件片段示例

   ```
   apiVersion: v1
   baseDomain: example.com
   credentialsMode: Manual
   # ...
   ```

3. 如果您之前还没有创建安装清单文件，请运行以下命令：

   ```
   $ openshift-install create manifests --dir <installation_directory>
   ```

   其中 **<installation_directory>** 是安装程序在其中创建文件的目录。

4. 运行以下命令，将 **ccoctl** 工具生成的清单复制到安装程序创建的 **manifests** 目录中：

   ```
   $ cp /<path_to_ccoctl_output_dir>/manifests/* ./manifests/
   ```

5. 运行以下命令，将 **ccoctl** 工具在 **tls** 目录中生成的私钥复制到安装目录中：

```
$ cp -a /<path_to_ccoctl_output_dir>/tls .
```

9.9.10. 部署集群

您可以在兼容云平台上安装 OpenShift Container Platform。

重要

在初始安装过程中，您只能运行安装程序的 **create cluster** 命令一次。

先决条件

- 您已使用托管集群的云平台配置了帐户。
- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。
- 已确认主机上的云供应商帐户具有部署集群的正确权限。权限不正确的帐户会导致安装过程失败，并显示包括缺失权限的错误消息。

流程

1. 删除所有不使用您为集群配置的 GCP 帐户的服务帐户密钥的现有 GCP 凭证，这些凭证存储在以下位置：
   - **GOOGLE_CREDENTIALS**、**GOOGLE_CLOUD_KEYFILE_JSON** 或 **GCLOUD_KEYFILE_JSON** 环境变量
   - The ~/**.gcp/osServiceAccount.json** file
   - **gcloud cli** 默认凭证
2. 进入包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定自定义 **./install-config.yaml** 文件的位置。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不是 **info**。

3. 可选：您可以减少用来安装集群的服务帐户的权限数量。
   - 如果将 **Owner** 角色分配给您的服务帐户，您可以删除该角色并将其替换为 **Viewer** 角色。
   - 如果包含 **Service Account Key Admin** 角色，您可以将其删除。

验证

当集群部署成功完成时：

- 终端会显示用于访问集群的说明，包括指向 Web 控制台和 **kubeadmin** 用户的凭证的链接。
- 凭证信息还会输出到 **<installation_directory>/.openshift_install.log**.

重要

不要删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

9.9.11. 使用 CLI 登录集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

其他资源

- 如需有关 访问和了解 OpenShift Container Platform Web 控制台的更多详情，请参阅 访问 Web 控制台。

9.9.12. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于 远程健康监控

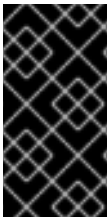
9.9.13. 后续步骤

- 自定义集群。
- 如果需要，您可以选择 不使用远程健康报告。

9.10. 使用 DEPLOYMENT MANAGER 模板在 GCP 中的用户置备的基础架构上安装集群

在 OpenShift Container Platform 版本 4.16 中，您可以使用您提供的基础架构在 Google Cloud Platform(GCP)上安装集群。

此处概述了进行用户提供基础架构安装的步骤。提供的几个 Deployment Manager 模板可帮助完成这些步骤，或者帮助您自行建模。您还可以自由选择通过其他方法创建所需的资源。

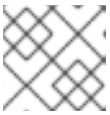


重要

执行用户置备的基础架构安装的步骤仅作为示例。使用您提供的基础架构安装集群需要了解云供应商和 OpenShift Container Platform 的安装过程。提供的几个 Deployment Manager 模板可帮助完成这些步骤，或者帮助您自行建模。您还可以自由选择通过其他方法创建所需的资源；模板仅作示例之用。

9.10.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读有关 选择集群安装方法的文档，并为用户准备它。
- 如果您使用防火墙并计划使用 Telemetry 服务，则将防火墙配置为允许集群需要访问的站点。
- 如果环境中无法访问云身份和访问管理(IAM)API，或者不想将管理员级别的凭证 secret 存储在 **kube-system** 命名空间中，您可以 手动创建和维护 IAM 凭证。



注意

如果要配置代理，请务必查看此站点列表。

9.10.2. 证书签名请求管理

在使用您置备的基础架构时，集群只能有限地访问自动机器管理，因此您必须提供一种在安装后批准集群证书签名请求 (CSR) 的机制。**kube-controller-manager** 只能批准 kubelet 客户端 CSR。 **machine-approver** 无法保证使用 kubelet 凭证请求的提供证书的有效性，因为它不能确认是正确的机器发出了该请求。您必须决定并实施一种方法，以验证 kubelet 提供证书请求的有效性并进行批准。

9.10.3. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类型的基础架构上执行受限网络安装。在此过程中，您可以下载所需的内容，并使用它为镜像 registry 填充安装软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群前，您要更新镜像 registry 的内容。

9.10.4. 配置 GCP 项目

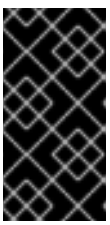
在安装 OpenShift Container Platform 之前，您必须配置 Google Cloud Platform(GCP)项目来托管它。

9.10.4.1. 创建 GCP 项目

要安装 OpenShift Container Platform，您必须在 Google Cloud Platform(GCP)帐户中创建项目来托管集群。

流程

- 创建一个项目来托管 OpenShift Container Platform 集群。请参阅 GCP 文档中的创建和管理 项目。



重要

如果您使用安装程序置备的基础架构，您的 GCP 项目必须使用 Premium Network Service Tier。使用安装程序安装的集群不支持 Standard Network Service Tier。安装程序为 **api-int.<cluster_name>.<base_domain>** URL 配置内部负载均衡；内部负载均衡需要 Premium Tier。

9.10.4.2. 在 GCP 中启用 API 服务

Google Cloud Platform(GCP)项目需要访问多个 API 服务来完成 OpenShift Container Platform 安装。

先决条件

- 已创建一个项目来托管集群。

流程

- 在托管集群的项目中启用以下所需的 API 服务。您还可以启用安装不需要的可选 API 服务。请参阅 GCP 文档中的 启用服务。

表 9.21. 所需的 API 服务

API 服务	控制台服务名称
Compute Engine API	**compute.googleapis.com**
Cloud Resource Manager API	**cloudresourcemanager.googleapis.com**
Google DNS API	**dns.googleapis.com**
IAM Service Account Credentials API	**iamcredentials.googleapis.com**
Identity and Access Management(IAM)API	**iam.googleapis.com**
Service Usage API	**serviceusage.googleapis.com**

表 9.22. 可选 API 服务

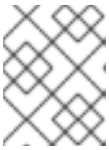
API 服务	控制台服务名称
Cloud Deployment Manager V2 API	**deploymentmanager.googleapis.com**
Google Cloud API	**cloudapis.googleapis.com**
服务管理 API	**servicemanagement.googleapis.com**
Google Cloud Storage JSON API	**storage-api.googleapis.com**
Cloud Storage	**storage-component.googleapis.com**

9.10.4.3. 为 GCP 配置 DNS

要安装 OpenShift Container Platform，您使用的 Google Cloud Platform(GCP)帐户必须在托管 OpenShift Container Platform 集群的同一项目中有一个专用的公共托管区。此区域必须对域具有权威。DNS 服务为集群外部连接提供集群 DNS 解析和名称查询。

流程

1. 确定您的域或子域，以及注册商。您可以转移现有的域和注册商，或通过 GCP 或其他来源获取新的域和注册商。



注意

如果您购买了新的域，则需要时间来传播相关的 DNS 更改。有关通过 Google 购买域的更多信息，请参阅 Google Domains。

2. 在 GCP 项目中为您的域或子域创建一个公共托管区。请参阅 GCP 文档中的 创建公共区。使用适当的根域，如 **openshiftcorp.com 或** 子域，如 cluster **.openshiftcorp.com**。

3. 从托管区域记录中提取新的权威名称服务器。请参阅 GCP 文档中的 查找您的云 DNS 名称服务器。
您通常有四个名称服务器。

4. 更新域所用名称服务器的注册商记录。例如，如果您将域注册到 Google Domains，请参阅 Google Domains 帮助中的以下主题： 如何切换到自定义名称服务器。

5. 如果您将根域迁移到 Google Cloud DNS，请迁移您的 DNS 记录。请参阅 GCP 文档中的 Migrating to Cloud DNS。

6. 如果您使用子域，请按照贵公司的步骤将其委派记录添加到父域。这个过程可能包括对您公司的 IT 部门或控制您公司的根域和 DNS 服务的部门发出的请求。

9.10.4.4. GCP 帐户限值

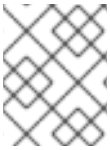
OpenShift Container Platform 集群使用许多 Google Cloud Platform(GCP)组件，但默认的 配额 不会影响您安装默认 OpenShift Container Platform 集群的能力。

默认集群包含三台计算和三台 control plane 机器，它使用以下资源。请注意，有些资源只在 bootstrap 过程中需要，并在集群部署后删除。

表 9.23. 默认集群中使用的 GCP 资源

service	组件	位置	所需的资源总数	bootstrap 后删除的资源
服务帐户	IAM	全局	6	1
防火墙规则	Networking	全局	11	1
转发规则	Compute	全局	2	0
健康检查	Compute	全局	2	0
镜像	Compute	全局	1	0
网络	Networking	全局	1	0
路由器	Networking	全局	1	0
Routes	Networking	全局	2	0

service	组件	位置	所需的资源总数	bootstrap 后删除的资源
子网	Compute	全局	2	0
目标池	Networking	全局	2	0



注意

如果在安装过程中任何配额不足，安装程序会显示一个错误信息，包括超过哪个配额，以及显示区域。

请考虑您的集群的实际大小、预定的集群增长，以及来自与您的帐户关联的其他集群的使用情况。CPU、静态 IP 地址和持久磁盘 SSD（存储）配额是最可能不足的。

如果您计划在以下区域之一部署集群，您将超过最大存储配额，并可能会超过 CPU 配额限制：

- **asia-east2**
- **asia-northeast2**
- **asia-south1**
- **australia-southeast1**
- **europe-north1**
- **europe-west2**
- **europe-west3**
- **europe-west6**
- **northamerica-northeast1**
- **southamerica-east1**
- **us-west2**

您可以从 GCP 控制台 增加资源配额，但可能需要提交一个支持问题单。务必提前规划集群大小，以便在安装 OpenShift Container Platform 集群前有足够的时间来等待支持问题单被处理。

9.10.4.5. 在 GCP 中创建服务帐户

OpenShift Container Platform 需要一个 Google Cloud Platform(GCP)服务帐户，它提供访问 Google API 中数据的验证和授权。如果您没有包含项目中所需角色的现有 IAM 服务帐户，您必须创建一个。

先决条件

- 已创建一个项目来托管集群。

流程

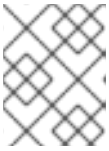
1. 在用于托管 OpenShift Container Platform 集群的项目中创建一个服务帐户。请参阅 GCP 文档中的 创建服务帐户。

2. 为服务帐户授予适当的权限。您可以逐一授予权限，也可以为其分配 **Owner** 角色。请参阅将角色转换到特定资源的服务帐户。

注意

将服务帐户设置为项目的所有者是获取所需权限的最简单方法，这意味着该服务帐户对项目有完全的控制权。您必须确定提供这种能力所带来的风险是否可以接受。

3. 您可以使用 JSON 格式创建服务帐户密钥，或将服务帐户附加到 GCP 虚拟机。请参阅 GCP 文档中的创建服务帐户密钥以及为实例创建并启用服务帐户。
您必须有一个服务帐户密钥或带有附加服务帐户的虚拟机来创建集群。



注意

如果您使用附加服务帐户的虚拟机来创建集群，则必须在安装前在 **install-config.yaml** 文件中设置 **credentialsMode: Manual**。

9.10.4.6. 所需的 GCP 角色

将 **Owner** 角色附加到您创建的服务帐户时，您可以为该服务帐户授予所有权限，包括安装 OpenShift Container Platform 所需的权限。如果机构的安全策略需要更严格的权限集，您可以创建具有以下权限的服务帐户：如果您将集群部署到现有的虚拟私有云 (VPC) 中，则服务帐户不需要某些网络权限，如以下列表中记录：

安装程序所需的角色

- Compute Admin
- Role Administrator
- Security Admin
- Service Account Admin
- Service Account Key Admin
- Service Account User
- Storage Admin

安装过程中创建网络资源所需的角色

- DNS Administrator

在 passthrough 模式中使用 Cloud Credential Operator 所需的角色

- Compute Load Balancer Admin

用户置备的 GCP 基础架构所需的角色

- Deployment Manager Editor

以下角色应用到 control plane 和计算机器使用的服务帐户：

表 9.24. GCP 服务帐户角色

帐户	角色
Control Plane	**roles/compute.instanceAdmin**
	roles/compute.networkAdmin
	roles/compute.securityAdmin
	roles/storage.admin
	roles/iam.serviceAccountUser
Compute	**roles/compute.viewer**
	roles/storage.admin

9.10.4.7. 用户置备的基础架构所需的 GCP 权限

将 **Owner** 角色附加到您创建的服务帐户时，您可以为该服务帐户授予所有权限，包括安装 OpenShift Container Platform 所需的权限。

如果机构的安全策略需要更严格的权限集，您可以创建具有所需权限的自定义角色。用户置备的基础架构需要以下权限来创建和删除 OpenShift Container Platform 集群。

例 9.51. 创建网络资源所需的权限

- **compute.addresses.create**
- **compute.addresses.createInternal**
- **compute.addresses.delete**
- **compute.addresses.get**
- **compute.addresses.list**
- **compute.addresses.use**
- **compute.addresses.useInternal**
- **compute.firewalls.create**
- **compute.firewalls.delete**
- **compute.firewalls.get**
- **compute.firewalls.list**
- **compute.forwardingRules.create**

- **compute.forwardingRules.get**
- **compute.forwardingRules.list**
- **compute.forwardingRules.setLabels**
- **compute.networks.create**
- **compute.networks.get**
- **compute.networks.list**
- **compute.networks.updatePolicy**
- **compute.routers.create**
- **compute.routers.get**
- **compute.routers.list**
- **compute.routers.update**
- **compute.routes.list**
- **compute.subnetworks.create**
- **compute.subnetworks.get**
- **compute.subnetworks.list**
- **compute.subnetworks.use**
- **compute.subnetworks.useExternalIp**

例 9.52. 创建负载均衡器资源所需的权限

- **compute.regionBackendServices.create**
- **compute.regionBackendServices.get**
- **compute.regionBackendServices.list**
- **compute.regionBackendServices.update**
- **compute.regionBackendServices.use**
- **compute.targetPools.addInstance**
- **compute.targetPools.create**
- **compute.targetPools.get**
- **compute.targetPools.list**
- **compute.targetPools.removeInstance**
- **compute.targetPools.use**

例 9.53. 创建 DNS 资源所需的权限

- **dns.changes.create**
- **dns.changes.get**
- **dns.managedZones.create**
- **dns.managedZones.get**
- **dns.managedZones.list**
- **dns.networks.bindPrivateDNSZone**
- **dns.resourceRecordSets.create**
- **dns.resourceRecordSets.list**
- **dns.resourceRecordSets.update**

例 9.54. 创建服务帐户资源所需的权限

- **iam.serviceAccountKeys.create**
- **iam.serviceAccountKeys.delete**
- **iam.serviceAccountKeys.get**
- **iam.serviceAccountKeys.list**
- **iam.serviceAccounts.actAs**
- **iam.serviceAccounts.create**
- **iam.serviceAccounts.delete**
- **iam.serviceAccounts.get**
- **iam.serviceAccounts.list**
- **resourcemanager.projects.get**
- **resourcemanager.projects.getIamPolicy**
- **resourcemanager.projects.setIamPolicy**

例 9.55. 创建计算资源所需的权限

- **compute.disks.create**
- **compute.disks.get**
- **compute.disks.list**

- compute.instanceGroups.create
- compute.instanceGroups.delete
- compute.instanceGroups.get
- compute.instanceGroups.list
- compute.instanceGroups.update
- compute.instanceGroups.use
- compute.instances.create
- compute.instances.delete
- compute.instances.get
- compute.instances.list
- compute.instances.setLabels
- compute.instances.setMetadata
- compute.instances.setServiceAccount
- compute.instances.setTags
- compute.instances.use
- compute.machineTypes.get
- compute.machineTypes.list

例 9.56. 创建存储资源需要

- storage.buckets.create
- storage.buckets.delete
- storage.buckets.get
- storage.buckets.list
- storage.objects.create
- storage.objects.delete
- storage.objects.get
- storage.objects.list

例 9.57. 创建健康检查资源所需的权限

- compute.healthChecks.create

- compute.healthChecks.get
- compute.healthChecks.list
- compute.healthChecks.useReadOnly
- compute.httpHealthChecks.create
- compute.httpHealthChecks.get
- compute.httpHealthChecks.list
- compute.httpHealthChecks.useReadOnly

例 9.58. 获取 GCP 区域和区域相关信息所需的权限

- compute.globalOperations.get
- compute.regionOperations.get
- compute.regions.list
- compute.zoneOperations.get
- compute.zones.get
- compute.zones.list

例 9.59. 检查服务和配额所需的权限

- monitoring.timeSeries.list
- serviceusage.quotas.get
- serviceusage.services.list

例 9.60. 安装所需的 IAM 权限

- iam.roles.get

例 9.61. 安装所需的镜像权限

- compute.images.create
- compute.images.delete
- compute.images.get
- compute.images.list

例 9.62. 运行收集 bootstrap 的可选权限

- **compute.instances.getSerialPortOutput**

例 9.63. 删除网络资源所需的权限

- **compute.addresses.delete**
- **compute.addresses.deleteInternal**
- **compute.addresses.list**
- **compute.firewalls.delete**
- **compute.firewalls.list**
- **compute.forwardingRules.delete**
- **compute.forwardingRules.list**
- **compute.networks.delete**
- **compute.networks.list**
- **compute.networks.updatePolicy**
- **compute.routers.delete**
- **compute.routers.list**
- **compute.routes.list**
- **compute.subnetworks.delete**
- **compute.subnetworks.list**

例 9.64. 删除负载均衡器资源所需的权限

- **compute.regionBackendServices.delete**
- **compute.regionBackendServices.list**
- **compute.targetPools.delete**
- **compute.targetPools.list**

例 9.65. 删除 DNS 资源所需的权限

- **dns.changes.create**
- **dns.managedZones.delete**
- **dns.managedZones.get**

- **dns.managedZones.list**
- **dns.resourceRecordSets.delete**
- **dns.resourceRecordSets.list**

例 9.66. 删除服务帐户资源所需的权限

- **iam.serviceAccounts.delete**
- **iam.serviceAccounts.get**
- **iam.serviceAccounts.list**
- **resourcemanager.projects.getIamPolicy**
- **resourcemanager.projects.setIamPolicy**

例 9.67. 删除计算资源所需的权限

- **compute.disks.delete**
- **compute.disks.list**
- **compute.instanceGroups.delete**
- **compute.instanceGroups.list**
- **compute.instances.delete**
- **compute.instances.list**
- **compute.instances.stop**
- **compute.machineTypes.list**

例 9.68. 删除存储资源需要

- **storage.buckets.delete**
- **storage.buckets.getIamPolicy**
- **storage.buckets.list**
- **storage.objects.delete**
- **storage.objects.list**

例 9.69. 删除健康检查资源所需的权限

- **compute.healthChecks.delete**

- **compute.healthChecks.list**
- **compute.httpHealthChecks.delete**
- **compute.httpHealthChecks.list**

例 9.70. 删除所需的镜像权限

- **compute.images.delete**
- **compute.images.list**

例 9.71. 获取区域相关信息所需的权限

- **compute.regions.get**

例 9.72. 所需的 Deployment Manager 权限

- **deploymentmanager.deployments.create**
- **deploymentmanager.deployments.delete**
- **deploymentmanager.deployments.get**
- **deploymentmanager.deployments.list**
- **deploymentmanager.manifests.get**
- **deploymentmanager.operations.get**
- **deploymentmanager.resources.list**

其他资源

- 优化存储

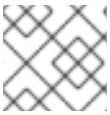
9.10.4.8. 支持的 GCP 区域

您可以将 OpenShift Container Platform 集群部署到以下 Google Cloud Platform(GCP)区域：

- **africa-south1** (Johannesburg, South Africa)
- **asia-east1** (Changhua County, Taiwan)
- **asia-east2** (Hong Kong)
- **asia-northeast1** (Tokyo, Japan)
- **asia-northeast2** (Osaka, Japan)
- **asia-northeast3** (Seoul, South Korea)

- **asia-south1** (Mumbai, India)
- **asia-south2** (Delhi, India)
- **asia-southeast1** (Jurong West, Singapore)
- **asia-southeast2** (Jakarta, Indonesia)
- **aia-southeast1(** Sydney, Australia)
- **australia-southeast2** (Melbourne, Australia)
- **europe-central2** (Warsaw, USA)
- **europe-north1** (Hamina, Finland)
- **europe-southwest1** (Madrid, Spain)
- **europe-west1** (St. Ghislain, Belgium)
- **europe-west2** (London, England, UK)
- **europe-west3** (Frankfurt, Germany)
- **europe-west4** (Eemshaven, Netherlands)
- **europe-west6** (Zürich, Switzerland)
- **europe-west8** (Milan, Italy)
- **europe-west9** (Paris, France)
- **europe-west12** (Turin, Italy)
- **me-central1** (Doha, Qatar, Middle East)
- **me-central2** (Dammam, Saudi Arabia, Middle East)
- **me-west1** (Tel Aviv, Israel)
- **northamerica-northeast1** (Montréal, Québec, Canada)
- **northamerica-northeast2** (Toronto, Ontario, Canada)
- **southamerica-east1** (São Paulo, Brazil)
- **southamerica-west1** (Santiago, Chile)
- **us-central1** (Council Bluffs, Iowa, USA)
- **us-east1** (Moncks Corner, South Carolina, USA)
- **us-east4** (Ashburn, North Virginia, USA)
- **us-east5** (Columbus, Ohio)
- **us-south1** (Dallas, Texas)

- **us-west1** (The Dalles, Oregon, USA)
- **us-west2** (Los Angeles, California, USA)
- **us-west3** (Salt Lake City, Utah, USA)
- **us-west4** (Las Vegas, Nevada, USA)



注意

要确定地区和区域中可以使用哪些机器类型实例，请参阅 Google 文档。

9.10.4.9. 为 GCP 安装和配置 CLI 工具

要使用用户置备的基础架构在 Google Cloud Platform(GCP)上安装 OpenShift Container Platform，您必须为 GCP 安装和配置 CLI 工具。

先决条件

- 已创建一个项目来托管集群。
- 您创建了服务帐户并授予其所需的权限。

流程

1. 在 **$PATH** 中安装以下二进制文件：
   - **gcloud**
   - **gsutil**

   请参阅 GCP 文档中的 安装最新的 Cloud SDK 版本。
2. 使用 **gcloud** 工具及您配置的服务帐户进行身份验证。
   请参阅 GCP 文档中的 使用服务帐户授权。

9.10.5. 具有用户置备基础架构的集群的要求

对于包含用户置备的基础架构的集群，您必须部署所有所需的机器。

本节论述了在用户置备的基础架构上部署 OpenShift Container Platform 的要求。

9.10.5.1. 集群安装所需的机器

最小的 OpenShift Container Platform 集群需要以下主机：

表 9.25. 最低所需的主机

主机	描述
一个临时 bootstrap 机器	集群需要 bootstrap 机器在三台 control plane 机器上部署 OpenShift Container Platform 集群。您可在安装集群后删除 bootstrap 机器。

主机	描述
三台 control plane 机器	control plane 机器运行组成 control plane 的 Kubernetes 和 OpenShift Container Platform 服务。
至少两台计算机器，也称为 worker 机器。	OpenShift Container Platform 用户请求的工作负载在计算机器上运行。

重要

要保持集群的高可用性，请将独立的物理主机用于这些集群机器。

bootstrap 和 control plane 机器必须使用 Red Hat Enterprise Linux CoreOS(RHCOS)作为操作系统。但是，计算机器可以在 Red Hat Enterprise Linux CoreOS(RHCOS)、Red Hat Enterprise Linux(RHEL) 8.6 和更高的版本。

请注意，RHCOS 基于 Red Hat Enterprise Linux(RHEL) 9.2，并继承其所有硬件认证和要求。查看 红帽企业 Linux 技术功能和限制。

9.10.5.2. 集群安装的最低资源要求

每台集群机器都必须满足以下最低要求：

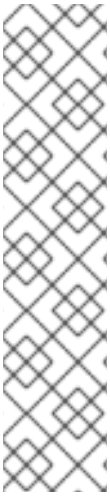
表 9.26. 最低资源要求

机器	操作系统	vCPU [1]	虚拟内存	Storage	每秒输入/输出(IOPS) [2]
bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane（控制平面）	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS、RHEL 8.6 及更新版本 [3]	2	8 GB	100 GB	300

1. 当未启用并发多线程(SMT)或超线程时，一个 vCPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比例：（每个内核数的线程）× sockets = vCPU。

2. OpenShift Container Platform 和 Kubernetes 对磁盘性能非常敏感，建议使用更快的存储速度，特别是 control plane 节点上需要 10 ms p99 fsync 持续时间的 etcd。请注意，在许多云平台上，存储大小和 IOPS 可一起扩展，因此您可能需要过度分配存储卷来获取足够的性能。

3. 与所有用户置备的安装一样，如果您选择在集群中使用 RHEL 计算机器，则负责所有操作系统生命周期管理和维护，包括执行系统更新、应用补丁和完成所有其他必要的任务。RHEL 7 计算机器的使用已弃用，并已在 OpenShift Container Platform 4.10 及更新的版本中删除。



注意

从 OpenShift Container Platform 版本 4.13 开始，RHCOS 基于 RHEL 版本 9.2，它更新了微架构要求。以下列表包含每个架构需要的最小指令集架构 (ISA)：

- x86-64 体系结构需要 x86-64-v2 ISA
- ARM64 架构需要 ARMv8.0-A ISA
- IBM Power 架构需要 Power 9 ISA
- s390x 架构需要 z14 ISA

如需更多信息，请参阅 RHEL 架构。

如果平台的实例类型满足集群机器的最低要求，则 OpenShift Container Platform 支持使用它。

其他资源

- 优化存储

9.10.5.3. 为 GCP 测试的实例类型

以下 Google Cloud Platform 实例类型已使用 OpenShift Container Platform 测试。

例 9.73. 机器系列

- **A2**
- **A3**
- **C2**
- **C2D**
- **C3**
- **C3D**
- **E2**
- **M1**
- **N1**
- **N2**
- **N2D**
- **Tau T2D**

9.10.5.4. 在 64 位 ARM 基础架构上为 GCP 测试的实例类型

以下 Google Cloud Platform (GCP) 64 位 ARM 实例类型已使用 OpenShift Container Platform 测试。

例 9.74. 64 位 ARM 机器的机器系列

- **Tau T2A**

9.10.5.5. 使用自定义机器类型

支持使用自定义机器类型来安装 OpenShift Container Platform 集群。

使用自定义机器类型时请考虑以下几点：

- 与预定义的实例类型类似，自定义机器类型必须满足 control plane 和计算机器的最低资源要求。如需更多信息，请参阅"集群安装的资源要求"。
- 自定义机器类型的名称必须遵循以下语法：
  custom-<number_of_cpus>-<amount_of_memory_in_mb>

  例如，**custom-6-20480**。

9.10.6. 为 GCP 创建安装文件

要使用用户置备的基础架构在 Google Cloud Platform(GCP)上安装 OpenShift Container Platform，您必须生成安装程序部署集群所需的文件，并进行修改，以便集群只创建要使用的机器。您可以生成并自定义 **install-config.yaml** 文件、Kubernetes 清单和 Ignition 配置文件。您还可以选择在安装准备阶段首先设置独立 **var** 分区。

9.10.6.1. 可选：创建独立 /var 分区

建议安装程序将 OpenShift Container Platform 的磁盘分区保留给安装程序。然而，在有些情况下您可能需要在文件系统的一部分中创建独立分区。

OpenShift Container Platform 支持添加单个分区来将存储附加到 **/var** 分区或 **/var** 的子目录中。例如：

- **/var/lib/containers** ：保存随着系统中添加更多镜像和容器而增长的容器相关内容。
- **/var/lib/etcd** ：保存您可能希望独立保留的数据，比如 etcd 存储的性能优化。
- **/var** ：保存您可能希望独立保留的数据，以满足审计等目的。

通过单独存储 **/var** 目录的内容，可以更轻松地根据需要为区域扩展存储，并在以后重新安装 OpenShift Container Platform，并保持该数据的完整性。使用这个方法，您不必再次拉取所有容器，在更新系统时也不必复制大量日志文件。

因为 **/var** 在进行一个全新的 Red Hat Enterprise Linux CoreOS（RHCOS）安装前必需存在，所以这个流程会在 OpenShift Container Platform 安装过程的 **openshift-install** 准备阶段插入一个创建的机器配置清单的机器配置来设置独立的 **/var** 分区。

重要

如果按照以下步骤在此流程中创建独立 **/var** 分区，则不需要再次创建 Kubernetes 清单和 Ignition 配置文件，如本节所述。

流程

1. 创建存放 OpenShift Container Platform 安装文件的目录：

```
$ mkdir $HOME/clusterconfig
```

2. 运行 **openshift-install**，以在 **manifest** 和 **openshift** 子目录中创建一组文件。在系统提示时回答系统问题：

```
$ openshift-install create manifests --dir $HOME/clusterconfig
```

输出示例

```
? SSH Public Key ...
INFO Credentials loaded from the "myprofile" profile in file "/home/myuser/.aws/credentials"
INFO Consuming Install Config from target directory
INFO Manifests created in: $HOME/clusterconfig/manifests and
$HOME/clusterconfig/openshift
```

3. 可选：确认安装程序在 **clusterconfig/openshift** 目录中创建了清单：

```
$ ls $HOME/clusterconfig/openshift/
```

输出示例

```
99_kubeadmin-password-secret.yaml
99_openshift-cluster-api_master-machines-0.yaml
99_openshift-cluster-api_master-machines-1.yaml
99_openshift-cluster-api_master-machines-2.yaml
...
```

4. 创建用于配置额外分区的 Butane 配置。例如，将文件命名为 **$HOME/clusterconfig/98-var-partition.bu**，将磁盘设备名称改为 **worker** 系统上存储设备的名称，并根据情况设置存储大小。这个示例将 **/var** 目录放在一个单独的分区中：

```
variant: openshift
version: 4.16.0
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
storage:
  disks:
  - device: /dev/disk/by-id/<device_name> 1
    partitions:
    - label: var
      start_mib: <partition_start_offset> 2
      size_mib: <partition_size> 3
      number: 5
  filesystems:
    - device: /dev/disk/by-partlabel/var
      path: /var
      format: xfs
      mount_options: [defaults, prjquota] 4
      with_mount_unit: true
```

1 要分区的磁盘的存储设备名称。

2 在引导磁盘中添加数据分区时，推荐最少使用 25000 MiB(Mebibytes)。root 文件系统会自动调整大小以填充所有可用空间（最多到指定的偏移值）。如果没有指定值，或者指定的值小于推荐的最小值，则生成的 root 文件系统会太小，而在以后进行的 RHCOS 重新安装可能会覆盖数据分区的开始部分。

3 以兆字节为单位的数据分区大小。

4 对于用于容器存储的文件系统，必须启用 **prjquota** 挂载选项。

注意

当创建单独的 **/var** 分区时，如果不同的实例类型没有相同的设备名称，则无法为 worker 节点使用不同的实例类型。

5. 从 Butane 配置创建一个清单，并将它保存到 **clusterconfig/openshift** 目录中。例如，运行以下命令：

```
$ butane $HOME/clusterconfig/98-var-partition.bu -o $HOME/clusterconfig/openshift/98-var-partition.yaml
```

6. 再次运行 **openshift-install**，从 **manifest** 和 **openshift** 子目录中的一组文件创建 Ignition 配置：

```
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
auth  bootstrap.ign  master.ign  metadata.json  worker.ign
```

现在，您可以使用 Ignition 配置文件作为安装程序的输入来安装 Red Hat Enterprise Linux CoreOS(RHCOS)系统。

9.10.6.2. 创建安装配置文件

您可以自定义在 Google Cloud Platform(GCP)上安装的 OpenShift Container Platform 集群。

先决条件

- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。

流程

1. 创建 **install-config.yaml** 文件。
   a. 进入包含安装程序的目录并运行以下命令：

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定要存储安装程序创建的文件的目录名称。

在指定目录时：

- 验证该目录是否具有**执行**权限。在安装目录中运行 Terraform 二进制文件需要这个权限。
- 使用空目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

b. 在提示符处，提供云的配置详情：

i. 可选：选择用于访问集群机器的 SSH 密钥。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

ii. 选择 **gcp** 作为目标平台。

iii. 如果您没有为计算机上的 GCP 帐户配置服务帐户密钥，则必须从 GCP 获取它，并粘贴文件的内容或输入文件的绝对路径。

iv. 选择要在其中置备集群的项目 ID。默认值由您配置的服务帐户指定。

v. 选择要将集群部署到的区域。

vi. 选择集群要部署到的基域。基域与您为集群创建的公共 DNS 区对应。

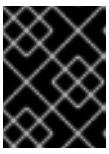
vii. 为集群输入描述性名称。

2. 修改 **install-config.yaml** 文件。您可以在"安装配置参数"部分找到有关可用参数的更多信息。

注意

如果要安装三节点集群，请确保将 **compute.replicas** 参数设置为 **0**。这样可确保集群的 control plane 可以调度。如需更多信息，请参阅"在 GCP 上安装三节点集群"。

3. 备份 **install-config.yaml** 文件，以便您可以使用它安装多个集群。



重要

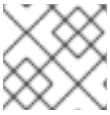
install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用此文件，必须现在备份。

其他资源

- GCP 的安装配置参数

9.10.6.3. 启用屏蔽虚拟机

您可在安装集群时使用 Shielded 虚拟机。Shielded 虚拟机具有额外的安全功能，包括安全引导、固件和完整性监控和 rootkit 检测。如需更多信息，请参阅 Google 文档中有关 Shielded 虚拟机 的文档。



注意

目前，在具有 64 位 ARM 基础架构的集群中不支持 Shielded 虚拟机。

先决条件

- 您已创建了 **install-config.yaml** 文件。

流程

- 在部署集群前，使用文本编辑器编辑 **install-config.yaml** 文件并添加以下部分之一：

  a. 仅将屏蔽的虚拟机用于 control plane 机器：

```
controlPlane:
  platform:
    gcp:
      secureBoot: Enabled
```

  b. 仅将屏蔽的虚拟机用于计算机器：

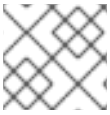
```
compute:
- platform:
    gcp:
      secureBoot: Enabled
```

  c. 将屏蔽的虚拟机用于所有机器：

```
platform:
  gcp:
    defaultMachinePlatform:
      secureBoot: Enabled
```

9.10.6.4. 启用机密虚拟机

您可在安装集群时使用机密虚拟机。机密虚拟机在处理数据时加密数据。如需更多信息，请参阅 Google 文档中有关 机密计算的内容。您可以同时启用机密虚拟机和 Shielded 虚拟机，虽然它们不相互依赖。



注意

64 位 ARM 架构目前不支持机密虚拟机。

先决条件

- 您已创建了 **install-config.yaml** 文件。

流程

- 在部署集群前，使用文本编辑器编辑 **install-config.yaml** 文件并添加以下部分之一：

  a. 仅将机密虚拟机用于 control plane 机器：

```
controlPlane:
  platform:
```

```
gcp:
  confidentialCompute: Enabled 1
  type: n2d-standard-8 2
  onHostMaintenance: Terminate 3
```

1 启用机密虚拟机。

2 指定支持机密虚拟机的机器类型。机密虚拟机需要 N2D 或 C2D 系列机器类型。有关支持的机器类型的更多信息，请参阅支持的操作系统和机器类型。

3 指定主机维护事件期间虚拟机的行为，如硬件或软件更新。对于使用机密虚拟机的机器，此值必须设置为 **Terminate**，这会停止虚拟机。机密虚拟机不支持实时迁移。

b. 仅将机密虚拟机用于计算机器：

```
compute:
- platform:
    gcp:
      confidentialCompute: Enabled
      type: n2d-standard-8
      onHostMaintenance: Terminate
```

c. 将机密虚拟机用于所有机器：

```
platform:
  gcp:
    defaultMachinePlatform:
      confidentialCompute: Enabled
      type: n2d-standard-8
      onHostMaintenance: Terminate
```

9.10.6.5. 在安装过程中配置集群范围的代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 **Proxy** 对象的 **spec.noProxy** 字段中添加站点来绕过代理。

注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，**Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 **.** 以仅匹配子域。例如，**.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代理。
4. 如果提供，安装程序会在 **openshift-config** 命名空间中生成名为 **user-ca-bundle** 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 **trusted-ca-bundle** 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， **Proxy** 对象的 **trustedCA** 字段中也会引用此配置映射。**additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。
5. 可选：决定 **Proxy** 对象的配置以引用 **trustedCA** 字段中 **user-ca-bundle** 配置映射的策略。允许的值是 **Proxyonly** 和 **Always**。仅在配置了 **http/https** 代理时，使用 **Proxyonly** 引用 **user-ca-bundle** 配置映射。使用 **Always** 始终引用 **user-ca-bundle** 配置映射。默认值为 **Proxyonly**。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。

注意

如果安装程序超时，重启并使用安装程序的 **wait-for** 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. 保存该文件并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 cluster 的集群范围代理，该代理 **使用** 提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

9.10.6.6. 创建 Kubernetes 清单和 Ignition 配置文件

由于您必须修改一些集群定义文件并手动启动集群机器，因此您必须生成 Kubernetes 清单和 Ignition 配置文件来配置机器。

安装配置文件转换为 Kubernetes 清单。清单嵌套到 Ignition 配置文件中，稍后用于配置集群机器。

重要

- OpenShift Container Platform 安装程序生成的 Ignition 配置文件包含 24 小时后过期的证书，然后在该时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

先决条件

- 已获得 OpenShift Container Platform 安装程序。
- 已创建 **install-config.yaml** 安装配置文件。

流程

1. 进入包含 OpenShift Container Platform 安装程序的目录，并为集群生成 Kubernetes 清单：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定包含您创建的 **install-config.yaml** 文件的安装目录。

2. 删除定义 control plane 机器的 Kubernetes 清单文件：

```
$ rm -f <installation_directory>/openshift/99_openshift-cluster-api_master-machines-*.yaml
```

通过删除这些文件，您可以防止集群自动生成 control plane 机器。

3. 删除定义 control plane 机器集的 Kubernetes 清单文件：

```
$ rm -f <installation_directory>/openshift/99_openshift-machine-api_master-control-plane-machine-set.yaml
```

4. 可选：如果您不希望集群置备计算机器，请删除定义 worker 机器的 Kubernetes 清单文件：

```
$ rm -f <installation_directory>/openshift/99_openshift-cluster-api_worker-machineset-*.yaml
```

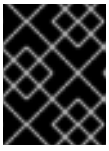
重要

如果在用户置备的基础架构上安装集群时禁用了 **MachineAPI** 功能，则必须删除定义 worker 机器的 Kubernetes 清单文件。否则，集群将无法安装。

由于您要自行创建和管理 worker 机器，因此不需要初始化这些机器。

警告

如果您要安装一个三节点集群，请跳过以下步骤，以便可以调度 control plane 节点。



重要

当您将 control plane 节点从默认的不可调度配置为可以调度时，需要额外的订阅。这是因为 control plane 节点变为计算节点。

5. 检查 **<installation_directory>/manifests/cluster-scheduler-02-config.yml Kubernetes 清单文件中的 mastersSchedulable** 参数是否已设置为 **false**。此设置可防止在 control plane 机器上调度 pod：

   a. 打开 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** 文件。

   b. 找到 **mastersSchedulable** 参数，并确保它被设置为 **false**。

   c. 保存并退出 文件。

6. 可选：如果您不希望 Ingress Operator 代表您创建 DNS 记录，请删除 **<installation_directory>/manifests/cluster-dns-02-config.yml** DNS 配置文件中的 **privateZone 和 publicZone** 部分：

```
apiVersion: config.openshift.io/v1
kind: DNS
metadata:
  creationTimestamp: null
  name: cluster
spec:
  baseDomain: example.openshift.com
  privateZone: 1
    id: mycluster-100419-private-zone
  publicZone: 2
    id: example.openshift.com
status: {}
```

1 2 完全删除此部分。

   如果这样做，您必须在后续步骤中手动添加入口 DNS 记录。

7. 要创建 Ignition 配置文件，请从包含安装程序的目录运行以下命令：

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定相同的安装目录。

为安装目录中的 bootstrap、control plane 和计算节点创建 Ignition 配置文件。**kubeadmin-password** 和 **kubeconfig** 文件在 **./<installation_directory>/auth** 目录中创建：

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

其他资源

- 可选：添加入口 DNS 记录

9.10.7. 导出常用变量

9.10.7.1. 提取基础架构名称

Ignition 配置文件包含一个唯一集群标识符，您可以使用它在 Google Cloud Platform(GCP)中唯一地标识您的集群。基础架构名称还用于在 OpenShift Container Platform 安装过程中定位适当的 GCP 资源。提供的 Deployment Manager 模板包含对此基础架构名称的引用，因此您必须提取它。

先决条件

- 已获取 OpenShift Container Platform 安装程序和集群的 pull secret。
- 已为集群生成 Ignition 配置文件。
- 已安装 **jq** 软件包。

流程

- 要从 Ignition 配置文件元数据中提取和查看基础架构名称，请运行以下命令：

```
$ jq -r .infraID <installation_directory>/metadata.json 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

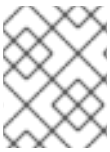
输出示例

```
openshift-vw9j6 1
```

1 此命令的输出是您的集群名称和随机字符串。

9.10.7.2. 为 Deployment Manager 模板导出常用变量

您必须导出与提供的 Deployment Manager 模板一起使用的一组常用变量，它们有助于在 Google Cloud Platform(GCP)上完成用户提供的基础架构安装。



注意

特定的 Deployment Manager 模板可能还需要其他导出变量，这些变量在相关程序中详细介绍。

先决条件

- 获取 OpenShift Container Platform 安装程序和集群的 pull secret。
- 为集群生成 Ignition 配置文件。
- 安装 **jq** 软件包。

流程

1. 导出由提供的 Deployment Manager 模板使用的以下常用变量：

```
$ export BASE_DOMAIN='<base_domain>'
$ export BASE_DOMAIN_ZONE_NAME='<base_domain_zone_name>'
$ export NETWORK_CIDR='10.0.0.0/16'
$ export MASTER_SUBNET_CIDR='10.0.0.0/17'
$ export WORKER_SUBNET_CIDR='10.0.128.0/17'

$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
$ export CLUSTER_NAME=`jq -r .clusterName <installation_directory>/metadata.json`
$ export INFRA_ID=`jq -r .infraID <installation_directory>/metadata.json`
$ export PROJECT_NAME=`jq -r .gcp.projectID <installation_directory>/metadata.json`
$ export REGION=`jq -r .gcp.region <installation_directory>/metadata.json`
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

9.10.8. 在 GCP 中创建 VPC

您必须在 Google Cloud Platform(GCP)中创建一个 VPC，供您的 OpenShift Container Platform 集群使用。您可以自定义 VPC 来满足您的要求。创建 VPC 的一种方法是修改提供的 Deployment Manager 模板。

注意

如果不使用提供的 Deployment Manager 模板来创建 GCP 基础架构，您必须检查提供的信息并手动创建基础架构。如果您的集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 GCP 帐户。
- 为集群生成 Ignition 配置文件。

流程

1. 复制 **VPC 的 Deployment Manager 模板一节中的模板，** 并将它以 **01_vpc.py** 形式保存到计算机上。此模板描述了集群所需的 VPC。

2. 创建 **01_vpc.yaml** 资源定义文件：

```
$ cat <<EOF >01_vpc.yaml
imports:
- path: 01_vpc.py

resources:
- name: cluster-vpc
  type: 01_vpc.py
  properties:
    infra_id: '${INFRA_ID}' 1
    region: '${REGION}' 2
    master_subnet_cidr: '${MASTER_SUBNET_CIDR}' 3
    worker_subnet_cidr: '${WORKER_SUBNET_CIDR}' 4
EOF
```

1. INFRA **_ID 是** 提取步骤 **中的 INFRA_ID** 基础架构名称。
2. **region** 是集群要部署到的区域，如 **us-central1**。
3. **master_subnet_cidr** 是 master 子网的 CIDR，如 **10.0.0.0/17**。
4. **worker_subnet_cidr** 是 worker 子网的 CIDR，例如 **10.0.128.0/17**。

3. 使用 **gcloud** CLI 创建部署：

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-vpc --config 01_vpc.yaml
```

9.10.8.1. VPC 的 Deployment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的 VPC：

例 9.75. 01_VPC.py Deployment Manager 模板

```
def GenerateConfig(context):

  resources = [{
    'name': context.properties['infra_id'] + '-network',
    'type': 'compute.v1.network',
    'properties': {
        'region': context.properties['region'],
        'autoCreateSubnetworks': False
    }
  }, {
    'name': context.properties['infra_id'] + '-master-subnet',
    'type': 'compute.v1.subnetwork',
    'properties': {
        'region': context.properties['region'],
        'network': '$(ref.' + context.properties['infra_id'] + '-network.selfLink)',
```

```
            'ipCidrRange': context.properties['master_subnet_cidr']
        }
    }, {
        'name': context.properties['infra_id'] + '-worker-subnet',
        'type': 'compute.v1.subnetwork',
        'properties': {
            'region': context.properties['region'],
            'network': '$(ref.' + context.properties['infra_id'] + '-network.selfLink)',
            'ipCidrRange': context.properties['worker_subnet_cidr']
        }
    }, {
        'name': context.properties['infra_id'] + '-router',
        'type': 'compute.v1.router',
        'properties': {
            'region': context.properties['region'],
            'network': '$(ref.' + context.properties['infra_id'] + '-network.selfLink)',
            'nats': [{
                'name': context.properties['infra_id'] + '-nat-master',
                'natIpAllocateOption': 'AUTO_ONLY',
                'minPortsPerVm': 7168,
                'sourceSubnetworkIpRangesToNat': 'LIST_OF_SUBNETWORKS',
                'subnetworks': [{
                    'name': '$(ref.' + context.properties['infra_id'] + '-master-subnet.selfLink)',
                    'sourceIpRangesToNat': ['ALL_IP_RANGES']
                }]
            }, {
                'name': context.properties['infra_id'] + '-nat-worker',
                'natIpAllocateOption': 'AUTO_ONLY',
                'minPortsPerVm': 512,
                'sourceSubnetworkIpRangesToNat': 'LIST_OF_SUBNETWORKS',
                'subnetworks': [{
                    'name': '$(ref.' + context.properties['infra_id'] + '-worker-subnet.selfLink)',
                    'sourceIpRangesToNat': ['ALL_IP_RANGES']
                }]
            }]
        }
    }]

    return {'resources': resources}
```

9.10.9. 用户置备的基础架构对网络的要求

所有 Red Hat Enterprise Linux CoreOS(RHCOS)机器都需要在启动时在 **initramfs** 中配置联网，以获取它们的 Ignition 配置文件。

9.10.9.1. 通过 DHCP 设置集群节点主机名

在 Red Hat Enterprise Linux CoreOS(RHCOS)机器上，主机名是通过 NetworkManager 设置的。默认情况下，机器通过 DHCP 获取其主机名。如果主机名不是由 DHCP 提供，请通过内核参数或者其它方法进行静态设置，请通过反向 DNS 查找获取。反向 DNS 查找在网络初始化后进行，可能需要一些时间来解决。其他系统服务可以在此之前启动，并将主机名检测为 **localhost** 或类似的内容。您可以使用 DHCP 为每个集群节点提供主机名来避免这种情况。

另外，通过 DHCP 设置主机名可以绕过实施 DNS split-horizon 的环境中的手动 DNS 记录名称配置错误。

9.10.9.2. 网络连接要求

您必须配置机器之间的网络连接，以允许 OpenShift Container Platform 集群组件进行通信。每台机器都必须能够解析集群中所有其他机器的主机名。

本节详细介绍了所需的端口。

重要

在连接的 OpenShift Container Platform 环境中，所有节点都需要访问互联网才能为平台容器拉取镜像，并向红帽提供遥测数据。

表 9.27. 用于全机器到所有机器通信的端口

协议	port	描述
ICMP	N/A	网络可访问性测试
TCP	**1936**	指标
	9000-**9999**	主机级别的服务，包括端口 9**100**-**9101** 上的节点导出器，以及端口 **9099** 上的 Cluster Version Operator。
	10250-**10259**	Kubernetes 保留的默认端口
UDP	**4789**	VXLAN
	6081	Geneve
	9000-**9999**	主机级别的服务，包括端口 **9100**到**9101** 上的节点导出器。
	500	IPsec IKE 数据包
	4500	IPsec NAT-T 数据包
	123	UDP 端口 **123** 上的网络时间协议 (NTP) 如果配置了外部 NTP 时间服务器，需要打开 UDP 端口 **123**。
TCP/UDP	**30000**-**32767**	Kubernetes 节点端口
ESP	N/A	IPsec Encapsulating Security Payload(ESP)

表 9.28. 用于所有机器控制平面通信的端口

协议	port	描述
TCP	**6443**	Kubernetes API

表 9.29. control plane 机器用于 control plane 机器通信的端口

协议	port	描述
TCP	**2379-2380**	etcd 服务器和对等端口

9.10.10. 在 GCP 中创建负载均衡器

您必须在 Google Cloud Platform(GCP)中配置负载均衡器，供您的 OpenShift Container Platform 集群使用。创建这些组件的一种方法是修改提供的 Deployment Manager 模板。

注意

如果不使用提供的 Deployment Manager 模板来创建 GCP 基础架构，您必须检查提供的信息并手动创建基础架构。如果您的集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 GCP 帐户。
- 为集群生成 Ignition 配置文件。
- 在 GCP 中创建和配置 VPC 及相关子网。

流程

1. 复制 **内部负载均衡器的 Deployment Manager 模板一节中的模板**，并将它以 **02_lb_int.py** 形式保存到计算机上。此模板描述了集群所需的内部负载均衡对象。
2. 对于外部集群，还要复制本主题 **的外部负载均衡器的 Deployment Manager 模板** 一节中的模板，并将它以 **02_lb_ext.py** 形式保存到计算机上。此模板描述了集群所需的外部负载均衡对象。
3. 导出部署模板使用的变量：
   a. 导出集群网络位置：

```
$ export CLUSTER_NETWORK=(`gcloud compute networks describe ${INFRA_ID}-network --format json | jq -r .selfLink`)
```

   b. 导出 control plane 子网位置：

```
$ export CONTROL_SUBNET=(`gcloud compute networks subnets describe ${INFRA_ID}-master-subnet --region=${REGION} --format json | jq -r .selfLink`)
```

   c. 导出集群使用的三个区：

```
$ export ZONE_0=(`gcloud compute regions describe ${REGION} --format=json | jq -r .zones[0] | cut -d "/" -f9`)
```

```
$ export ZONE_1=(`gcloud compute regions describe ${REGION} --format=json | jq -r .zones[1] | cut -d "/" -f9`)
```

```
$ export ZONE_2=(`gcloud compute regions describe ${REGION} --format=json | jq -r .zones[2] | cut -d "/" -f9`)
```

4. 创建 **02_infra.yaml** 资源定义文件：

```
$ cat <<EOF >02_infra.yaml
imports:
- path: 02_lb_ext.py
- path: 02_lb_int.py (1)
resources:
- name: cluster-lb-ext (2)
  type: 02_lb_ext.py
  properties:
    infra_id: '${INFRA_ID}' (3)
    region: '${REGION}' (4)
- name: cluster-lb-int
  type: 02_lb_int.py
  properties:
    cluster_network: '${CLUSTER_NETWORK}'
    control_subnet: '${CONTROL_SUBNET}' (5)
    infra_id: '${INFRA_ID}'
    region: '${REGION}'
    zones: (6)
    - '${ZONE_0}'
    - '${ZONE_1}'
    - '${ZONE_2}'
EOF
```

(1) (2) 仅在部署外部集群时需要。

(3) INFRA **_ID 是** 提取步骤 **中的 INFRA_ID** 基础架构名称。

(4) **region** 是集群要部署到的区域，如 **us-central1**。

(5) **control_subnet** 是控制子网的 URI。

(6) **zones** 是 control plane 实例要部署到的区域，如 **us-east1-b**、**us-east1-c** 和 **us-east1-d**。

5. 使用 **gcloud** CLI 创建部署：

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-infra --config 02_infra.yaml
```

6. 导出集群 IP 地址：

```
$ export CLUSTER_IP=(`gcloud compute addresses describe ${INFRA_ID}-cluster-ip --region=${REGION} --format json | jq -r .address`)
```

7. 对于外部集群，还要导出集群公共 IP 地址：

```
$ export CLUSTER_PUBLIC_IP=(`gcloud compute addresses describe ${INFRA_ID}-cluster-public-ip --region=${REGION} --format json | jq -r .address`)
```

9.10.10.1. 外部负载均衡器的 Deployment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的外部负载均衡器：

例 9.76. 02_lb_ext.py Deployment Manager 模板

```
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-cluster-public-ip',
        'type': 'compute.v1.address',
        'properties': {
            'region': context.properties['region']
        }
    }, {
        # Refer to docs/dev/kube-apiserver-health-check.md on how to correctly setup health check probe for kube-apiserver
        'name': context.properties['infra_id'] + '-api-http-health-check',
        'type': 'compute.v1.httpHealthCheck',
        'properties': {
            'port': 6080,
            'requestPath': '/readyz'
        }
    }, {
        'name': context.properties['infra_id'] + '-api-target-pool',
        'type': 'compute.v1.targetPool',
        'properties': {
            'region': context.properties['region'],
            'healthChecks': ['$(ref.' + context.properties['infra_id'] + '-api-http-health-check.selfLink)'],
            'instances': []
        }
    }, {
        'name': context.properties['infra_id'] + '-api-forwarding-rule',
        'type': 'compute.v1.forwardingRule',
        'properties': {
            'region': context.properties['region'],
            'IPAddress': '$(ref.' + context.properties['infra_id'] + '-cluster-public-ip.selfLink)',
            'target': '$(ref.' + context.properties['infra_id'] + '-api-target-pool.selfLink)',
            'portRange': '6443'
        }
    }]

    return {'resources': resources}
```

9.10.10.2. 内部负载均衡器的 Deployment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的内部负载均衡器：

例 9.77. 02_lb_int.py Deployment Manager 模板

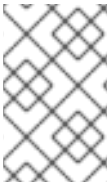
```
def GenerateConfig(context):

    backends = []
    for zone in context.properties['zones']:
        backends.append({
            'group': '$(ref.' + context.properties['infra_id'] + '-master-' + zone + '-ig' + '.selfLink)'
        })

    resources = [{
        'name': context.properties['infra_id'] + '-cluster-ip',
        'type': 'compute.v1.address',
        'properties': {
            'addressType': 'INTERNAL',
            'region': context.properties['region'],
            'subnetwork': context.properties['control_subnet']
        }
    }, {
        # Refer to docs/dev/kube-apiserver-health-check.md on how to correctly setup health check
probe for kube-apiserver
        'name': context.properties['infra_id'] + '-api-internal-health-check',
        'type': 'compute.v1.healthCheck',
        'properties': {
            'httpsHealthCheck': {
                'port': 6443,
                'requestPath': '/readyz'
            },
            'type': "HTTPS"
        }
    }, {
        'name': context.properties['infra_id'] + '-api-internal-backend-service',
        'type': 'compute.v1.regionBackendService',
        'properties': {
            'backends': backends,
            'healthChecks': ['$(ref.' + context.properties['infra_id'] + '-api-internal-health-
check.selfLink)'],
            'loadBalancingScheme': 'INTERNAL',
            'region': context.properties['region'],
            'protocol': 'TCP',
            'timeoutSec': 120
        }
    }, {
        'name': context.properties['infra_id'] + '-api-internal-forwarding-rule',
        'type': 'compute.v1.forwardingRule',
        'properties': {
            'backendService': '$(ref.' + context.properties['infra_id'] + '-api-internal-backend-
service.selfLink)',
            'IPAddress': '$(ref.' + context.properties['infra_id'] + '-cluster-ip.selfLink)',
            'loadBalancingScheme': 'INTERNAL',
            'ports': ['6443','22623'],
            'region': context.properties['region'],
            'subnetwork': context.properties['control_subnet']
```

```
    }
}]

for zone in context.properties['zones']:
    resources.append({
        'name': context.properties['infra_id'] + '-master-' + zone + '-ig',
        'type': 'compute.v1.instanceGroup',
        'properties': {
            'namedPorts': [
                {
                    'name': 'ignition',
                    'port': 22623
                }, {
                    'name': 'https',
                    'port': 6443
                }
            ],
            'network': context.properties['cluster_network'],
            'zone': zone
        }
    })

return {'resources': resources}
```

创建外部集群时，除了 **02_lb_ext.py** 模板外，还需要此模板。

9.10.11. 在 GCP 中创建私有 DNS 区域

您必须在 Google Cloud Platform(GCP)中配置私有 DNS 区，供您的 OpenShift Container Platform 集群使用。创建此组件的一种方法是修改提供的 Deployment Manager 模板。



注意

如果不使用提供的 Deployment Manager 模板来创建 GCP 基础架构，您必须检查提供的信息并手动创建基础架构。如果您的集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 GCP 帐户。
- 为集群生成 Ignition 配置文件。
- 在 GCP 中创建和配置 VPC 及相关子网。

流程

1. 复制 **私有 DNS 的 Deployment Manager 模板** 一节中的模板，并将它以 **02_dns.py** 形式保存到计算机上。此模板描述了集群所需的私有 DNS 对象。
2. 创建 **02_dns.yaml** 资源定义文件：

```
$ cat <<EOF >02_dns.yaml
imports:
```

```
- path: 02_dns.py

resources:
- name: cluster-dns
  type: 02_dns.py
  properties:
    infra_id: '${INFRA_ID}' 1
    cluster_domain: '${CLUSTER_NAME}.${BASE_DOMAIN}' 2
    cluster_network: '${CLUSTER_NETWORK}' 3
EOF
```

1. INFRA _ID 是 提取步骤 中的 **INFRA_ID** 基础架构名称。
2. **cluster_domain** 是集群的域，如 **openshift.example.com**。
3. **cluster_network** 是集群网络的 **selfLink** URL。

3. 使用 **gcloud** CLI 创建部署：

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-dns --config 02_dns.yaml
```

4. 由于 Deployment Manager 的限制，模板不会创建 DNS 条目，因此您必须手动创建它们：

   a. 添加内部 DNS 条目：

```
$ if [ -f transaction.yaml ]; then rm transaction.yaml; fi
$ gcloud dns record-sets transaction start --zone ${INFRA_ID}-private-zone
$ gcloud dns record-sets transaction add ${CLUSTER_IP} --name
api.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 60 --type A --zone ${INFRA_ID}-
private-zone
$ gcloud dns record-sets transaction add ${CLUSTER_IP} --name api-
int.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 60 --type A --zone ${INFRA_ID}-
private-zone
$ gcloud dns record-sets transaction execute --zone ${INFRA_ID}-private-zone
```

   b. 对于外部集群，还要添加外部 DNS 条目：

```
$ if [ -f transaction.yaml ]; then rm transaction.yaml; fi
$ gcloud dns record-sets transaction start --zone ${BASE_DOMAIN_ZONE_NAME}
$ gcloud dns record-sets transaction add ${CLUSTER_PUBLIC_IP} --name
api.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 60 --type A --zone
${BASE_DOMAIN_ZONE_NAME}
$ gcloud dns record-sets transaction execute --zone ${BASE_DOMAIN_ZONE_NAME}
```

9.10.11.1. 私有 DNS 的 Deployment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的私有 DNS：

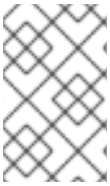
例 9.78. 02_DNS.py Deployment Manager 模板

```
def GenerateConfig(context):
```

```
resources = [{
    'name': context.properties['infra_id'] + '-private-zone',
    'type': 'dns.v1.managedZone',
    'properties': {
        'description': '',
        'dnsName': context.properties['cluster_domain'] + '.',
        'visibility': 'private',
        'privateVisibilityConfig': {
            'networks': [{
                'networkUrl': context.properties['cluster_network']
            }]
        }
    }
}]

return {'resources': resources}
```

9.10.12. 在 GCP 中创建防火墙规则

您必须在 Google Cloud Platform(GCP)中创建防火墙规则，供您的 OpenShift Container Platform 集群使用。创建这些组件的一种方法是修改提供的 Deployment Manager 模板。



注意

如果不使用提供的 Deployment Manager 模板来创建 GCP 基础架构，您必须检查提供的信息并手动创建基础架构。如果您的集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 GCP 帐户。
- 为集群生成 Ignition 配置文件。
- 在 GCP 中创建和配置 VPC 及相关子网。

流程

1. 复制 **防火墙规则的 Deployment Manager 模板一节中的模板**，并将它以 **03_firewall.py** 形式保存到计算机上。此模板描述了集群所需的安全组。
2. 创建 **03_firewall.yaml** 资源定义文件：

```
$ cat <<EOF >03_firewall.yaml
imports:
- path: 03_firewall.py

resources:
- name: cluster-firewall
  type: 03_firewall.py
  properties:
    allowed_external_cidr: '0.0.0.0/0' (1)
    infra_id: '${INFRA_ID}' (2)
```

```
  cluster_network: '${CLUSTER_NETWORK}' (3)
  network_cidr: '${NETWORK_CIDR}' (4)
EOF
```

1. **allowed_external_cidr** 是 CIDR 范围，可访问集群 API 和 SSH 到 bootstrap 主机。对于内部集群，将此值设置为 **${NETWORK_CIDR}**。
2. INFRA **_ID 是** 提取步骤 **中的 INFRA_ID** 基础架构名称。
3. **cluster_network** 是集群网络的 **selfLink** URL。
4. **NETWORK_CIDR 是** VPC 网络的 CIDR，如 **10.0.0.0/16**。

3. 使用 **gcloud** CLI 创建部署：

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-firewall --config
03_firewall.yaml
```

9.10.12.1. 防火墙规则的 Deployment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的防火墙破坏：

例 9.79. 03_firewall.py Deployment Manager 模板

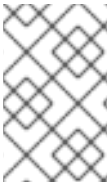
```
def GenerateConfig(context):

  resources = [{
    'name': context.properties['infra_id'] + '-bootstrap-in-ssh',
    'type': 'compute.v1.firewall',
    'properties': {
      'network': context.properties['cluster_network'],
      'allowed': [{
        'IPProtocol': 'tcp',
        'ports': ['22']
      }],
      'sourceRanges': [context.properties['allowed_external_cidr']],
      'targetTags': [context.properties['infra_id'] + '-bootstrap']
    }
  }, {
    'name': context.properties['infra_id'] + '-api',
    'type': 'compute.v1.firewall',
    'properties': {
      'network': context.properties['cluster_network'],
      'allowed': [{
        'IPProtocol': 'tcp',
        'ports': ['6443']
      }],
      'sourceRanges': [context.properties['allowed_external_cidr']],
      'targetTags': [context.properties['infra_id'] + '-master']
    }
  }, {
    'name': context.properties['infra_id'] + '-health-checks',
    'type': 'compute.v1.firewall',
```

```
        'properties': {
            'network': context.properties['cluster_network'],
            'allowed': [{
                'IPProtocol': 'tcp',
                'ports': ['6080', '6443', '22624']
            }],
            'sourceRanges': ['35.191.0.0/16', '130.211.0.0/22', '209.85.152.0/22', '209.85.204.0/22'],
            'targetTags': [context.properties['infra_id'] + '-master']
        }
    }, {
        'name': context.properties['infra_id'] + '-etcd',
        'type': 'compute.v1.firewall',
        'properties': {
            'network': context.properties['cluster_network'],
            'allowed': [{
                'IPProtocol': 'tcp',
                'ports': ['2379-2380']
            }],
            'sourceTags': [context.properties['infra_id'] + '-master'],
            'targetTags': [context.properties['infra_id'] + '-master']
        }
    }, {
        'name': context.properties['infra_id'] + '-control-plane',
        'type': 'compute.v1.firewall',
        'properties': {
            'network': context.properties['cluster_network'],
            'allowed': [{
                'IPProtocol': 'tcp',
                'ports': ['10257']
            },{
                'IPProtocol': 'tcp',
                'ports': ['10259']
            },{
                'IPProtocol': 'tcp',
                'ports': ['22623']
            }],
            'sourceTags': [
                context.properties['infra_id'] + '-master',
                context.properties['infra_id'] + '-worker'
            ],
            'targetTags': [context.properties['infra_id'] + '-master']
        }
    }, {
        'name': context.properties['infra_id'] + '-internal-network',
        'type': 'compute.v1.firewall',
        'properties': {
            'network': context.properties['cluster_network'],
            'allowed': [{
                'IPProtocol': 'icmp'
            },{
                'IPProtocol': 'tcp',
                'ports': ['22']
            }],
            'sourceRanges': [context.properties['network_cidr']],
            'targetTags': [
                context.properties['infra_id'] + '-master',
```

```
                context.properties['infra_id'] + '-worker'
            ]
        }
    }, {
        'name': context.properties['infra_id'] + '-internal-cluster',
        'type': 'compute.v1.firewall',
        'properties': {
            'network': context.properties['cluster_network'],
            'allowed': [{
                'IPProtocol': 'udp',
                'ports': ['4789', '6081']
            },{
                'IPProtocol': 'udp',
                'ports': ['500', '4500']
            },{
                'IPProtocol': 'esp',
            },{
                'IPProtocol': 'tcp',
                'ports': ['9000-9999']
            },{
                'IPProtocol': 'udp',
                'ports': ['9000-9999']
            },{
                'IPProtocol': 'tcp',
                'ports': ['10250']
            },{
                'IPProtocol': 'tcp',
                'ports': ['30000-32767']
            },{
                'IPProtocol': 'udp',
                'ports': ['30000-32767']
            }],
            'sourceTags': [
                context.properties['infra_id'] + '-master',
                context.properties['infra_id'] + '-worker'
            ],
            'targetTags': [
                context.properties['infra_id'] + '-master',
                context.properties['infra_id'] + '-worker'
            ]
        }
    }]

    return {'resources': resources}
```

9.10.13. 在 GCP 中创建 IAM 角色

您必须在 Google Cloud Platform(GCP)中创建 IAM 角色，供您的 OpenShift Container Platform 集群使用。创建这些组件的一种方法是修改提供的 Deployment Manager 模板。



注意

如果不使用提供的 Deployment Manager 模板来创建 GCP 基础架构，您必须检查提供的信息并手动创建基础架构。如果您的集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 GCP 帐户。
- 为集群生成 Ignition 配置文件。
- 在 GCP 中创建和配置 VPC 及相关子网。

流程

1. 复制 **IAM 角色的 Deployment Manager 模板一节中的模板**，并将它以 **03_iam.py** 形式保存到计算机上。此模板描述了集群所需的 IAM 角色。
2. 创建 **03_iam.yaml** 资源定义文件：

```
$ cat <<EOF >03_iam.yaml
imports:
- path: 03_iam.py
resources:
- name: cluster-iam
  type: 03_iam.py
  properties:
    infra_id: '${INFRA_ID}' (1)
EOF
```

(1) INFRA _ID 是 提取步骤 中的 INFRA_ID 基础架构名称。

3. 使用 **gcloud** CLI 创建部署：

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-iam --config 03_iam.yaml
```

4. 导出 master 服务帐户的变量：

```
$ export MASTER_SERVICE_ACCOUNT=(`gcloud iam service-accounts list --filter
"email~^${INFRA_ID}-m@${PROJECT_NAME}." --format json | jq -r '.[0].email'`)
```

5. 导出 worker 服务帐户的变量：

```
$ export WORKER_SERVICE_ACCOUNT=(`gcloud iam service-accounts list --filter
"email~^${INFRA_ID}-w@${PROJECT_NAME}." --format json | jq -r '.[0].email'`)
```

6. 导出托管计算机器的子网的变量：

```
$ export COMPUTE_SUBNET=(`gcloud compute networks subnets describe ${INFRA_ID}-
worker-subnet --region=${REGION} --format json | jq -r .selfLink`)
```

7. 由于 Deployment Manager 的限制，模板不会创建策略绑定，因此您必须手动创建它们：

```
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/compute.instanceAdmin"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/compute.networkAdmin"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/compute.securityAdmin"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/iam.serviceAccountUser"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/storage.admin"

$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${WORKER_SERVICE_ACCOUNT}" --role "roles/compute.viewer"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${WORKER_SERVICE_ACCOUNT}" --role "roles/storage.admin"
```

8. 创建服务帐户密钥并将其存储在本地以供以后使用：

```
$ gcloud iam service-accounts keys create service-account-key.json --iam-
account=${MASTER_SERVICE_ACCOUNT}
```

9.10.13.1. IAM 角色的 Deployment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的 IAM 角色：

例 9.80. 03_IAM.py Deployment Manager 模板

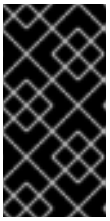
```
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-master-node-sa',
        'type': 'iam.v1.serviceAccount',
        'properties': {
            'accountId': context.properties['infra_id'] + '-m',
            'displayName': context.properties['infra_id'] + '-master-node'
        }
    }, {
        'name': context.properties['infra_id'] + '-worker-node-sa',
        'type': 'iam.v1.serviceAccount',
        'properties': {
            'accountId': context.properties['infra_id'] + '-w',
            'displayName': context.properties['infra_id'] + '-worker-node'
        }
    }]

    return {'resources': resources}
```

9.10.14. 为 GCP 基础架构创建 RHCOS 集群镜像

您必须对 OpenShift Container Platform 节点的 Google Cloud Platform(GCP)使用有效的 Red Hat Enterprise Linux CoreOS(RHCOS)镜像。

流程

1. 从 RHCOS 镜像镜像页面获取 RHCOS 镜像。



重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每个发行版本而改变。您必须下载最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。如果可用，请使用与 OpenShift Container Platform 版本匹配的镜像版本。

文件名包含 OpenShift Container Platform 版本号，格式为 **rhcos-<version>-<arch>-gcp.<arch>.tar.gz**。

2. 创建 Google 存储桶：

```
$ gsutil mb gs://<bucket_name>
```

3. 将 RHCOS 镜像上传到 Google 存储桶：

```
$ gsutil cp <downloaded_image_file_path>/rhcos-<version>-x86_64-gcp.x86_64.tar.gz
gs://<bucket_name>
```

4. 将上传的 RHCOS 镜像位置导出为变量：

```
$ export IMAGE_SOURCE=gs://<bucket_name>/rhcos-<version>-x86_64-gcp.x86_64.tar.gz
```

5. 创建集群镜像：

```
$ gcloud compute images create "${INFRA_ID}-rhcos-image" \
    --source-uri="${IMAGE_SOURCE}"
```

9.10.15. 在 GCP 中创建 bootstrap 机器

您必须在 Google Cloud Platform(GCP)中创建 bootstrap 机器，以便在 OpenShift Container Platform 集群初始化过程中使用。创建此机器的一种方法是修改提供的 Deployment Manager 模板。

注意

如果不使用提供的 Deployment Manager 模板来创建 bootstrap 机器，您必须检查提供的信息并手动创建基础架构。如果您的集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 GCP 帐户。
- 为集群生成 Ignition 配置文件。
- 在 GCP 中创建和配置 VPC 及相关子网。
- 在 GCP 中创建和配置联网与负载均衡器。

- 创建 control plane 和计算角色。
- 确保已安装 pyOpenSSL。

流程

1. 复制 **bootstrap 机器的 Deployment Manager 模板一节中的模板**，并将它以 **04_bootstrap.py** 形式保存到计算机上。此模板描述了集群所需的 bootstrap 机器。
2. 导出安装程序所需的 Red Hat Enterprise Linux CoreOS(RHCOS)镜像的位置：

```
$ export CLUSTER_IMAGE=(`gcloud compute images describe ${INFRA_ID}-rhcos-image --format json | jq -r .selfLink`)
```

3. 创建存储桶并上传 **bootstrap.ign** 文件：

```
$ gsutil mb gs://${INFRA_ID}-bootstrap-ignition
```

```
$ gsutil cp <installation_directory>/bootstrap.ign gs://${INFRA_ID}-bootstrap-ignition/
```

4. 为 bootstrap 实例创建一个签名的 URL，用于访问 Ignition 配置。将输出中的 URL 导出为变量：

```
$ export BOOTSTRAP_IGN=`gsutil signurl -d 1h service-account-key.json gs://${INFRA_ID}-bootstrap-ignition/bootstrap.ign | grep "^gs:" | awk '{print $5}'`
```

5. 创建 **04_bootstrap.yaml** 资源定义文件：

```
$ cat <<EOF >04_bootstrap.yaml
imports:
- path: 04_bootstrap.py

resources:
- name: cluster-bootstrap
  type: 04_bootstrap.py
  properties:
    infra_id: '${INFRA_ID}' 1
    region: '${REGION}' 2
    zone: '${ZONE_0}' 3

    cluster_network: '${CLUSTER_NETWORK}' 4
    control_subnet: '${CONTROL_SUBNET}' 5
    image: '${CLUSTER_IMAGE}' 6
    machine_type: 'n1-standard-4' 7
    root_volume_size: '128' 8

    bootstrap_ign: '${BOOTSTRAP_IGN}' 9
EOF
```

1. INFRA **_ID 是** 提取步骤 **中的 INFRA_ID** 基础架构名称。
2. **region** 是集群要部署到的区域，如 **us-central1**。
3. **zone** 是 bootstrap 实例要部署到的区域，如 **us-central1-b**。

4. **cluster_network** 是集群网络的 **selfLink** URL。
5. **control_subnet** 是控制子网的 **selfLink** URL。
6. **image** 是 RHCOS 镜像的 **selfLink** URL。
7. **machine_type** 是实例的机器类型，如 **n1-standard-4**。
8. **root_volume_size** 是 bootstrap 计算机的引导磁盘大小。
9. **bootstrap_ign** 是创建签名 URL 时的 URL 输出。

6. 使用 **gcloud** CLI 创建部署：

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-bootstrap --config
04_bootstrap.yaml
```

7. 由于 Deployment Manager 的限制，模板无法管理负载均衡器成员资格，因此您必须手动添加 bootstrap 机器。
   a. 将 bootstrap 实例添加到内部负载均衡器实例组中：

```
$ gcloud compute instance-groups unmanaged add-instances \
    ${INFRA_ID}-bootstrap-ig --zone=${ZONE_0} --instances=${INFRA_ID}-bootstrap
```

   b. 将 bootstrap 实例组添加到内部负载均衡器后端服务中：

```
$ gcloud compute backend-services add-backend \
    ${INFRA_ID}-api-internal-backend-service --region=${REGION} --instance-
group=${INFRA_ID}-bootstrap-ig --instance-group-zone=${ZONE_0}
```

9.10.15.1. bootstrap 机器的 Deployment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的 bootstrap 机器：

例 9.81. 04_bootstrap.py Deployment Manager 模板

```
def GenerateConfig(context):

  resources = [{
    'name': context.properties['infra_id'] + '-bootstrap-public-ip',
    'type': 'compute.v1.address',
    'properties': {
      'region': context.properties['region']
    }
  }, {
    'name': context.properties['infra_id'] + '-bootstrap',
    'type': 'compute.v1.instance',
    'properties': {
      'disks': [{
        'autoDelete': True,
        'boot': True,
        'initializeParams': {
```

```
                'diskSizeGb': context.properties['root_volume_size'],
                'sourceImage': context.properties['image']
            }
        }],
        'machineType': 'zones/' + context.properties['zone'] + '/machineTypes/' +
context.properties['machine_type'],
        'metadata': {
            'items': [{
                'key': 'user-data',
                'value': '{"ignition":{"config":{"replace":{"source":"' + context.properties['bootstrap_ign']
+ '"}},"version":"3.2.0"}}',
            }]
        },
        'networkInterfaces': [{
            'subnetwork': context.properties['control_subnet'],
            'accessConfigs': [{
                'natIP': '$(ref.' + context.properties['infra_id'] + '-bootstrap-public-ip.address)'
            }]
        }],
        'tags': {
            'items': [
                context.properties['infra_id'] + '-master',
                context.properties['infra_id'] + '-bootstrap'
            ]
        },
        'zone': context.properties['zone']
    }
}, {
    'name': context.properties['infra_id'] + '-bootstrap-ig',
    'type': 'compute.v1.instanceGroup',
    'properties': {
        'namedPorts': [
            {
                'name': 'ignition',
                'port': 22623
            }, {
                'name': 'https',
                'port': 6443
            }
        ],
        'network': context.properties['cluster_network'],
        'zone': context.properties['zone']
    }
}]

return {'resources': resources}
```

9.10.16. 在 GCP 中创建 control plane 机器

您必须在 Google Cloud Platform(GCP)中创建 control plane 机器，供您的集群使用。创建这些机器的一种方法是修改提供的 Deployment Manager 模板。

注意

如果不使用提供的 Deployment Manager 模板来创建 control plane 机器，您必须检查提供的信息并手动创建基础架构。如果您的集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 GCP 帐户。
- 为集群生成 Ignition 配置文件。
- 在 GCP 中创建和配置 VPC 及相关子网。
- 在 GCP 中创建和配置联网与负载均衡器。
- 创建 control plane 和计算角色。
- 创建 bootstrap 机器。

流程

1. 复制 **control plane 机器的 Deployment Manager 模板一节中的模板**，并将它以 **05_control_plane.py** 形式保存到计算机上。此模板描述了集群所需的 control plane 机器。
2. 导出资源定义所需的以下变量：

```
$ export MASTER_IGNITION=`cat <installation_directory>/master.ign`
```

3. 创建 **05_control_plane.yaml** 资源定义文件：

```
$ cat <<EOF >05_control_plane.yaml
imports:
- path: 05_control_plane.py

resources:
- name: cluster-control-plane
  type: 05_control_plane.py
  properties:
    infra_id: '${INFRA_ID}' 1
    zones: 2
    - '${ZONE_0}'
    - '${ZONE_1}'
    - '${ZONE_2}'

    control_subnet: '${CONTROL_SUBNET}' 3
    image: '${CLUSTER_IMAGE}' 4
    machine_type: 'n1-standard-4' 5
    root_volume_size: '128'
    service_account_email: '${MASTER_SERVICE_ACCOUNT}' 6

    ignition: '${MASTER_IGNITION}' 7
EOF
```

1. INFRA **_ID 是** 提取步骤 **中的 INFRA_ID** 基础架构名称。
2. **zones** 是 control plane 实例要部署到的区域，如 **us-central1-a**、**us-central1-b** 和 **us-central1-c**。
3. **control_subnet** 是控制子网的 **selfLink** URL。
4. **image** 是 RHCOS 镜像的 **selfLink** URL。
5. **machine_type** 是实例的机器类型，如 **n1-standard-4**。
6. **service_account_email** 是您创建的 master 服务帐户的电子邮件地址。
7. **Ignition** 是 **master.ign** 文件的内容。

4. 使用 **gcloud** CLI 创建部署：

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-control-plane --config
05_control_plane.yaml
```

5. 由于 Deployment Manager 的限制，模板无法管理负载均衡器成员资格，因此您必须手动添加 control plane 机器。
   - 运行以下命令，将 control plane 机器添加到适当的实例组中：

```
$ gcloud compute instance-groups unmanaged add-instances ${INFRA_ID}-master-
${ZONE_0}-ig --zone=${ZONE_0} --instances=${INFRA_ID}-master-0
```

```
$ gcloud compute instance-groups unmanaged add-instances ${INFRA_ID}-master-
${ZONE_1}-ig --zone=${ZONE_1} --instances=${INFRA_ID}-master-1
```

```
$ gcloud compute instance-groups unmanaged add-instances ${INFRA_ID}-master-
${ZONE_2}-ig --zone=${ZONE_2} --instances=${INFRA_ID}-master-2
```

   - 对于外部集群，还必须运行以下命令将 control plane 机器添加到目标池中：

```
$ gcloud compute target-pools add-instances ${INFRA_ID}-api-target-pool --instances-
zone="${ZONE_0}" --instances=${INFRA_ID}-master-0
```

```
$ gcloud compute target-pools add-instances ${INFRA_ID}-api-target-pool --instances-
zone="${ZONE_1}" --instances=${INFRA_ID}-master-1
```

```
$ gcloud compute target-pools add-instances ${INFRA_ID}-api-target-pool --instances-
zone="${ZONE_2}" --instances=${INFRA_ID}-master-2
```

9.10.16.1. control plane 机器的 Deployment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的 control plane 机器：

例 9.82. 05_control_plane.py Deployment Manager 模板

```
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-master-0',
        'type': 'compute.v1.instance',
        'properties': {
            'disks': [{
                'autoDelete': True,
                'boot': True,
                'initializeParams': {
                    'diskSizeGb': context.properties['root_volume_size'],
                    'diskType': 'zones/' + context.properties['zones'][0] + '/diskTypes/pd-ssd',
                    'sourceImage': context.properties['image']
                }
            }],
            'machineType': 'zones/' + context.properties['zones'][0] + '/machineTypes/' +
context.properties['machine_type'],
            'metadata': {
                'items': [{
                    'key': 'user-data',
                    'value': context.properties['ignition']
                }]
            },
            'networkInterfaces': [{
                'subnetwork': context.properties['control_subnet']
            }],
            'serviceAccounts': [{
                'email': context.properties['service_account_email'],
                'scopes': ['https://www.googleapis.com/auth/cloud-platform']
            }],
            'tags': {
                'items': [
                    context.properties['infra_id'] + '-master',
                ]
            },
            'zone': context.properties['zones'][0]
        }
    }, {
        'name': context.properties['infra_id'] + '-master-1',
        'type': 'compute.v1.instance',
        'properties': {
            'disks': [{
                'autoDelete': True,
                'boot': True,
                'initializeParams': {
                    'diskSizeGb': context.properties['root_volume_size'],
                    'diskType': 'zones/' + context.properties['zones'][1] + '/diskTypes/pd-ssd',
                    'sourceImage': context.properties['image']
                }
            }],
            'machineType': 'zones/' + context.properties['zones'][1] + '/machineTypes/' +
context.properties['machine_type'],
            'metadata': {
                'items': [{
                    'key': 'user-data',
                    'value': context.properties['ignition']
```

```
            }]
        },
        'networkInterfaces': [{
            'subnetwork': context.properties['control_subnet']
        }],
        'serviceAccounts': [{
            'email': context.properties['service_account_email'],
            'scopes': ['https://www.googleapis.com/auth/cloud-platform']
        }],
        'tags': {
            'items': [
                context.properties['infra_id'] + '-master',
            ]
        },
        'zone': context.properties['zones'][1]
    }
}, {
    'name': context.properties['infra_id'] + '-master-2',
    'type': 'compute.v1.instance',
    'properties': {
        'disks': [{
            'autoDelete': True,
            'boot': True,
            'initializeParams': {
                'diskSizeGb': context.properties['root_volume_size'],
                'diskType': 'zones/' + context.properties['zones'][2] + '/diskTypes/pd-ssd',
                'sourceImage': context.properties['image']
            }
        }],
        'machineType': 'zones/' + context.properties['zones'][2] + '/machineTypes/' +
context.properties['machine_type'],
        'metadata': {
            'items': [{
                'key': 'user-data',
                'value': context.properties['ignition']
            }]
        },
        'networkInterfaces': [{
            'subnetwork': context.properties['control_subnet']
        }],
        'serviceAccounts': [{
            'email': context.properties['service_account_email'],
            'scopes': ['https://www.googleapis.com/auth/cloud-platform']
        }],
        'tags': {
            'items': [
                context.properties['infra_id'] + '-master',
            ]
        },
        'zone': context.properties['zones'][2]
    }
}]

return {'resources': resources}
```

9.10.17. 等待 bootstrap 完成并删除 GCP 中的 bootstrap 资源

在 Google Cloud Platform(GCP)中创建所有所需的基础架构后，请等待您通过安装程序生成的 Ignition 配置文件所置备的机器上完成 bootstrap 过程。

先决条件

- 配置 GCP 帐户。
- 为集群生成 Ignition 配置文件。
- 在 GCP 中创建和配置 VPC 及相关子网。
- 在 GCP 中创建和配置联网与负载均衡器。
- 创建 control plane 和计算角色。
- 创建 bootstrap 机器。
- 创建 control plane 机器。

流程

1. 进入包含安装程序的目录并运行以下命令：

```
$ ./openshift-install wait-for bootstrap-complete --dir <installation_directory> \ 1
    --log-level info 2
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不是 **info**。

如果命令退出时没有 **FATAL** 警告，则您的生产环境 control plane 已被初始化。

2. 删除 bootstrap 资源：

```
$ gcloud compute backend-services remove-backend ${INFRA_ID}-api-internal-backend-
service --region=${REGION} --instance-group=${INFRA_ID}-bootstrap-ig --instance-group-
zone=${ZONE_0}
```

```
$ gsutil rm gs://${INFRA_ID}-bootstrap-ignition/bootstrap.ign
```

```
$ gsutil rb gs://${INFRA_ID}-bootstrap-ignition
```

```
$ gcloud deployment-manager deployments delete ${INFRA_ID}-bootstrap
```

9.10.18. 在 GCP 中创建额外的 worker 机器

您可以通过分散启动各个实例或利用集群外自动化流程（如自动扩展组），在 Google Cloud Platform(GCP)中为您的集群创建 worker 机器。您还可以利用 OpenShift Container Platform 中的内置集群扩展机制和机器 API。

注意

如果您要安装三节点集群，请跳过这一步。三节点集群包含三个 control plane 机器，它们也可以充当计算机器。

在本例中，您要使用 Deployment Manager 模板手动启动一个实例。通过在 文件中包括类型为 **06_worker.py** 的其他资源，即可启动其他实例。

注意

如果不使用提供的 Deployment Manager 模板来创建 worker 机器，您必须检查提供的信息并手动创建基础架构。如果您的集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 GCP 帐户。
- 为集群生成 Ignition 配置文件。
- 在 GCP 中创建和配置 VPC 及相关子网。
- 在 GCP 中创建和配置联网与负载均衡器。
- 创建 control plane 和计算角色。
- 创建 bootstrap 机器。
- 创建 control plane 机器。

流程

1. 复制 **worker 机器的 Deployment Manager 模板一节中的模板**，并将它以 **06_worker.py** 形式保存到计算机上。此模板描述了集群所需的 worker 机器。
2. 导出资源定义使用的变量。
   a. 导出托管计算机器的子网：

```
$ export COMPUTE_SUBNET=(`gcloud compute networks subnets describe
${INFRA_ID}-worker-subnet --region=${REGION} --format json | jq -r .selfLink`)
```

   b. 为您的服务帐户导出电子邮件地址：

```
$ export WORKER_SERVICE_ACCOUNT=(`gcloud iam service-accounts list --filter
"email~^${INFRA_ID}-w@${PROJECT_NAME}." --format json | jq -r '.[0].email'`)
```

   c. 导出计算机器 Ignition 配置文件的位置：

```
$ export WORKER_IGNITION=`cat <installation_directory>/worker.ign`
```

3. 创建 **06_worker.yaml** 资源定义文件：

```
$ cat <<EOF >06_worker.yaml
```

```
imports:
- path: 06_worker.py

resources:
- name: 'worker-0' (1)
  type: 06_worker.py
  properties:
    infra_id: '${INFRA_ID}' (2)
    zone: '${ZONE_0}' (3)
    compute_subnet: '${COMPUTE_SUBNET}' (4)
    image: '${CLUSTER_IMAGE}' (5)
    machine_type: 'n1-standard-4' (6)
    root_volume_size: '128'
    service_account_email: '${WORKER_SERVICE_ACCOUNT}' (7)
    ignition: '${WORKER_IGNITION}' (8)
- name: 'worker-1'
  type: 06_worker.py
  properties:
    infra_id: '${INFRA_ID}' (9)
    zone: '${ZONE_1}' (10)
    compute_subnet: '${COMPUTE_SUBNET}' (11)
    image: '${CLUSTER_IMAGE}' (12)
    machine_type: 'n1-standard-4' (13)
    root_volume_size: '128'
    service_account_email: '${WORKER_SERVICE_ACCOUNT}' (14)
    ignition: '${WORKER_IGNITION}' (15)
EOF
```

(1) **name** 是 worker 机器的名称，如 **worker-0**。

(2) (9) INFRA _**ID 是** 提取步骤 **中的 INFRA_ID** 基础架构名称。

(3) (10) **zone** 是 worker 机器要部署到的区域，如 **us-central1-a**。

(4) (11) **compute_subnet** 是计算子网的 **selfLink** URL。

(5) (12) **image** 是 RHCOS 镜像的 **selfLink** URL。[1]

(6) (13) **machine_type** 是实例的机器类型，如 **n1-standard-4**。

(7) (14) **service_account_email** 是您创建的 worker 服务帐户的电子邮件地址。

(8) (15) **Ignition** 是 **worker.ign** 文件的内容。

4. 可选：如果要启动更多实例，请在 **06_worker.yaml** 资源定义文件中包含类型为 **06_worker.py** 的其他资源。

5. 使用 **gcloud** CLI 创建部署：

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-worker --config
06_worker.yaml
```

1. 要使用 GCP Marketplace 镜像，请指定要使用的功能：

- OpenShift Container Platform: **https://www.googleapis.com/compute/v1/projects/redhat-marketplace-public/global/images/redhat-coreos-ocp-413-x86-64-202305021736**
- OpenShift Platform Plus: **https://www.googleapis.com/compute/v1/projects/redhat-marketplace-public/global/images/redhat-coreos-opp-413-x86-64-202305021736**
- OpenShift Kubernetes Engine: **https://www.googleapis.com/compute/v1/projects/redhat-marketplace-public/global/images/redhat-coreos-oke-413-x86-64-202305021736**

9.10.18.1. worker 机器的 Deployment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的 worker 机器：

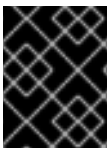
例 9.83. 06_worker.py Deployment Manager 模板

```
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-' + context.env['name'],
        'type': 'compute.v1.instance',
        'properties': {
            'disks': [{
                'autoDelete': True,
                'boot': True,
                'initializeParams': {
                    'diskSizeGb': context.properties['root_volume_size'],
                    'sourceImage': context.properties['image']
                }
            }],
            'machineType': 'zones/' + context.properties['zone'] + '/machineTypes/' +
context.properties['machine_type'],
            'metadata': {
                'items': [{
                    'key': 'user-data',
                    'value': context.properties['ignition']
                }]
            },
            'networkInterfaces': [{
                'subnetwork': context.properties['compute_subnet']
            }],
            'serviceAccounts': [{
                'email': context.properties['service_account_email'],
                'scopes': ['https://www.googleapis.com/auth/cloud-platform']
            }],
            'tags': {
                'items': [
                    context.properties['infra_id'] + '-worker',
                ]
            },
            'zone': context.properties['zone']
        }
```

```
    }]

    return {'resources': resources}
```

9.10.19. 安装 OpenShift CLI

您可以安装 OpenShift CLI(**oc**)来使用命令行界面与 OpenShift Container Platform 进行交互。您可以在 Linux、Windows 或 macOS 上安装 **oc**。



重要

如果安装了旧版本的 **oc**，则可能无法使用 OpenShift Container Platform 4.16 中的所有命令。下载并安装新版本的 **oc**。

在 Linux 上安装 OpenShift CLI
您可以按照以下流程在 Linux 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **产品变体** 下拉列表中选择架构。
3. 从 **版本** 下拉列表中选择适当的版本。
4. 点 **OpenShift v4.16 Linux Client** 条目旁的 **Download Now** 来保存文件。
5. 解包存档：

```
$ tar xvf <file>
```

6. 将 **oc** 二进制文件放到 **PATH 中的目录中**。
   要查看您的 **PATH**，请执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

在 Windows 上安装 OpenShift CLI
您可以按照以下流程在 Windows 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 Windows Client** 条目旁的 **Download Now** 来保存文件。

4. 使用 ZIP 程序解压存档。
5. 将 **oc** 二进制文件移到 **PATH 中的目录中**。
   要查看您的 **PATH**，请打开命令提示并执行以下命令：

```
C:\> path
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

在 macOS 上安装 OpenShift CLI
您可以按照以下流程在 macOS 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 macOS Client** 条目旁的 **Download Now** 来保存文件。

注意

对于 macOS arm64，请选择 **OpenShift v4.16 macOS arm64 Client** 条目。

4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
   要查看您的 **PATH**，请打开终端并执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

9.10.20. 使用 CLI 登录集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

9.10.21. 批准机器的证书签名请求

当您将机器添加到集群时，会为您添加的每台机器生成两个待处理证书签名请求(CSR)。您必须确认这些 CSR 已获得批准，或根据需要自行批准。必须首先批准客户端请求，然后批准服务器请求。

先决条件

- 您已将机器添加到集群中。

流程

1. 确认集群可以识别这些机器：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0 Ready    master  63m  v1.29.4
master-1 Ready    master  63m  v1.29.4
master-2 Ready    master  64m  v1.29.4
```

输出中列出了您创建的所有机器。

注意

在有些 CSR 被批准前，前面的输出可能不包括计算节点（也称为 worker 节点）。

2. 检查待处理的 CSR，并确保添加到集群中的每台机器都有 **Pending** 或 **Approved** 状态的客户端请求：

```
$ oc get csr
```

输出示例

```
```

```
NAME        AGE     REQUESTOR                                                                   CONDITION
csr-8b2br   15m     system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
csr-8vnps   15m     system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
...
```

在本例中，两台机器加入集群。您可能会在列表中看到更多已批准的 CSR。

3. 如果 CSR 没有获得批准，在您添加的机器的所有待处理 CSR 都处于 **Pending 状态** 后，请批准集群机器的 CSR：

注意

由于 CSR 会自动轮转，因此请在将机器添加到集群后一小时内批准您的 CSR。如果没有在一小时内批准它们，证书将会轮转，每个节点会存在多个证书。您必须批准所有这些证书。批准客户端 CSR 后，Kubelet 为服务证书创建一个二级 CSR，这需要手动批准。然后，如果 Kubelet 请求具有相同参数的新证书，则后续提供证书续订请求由 **machine-approver** 自动批准。

注意

对于在未启用机器 API 的平台上运行的集群，如裸机和其他用户置备的基础架构，您必须实施一种方法来自动批准 kubelet 提供证书请求(CSR)。如果没有批准请求，则 **oc exec**、oc**rsh** 和 **oc logs** 命令将无法成功，因为 API 服务器连接到 kubelet 时需要服务证书。与 Kubelet 端点联系的任何操作都需要此证书批准。该方法必须监视新的 CSR，确认 CSR 由 system: **node** 或 **system:admin** 组中的 **node-bootstrapper** 服务帐户提交，并确认节点的身份。

- 要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```

注意

在有些 CSR 被批准前，一些 Operator 可能无法使用。

4. 现在，您的客户端请求已被批准，您必须查看添加到集群中的每台机器的服务器请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE     REQUESTOR                                                                   CONDITION
```

```
csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

5. 如果剩余的 CSR 没有被批准，且处于 **Pending** 状态，请批准集群机器的 CSR：

   - 要单独批准，请对每个有效的 CSR 运行以下命令：

     ```
     $ oc adm certificate approve <csr_name> 1
     ```

     1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

   - 要批准所有待处理的 CSR，请运行以下命令：

     ```
     $ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
     {{end}}{{end}}' | xargs oc adm certificate approve
     ```

6. 批准所有客户端和服务器 CSR 后，机器将 处于 **Ready 状态**。运行以下命令验证：

   ```
   $ oc get nodes
   ```

   输出示例

   ```
   NAME      STATUS    ROLES   AGE  VERSION
   master-0  Ready     master  73m  v1.29.4
   master-1  Ready     master  73m  v1.29.4
   master-2  Ready     master  74m  v1.29.4
   worker-0  Ready     worker  11m  v1.29.4
   worker-1  Ready     worker  11m  v1.29.4
   ```

   注意

   批准服务器 CSR 后可能需要几分钟时间让机器过渡到 **Ready 状态**。

其他信息

- 如需有关 CSR 的更多信息，请参阅 证书签名请求。

9.10.22. 可选：添加入口 DNS 记录

如果在创建 Kubernetes 清单并生成 Ignition 配置时删除了 DNS 区配置，您必须手动创建指向入口负载均衡器的 DNS 记录。您可以创建一个通配符 ***.apps.{baseDomain}.** 或特定的记录。您可以根据要求使用 A、CNAME 和其他记录。

先决条件

- 配置 GCP 帐户。
- 在创建 Kubernetes 清单并生成 Ignition 配置时，删除 DNS 区配置。

- 在 GCP 中创建和配置 VPC 及相关子网。
- 在 GCP 中创建和配置联网与负载均衡器。
- 创建 control plane 和计算角色。
- 创建 bootstrap 机器。
- 创建 control plane 机器。
- 创建 worker 机器。

流程

1. 等待入口路由器创建负载均衡器并填充 **EXTERNAL-IP** 字段：

```
$ oc -n openshift-ingress get service router-default
```

输出示例

```
NAME            TYPE           CLUSTER-IP      EXTERNAL-IP     PORT(S)                      AGE
router-default  LoadBalancer  172.30.18.154  35.233.157.184
80:32288/TCP,443:31215/TCP  98
```

2. 在您的区中添加 A 记录：
   - 使用 A 记录：
     i. 导出路由器 IP 地址的变量：

```
$ export ROUTER_IP=`oc -n openshift-ingress get service router-default --no-
headers | awk '{print $4}'`
```

     ii. 在私有区中添加 A 记录：

```
$ if [ -f transaction.yaml ]; then rm transaction.yaml; fi
$ gcloud dns record-sets transaction start --zone ${INFRA_ID}-private-zone
$ gcloud dns record-sets transaction add ${ROUTER_IP} --name
\*.apps.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 300 --type A --zone
${INFRA_ID}-private-zone
$ gcloud dns record-sets transaction execute --zone ${INFRA_ID}-private-zone
```

     iii. 对于外部集群，还要在公共区中添加 A 记录：

```
$ if [ -f transaction.yaml ]; then rm transaction.yaml; fi
$ gcloud dns record-sets transaction start --zone ${BASE_DOMAIN_ZONE_NAME}
$ gcloud dns record-sets transaction add ${ROUTER_IP} --name
\*.apps.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 300 --type A --zone
${BASE_DOMAIN_ZONE_NAME}
$ gcloud dns record-sets transaction execute --zone
${BASE_DOMAIN_ZONE_NAME}
```

   - 要添加特定域而不使用通配符，请为集群的每个当前路由创建条目：

```
$ oc get --all-namespaces -o jsonpath='{range .items[*]}{range .status.ingress[*]}{.host}
{"\n"}{end}{end}' routes
```

输出示例

```
oauth-openshift.apps.your.cluster.domain.example.com
console-openshift-console.apps.your.cluster.domain.example.com
downloads-openshift-console.apps.your.cluster.domain.example.com
alertmanager-main-openshift-monitoring.apps.your.cluster.domain.example.com
prometheus-k8s-openshift-monitoring.apps.your.cluster.domain.example.com
```

9.10.23. 在用户置备的基础架构上完成 GCP 安装

在 Google Cloud Platform(GCP)用户置备的基础架构上启动 OpenShift Container Platform 安装后，您可以监控集群事件，直到集群就绪。

先决条件

- 在用户置备的 GCP 基础架构上为 OpenShift Container Platform 集群部署 bootstrap 机器。
- 安装 **oc** CLI 并登录。

流程

1. 完成集群安装：

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

输出示例

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

重要

- 安装程序生成的 Ignition 配置文件包含 24 小时后过期的证书，然后在该时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

2. 观察集群的运行状态。

   a. 运行以下命令查看当前的集群版本和状态：

```
$ oc get clusterversion
```

输出示例

```
NAME      VERSION   AVAILABLE   PROGRESSING   SINCE   STATUS
version             False       True          24m     Working towards 4.5.4: 99% complete
```

b. 运行以下命令，查看 control plane 上由 Cluster Version Operator(CVO)管理的 Operator：

```
$ oc get clusteroperators
```

输出示例

```
NAME                                       VERSION   AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.5.4     True        False         False      7m56s
cloud-credential                           4.5.4     True        False         False      31m
cluster-autoscaler                         4.5.4     True        False         False      16m
console                                    4.5.4     True        False         False      10m
csi-snapshot-controller                    4.5.4     True        False         False      16m
dns                                        4.5.4     True        False         False      22m
etcd                                       4.5.4     False       False         False      25s
image-registry                             4.5.4     True        False         False      16m
ingress                                    4.5.4     True        False         False      16m
insights                                   4.5.4     True        False         False      17m
kube-apiserver                             4.5.4     True        False         False      19m
kube-controller-manager                    4.5.4     True        False         False      20m
kube-scheduler                             4.5.4     True        False         False      20m
kube-storage-version-migrator              4.5.4     True        False         False      16m
machine-api                                4.5.4     True        False         False      22m
machine-config                             4.5.4     True        False         False      22m
marketplace                                4.5.4     True        False         False      16m
monitoring                                 4.5.4     True        False         False      10m
network                                    4.5.4     True        False         False      23m
node-tuning                                4.5.4     True        False         False      23m
openshift-apiserver                        4.5.4     True        False         False      17m
openshift-controller-manager               4.5.4     True        False         False      15m
openshift-samples                          4.5.4     True        False         False      16m
operator-lifecycle-manager                 4.5.4     True        False         False      22m
operator-lifecycle-manager-catalog         4.5.4     True        False         False      22m
operator-lifecycle-manager-packageserver   4.5.4     True        False         False      18m
service-ca                                 4.5.4     True        False         False      23m
service-catalog-apiserver                  4.5.4     True        False         False      23m
service-catalog-controller-manager         4.5.4     True        False         False      23m
storage                                    4.5.4     True        False         False      17m
```

c. 运行以下命令来查看您的集群 pod：

```
$ oc get pods --all-namespaces
```

输出示例

```
NAMESPACE                                  NAME
```

```
READY   STATUS    RESTARTS   AGE
kube-system                                        etcd-member-ip-10-0-3-111.us-east-
2.compute.internal             1/1     Running   0          35m
kube-system                                        etcd-member-ip-10-0-3-239.us-east-
2.compute.internal             1/1     Running   0          37m
kube-system                                        etcd-member-ip-10-0-3-24.us-east-
2.compute.internal             1/1     Running   0          35m
openshift-apiserver-operator                       openshift-apiserver-operator-6d6674f4f4-
h7t2t                          1/1     Running   1          37m
openshift-apiserver                                apiserver-fm48r
1/1     Running   0          30m
openshift-apiserver                                apiserver-fxkvv
1/1     Running   0          29m
openshift-apiserver                                apiserver-q85nm
1/1     Running   0          29m
...
openshift-service-ca-operator                      openshift-service-ca-operator-66ff6dc6cd-
9r257                          1/1     Running   0          37m
openshift-service-ca                               apiservice-cabundle-injector-695b6bcbc-cl5hm
1/1     Running   0          35m
openshift-service-ca                               configmap-cabundle-injector-8498544d7-
25qn6                          1/1     Running   0          35m
openshift-service-ca                               service-serving-cert-signer-6445fc9c6-wqdqn
1/1     Running   0          35m
openshift-service-catalog-apiserver-operator       openshift-service-catalog-apiserver-
operator-549f44668b-b5q2w      1/1     Running   0          32m
openshift-service-catalog-controller-manager-operator   openshift-service-catalog-
controller-manager-operator-b78cr2lnm   1/1     Running   0          31m
```

当当前集群版本是 **AVAILABLE** 时，安装已完成。

9.10.24. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于 远程健康监控

9.10.25. 后续步骤

- 自定义集群。
- 如果需要，您可以选择 不使用远程健康报告。
- 为 GCP 上的 Ingress Controller 配置全局访问 。

9.11. 使用 DEPLOYMENT MANAGER 模板在 GCP 上将集群安装到共享 VPC 中

在 OpenShift Container Platform 版本 4.16 中，您可以使用您提供的基础架构在 Google Cloud Platform (GCP) 上安装共享 VPC 的集群。在这种情况下，安装到共享 VPC 的集群是配置为使用 VPC 的集群，它与部署集群的项目不同。

共享 VPC 可让组织将资源从多个项目连接到一个通用 VPC 网络。您可以使用来自该网络的内部 IP，在组织内安全、高效地通信。有关共享 VPC 的更多信息，请参阅 GCP 文档中的 共享 VPC 概述。

此处概述了在共享 VPC 中执行用户提供基础架构安装的步骤。提供的几个 Deployment Manager 模板可帮助完成这些步骤，或者帮助您自行建模。您还可以自由选择通过其他方法创建所需的资源。

重要

执行用户置备的基础架构安装的步骤仅作为示例。使用您提供的基础架构安装集群需要了解云供应商和 OpenShift Container Platform 的安装过程。提供的几个 Deployment Manager 模板可帮助完成这些步骤，或者帮助您自行建模。您还可以自由选择通过其他方法创建所需的资源；模板仅作示例之用。

9.11.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读有关 选择集群安装方法的文档，并为用户准备它。
- 如果您使用防火墙并计划使用 Telemetry 服务，则将防火墙配置为允许集群需要访问的站点。
- 如果环境中无法访问云身份和访问管理(IAM)API，或者不想将管理员级别的凭证 secret 存储在 **kube-system** 命名空间中，您可以 手动创建和维护 IAM 凭证。

注意

如果要配置代理，请务必查看此站点列表。

9.11.2. 证书签名请求管理

在使用您置备的基础架构时，集群只能有限地访问自动机器管理，因此您必须提供一种在安装后批准集群证书签名请求 (CSR) 的机制。**kube-controller-manager** 只能批准 kubelet 客户端 CSR。 **machine-approver** 无法保证使用 kubelet 凭证请求的提供证书的有效性，因为它不能确认是正确的机器发出了该请求。您必须决定并实施一种方法，以验证 kubelet 提供证书请求的有效性并进行批准。

9.11.3. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类型的基础架构上执行受限网络安装。在此过程中，您可以下载所需的内容，并使用它为镜像 registry 填充安装软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群前，您要更新镜像 registry 的内容。

9.11.4. 配置托管集群的 GCP 项目

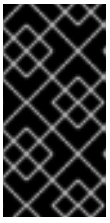
在安装 OpenShift Container Platform 之前，您必须配置 Google Cloud Platform(GCP)项目来托管它。

9.11.4.1. 创建 GCP 项目

要安装 OpenShift Container Platform，您必须在 Google Cloud Platform(GCP)帐户中创建项目来托管集群。

流程

- 创建一个项目来托管 OpenShift Container Platform 集群。请参阅 GCP 文档中的创建和管理 项目。



重要

如果您使用安装程序置备的基础架构，您的 GCP 项目必须使用 Premium Network Service Tier。使用安装程序安装的集群不支持 Standard Network Service Tier。安装程序为 **api-int.<cluster_name>.<base_domain>** URL 配置内部负载均衡；内部负载均衡需要 Premium Tier。

9.11.4.2. 在 GCP 中启用 API 服务

Google Cloud Platform(GCP)项目需要访问多个 API 服务来完成 OpenShift Container Platform 安装。

先决条件

- 已创建一个项目来托管集群。

流程

- 在托管集群的项目中启用以下所需的 API 服务。您还可以启用安装不需要的可选 API 服务。请参阅 GCP 文档中的 启用服务。

表 9.30. 所需的 API 服务

API 服务	控制台服务名称
Compute Engine API	**compute.googleapis.com**
Cloud Resource Manager API	**cloudresourcemanager.googleapis.com**
Google DNS API	**dns.googleapis.com**
IAM Service Account Credentials API	**iamcredentials.googleapis.com**

API 服务	控制台服务名称
Identity and Access Management(IAM)API	**iam.googleapis.com**
Service Usage API	**serviceusage.googleapis.com**

表 9.31. 可选 API 服务

API 服务	控制台服务名称
Cloud Deployment Manager V2 API	**deploymentmanager.googleapis.com**
Google Cloud API	**cloudapis.googleapis.com**
服务管理 API	**servicemanagement.googleapis.com**
Google Cloud Storage JSON API	**storage-api.googleapis.com**
Cloud Storage	**storage-component.googleapis.com**

9.11.4.3. GCP 帐户限值

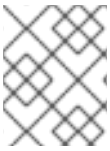
OpenShift Container Platform 集群使用许多 Google Cloud Platform(GCP)组件，但默认的 配额 不会影响您安装默认 OpenShift Container Platform 集群的能力。

默认集群包含三台计算和三台 control plane 机器，它使用以下资源。请注意，有些资源只在 bootstrap 过程中需要，并在集群部署后删除。

表 9.32. 默认集群中使用的 GCP 资源

service	组件	位置	所需的资源总数	bootstrap 后删除的资源
服务帐户	IAM	全局	6	1
防火墙规则	Networking	全局	11	1
转发规则	Compute	全局	2	0
健康检查	Compute	全局	2	0
镜像	Compute	全局	1	0
网络	Networking	全局	1	0

service	组件	位置	所需的资源总数	bootstrap 后删除的资源
路由器	Networking	全局	1	0
Routes	Networking	全局	2	0
子网	Compute	全局	2	0
目标池	Networking	全局	2	0



注意

如果在安装过程中任何配额不足，安装程序会显示一个错误信息，包括超过哪个配额，以及显示区域。

请考虑您的集群的实际大小、预定的集群增长，以及来自与您的帐户关联的其他集群的使用情况。CPU、静态 IP 地址和持久磁盘 SSD（存储）配额是最可能不足的。

如果您计划在以下区域之一部署集群，您将超过最大存储配额，并可能会超过 CPU 配额限制：

- **asia-east2**
- **asia-northeast2**
- **asia-south1**
- **australia-southeast1**
- **europe-north1**
- **europe-west2**
- **europe-west3**
- **europe-west6**
- **northamerica-northeast1**
- **southamerica-east1**
- **us-west2**

您可以从 GCP 控制台 增加资源配额，但可能需要提交一个支持问题单。务必提前规划集群大小，以便在安装 OpenShift Container Platform 集群前有足够的时间来等待支持问题单被处理。

9.11.4.4. 在 GCP 中创建服务帐户

OpenShift Container Platform 需要一个 Google Cloud Platform(GCP)服务帐户，它提供访问 Google API 中数据的验证和授权。如果您没有包含项目中所需角色的现有 IAM 服务帐户，您必须创建一个。

先决条件

- 已创建一个项目来托管集群。

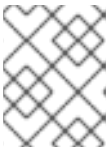
流程

1. 在用于托管 OpenShift Container Platform 集群的项目中创建一个服务帐户。请参阅 GCP 文档中的 创建服务帐户。
2. 为服务帐户授予适当的权限。您可以逐一授予权限，也可以为其分配 **Owner** 角色。请参阅将角色转换到特定资源的服务帐户。

注意

将服务帐户设置为项目的所有者是获取所需权限的最简单方法，这意味着该服务帐户对项目有完全的控制权。您必须确定提供这种能力所带来的风险是否可以接受。

3. 您可以使用 JSON 格式创建服务帐户密钥，或将服务帐户附加到 GCP 虚拟机。请参阅 GCP 文档中的创建服务帐户密钥以及为实例创建并启用服务帐户。
您必须有一个服务帐户密钥或带有附加服务帐户的虚拟机来创建集群。



注意

如果您使用附加服务帐户的虚拟机来创建集群，则必须在安装前在 **install-config.yaml** 文件中设置 **credentialsMode: Manual**。

9.11.4.4.1. 所需的 GCP 角色

将 **Owner** 角色附加到您创建的服务帐户时，您可以为该服务帐户授予所有权限，包括安装 OpenShift Container Platform 所需的权限。如果机构的安全策略需要更严格的权限集，您可以创建具有以下权限的服务帐户：如果您将集群部署到现有的虚拟私有云 (VPC) 中，则服务帐户不需要某些网络权限，如以下列表中记录：

安装程序所需的角色

- Compute Admin
- Role Administrator
- Security Admin
- Service Account Admin
- Service Account Key Admin
- Service Account User
- Storage Admin

安装过程中创建网络资源所需的角色

- DNS Administrator

在 passthrough 模式中使用 Cloud Credential Operator 所需的角色

- Compute Load Balancer Admin

用户置备的 GCP 基础架构所需的角色

- Deployment Manager Editor

以下角色应用到 control plane 和计算机器使用的服务帐户：

表 9.33. GCP 服务帐户角色

帐户	角色
Control Plane	**roles/compute.instanceAdmin**
	roles/compute.networkAdmin
	roles/compute.securityAdmin
	roles/storage.admin
	roles/iam.serviceAccountUser
Compute	**roles/compute.viewer**
	roles/storage.admin

9.11.4.5. 支持的 GCP 区域

您可以将 OpenShift Container Platform 集群部署到以下 Google Cloud Platform(GCP)区域：

- **africa-south1** (Johannesburg, South Africa)
- **asia-east1** (Changhua County, Taiwan)
- **asia-east2** (Hong Kong)
- **asia-northeast1** (Tokyo, Japan)
- **asia-northeast2** (Osaka, Japan)
- **asia-northeast3** (Seoul, South Korea)
- **asia-south1** (Mumbai, India)
- **asia-south2** (Delhi, India)
- **asia-southeast1** (Jurong West, Singapore)
- **asia-southeast2** (Jakarta, Indonesia)
- **aia-southeast1(** Sydney, Australia)
- **australia-southeast2** (Melbourne, Australia)
- **europe-central2** (Warsaw, USA)

- **europe-north1** (Hamina, Finland)
- **europe-southwest1** (Madrid, Spain)
- **europe-west1** (St. Ghislain, Belgium)
- **europe-west2** (London, England, UK)
- **europe-west3** (Frankfurt, Germany)
- **europe-west4** (Eemshaven, Netherlands)
- **europe-west6** (Zürich, Switzerland)
- **europe-west8** (Milan, Italy)
- **europe-west9** (Paris, France)
- **europe-west12** (Turin, Italy)
- **me-central1** (Doha, Qatar, Middle East)
- **me-central2** (Dammam, Saudi Arabia, Middle East)
- **me-west1** (Tel Aviv, Israel)
- **northamerica-northeast1** (Montréal, Québec, Canada)
- **northamerica-northeast2** (Toronto, Ontario, Canada)
- **southamerica-east1** (São Paulo, Brazil)
- **southamerica-west1** (Santiago, Chile)
- **us-central1** (Council Bluffs, Iowa, USA)
- **us-east1** (Moncks Corner, South Carolina, USA)
- **us-east4** (Ashburn, North Virginia, USA)
- **us-east5** (Columbus, Ohio)
- **us-south1** (Dallas, Texas)
- **us-west1** (The Dalles, Oregon, USA)
- **us-west2** (Los Angeles, California, USA)
- **us-west3** (Salt Lake City, Utah, USA)
- **us-west4** (Las Vegas, Nevada, USA)

注意

要确定地区和区域中可以使用哪些机器类型实例，请参阅 Google 文档。

9.11.4.6. 为 GCP 安装和配置 CLI 工具

要使用用户置备的基础架构在 Google Cloud Platform(GCP)上安装 OpenShift Container Platform，您必须为 GCP 安装和配置 CLI 工具。

先决条件

- 已创建一个项目来托管集群。
- 您创建了服务帐户并授予其所需的权限。

流程

1. 在 **$PATH** 中安装以下二进制文件：
   - **gcloud**
   - **gsutil**

   请参阅 GCP 文档中的 安装最新的 Cloud SDK 版本。
2. 使用 **gcloud** 工具及您配置的服务帐户进行身份验证。
   请参阅 GCP 文档中的 使用服务帐户授权。

9.11.5. 具有用户置备基础架构的集群的要求

对于包含用户置备的基础架构的集群，您必须部署所有所需的机器。

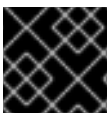
本节论述了在用户置备的基础架构上部署 OpenShift Container Platform 的要求。

9.11.5.1. 集群安装所需的机器

最小的 OpenShift Container Platform 集群需要以下主机：

表 9.34. 最低所需的主机

主机	描述
一个临时 bootstrap 机器	集群需要 bootstrap 机器在三台 control plane 机器上部署 OpenShift Container Platform 集群。您可在安装集群后删除 bootstrap 机器。
三台 control plane 机器	control plane 机器运行组成 control plane 的 Kubernetes 和 OpenShift Container Platform 服务。
至少两台计算机器，也称为 worker 机器。	OpenShift Container Platform 用户请求的工作负载在计算机器上运行。



重要

要保持集群的高可用性，请将独立的物理主机用于这些集群机器。

bootstrap 和 control plane 机器必须使用 Red Hat Enterprise Linux CoreOS(RHCOS)作为操作系统。但是，计算机器可以在 Red Hat Enterprise Linux CoreOS(RHCOS)、Red Hat Enterprise Linux(RHEL) 8.6 和更高的版本。

请注意，RHCOS 基于 Red Hat Enterprise Linux(RHEL) 9.2，并继承其所有硬件认证和要求。查看 红帽企业 Linux 技术功能和限制。

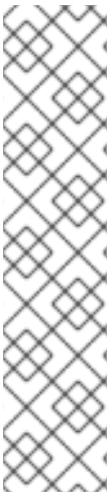
9.11.5.2. 集群安装的最低资源要求

每台集群机器都必须满足以下最低要求：

表 9.35. 最低资源要求

机器	操作系统	vCPU [1]	虚拟内存	Storage	每秒输入/输出 (IOPS) [2]
bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane（控制平面）	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS、RHEL 8.6 及更新版本 [3]	2	8 GB	100 GB	300

1. 当未启用并发多线程(SMT)或超线程时，一个 vCPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比例：（每个内核数的线程）× sockets = vCPU。
2. OpenShift Container Platform 和 Kubernetes 对磁盘性能非常敏感，建议使用更快的存储速度，特别是 control plane 节点上需要 10 ms p99 fsync 持续时间的 etcd。请注意，在许多云平台上，存储大小和 IOPS 可一起扩展，因此您可能需要过度分配存储卷来获取足够的性能。
3. 与所有用户置备的安装一样，如果您选择在集群中使用 RHEL 计算机器，则负责所有操作系统生命周期管理和维护，包括执行系统更新、应用补丁和完成所有其他必要的任务。RHEL 7 计算机器的使用已弃用，并已在 OpenShift Container Platform 4.10 及更新的版本中删除。



注意

从 OpenShift Container Platform 版本 4.13 开始，RHCOS 基于 RHEL 版本 9.2，它更新了微架构要求。以下列表包含每个架构需要的最小指令集架构 (ISA)：

- x86-64 体系结构需要 x86-64-v2 ISA
- ARM64 架构需要 ARMv8.0-A ISA
- IBM Power 架构需要 Power 9 ISA
- s390x 架构需要 z14 ISA

如需更多信息，请参阅 RHEL 架构。

如果平台的实例类型满足集群机器的最低要求，则 OpenShift Container Platform 支持使用它。

其他资源

- 优化存储

9.11.5.3. 为 GCP 测试的实例类型

以下 Google Cloud Platform 实例类型已使用 OpenShift Container Platform 测试。

例 9.84. 机器系列

- **A2**
- **A3**
- **C2**
- **C2D**
- **C3**
- **C3D**
- **E2**
- **M1**
- **N1**
- **N2**
- **N2D**
- **Tau T2D**

9.11.5.4. 使用自定义机器类型

支持使用自定义机器类型来安装 OpenShift Container Platform 集群。

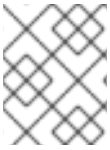
使用自定义机器类型时请考虑以下几点：

- 与预定义的实例类型类似，自定义机器类型必须满足 control plane 和计算机器的最低资源要求。如需更多信息，请参阅"集群安装的资源要求"。
- 自定义机器类型的名称必须遵循以下语法：
  custom-<number_of_cpus>-<amount_of_memory_in_mb>

  例如，**custom-6-20480**。

9.11.6. 配置托管共享 VPC 网络的 GCP 项目

如果使用共享的 Virtual Private Cloud(VPC)在 Google Cloud Platform(GCP)中托管 OpenShift Container Platform 集群，您必须配置托管它的项目。



注意

如果您已有托管共享 VPC 网络的项目，请查看本节以确保项目满足安装 OpenShift Container Platform 集群的所有要求。

流程

1. 创建一个项目来托管您的 OpenShift Container Platform 集群的共享 VPC。请参阅 GCP 文档中的创建和管理 项目。
2. 在托管共享 VPC 的项目中创建服务帐户。请参阅 GCP 文档中的 创建服务帐户。
3. 为服务帐户授予适当的权限。您可以逐一授予权限，也可以为其分配 **Owner** 角色。请参阅将角色转换到特定资源的服务帐户。

注意

将服务帐户设置为项目的所有者是获取所需权限的最简单方法，这意味着该服务帐户对项目有完全的控制权。您必须确定提供这种能力所带来的风险是否可以接受。

托管共享 VPC 网络的项目的服务帐户需要以下角色：

- Compute 网络用户
- Compute Security Admin
- Deployment Manager Editor
- DNS Administrator
- Security Admin
- 网络管理管理员

9.11.6.1. 为 GCP 配置 DNS

要安装 OpenShift Container Platform，您使用的 Google Cloud Platform(GCP)帐户必须在项目中有一个专用的公共托管区，托管您将集群安装到的共享 VPC 中。此区域必须对域具有权威。DNS 服务为集群外部连接提供集群 DNS 解析和名称查询。

流程

1. 确定您的域或子域，以及注册商。您可以转移现有的域和注册商，或通过 GCP 或其他来源获取新的域和注册商。

注意

如果您购买了新的域，则需要时间来传播相关的 DNS 更改。有关通过 Google 购买域的更多信息，请参阅 Google Domains。

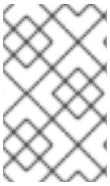
2. 在 GCP 项目中为您的域或子域创建一个公共托管区。请参阅 GCP 文档中的 创建公共区。
使用适当的根域，如 **openshiftcorp.com 或** 子域，如 cluster **.openshiftcorp.com**。
3. 从托管区域记录中提取新的权威名称服务器。请参阅 GCP 文档中的 查找您的云 DNS 名称服务器。
您通常有四个名称服务器。
4. 更新域所用名称服务器的注册商记录。例如，如果您将域注册到 Google Domains，请参阅 Google Domains 帮助中的以下主题： 如何切换到自定义名称服务器。

5. 如果您将根域迁移到 Google Cloud DNS，请迁移您的 DNS 记录。请参阅 GCP 文档中的 Migrating to Cloud DNS。

6. 如果您使用子域，请按照贵公司的步骤将其委派记录添加到父域。这个过程可能包括对您公司的 IT 部门或控制您公司的根域和 DNS 服务的部门发出的请求。

9.11.6.2. 在 GCP 中创建 VPC

您必须在 Google Cloud Platform(GCP)中创建一个 VPC，供您的 OpenShift Container Platform 集群使用。您可以自定义 VPC 来满足您的要求。创建 VPC 的一种方法是修改提供的 Deployment Manager 模板。



注意

如果不使用提供的 Deployment Manager 模板来创建 GCP 基础架构，您必须检查提供的信息并手动创建基础架构。如果您的集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 GCP 帐户。

流程

1. 复制 **VPC 的 Deployment Manager 模板一节中的模板，** 并将它以 **01_vpc.py** 形式保存到计算机上。此模板描述了集群所需的 VPC。

2. 导出资源定义所需的以下变量：

   a. 导出 control plane CIDR:

   ```
   $ export MASTER_SUBNET_CIDR='10.0.0.0/17'
   ```

   b. 导出计算 CIDR:

   ```
   $ export WORKER_SUBNET_CIDR='10.0.128.0/17'
   ```

   c. 将部署 VPC 网络和集群的区域导出到：

   ```
   $ export REGION='<region>'
   ```

3. 导出托管共享 VPC 的项目 ID 的变量：

   ```
   $ export HOST_PROJECT=<host_project>
   ```

4. 导出属于主机项目的服务帐户电子邮件的变量：

   ```
   $ export HOST_PROJECT_ACCOUNT=<host_service_account_email>
   ```

5. 创建 **01_vpc.yaml** 资源定义文件：

   ```
   $ cat <<EOF >01_vpc.yaml
   imports:
   ```

```
- path: 01_vpc.py

resources:
- name: cluster-vpc
  type: 01_vpc.py
  properties:
    infra_id: '<prefix>' 1
    region: '${REGION}' 2
    master_subnet_cidr: '${MASTER_SUBNET_CIDR}' 3
    worker_subnet_cidr: '${WORKER_SUBNET_CIDR}' 4
EOF
```

1. **NFRA_** ID 是网络名称的前缀。
2. **region** 是集群要部署到的区域，如 **us-central1**。
3. **master_subnet_cidr** 是 master 子网的 CIDR，如 **10.0.0.0/17**。
4. **worker_subnet_cidr** 是 worker 子网的 CIDR，例如 **10.0.128.0/17**。

6. 使用 **gcloud** CLI 创建部署：

```
$ gcloud deployment-manager deployments create <vpc_deployment_name> --config
01_vpc.yaml --project ${HOST_PROJECT} --account ${HOST_PROJECT_ACCOUNT} 1
```

1. 对于 **<vpc_deployment_name>**，请指定要部署的 VPC 名称。

7. 导出其他组件需要的 VPC 变量：
   a. 导出主机项目网络的名称：

```
$ export HOST_PROJECT_NETWORK=<vpc_network>
```

   b. 导出主机项目 control plane 子网的名称：

```
$ export HOST_PROJECT_CONTROL_SUBNET=<control_plane_subnet>
```

   c. 导出主机项目计算子网的名称：

```
$ export HOST_PROJECT_COMPUTE_SUBNET=<compute_subnet>
```

8. 设置共享 VPC。请参阅 GCP 文档中的 设置共享 VPC。

9.11.6.2.1. VPC 的 Deployment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的 VPC：

例 9.85. 01_VPC.py Deployment Manager 模板

```
def GenerateConfig(context):

    resources = [{
```

```
        'name': context.properties['infra_id'] + '-network',
        'type': 'compute.v1.network',
        'properties': {
            'region': context.properties['region'],
            'autoCreateSubnetworks': False
        }
    }, {
        'name': context.properties['infra_id'] + '-master-subnet',
        'type': 'compute.v1.subnetwork',
        'properties': {
            'region': context.properties['region'],
            'network': '$(ref.' + context.properties['infra_id'] + '-network.selfLink)',
            'ipCidrRange': context.properties['master_subnet_cidr']
        }
    }, {
        'name': context.properties['infra_id'] + '-worker-subnet',
        'type': 'compute.v1.subnetwork',
        'properties': {
            'region': context.properties['region'],
            'network': '$(ref.' + context.properties['infra_id'] + '-network.selfLink)',
            'ipCidrRange': context.properties['worker_subnet_cidr']
        }
    }, {
        'name': context.properties['infra_id'] + '-router',
        'type': 'compute.v1.router',
        'properties': {
            'region': context.properties['region'],
            'network': '$(ref.' + context.properties['infra_id'] + '-network.selfLink)',
            'nats': [{
                'name': context.properties['infra_id'] + '-nat-master',
                'natIpAllocateOption': 'AUTO_ONLY',
                'minPortsPerVm': 7168,
                'sourceSubnetworkIpRangesToNat': 'LIST_OF_SUBNETWORKS',
                'subnetworks': [{
                    'name': '$(ref.' + context.properties['infra_id'] + '-master-subnet.selfLink)',
                    'sourceIpRangesToNat': ['ALL_IP_RANGES']
                }]
            }, {
                'name': context.properties['infra_id'] + '-nat-worker',
                'natIpAllocateOption': 'AUTO_ONLY',
                'minPortsPerVm': 512,
                'sourceSubnetworkIpRangesToNat': 'LIST_OF_SUBNETWORKS',
                'subnetworks': [{
                    'name': '$(ref.' + context.properties['infra_id'] + '-worker-subnet.selfLink)',
                    'sourceIpRangesToNat': ['ALL_IP_RANGES']
                }]
            }]
        }
    }]

    return {'resources': resources}
```

9.11.7. 为 GCP 创建安装文件

要使用用户置备的基础架构在 Google Cloud Platform(GCP)上安装 OpenShift Container Platform，您必须生成安装程序部署集群所需的文件，并进行修改，以便集群只创建要使用的机器。您可以生成并自定义 **install-config.yaml** 文件、Kubernetes 清单和 Ignition 配置文件。您还可以选择在安装准备阶段首先设置独立 **var** 分区。

9.11.7.1. 手动创建安装配置文件

安装集群要求您手动创建安装配置文件。

先决条件

- 您在本地机器上有一个 SSH 公钥来提供给安装程序。该密钥将用于在集群节点上进行 SSH 身份验证，以进行调试和灾难恢复。
- 已获取 OpenShift Container Platform 安装程序和集群的 pull secret。

流程

1. 创建一个安装目录来存储所需的安装资产：

```
$ mkdir <installation_directory>
```

重要

您必须创建一个目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

2. 自定义提供的 **install-config.yaml** 文件模板示例，并将其保存在 **<installation_directory>** 中。

注意

此配置文件必须命名为 **install-config.yaml**。

3. 备份 **install-config.yaml** 文件，以便您可以使用它安装多个集群。

重要

install-config.yaml 文件会在安装过程的下一步中使用。现在必须备份它。

其他资源

- GCP 的安装配置参数

9.11.7.2. 启用屏蔽虚拟机

您可在安装集群时使用 Shielded 虚拟机。Shielded 虚拟机具有额外的安全功能，包括安全引导、固件和完整性监控和 rootkit 检测。如需更多信息，请参阅 Google 文档中有关 Shielded 虚拟机 的文档。

注意

目前，在具有 64 位 ARM 基础架构的集群中不支持 Shielded 虚拟机。

先决条件

- 您已创建了 **install-config.yaml** 文件。

流程

- 在部署集群前，使用文本编辑器编辑 **install-config.yaml** 文件并添加以下部分之一：

  a. 仅将屏蔽的虚拟机用于 control plane 机器：

```
controlPlane:
  platform:
    gcp:
      secureBoot: Enabled
```

  b. 仅将屏蔽的虚拟机用于计算机器：

```
compute:
- platform:
    gcp:
      secureBoot: Enabled
```

  c. 将屏蔽的虚拟机用于所有机器：

```
platform:
  gcp:
    defaultMachinePlatform:
      secureBoot: Enabled
```

9.11.7.3. 启用机密虚拟机

您可在安装集群时使用机密虚拟机。机密虚拟机在处理数据时加密数据。如需更多信息，请参阅 Google 文档中有关 机密计算的内容。您可以同时启用机密虚拟机和 Shielded 虚拟机，虽然它们不相互依赖。

注意

64 位 ARM 架构目前不支持机密虚拟机。

先决条件

- 您已创建了 **install-config.yaml** 文件。

流程

- 在部署集群前，使用文本编辑器编辑 **install-config.yaml** 文件并添加以下部分之一：

  a. 仅将机密虚拟机用于 control plane 机器：

```
controlPlane:
  platform:
```

```
    gcp:
      confidentialCompute: Enabled 1
      type: n2d-standard-8 2
      onHostMaintenance: Terminate 3
```

1 启用机密虚拟机。

2 指定支持机密虚拟机的机器类型。机密虚拟机需要 N2D 或 C2D 系列机器类型。有关支持的机器类型的更多信息，请参阅支持的操作系统和机器类型。

3 指定主机维护事件期间虚拟机的行为，如硬件或软件更新。对于使用机密虚拟机的机器，此值必须设置为 **Terminate**，这会停止虚拟机。机密虚拟机不支持实时迁移。

b. 仅将机密虚拟机用于计算机器：

```
compute:
- platform:
    gcp:
      confidentialCompute: Enabled
      type: n2d-standard-8
      onHostMaintenance: Terminate
```

c. 将机密虚拟机用于所有机器：

```
platform:
  gcp:
    defaultMachinePlatform:
      confidentialCompute: Enabled
      type: n2d-standard-8
      onHostMaintenance: Terminate
```

9.11.7.4. GCP 的自定义 install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多详情，或修改所需参数的值。

重要

此示例 YAML 文件仅供参考。您必须使用安装程序来获取 **install-config.yaml** 文件，并进行修改。

```
apiVersion: v1
baseDomain: example.com 1
controlPlane: 2
  hyperthreading: Enabled 3 4
  name: master
  platform:
    gcp:
      type: n2-standard-4
      zones:
      - us-central1-a
      - us-central1-c
```

```
      tags: 5
      - control-plane-tag1
      - control-plane-tag2
    replicas: 3
  compute: 6
  - hyperthreading: Enabled 7
    name: worker
    platform:
      gcp:
        type: n2-standard-4
        zones:
        - us-central1-a
        - us-central1-c
        tags: 8
        - compute-tag1
        - compute-tag2
    replicas: 0
  metadata:
    name: test-cluster
  networking:
    clusterNetwork:
    - cidr: 10.128.0.0/14
      hostPrefix: 23
    machineNetwork:
    - cidr: 10.0.0.0/16
    networkType: OVNKubernetes 9
    serviceNetwork:
    - 172.30.0.0/16
  platform:
    gcp:
      defaultMachinePlatform:
        tags: 10
        - global-tag1
        - global-tag2
      projectID: openshift-production 11
      region: us-central1 12
  pullSecret: '{"auths": ...}'
  fips: false 13
  sshKey: ssh-ed25519 AAAA... 14
  publish: Internal 15
```

1 指定主机项目上的公共 DNS。

2 6 如果没有提供这些参数和值，安装程序会提供默认值。

3 7 **controlPlane** 部分是一个单个映射，但 compute 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，controlPlane **部分** 的第一行则不以连字符开头。虽然这两个部分目前都定义了单一机器池，但未来的 OpenShift Container Platform 版本可能在安装过程中支持定义多个计算池。仅使用一个 control plane 池。

4 是否要启用或禁用并发多线程或 **超线程**。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将 参数值设置为 **Disabled** 来禁用它。如果在某些集群机器中禁用并发多线程，则必须在所有集群机器中禁用它。

重要

如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。如果您禁用并发多线程，请为您的机器使用较大的类型，如 **n1-standard-8**。

5 8 10 可选：要应用到 control plane 或计算机器集的一组网络标签。**platform.gcp.defaultMachinePlatform.tags** 参数将应用到 control plane 和计算机器。如果设置了 **compute.platform.gcp.tags** 或 **controlPlane.platform.gcp.tags** 参数，它们会覆盖 **platform.gcp.defaultMachinePlatform.tags** 参数。

9 要安装的集群网络插件。默认值 **OVNKubernetes** 是唯一支持的值。

11 指定虚拟机实例所在的主项目。

12 指定 VPC 网络所在区域。

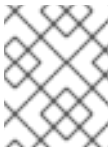
13 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 Red Hat Enterprise Linux (RHEL) 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅在 FIPS 模式中安装该系统。

当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS)时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。

14 您可以选择提供您用来访问集群中机器的 **sshKey** 值。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

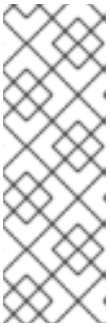
15 如何发布集群的面向用户的端点。将 **publish** 设置为 **Internal** 以部署一个私有集群，它不能被互联网访问。默认值为 **External**。要在使用您置备的基础架构的集群中使用共享 VPC，必须将 **publish** 设置为 **Internal**。安装程序将不再能够访问主机项目中的基域的公共 DNS 区域。

9.11.7.5. 在安装过程中配置集群范围的代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 **Proxy** 对象的 **spec.noProxy** 字段中添加站点来绕过代理。



注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，**Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

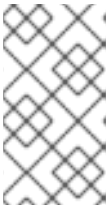
1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 **.** 以仅匹配子域。例如，**.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代理。
4. 如果提供，安装程序会在 **openshift-config** 命名空间中生成名为 **user-ca-bundle** 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 **trusted-ca-bundle** 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， **Proxy** 对象的 **trustedCA** 字段中也会引用此配置映射。**additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。
5. 可选：决定 **Proxy** 对象的配置以引用 **trustedCA** 字段中 **user-ca-bundle** 配置映射的策略。允许的值是 **Proxyonly** 和 **Always**。仅在配置了 **http/https** 代理时，使用 **Proxyonly** 引用 **user-ca-bundle** 配置映射。使用 **Always** 始终引用 **user-ca-bundle** 配置映射。默认值为 **Proxyonly**。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。



注意

如果安装程序超时，重启并使用安装程序的 **wait-for** 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. 保存该文件并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 cluster 的集群范围代理，该代理 **使用** 提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

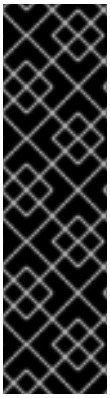
注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

9.11.7.6. 创建 Kubernetes 清单和 Ignition 配置文件

由于您必须修改一些集群定义文件并手动启动集群机器，因此您必须生成 Kubernetes 清单和 Ignition 配置文件来配置机器。

安装配置文件转换为 Kubernetes 清单。清单嵌套到 Ignition 配置文件中，稍后用于配置集群机器。



重要

- OpenShift Container Platform 安装程序生成的 Ignition 配置文件包含 24 小时后过期的证书，然后在该时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

先决条件

- 已获得 OpenShift Container Platform 安装程序。
- 已创建 **install-config.yaml** 安装配置文件。

流程

1. 进入包含 OpenShift Container Platform 安装程序的目录，并为集群生成 Kubernetes 清单：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定包含您创建的 **install-config.yaml** 文件的安装目录。

2. 删除定义 control plane 机器的 Kubernetes 清单文件：

```
$ rm -f <installation_directory>/openshift/99_openshift-cluster-api_master-machines-*.yaml
```

通过删除这些文件，您可以防止集群自动生成 control plane 机器。

3. 删除定义 control plane 机器集的 Kubernetes 清单文件：

```
$ rm -f <installation_directory>/openshift/99_openshift-machine-api_master-control-plane-machine-set.yaml
```

4. 删除定义 worker 机器的 Kubernetes 清单文件：

```
$ rm -f <installation_directory>/openshift/99_openshift-cluster-api_worker-machineset-*.yaml
```

由于您要自行创建和管理 worker 机器，因此不需要初始化这些机器。

5. 检查 **<installation_directory>/manifests/cluster-scheduler-02-config.yml Kubernetes 清单文件中的 mastersSchedulable** 参数是否已设置为 **false**。此设置可防止在 control plane 机器上调度 pod：

   a. 打开 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** 文件。

   b. 找到 **mastersSchedulable** 参数，并确保它被设置为 **false**。

   c. 保存并退出 文件。

6. 删除 **<installation_directory>/manifests/cluster-dns-02-config.yml** DNS 配置文件中的 **privateZone** 部分：

```
apiVersion: config.openshift.io/v1
kind: DNS
metadata:
  creationTimestamp: null
  name: cluster
spec:
  baseDomain: example.openshift.com
  privateZone: 1
    id: mycluster-100419-private-zone
status: {}
```

1 完全删除此部分。

7. 配置 VPC 的云供应商。

   a. 打开 **<installation_directory>/manifests/cloud-provider-config.yaml** 文件。

   b. 添加 **network-project-id** 参数，并将其值设置为托管共享 VPC 网络的项目 ID。

   c. 添加 **network-name** 参数，并将其值设置为托管 OpenShift Container Platform 集群的共享 VPC 网络的名称。

   d. 将 **subnetwork-name** 参数的值替换为托管计算机器的共享 VPC 子网的值。

<installation_directory>/manifests/cloud-provider-config.yaml 的内容类似以下示例：

```
config: |+
  [global]
  project-id      = example-project
```

```
  regional       = true
  multizone      = true
  node-tags      = opensh-ptzzx-master
  node-tags      = opensh-ptzzx-worker
  node-instance-prefix = opensh-ptzzx
  external-instance-groups-prefix = opensh-ptzzx
  network-project-id = example-shared-vpc
  network-name    = example-network
  subnetwork-name = example-worker-subnet
```

8. 如果您部署了一个不属于私有网络的集群，打开 **<installation_directory>/manifests/cluster-ingress-default-ingresscontroller.yaml** 文件，并将 **scope** 参数的值替换为 **External**。该文件类似于以下示例：

```
apiVersion: operator.openshift.io/v1
kind: IngressController
metadata:
  creationTimestamp: null
  name: default
  namespace: openshift-ingress-operator
spec:
  endpointPublishingStrategy:
    loadBalancer:
      scope: External
    type: LoadBalancerService
status:
  availableReplicas: 0
  domain: ''
  selector: ''
```

9. 要创建 Ignition 配置文件，请从包含安装程序的目录运行以下命令：

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定相同的安装目录。

为安装目录中的 bootstrap、control plane 和计算节点创建 Ignition 配置文件。**kubeadmin-password** 和 **kubeconfig** 文件在 **./<installation_directory>/auth** 目录中创建：

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

9.11.8. 导出常用变量

9.11.8.1. 提取基础架构名称

Ignition 配置文件包含一个唯一集群标识符，您可以使用它在 Google Cloud Platform(GCP)中唯一地标识您的集群。基础架构名称还用于在 OpenShift Container Platform 安装过程中定位适当的 GCP 资源。提供的 Deployment Manager 模板包含对此基础架构名称的引用，因此您必须提取它。

先决条件

- 已获取 OpenShift Container Platform 安装程序和集群的 pull secret。
- 已为集群生成 Ignition 配置文件。
- 已安装 **jq** 软件包。

流程

- 要从 Ignition 配置文件元数据中提取和查看基础架构名称，请运行以下命令：

```
$ jq -r .infraID <installation_directory>/metadata.json 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

输出示例

```
openshift-vw9j6 1
```

1 此命令的输出是您的集群名称和随机字符串。

9.11.8.2. 为 Deployment Manager 模板导出常用变量

您必须导出与提供的 Deployment Manager 模板一起使用的一组常用变量，它们有助于在 Google Cloud Platform(GCP)上完成用户提供的基础架构安装。

注意

特定的 Deployment Manager 模板可能还需要其他导出变量，这些变量在相关程序中详细介绍。

先决条件

- 获取 OpenShift Container Platform 安装程序和集群的 pull secret。
- 为集群生成 Ignition 配置文件。
- 安装 **jq** 软件包。

流程

1. 导出由提供的 Deployment Manager 模板使用的以下常用变量：

```
$ export BASE_DOMAIN='<base_domain>' 1
$ export BASE_DOMAIN_ZONE_NAME='<base_domain_zone_name>' 2
$ export NETWORK_CIDR='10.0.0.0/16'
```

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 3
$ export CLUSTER_NAME=`jq -r .clusterName <installation_directory>/metadata.json`
$ export INFRA_ID=`jq -r .infraID <installation_directory>/metadata.json`
$ export PROJECT_NAME=`jq -r .gcp.projectID <installation_directory>/metadata.json`
```

1 2 提供主机项目的值。

3 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

9.11.9. 用户置备的基础架构对网络的要求

所有 Red Hat Enterprise Linux CoreOS(RHCOS)机器都需要在启动时在 **initramfs** 中配置联网，以获取它们的 Ignition 配置文件。

9.11.9.1. 通过 DHCP 设置集群节点主机名

在 Red Hat Enterprise Linux CoreOS(RHCOS)机器上，主机名是通过 NetworkManager 设置的。默认情况下，机器通过 DHCP 获取其主机名。如果主机名不是由 DHCP 提供，请通过内核参数或者其它方法进行静态设置，请通过反向 DNS 查找获取。反向 DNS 查找在网络初始化后进行，可能需要一些时间来解决。其他系统服务可以在此之前启动，并将主机名检测为 **localhost** 或类似的内容。您可以使用 DHCP 为每个集群节点提供主机名来避免这种情况。

另外，通过 DHCP 设置主机名可以绕过实施 DNS split-horizon 的环境中的手动 DNS 记录名称配置错误。

9.11.9.2. 网络连接要求

您必须配置机器之间的网络连接，以允许 OpenShift Container Platform 集群组件进行通信。每台机器都必须能够解析集群中所有其他机器的主机名。

本节详细介绍了所需的端口。

重要

在连接的 OpenShift Container Platform 环境中，所有节点都需要访问互联网才能为平台容器拉取镜像，并向红帽提供遥测数据。

表 9.36. 用于全机器到所有机器通信的端口

协议	port	描述
ICMP	N/A	网络可访问性测试
TCP	**1936**	指标
	9000-9999	主机级别的服务，包括端口 9**100-9101 上的节点导出器**，以及端口 **9099** 上的 Cluster Version Operator。
	10250-10259	Kubernetes 保留的默认端口
UDP	**4789**	VXLAN

协议	port	描述
	6081	Geneve
	9000-9999	主机级别的服务，包括端口 **9100**到**9101** 上的节点导出器。
	500	IPsec IKE 数据包
	4500	IPsec NAT-T 数据包
	123	UDP 端口 **123** 上的网络时间协议 (NTP) 如果配置了外部 NTP 时间服务器，需要打开 UDP 端口 **123**。
TCP/UDP	**30000-32767**	Kubernetes 节点端口
ESP	N/A	IPsec Encapsulating Security Payload(ESP)

表 9.37. 用于所有机器控制平面通信的端口

协议	port	描述
TCP	**6443**	Kubernetes API

表 9.38. control plane 机器用于 control plane 机器通信的端口

协议	port	描述
TCP	**2379-2380**	etcd 服务器和对等端口

9.11.10. 在 GCP 中创建负载均衡器

您必须在 Google Cloud Platform(GCP)中配置负载均衡器，供您的 OpenShift Container Platform 集群使用。创建这些组件的一种方法是修改提供的 Deployment Manager 模板。

注意

如果不使用提供的 Deployment Manager 模板来创建 GCP 基础架构，您必须检查提供的信息并手动创建基础架构。如果您的集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 GCP 帐户。
- 为集群生成 Ignition 配置文件。

- 在 GCP 中创建和配置 VPC 及相关子网。

流程

1. 复制 **内部负载均衡器的 Deployment Manager 模板一节中的模板**，并将它以 **02_lb_int.py** 形式保存到计算机上。此模板描述了集群所需的内部负载均衡对象。
2. 对于外部集群，还要复制本主题 **的外部负载均衡器的 Deployment Manager 模板** 一节中的模板，并将它以 **02_lb_ext.py** 形式保存到计算机上。此模板描述了集群所需的外部负载均衡对象。
3. 导出部署模板使用的变量：
   a. 导出集群网络位置：

```
$ export CLUSTER_NETWORK=(`gcloud compute networks describe
${HOST_PROJECT_NETWORK} --project ${HOST_PROJECT} --account
${HOST_PROJECT_ACCOUNT} --format json | jq -r .selfLink`)
```

   b. 导出 control plane 子网位置：

```
$ export CONTROL_SUBNET=(`gcloud compute networks subnets describe
${HOST_PROJECT_CONTROL_SUBNET} --region=${REGION} --project
${HOST_PROJECT} --account ${HOST_PROJECT_ACCOUNT} --format json | jq -r
.selfLink`)
```

   c. 导出集群使用的三个区：

```
$ export ZONE_0=(`gcloud compute regions describe ${REGION} --format=json | jq -r
.zones[0] | cut -d "/" -f9`)
```

```
$ export ZONE_1=(`gcloud compute regions describe ${REGION} --format=json | jq -r
.zones[1] | cut -d "/" -f9`)
```

```
$ export ZONE_2=(`gcloud compute regions describe ${REGION} --format=json | jq -r
.zones[2] | cut -d "/" -f9`)
```

4. 创建 **02_infra.yaml** 资源定义文件：

```
$ cat <<EOF >02_infra.yaml
imports:
- path: 02_lb_ext.py
- path: 02_lb_int.py (1)
resources:
- name: cluster-lb-ext (2)
  type: 02_lb_ext.py
  properties:
    infra_id: '${INFRA_ID}' (3)
    region: '${REGION}' (4)
- name: cluster-lb-int
  type: 02_lb_int.py
  properties:
    cluster_network: '${CLUSTER_NETWORK}'
    control_subnet: '${CONTROL_SUBNET}' (5)
```

```
    infra_id: '${INFRA_ID}'
    region: '${REGION}'
    zones: 6
    - '${ZONE_0}'
    - '${ZONE_1}'
    - '${ZONE_2}'
  EOF
```

1 2 仅在部署外部集群时需要。

3 INFRA _ID 是 提取步骤 中的 **INFRA_ID** 基础架构名称。

4 **region** 是集群要部署到的区域，如 **us-central1**。

5 **control_subnet** 是控制子网的 URI。

6 **zones** 是 control plane 实例要部署到的区域，如 **us-east1-b**、**us-east1-c** 和 **us-east1-d**。

5. 使用 **gcloud** CLI 创建部署：

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-infra --config 02_infra.yaml
```

6. 导出集群 IP 地址：

```
$ export CLUSTER_IP=(`gcloud compute addresses describe ${INFRA_ID}-cluster-ip --
region=${REGION} --format json | jq -r .address`)
```

7. 对于外部集群，还要导出集群公共 IP 地址：

```
$ export CLUSTER_PUBLIC_IP=(`gcloud compute addresses describe ${INFRA_ID}-cluster-
public-ip --region=${REGION} --format json | jq -r .address`)
```

9.11.10.1. 外部负载均衡器的 Deployment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的外部负载均衡器：

例 9.86. 02_lb_ext.py Deployment Manager 模板

```
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-cluster-public-ip',
        'type': 'compute.v1.address',
        'properties': {
            'region': context.properties['region']
        }
    }, {
        # Refer to docs/dev/kube-apiserver-health-check.md on how to correctly setup health check
probe for kube-apiserver
        'name': context.properties['infra_id'] + '-api-http-health-check',
        'type': 'compute.v1.httpHealthCheck',
        'properties': {
```

```
            'port': 6080,
            'requestPath': '/readyz'
        }
    }, {
        'name': context.properties['infra_id'] + '-api-target-pool',
        'type': 'compute.v1.targetPool',
        'properties': {
            'region': context.properties['region'],
            'healthChecks': ['$(ref.' + context.properties['infra_id'] + '-api-http-health-check.selfLink)'],
            'instances': []
        }
    }, {
        'name': context.properties['infra_id'] + '-api-forwarding-rule',
        'type': 'compute.v1.forwardingRule',
        'properties': {
            'region': context.properties['region'],
            'IPAddress': '$(ref.' + context.properties['infra_id'] + '-cluster-public-ip.selfLink)',
            'target': '$(ref.' + context.properties['infra_id'] + '-api-target-pool.selfLink)',
            'portRange': '6443'
        }
    }]

    return {'resources': resources}
```

9.11.10.2. 内部负载均衡器的 Deployment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的内部负载均衡器：

例 9.87. 02_lb_int.py Deployment Manager 模板

```
def GenerateConfig(context):

    backends = []
    for zone in context.properties['zones']:
        backends.append({
            'group': '$(ref.' + context.properties['infra_id'] + '-master-' + zone + '-ig' + '.selfLink)'
        })

    resources = [{
        'name': context.properties['infra_id'] + '-cluster-ip',
        'type': 'compute.v1.address',
        'properties': {
            'addressType': 'INTERNAL',
            'region': context.properties['region'],
            'subnetwork': context.properties['control_subnet']
        }
    }, {
        # Refer to docs/dev/kube-apiserver-health-check.md on how to correctly setup health check
probe for kube-apiserver
        'name': context.properties['infra_id'] + '-api-internal-health-check',
        'type': 'compute.v1.healthCheck',
        'properties': {
            'httpsHealthCheck': {
```

```
                'port': 6443,
                'requestPath': '/readyz'
            },
            'type': "HTTPS"
        }
    }, {
        'name': context.properties['infra_id'] + '-api-internal-backend-service',
        'type': 'compute.v1.regionBackendService',
        'properties': {
            'backends': backends,
            'healthChecks': ['$(ref.' + context.properties['infra_id'] + '-api-internal-health-
check.selfLink)'],
            'loadBalancingScheme': 'INTERNAL',
            'region': context.properties['region'],
            'protocol': 'TCP',
            'timeoutSec': 120
        }
    }, {
        'name': context.properties['infra_id'] + '-api-internal-forwarding-rule',
        'type': 'compute.v1.forwardingRule',
        'properties': {
            'backendService': '$(ref.' + context.properties['infra_id'] + '-api-internal-backend-
service.selfLink)',
            'IPAddress': '$(ref.' + context.properties['infra_id'] + '-cluster-ip.selfLink)',
            'loadBalancingScheme': 'INTERNAL',
            'ports': ['6443','22623'],
            'region': context.properties['region'],
            'subnetwork': context.properties['control_subnet']
        }
    }]

    for zone in context.properties['zones']:
        resources.append({
            'name': context.properties['infra_id'] + '-master-' + zone + '-ig',
            'type': 'compute.v1.instanceGroup',
            'properties': {
                'namedPorts': [
                    {
                        'name': 'ignition',
                        'port': 22623
                    }, {
                        'name': 'https',
                        'port': 6443
                    }
                ],
                'network': context.properties['cluster_network'],
                'zone': zone
            }
        })

    return {'resources': resources}
```

创建外部集群时，除了 **02_lb_ext.py** 模板外，还需要此模板。

9.11.11. 在 GCP 中创建私有 DNS 区域

您必须在 Google Cloud Platform(GCP)中配置私有 DNS 区，供您的 OpenShift Container Platform 集群使用。创建此组件的一种方法是修改提供的 Deployment Manager 模板。

注意

如果不使用提供的 Deployment Manager 模板来创建 GCP 基础架构，您必须检查提供的信息并手动创建基础架构。如果您的集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 GCP 帐户。
- 为集群生成 Ignition 配置文件。
- 在 GCP 中创建和配置 VPC 及相关子网。

流程

1. 复制 **私有 DNS 的 Deployment Manager 模板** 一节中的模板，并将它以 **02_dns.py** 形式保存到计算机上。此模板描述了集群所需的私有 DNS 对象。
2. 创建 **02_dns.yaml** 资源定义文件：

```
$ cat <<EOF >02_dns.yaml
imports:
- path: 02_dns.py

resources:
- name: cluster-dns
  type: 02_dns.py
  properties:
    infra_id: '${INFRA_ID}' (1)
    cluster_domain: '${CLUSTER_NAME}.${BASE_DOMAIN}' (2)
    cluster_network: '${CLUSTER_NETWORK}' (3)
EOF
```

(1) INFRA **_ID 是** 提取步骤 **中的 INFRA_ID** 基础架构名称。

(2) **cluster_domain** 是集群的域，如 **openshift.example.com**。

(3) **cluster_network** 是集群网络的 **selfLink** URL。

3. 使用 **gcloud** CLI 创建部署：

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-dns --config 02_dns.yaml --
project ${HOST_PROJECT} --account ${HOST_PROJECT_ACCOUNT}
```

4. 由于 Deployment Manager 的限制，模板不会创建 DNS 条目，因此您必须手动创建它们：
   a. 添加内部 DNS 条目：

```
$ if [ -f transaction.yaml ]; then rm transaction.yaml; fi
$ gcloud dns record-sets transaction start --zone ${INFRA_ID}-private-zone --project
${HOST_PROJECT} --account ${HOST_PROJECT_ACCOUNT}
$ gcloud dns record-sets transaction add ${CLUSTER_IP} --name
api.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 60 --type A --zone ${INFRA_ID}-
private-zone --project ${HOST_PROJECT} --account ${HOST_PROJECT_ACCOUNT}
$ gcloud dns record-sets transaction add ${CLUSTER_IP} --name api-
int.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 60 --type A --zone ${INFRA_ID}-
private-zone --project ${HOST_PROJECT} --account ${HOST_PROJECT_ACCOUNT}
$ gcloud dns record-sets transaction execute --zone ${INFRA_ID}-private-zone --project
${HOST_PROJECT} --account ${HOST_PROJECT_ACCOUNT}
```

b. 对于外部集群，还要添加外部 DNS 条目：

```
$ if [ -f transaction.yaml ]; then rm transaction.yaml; fi
$ gcloud --account=${HOST_PROJECT_ACCOUNT} --project=${HOST_PROJECT} dns
record-sets transaction start --zone ${BASE_DOMAIN_ZONE_NAME}
$ gcloud --account=${HOST_PROJECT_ACCOUNT} --project=${HOST_PROJECT} dns
record-sets transaction add ${CLUSTER_PUBLIC_IP} --name
api.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 60 --type A --zone
${BASE_DOMAIN_ZONE_NAME}
$ gcloud --account=${HOST_PROJECT_ACCOUNT} --project=${HOST_PROJECT} dns
record-sets transaction execute --zone ${BASE_DOMAIN_ZONE_NAME}
```

9.11.11.1. 私有 DNS 的 Deployment Manager 模板

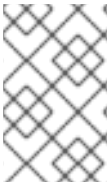
您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的私有 DNS：

例 9.88. 02_DNS.py Deployment Manager 模板

```
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-private-zone',
        'type': 'dns.v1.managedZone',
        'properties': {
            'description': '',
            'dnsName': context.properties['cluster_domain'] + '.',
            'visibility': 'private',
            'privateVisibilityConfig': {
                'networks': [{
                    'networkUrl': context.properties['cluster_network']
                }]
            }
        }
    }]

    return {'resources': resources}
```

9.11.12. 在 GCP 中创建防火墙规则

您必须在 Google Cloud Platform(GCP)中创建防火墙规则，供您的 OpenShift Container Platform 集群使用。创建这些组件的一种方法是修改提供的 Deployment Manager 模板。



注意

如果不使用提供的 Deployment Manager 模板来创建 GCP 基础架构，您必须检查提供的信息并手动创建基础架构。如果您的集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 GCP 帐户。
- 为集群生成 Ignition 配置文件。
- 在 GCP 中创建和配置 VPC 及相关子网。

流程

1. 复制 **防火墙规则的 Deployment Manager 模板一节中的模板**，并将它以 **03_firewall.py** 形式保存到计算机上。此模板描述了集群所需的安全组。
2. 创建 **03_firewall.yaml** 资源定义文件：

```
$ cat <<EOF >03_firewall.yaml
imports:
- path: 03_firewall.py

resources:
- name: cluster-firewall
  type: 03_firewall.py
  properties:
    allowed_external_cidr: '0.0.0.0/0' (1)
    infra_id: '${INFRA_ID}' (2)
    cluster_network: '${CLUSTER_NETWORK}' (3)
    network_cidr: '${NETWORK_CIDR}' (4)
EOF
```

(1) **allowed_external_cidr** 是 CIDR 范围，可访问集群 API 和 SSH 到 bootstrap 主机。对于内部集群，将此值设置为 **${NETWORK_CIDR}**。

(2) INFRA **_ID 是** 提取步骤 **中的 INFRA_ID** 基础架构名称。

(3) **cluster_network** 是集群网络的 **selfLink** URL。

(4) **NETWORK_CIDR 是** VPC 网络的 CIDR，如 **10.0.0.0/16**。

3. 使用 **gcloud** CLI 创建部署：

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-firewall --config
03_firewall.yaml --project ${HOST_PROJECT} --account ${HOST_PROJECT_ACCOUNT}
```

9.11.12.1. 防火墙规则的 Deployment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的防火墙破坏：

例 9.89. 03_firewall.py Deployment Manager 模板

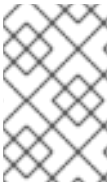
```
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-bootstrap-in-ssh',
        'type': 'compute.v1.firewall',
        'properties': {
            'network': context.properties['cluster_network'],
            'allowed': [{
                'IPProtocol': 'tcp',
                'ports': ['22']
            }],
            'sourceRanges': [context.properties['allowed_external_cidr']],
            'targetTags': [context.properties['infra_id'] + '-bootstrap']
        }
    }, {
        'name': context.properties['infra_id'] + '-api',
        'type': 'compute.v1.firewall',
        'properties': {
            'network': context.properties['cluster_network'],
            'allowed': [{
                'IPProtocol': 'tcp',
                'ports': ['6443']
            }],
            'sourceRanges': [context.properties['allowed_external_cidr']],
            'targetTags': [context.properties['infra_id'] + '-master']
        }
    }, {
        'name': context.properties['infra_id'] + '-health-checks',
        'type': 'compute.v1.firewall',
        'properties': {
            'network': context.properties['cluster_network'],
            'allowed': [{
                'IPProtocol': 'tcp',
                'ports': ['6080', '6443', '22624']
            }],
            'sourceRanges': ['35.191.0.0/16', '130.211.0.0/22', '209.85.152.0/22', '209.85.204.0/22'],
            'targetTags': [context.properties['infra_id'] + '-master']
        }
    }, {
        'name': context.properties['infra_id'] + '-etcd',
        'type': 'compute.v1.firewall',
        'properties': {
            'network': context.properties['cluster_network'],
            'allowed': [{
                'IPProtocol': 'tcp',
                'ports': ['2379-2380']
            }],
            'sourceTags': [context.properties['infra_id'] + '-master'],
            'targetTags': [context.properties['infra_id'] + '-master']
        }
    }, {
```

```
        'name': context.properties['infra_id'] + '-control-plane',
        'type': 'compute.v1.firewall',
        'properties': {
            'network': context.properties['cluster_network'],
            'allowed': [{
                'IPProtocol': 'tcp',
                'ports': ['10257']
            },{
                'IPProtocol': 'tcp',
                'ports': ['10259']
            },{
                'IPProtocol': 'tcp',
                'ports': ['22623']
            }],
            'sourceTags': [
                context.properties['infra_id'] + '-master',
                context.properties['infra_id'] + '-worker'
            ],
            'targetTags': [context.properties['infra_id'] + '-master']
        }
    }, {
        'name': context.properties['infra_id'] + '-internal-network',
        'type': 'compute.v1.firewall',
        'properties': {
            'network': context.properties['cluster_network'],
            'allowed': [{
                'IPProtocol': 'icmp'
            },{
                'IPProtocol': 'tcp',
                'ports': ['22']
            }],
            'sourceRanges': [context.properties['network_cidr']],
            'targetTags': [
                context.properties['infra_id'] + '-master',
                context.properties['infra_id'] + '-worker'
            ]
        }
    }, {
        'name': context.properties['infra_id'] + '-internal-cluster',
        'type': 'compute.v1.firewall',
        'properties': {
            'network': context.properties['cluster_network'],
            'allowed': [{
                'IPProtocol': 'udp',
                'ports': ['4789', '6081']
            },{
                'IPProtocol': 'udp',
                'ports': ['500', '4500']
            },{
                'IPProtocol': 'esp',
            },{
                'IPProtocol': 'tcp',
                'ports': ['9000-9999']
            },{
                'IPProtocol': 'udp',
                'ports': ['9000-9999']
```

```
            },{
                'IPProtocol': 'tcp',
                'ports': ['10250']
            },{
                'IPProtocol': 'tcp',
                'ports': ['30000-32767']
            },{
                'IPProtocol': 'udp',
                'ports': ['30000-32767']
            }],
            'sourceTags': [
                context.properties['infra_id'] + '-master',
                context.properties['infra_id'] + '-worker'
            ],
            'targetTags': [
                context.properties['infra_id'] + '-master',
                context.properties['infra_id'] + '-worker'
            ]
        }
    }]

    return {'resources': resources}
```

9.11.13. 在 GCP 中创建 IAM 角色

您必须在 Google Cloud Platform(GCP)中创建 IAM 角色，供您的 OpenShift Container Platform 集群使用。创建这些组件的一种方法是修改提供的 Deployment Manager 模板。



注意

如果不使用提供的 Deployment Manager 模板来创建 GCP 基础架构，您必须检查提供的信息并手动创建基础架构。如果您的集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 GCP 帐户。
- 为集群生成 Ignition 配置文件。
- 在 GCP 中创建和配置 VPC 及相关子网。

流程

1. 复制 **IAM 角色的 Deployment Manager 模板一节中的模板**，并将它以 **03_iam.py** 形式保存到计算机上。此模板描述了集群所需的 IAM 角色。
2. 创建 **03_iam.yaml** 资源定义文件：

```
$ cat <<EOF >03_iam.yaml
imports:
- path: 03_iam.py
resources:
```

```
- name: cluster-iam
  type: 03_iam.py
  properties:
    infra_id: '${INFRA_ID}' 1
EOF
```

1 INFRA _ID 是 提取步骤 中的 INFRA_ID 基础架构名称。

3. 使用 **gcloud** CLI 创建部署：

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-iam --config 03_iam.yaml
```

4. 导出 master 服务帐户的变量：

```
$ export MASTER_SERVICE_ACCOUNT=(`gcloud iam service-accounts list --filter
"email~^${INFRA_ID}-m@${PROJECT_NAME}." --format json | jq -r '.[0].email'`)
```

5. 导出 worker 服务帐户的变量：

```
$ export WORKER_SERVICE_ACCOUNT=(`gcloud iam service-accounts list --filter
"email~^${INFRA_ID}-w@${PROJECT_NAME}." --format json | jq -r '.[0].email'`)
```

6. 将安装程序所需的权限分配给托管 control plane 和计算子网的子网的服务帐户：

   a. 为 master 服务帐户授予托管共享 VPC 的项目 **networkViewer** 角色：

```
$ gcloud --account=${HOST_PROJECT_ACCOUNT} --project=${HOST_PROJECT}
projects add-iam-policy-binding ${HOST_PROJECT} --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role
"roles/compute.networkViewer"
```

   b. 将 **networkUser** 角色授予 control plane 子网的 master 服务帐户：

```
$ gcloud --account=${HOST_PROJECT_ACCOUNT} --project=${HOST_PROJECT}
compute networks subnets add-iam-policy-binding
"${HOST_PROJECT_CONTROL_SUBNET}" --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/compute.networkUser"
--region ${REGION}
```

   c. 将 **networkUser** 角色授予 control plane 子网的 worker 服务帐户：

```
$ gcloud --account=${HOST_PROJECT_ACCOUNT} --project=${HOST_PROJECT}
compute networks subnets add-iam-policy-binding
"${HOST_PROJECT_CONTROL_SUBNET}" --member
"serviceAccount:${WORKER_SERVICE_ACCOUNT}" --role
"roles/compute.networkUser" --region ${REGION}
```

   d. 将 **networkUser** 角色授予计算子网的 master 服务帐户：

```
$ gcloud --account=${HOST_PROJECT_ACCOUNT} --project=${HOST_PROJECT}
compute networks subnets add-iam-policy-binding
"${HOST_PROJECT_COMPUTE_SUBNET}" --member
```

```
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/compute.networkUser"
--region ${REGION}
```

e. 将 **networkUser** 角色授予计算子网的 worker 服务帐户：

```
$ gcloud --account=${HOST_PROJECT_ACCOUNT} --project=${HOST_PROJECT}
compute networks subnets add-iam-policy-binding
"${HOST_PROJECT_COMPUTE_SUBNET}" --member
"serviceAccount:${WORKER_SERVICE_ACCOUNT}" --role
"roles/compute.networkUser" --region ${REGION}
```

7. 由于 Deployment Manager 的限制，模板不会创建策略绑定，因此您必须手动创建它们：

```
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/compute.instanceAdmin"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/compute.networkAdmin"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/compute.securityAdmin"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/iam.serviceAccountUser"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/storage.admin"

$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${WORKER_SERVICE_ACCOUNT}" --role "roles/compute.viewer"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${WORKER_SERVICE_ACCOUNT}" --role "roles/storage.admin"
```

8. 创建服务帐户密钥并将其存储在本地以供以后使用：

```
$ gcloud iam service-accounts keys create service-account-key.json --iam-
account=${MASTER_SERVICE_ACCOUNT}
```

9.11.13.1. IAM 角色的 Deployment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的 IAM 角色：

例 9.90. 03_IAM.py Deployment Manager 模板

```
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-master-node-sa',
        'type': 'iam.v1.serviceAccount',
        'properties': {
            'accountId': context.properties['infra_id'] + '-m',
            'displayName': context.properties['infra_id'] + '-master-node'
        }
    }, {
        'name': context.properties['infra_id'] + '-worker-node-sa',
        'type': 'iam.v1.serviceAccount',
        'properties': {
```

```
            'accountId': context.properties['infra_id'] + '-w',
            'displayName': context.properties['infra_id'] + '-worker-node'
        }
    }]

    return {'resources': resources}
```

9.11.14. 为 GCP 基础架构创建 RHCOS 集群镜像

您必须对 OpenShift Container Platform 节点的 Google Cloud Platform(GCP)使用有效的 Red Hat Enterprise Linux CoreOS(RHCOS)镜像。

流程

1. 从 RHCOS 镜像镜像页面获取 RHCOS 镜像。

重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每个发行版本而改变。您必须下载最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。如果可用，请使用与 OpenShift Container Platform 版本匹配的镜像版本。

文件名包含 OpenShift Container Platform 版本号，格式为 **rhcos-<version>-<arch>-gcp.<arch>.tar.gz**。

2. 创建 Google 存储桶：

```
$ gsutil mb gs://<bucket_name>
```

3. 将 RHCOS 镜像上传到 Google 存储桶：

```
$ gsutil cp <downloaded_image_file_path>/rhcos-<version>-x86_64-gcp.x86_64.tar.gz
gs://<bucket_name>
```

4. 将上传的 RHCOS 镜像位置导出为变量：

```
$ export IMAGE_SOURCE=gs://<bucket_name>/rhcos-<version>-x86_64-gcp.x86_64.tar.gz
```

5. 创建集群镜像：

```
$ gcloud compute images create "${INFRA_ID}-rhcos-image" \
    --source-uri="${IMAGE_SOURCE}"
```

9.11.15. 在 GCP 中创建 bootstrap 机器

您必须在 Google Cloud Platform(GCP)中创建 bootstrap 机器，以便在 OpenShift Container Platform 集群初始化过程中使用。创建此机器的一种方法是修改提供的 Deployment Manager 模板。

注意

如果不使用提供的 Deployment Manager 模板来创建 bootstrap 机器，您必须检查提供的信息并手动创建基础架构。如果您的集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 GCP 帐户。
- 为集群生成 Ignition 配置文件。
- 在 GCP 中创建和配置 VPC 及相关子网。
- 在 GCP 中创建和配置联网与负载均衡器。
- 创建 control plane 和计算角色。
- 确保已安装 pyOpenSSL。

流程

1. 复制 **bootstrap 机器的 Deployment Manager 模板一节中的模板**，并将它以 **04_bootstrap.py** 形式保存到计算机上。此模板描述了集群所需的 bootstrap 机器。
2. 导出安装程序所需的 Red Hat Enterprise Linux CoreOS(RHCOS)镜像的位置：

```
$ export CLUSTER_IMAGE=(`gcloud compute images describe ${INFRA_ID}-rhcos-image --format json | jq -r .selfLink`)
```

3. 创建存储桶并上传 **bootstrap.ign** 文件：

```
$ gsutil mb gs://${INFRA_ID}-bootstrap-ignition
```

```
$ gsutil cp <installation_directory>/bootstrap.ign gs://${INFRA_ID}-bootstrap-ignition/
```

4. 为 bootstrap 实例创建一个签名的 URL，用于访问 Ignition 配置。将输出中的 URL 导出为变量：

```
$ export BOOTSTRAP_IGN=`gsutil signurl -d 1h service-account-key.json gs://${INFRA_ID}-bootstrap-ignition/bootstrap.ign | grep "^gs:" | awk '{print $5}'`
```

5. 创建 **04_bootstrap.yaml** 资源定义文件：

```
$ cat <<EOF >04_bootstrap.yaml
imports:
- path: 04_bootstrap.py

resources:
- name: cluster-bootstrap
  type: 04_bootstrap.py
  properties:
    infra_id: '${INFRA_ID}' (1)
    region: '${REGION}' (2)
    zone: '${ZONE_0}' (3)
```

```
  cluster_network: '${CLUSTER_NETWORK}' (4)
  control_subnet: '${CONTROL_SUBNET}' (5)
  image: '${CLUSTER_IMAGE}' (6)
  machine_type: 'n1-standard-4' (7)
  root_volume_size: '128' (8)

  bootstrap_ign: '${BOOTSTRAP_IGN}' (9)
EOF
```

(1) INFRA _ID 是 提取步骤 中的 **INFRA_ID** 基础架构名称。

(2) **region** 是集群要部署到的区域，如 **us-central1**。

(3) **zone** 是 bootstrap 实例要部署到的区域，如 **us-central1-b**。

(4) **cluster_network** 是集群网络的 **selfLink** URL。

(5) **control_subnet** 是控制子网的 **selfLink** URL。

(6) **image** 是 RHCOS 镜像的 **selfLink** URL。

(7) **machine_type** 是实例的机器类型，如 **n1-standard-4**。

(8) **root_volume_size** 是 bootstrap 计算机的引导磁盘大小。

(9) **bootstrap_ign** 是创建签名 URL 时的 URL 输出。

6. 使用 **gcloud** CLI 创建部署：

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-bootstrap --config
04_bootstrap.yaml
```

7. 将 bootstrap 实例添加到内部负载均衡器实例组中：

```
$ gcloud compute instance-groups unmanaged add-instances ${INFRA_ID}-bootstrap-ig --
zone=${ZONE_0} --instances=${INFRA_ID}-bootstrap
```

8. 将 bootstrap 实例组添加到内部负载均衡器后端服务中：

```
$ gcloud compute backend-services add-backend ${INFRA_ID}-api-internal-backend-service
--region=${REGION} --instance-group=${INFRA_ID}-bootstrap-ig --instance-group-
zone=${ZONE_0}
```

9.11.15.1. bootstrap 机器的 Deployment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的 bootstrap 机器：

例 9.91. 04_bootstrap.py Deployment Manager 模板

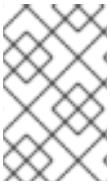
```
def GenerateConfig(context):
```

```
    resources = [{
        'name': context.properties['infra_id'] + '-bootstrap-public-ip',
        'type': 'compute.v1.address',
        'properties': {
            'region': context.properties['region']
        }
    }, {
        'name': context.properties['infra_id'] + '-bootstrap',
        'type': 'compute.v1.instance',
        'properties': {
            'disks': [{
                'autoDelete': True,
                'boot': True,
                'initializeParams': {
                    'diskSizeGb': context.properties['root_volume_size'],
                    'sourceImage': context.properties['image']
                }
            }],
            'machineType': 'zones/' + context.properties['zone'] + '/machineTypes/' +
context.properties['machine_type'],
            'metadata': {
                'items': [{
                    'key': 'user-data',
                    'value': '{"ignition":{"config":{"replace":{"source":"' + context.properties['bootstrap_ign']
+ '"}},"version":"3.2.0"}}',
                }]
            },
            'networkInterfaces': [{
                'subnetwork': context.properties['control_subnet'],
                'accessConfigs': [{
                    'natIP': '$(ref.' + context.properties['infra_id'] + '-bootstrap-public-ip.address)'
                }]
            }],
            'tags': {
                'items': [
                    context.properties['infra_id'] + '-master',
                    context.properties['infra_id'] + '-bootstrap'
                ]
            },
            'zone': context.properties['zone']
        }
    }, {
        'name': context.properties['infra_id'] + '-bootstrap-ig',
        'type': 'compute.v1.instanceGroup',
        'properties': {
            'namedPorts': [
                {
                    'name': 'ignition',
                    'port': 22623
                }, {
                    'name': 'https',
                    'port': 6443
                }
            ],
            'network': context.properties['cluster_network'],
            'zone': context.properties['zone']
```

```
    }
}]

return {'resources': resources}
```

9.11.16. 在 GCP 中创建 control plane 机器

您必须在 Google Cloud Platform(GCP)中创建 control plane 机器，供您的集群使用。创建这些机器的一种方法是修改提供的 Deployment Manager 模板。



注意

如果不使用提供的 Deployment Manager 模板来创建 control plane 机器，您必须检查提供的信息并手动创建基础架构。如果您的集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 GCP 帐户。
- 为集群生成 Ignition 配置文件。
- 在 GCP 中创建和配置 VPC 及相关子网。
- 在 GCP 中创建和配置联网与负载均衡器。
- 创建 control plane 和计算角色。
- 创建 bootstrap 机器。

流程

1. 复制 **control plane 机器的 Deployment Manager 模板一节中的模板**，并将它以 **05_control_plane.py** 形式保存到计算机上。此模板描述了集群所需的 control plane 机器。
2. 导出资源定义所需的以下变量：

```
$ export MASTER_IGNITION=`cat <installation_directory>/master.ign`
```

3. 创建 **05_control_plane.yaml** 资源定义文件：

```
$ cat <<EOF >05_control_plane.yaml
imports:
- path: 05_control_plane.py

resources:
- name: cluster-control-plane
  type: 05_control_plane.py
  properties:
    infra_id: '${INFRA_ID}' (1)
    zones: (2)
    - '${ZONE_0}'
    - '${ZONE_1}'
```

```
    - '${ZONE_2}'

    control_subnet: '${CONTROL_SUBNET}' 3
    image: '${CLUSTER_IMAGE}' 4
    machine_type: 'n1-standard-4' 5
    root_volume_size: '128'
    service_account_email: '${MASTER_SERVICE_ACCOUNT}' 6

    ignition: '${MASTER_IGNITION}' 7
EOF
```

1. INFRA _ID 是 提取步骤 中的 **INFRA_ID** 基础架构名称。
2. **zones** 是 control plane 实例要部署到的区域，如 **us-central1-a**、**us-central1-b** 和 **us-central1-c**。
3. **control_subnet** 是控制子网的 **selfLink** URL。
4. **image** 是 RHCOS 镜像的 **selfLink** URL。
5. **machine_type** 是实例的机器类型，如 **n1-standard-4**。
6. **service_account_email** 是您创建的 master 服务帐户的电子邮件地址。
7. **Ignition** 是 **master.ign** 文件的内容。

4. 使用 **gcloud** CLI 创建部署：

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-control-plane --config
05_control_plane.yaml
```

5. 由于 Deployment Manager 的限制，模板无法管理负载均衡器成员资格，因此您必须手动添加 control plane 机器。
   - 运行以下命令，将 control plane 机器添加到适当的实例组中：

```
$ gcloud compute instance-groups unmanaged add-instances ${INFRA_ID}-master-
${ZONE_0}-ig --zone=${ZONE_0} --instances=${INFRA_ID}-master-0
```

```
$ gcloud compute instance-groups unmanaged add-instances ${INFRA_ID}-master-
${ZONE_1}-ig --zone=${ZONE_1} --instances=${INFRA_ID}-master-1
```

```
$ gcloud compute instance-groups unmanaged add-instances ${INFRA_ID}-master-
${ZONE_2}-ig --zone=${ZONE_2} --instances=${INFRA_ID}-master-2
```

   - 对于外部集群，还必须运行以下命令将 control plane 机器添加到目标池中：

```
$ gcloud compute target-pools add-instances ${INFRA_ID}-api-target-pool --instances-
zone="${ZONE_0}" --instances=${INFRA_ID}-master-0
```

```
$ gcloud compute target-pools add-instances ${INFRA_ID}-api-target-pool --instances-
zone="${ZONE_1}" --instances=${INFRA_ID}-master-1
```

```
$ gcloud compute target-pools add-instances ${INFRA_ID}-api-target-pool --instances-
zone="${ZONE_2}" --instances=${INFRA_ID}-master-2
```

9.11.16.1. control plane 机器的 Deployment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的 control plane 机器：

例 9.92. 05_control_plane.py Deployment Manager 模板

```
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-master-0',
        'type': 'compute.v1.instance',
        'properties': {
            'disks': [{
                'autoDelete': True,
                'boot': True,
                'initializeParams': {
                    'diskSizeGb': context.properties['root_volume_size'],
                    'diskType': 'zones/' + context.properties['zones'][0] + '/diskTypes/pd-ssd',
                    'sourceImage': context.properties['image']
                }
            }],
            'machineType': 'zones/' + context.properties['zones'][0] + '/machineTypes/' +
context.properties['machine_type'],
            'metadata': {
                'items': [{
                    'key': 'user-data',
                    'value': context.properties['ignition']
                }]
            },
            'networkInterfaces': [{
                'subnetwork': context.properties['control_subnet']
            }],
            'serviceAccounts': [{
                'email': context.properties['service_account_email'],
                'scopes': ['https://www.googleapis.com/auth/cloud-platform']
            }],
            'tags': {
                'items': [
                    context.properties['infra_id'] + '-master',
                ]
            },
            'zone': context.properties['zones'][0]
        }
    }, {
        'name': context.properties['infra_id'] + '-master-1',
        'type': 'compute.v1.instance',
        'properties': {
            'disks': [{
                'autoDelete': True,
                'boot': True,
                'initializeParams': {
```

```
                'diskSizeGb': context.properties['root_volume_size'],
                'diskType': 'zones/' + context.properties['zones'][1] + '/diskTypes/pd-ssd',
                'sourceImage': context.properties['image']
            }
        }],
        'machineType': 'zones/' + context.properties['zones'][1] + '/machineTypes/' +
context.properties['machine_type'],
        'metadata': {
            'items': [{
                'key': 'user-data',
                'value': context.properties['ignition']
            }]
        },
        'networkInterfaces': [{
            'subnetwork': context.properties['control_subnet']
        }],
        'serviceAccounts': [{
            'email': context.properties['service_account_email'],
            'scopes': ['https://www.googleapis.com/auth/cloud-platform']
        }],
        'tags': {
            'items': [
                context.properties['infra_id'] + '-master',
            ]
        },
        'zone': context.properties['zones'][1]
    }
}, {
    'name': context.properties['infra_id'] + '-master-2',
    'type': 'compute.v1.instance',
    'properties': {
        'disks': [{
            'autoDelete': True,
            'boot': True,
            'initializeParams': {
                'diskSizeGb': context.properties['root_volume_size'],
                'diskType': 'zones/' + context.properties['zones'][2] + '/diskTypes/pd-ssd',
                'sourceImage': context.properties['image']
            }
        }],
        'machineType': 'zones/' + context.properties['zones'][2] + '/machineTypes/' +
context.properties['machine_type'],
        'metadata': {
            'items': [{
                'key': 'user-data',
                'value': context.properties['ignition']
            }]
        },
        'networkInterfaces': [{
            'subnetwork': context.properties['control_subnet']
        }],
        'serviceAccounts': [{
            'email': context.properties['service_account_email'],
            'scopes': ['https://www.googleapis.com/auth/cloud-platform']
        }],
        'tags': {
```

```
            'items': [
                context.properties['infra_id'] + '-master',
            ]
        },
        'zone': context.properties['zones'][2]
    }
}]

return {'resources': resources}
```

9.11.17. 等待 bootstrap 完成并删除 GCP 中的 bootstrap 资源

在 Google Cloud Platform(GCP)中创建所有所需的基础架构后，请等待您通过安装程序生成的 Ignition 配置文件所置备的机器上完成 bootstrap 过程。

先决条件

- 配置 GCP 帐户。
- 为集群生成 Ignition 配置文件。
- 在 GCP 中创建和配置 VPC 及相关子网。
- 在 GCP 中创建和配置联网与负载均衡器。
- 创建 control plane 和计算角色。
- 创建 bootstrap 机器。
- 创建 control plane 机器。

流程

1. 进入包含安装程序的目录并运行以下命令：

```
$ ./openshift-install wait-for bootstrap-complete --dir <installation_directory> \ 1
    --log-level info 2
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不是 **info**。

如果命令退出时没有 **FATAL** 警告，则您的生产环境 control plane 已被初始化。

2. 删除 bootstrap 资源：

```
$ gcloud compute backend-services remove-backend ${INFRA_ID}-api-internal-backend-
service --region=${REGION} --instance-group=${INFRA_ID}-bootstrap-ig --instance-group-
zone=${ZONE_0}
```

```
$ gsutil rm gs://${INFRA_ID}-bootstrap-ignition/bootstrap.ign
```

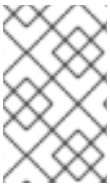
```
$ gsutil rb gs://${INFRA_ID}-bootstrap-ignition
```

```
$ gcloud deployment-manager deployments delete ${INFRA_ID}-bootstrap
```

9.11.18. 在 GCP 中创建额外的 worker 机器

您可以通过分散启动各个实例或利用集群外自动化流程（如自动扩展组），在 Google Cloud Platform(GCP)中为您的集群创建 worker 机器。您还可以利用 OpenShift Container Platform 中的内置集群扩展机制和机器 API。

在本例中，您要使用 Deployment Manager 模板手动启动一个实例。通过在 文件中包括类型为 **06_worker.py** 的其他资源，即可启动其他实例。



注意

如果不使用提供的 Deployment Manager 模板来创建 worker 机器，您必须检查提供的信息并手动创建基础架构。如果您的集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 GCP 帐户。
- 为集群生成 Ignition 配置文件。
- 在 GCP 中创建和配置 VPC 及相关子网。
- 在 GCP 中创建和配置联网与负载均衡器。
- 创建 control plane 和计算角色。
- 创建 bootstrap 机器。
- 创建 control plane 机器。

流程

1. 复制 **worker 机器的 Deployment Manager 模板一节中的模板**，并将它以 **06_worker.py** 形式保存到计算机上。此模板描述了集群所需的 worker 机器。
2. 导出资源定义使用的变量。
   a. 导出托管计算机器的子网：

```
$ export COMPUTE_SUBNET=(`gcloud compute networks subnets describe
${HOST_PROJECT_COMPUTE_SUBNET} --region=${REGION} --project
${HOST_PROJECT} --account ${HOST_PROJECT_ACCOUNT} --format json | jq -r
.selfLink`)
```

   b. 为您的服务帐户导出电子邮件地址：

```
$ export WORKER_SERVICE_ACCOUNT=(`gcloud iam service-accounts list --filter
"email~^${INFRA_ID}-w@${PROJECT_NAME}." --format json | jq -r '.[0].email'`)
```

c. 导出计算机器 Ignition 配置文件的位置：

```
$ export WORKER_IGNITION=`cat <installation_directory>/worker.ign`
```

3. 创建 **06_worker.yaml** 资源定义文件：

```
$ cat <<EOF >06_worker.yaml
imports:
- path: 06_worker.py

resources:
- name: 'worker-0' (1)
  type: 06_worker.py
  properties:
    infra_id: '${INFRA_ID}' (2)
    zone: '${ZONE_0}' (3)
    compute_subnet: '${COMPUTE_SUBNET}' (4)
    image: '${CLUSTER_IMAGE}' (5)
    machine_type: 'n1-standard-4' (6)
    root_volume_size: '128'
    service_account_email: '${WORKER_SERVICE_ACCOUNT}' (7)
    ignition: '${WORKER_IGNITION}' (8)
- name: 'worker-1'
  type: 06_worker.py
  properties:
    infra_id: '${INFRA_ID}' (9)
    zone: '${ZONE_1}' (10)
    compute_subnet: '${COMPUTE_SUBNET}' (11)
    image: '${CLUSTER_IMAGE}' (12)
    machine_type: 'n1-standard-4' (13)
    root_volume_size: '128'
    service_account_email: '${WORKER_SERVICE_ACCOUNT}' (14)
    ignition: '${WORKER_IGNITION}' (15)
EOF
```

(1) **name** 是 worker 机器的名称，如 **worker-0**。

(2) (9) INFRA _**ID 是** 提取步骤 **中的 INFRA_ID** 基础架构名称。

(3) (10) **zone** 是 worker 机器要部署到的区域，如 **us-central1-a**。

(4) (11) **compute_subnet** 是计算子网的 **selfLink** URL。

(5) (12) **image** 是 RHCOS 镜像的 **selfLink** URL。[1]

(6) (13) **machine_type** 是实例的机器类型，如 **n1-standard-4**。

(7) (14) **service_account_email** 是您创建的 worker 服务帐户的电子邮件地址。

(8) (15) **Ignition** 是 **worker.ign** 文件的内容。

4. 可选：如果要启动更多实例，请在 **06_worker.yaml** 资源定义文件中包含类型为 **06_worker.py** 的其他资源。

5. 使用 **gcloud** CLI 创建部署：

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-worker --config
06_worker.yaml
```

1. 要使用 GCP Marketplace 镜像，请指定要使用的功能：

- OpenShift Container Platform: **https://www.googleapis.com/compute/v1/projects/redhat-marketplace-public/global/images/redhat-coreos-ocp-413-x86-64-202305021736**
- OpenShift Platform Plus: **https://www.googleapis.com/compute/v1/projects/redhat-marketplace-public/global/images/redhat-coreos-opp-413-x86-64-202305021736**
- OpenShift Kubernetes Engine: **https://www.googleapis.com/compute/v1/projects/redhat-marketplace-public/global/images/redhat-coreos-oke-413-x86-64-202305021736**

9.11.18.1. worker 机器的 Deployment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的 worker 机器：

例 9.93. 06_worker.py Deployment Manager 模板

```
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-' + context.env['name'],
        'type': 'compute.v1.instance',
        'properties': {
            'disks': [{
                'autoDelete': True,
                'boot': True,
                'initializeParams': {
                    'diskSizeGb': context.properties['root_volume_size'],
                    'sourceImage': context.properties['image']
                }
            }],
            'machineType': 'zones/' + context.properties['zone'] + '/machineTypes/' +
context.properties['machine_type'],
            'metadata': {
                'items': [{
                    'key': 'user-data',
                    'value': context.properties['ignition']
                }]
            },
            'networkInterfaces': [{
                'subnetwork': context.properties['compute_subnet']
            }],
            'serviceAccounts': [{
                'email': context.properties['service_account_email'],
                'scopes': ['https://www.googleapis.com/auth/cloud-platform']
            }],
            'tags': {
```

```
            'items': [
                context.properties['infra_id'] + '-worker',
            ]
        },
        'zone': context.properties['zone']
    }
}]

return {'resources': resources}
```

9.11.19. 安装 OpenShift CLI

您可以安装 OpenShift CLI(**oc**)来使用命令行界面与 OpenShift Container Platform 进行交互。您可以在 Linux、Windows 或 macOS 上安装 **oc**。

重要

如果安装了旧版本的 **oc**，则可能无法使用 OpenShift Container Platform 4.16 中的所有命令。下载并安装新版本的 **oc**。

在 Linux 上安装 OpenShift CLI
您可以按照以下流程在 Linux 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **产品变体** 下拉列表中选择架构。
3. 从 **版本** 下拉列表中选择适当的版本。
4. 点 **OpenShift v4.16 Linux Client** 条目旁的 **Download Now** 来保存文件。
5. 解包存档：

```
$ tar xvf <file>
```

6. 将 **oc** 二进制文件放到 **PATH** 中的目录中。
   要查看您的 **PATH**，请执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

在 Windows 上安装 OpenShift CLI
您可以按照以下流程在 Windows 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 Windows Client** 条目旁的 **Download Now** 来保存文件。
4. 使用 ZIP 程序解压存档。
5. 将 **oc** 二进制文件移到 **PATH 中的目录中**。
   要查看您的 **PATH**，请打开命令提示并执行以下命令：

```
C:\> path
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

在 macOS 上安装 OpenShift CLI
您可以按照以下流程在 macOS 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 macOS Client** 条目旁的 **Download Now** 来保存文件。

注意

对于 macOS arm64，请选择 **OpenShift v4.16 macOS arm64 Client** 条目。

4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
   要查看您的 **PATH**，请打开终端并执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

9.11.20. 使用 CLI 登录集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

9.11.21. 批准机器的证书签名请求

当您将机器添加到集群时，会为您添加的每台机器生成两个待处理证书签名请求(CSR)。您必须确认这些 CSR 已获得批准，或根据需要自行批准。必须首先批准客户端请求，然后批准服务器请求。

先决条件

- 您已将机器添加到集群中。

流程

1. 确认集群可以识别这些机器：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0 Ready    master  63m  v1.29.4
master-1 Ready    master  63m  v1.29.4
master-2 Ready    master  64m  v1.29.4
```

输出中列出了您创建的所有机器。

注意

在有些 CSR 被批准前，前面的输出可能不包括计算节点（也称为 worker 节点）。

2. 检查待处理的 CSR，并确保添加到集群中的每台机器都有 **Pending** 或 **Approved** 状态的客户端请求：

```
$ oc get csr
```

输出示例

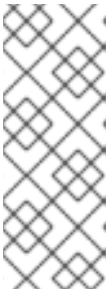
```
NAME        AGE    REQUESTOR                                                   CONDITION
csr-8b2br   15m    system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
csr-8vnps   15m    system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
...
```

在本例中，两台机器加入集群。您可能会在列表中看到更多已批准的 CSR。

3. 如果 CSR 没有获得批准，在您添加的机器的所有待处理 CSR 都处于 **Pending 状态** 后，请批准集群机器的 CSR：

注意

由于 CSR 会自动轮转，因此请在将机器添加到集群后一小时内批准您的 CSR。如果没有在一小时内批准它们，证书将会轮转，每个节点会存在多个证书。您必须批准所有这些证书。批准客户端 CSR 后，Kubelet 为服务证书创建一个二级 CSR，这需要手动批准。然后，如果 Kubelet 请求具有相同参数的新证书，则后续提供证书续订请求由 **machine-approver** 自动批准。



注意

对于在未启用机器 API 的平台上运行的集群，如裸机和其他用户置备的基础架构，您必须实施一种方法来自动批准 kubelet 提供证书请求(CSR)。如果没有批准请求，则 **oc exec**、oc**rsh** 和 **oc logs** 命令将无法成功，因为 API 服务器连接到 kubelet 时需要服务证书。与 Kubelet 端点联系的任何操作都需要此证书批准。该方法必须监视新的 CSR，确认 CSR 由 system: **node** 或 **system:admin** 组中的 **node-bootstrapper** 服务帐户提交，并确认节点的身份。

- 要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```

注意

在有些 CSR 被批准前，一些 Operator 可能无法使用。

4. 现在，您的客户端请求已被批准，您必须查看添加到集群中的每台机器的服务器请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE     REQUESTOR                                                   CONDITION
csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal       Pending
csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal       Pending
...
```

5. 如果剩余的 CSR 没有被批准，且处于 **Pending** 状态，请批准集群机器的 CSR：

- 要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs oc adm certificate approve
```

6. 批准所有客户端和服务器 CSR 后，机器将 处于 **Ready 状态**。运行以下命令验证：

```
$ oc get nodes
```

输出示例

```
NAME      STATUS   ROLES   AGE  VERSION
master-0  Ready    master  73m  v1.29.4
master-1  Ready    master  73m  v1.29.4
master-2  Ready    master  74m  v1.29.4
worker-0  Ready    worker  11m  v1.29.4
worker-1  Ready    worker  11m  v1.29.4
```

注意

批准服务器 CSR 后可能需要几分钟时间让机器过渡到 **Ready 状态**。

其他信息

- 如需有关 CSR 的更多信息，请参阅 证书签名请求。

9.11.22. 添加入口 DNS 记录

创建 Kubernetes 清单并生成 Ignition 配置时，DNS 区配置会被删除。您必须手动创建指向入口负载均衡器的 DNS 记录。您可以创建一个通配符 ***.apps.{baseDomain}.** 或特定的记录。您可以根据要求使用 A、CNAME 和其他记录。

先决条件

- 配置 GCP 帐户。
- 在创建 Kubernetes 清单并生成 Ignition 配置时，删除 DNS 区配置。
- 在 GCP 中创建和配置 VPC 及相关子网。
- 在 GCP 中创建和配置联网与负载均衡器。
- 创建 control plane 和计算角色。
- 创建 bootstrap 机器。
- 创建 control plane 机器。
- 创建 worker 机器。

流程

1. 等待入口路由器创建负载均衡器并填充 **EXTERNAL-IP** 字段：

```
$ oc -n openshift-ingress get service router-default
```

输出示例

```
NAME            TYPE           CLUSTER-IP     EXTERNAL-IP     PORT(S)                      AGE
router-default  LoadBalancer  172.30.18.154  35.233.157.184
80:32288/TCP,443:31215/TCP  98
```

2. 在您的区中添加 A 记录：
   - 使用 A 记录：
     i. 导出路由器 IP 地址的变量：

```
$ export ROUTER_IP=`oc -n openshift-ingress get service router-default --no-
headers | awk '{print $4}'`
```

     ii. 在私有区中添加 A 记录：

```
$ if [ -f transaction.yaml ]; then rm transaction.yaml; fi
$ gcloud dns record-sets transaction start --zone ${INFRA_ID}-private-zone --project
${HOST_PROJECT} --account ${HOST_PROJECT_ACCOUNT}
$ gcloud dns record-sets transaction add ${ROUTER_IP} --name
\*.apps.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 300 --type A --zone
${INFRA_ID}-private-zone --project ${HOST_PROJECT} --account
```

```
${HOST_PROJECT_ACCOUNT}
$ gcloud dns record-sets transaction execute --zone ${INFRA_ID}-private-zone --
project ${HOST_PROJECT} --account ${HOST_PROJECT_ACCOUNT}
```

iii. 对于外部集群，还要在公共区中添加 A 记录：

```
$ if [ -f transaction.yaml ]; then rm transaction.yaml; fi
$ gcloud dns record-sets transaction start --zone ${BASE_DOMAIN_ZONE_NAME} -
-project ${HOST_PROJECT} --account ${HOST_PROJECT_ACCOUNT}
$ gcloud dns record-sets transaction add ${ROUTER_IP} --name
\*.apps.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 300 --type A --zone
${BASE_DOMAIN_ZONE_NAME} --project ${HOST_PROJECT} --account
${HOST_PROJECT_ACCOUNT}
$ gcloud dns record-sets transaction execute --zone
${BASE_DOMAIN_ZONE_NAME} --project ${HOST_PROJECT} --account
${HOST_PROJECT_ACCOUNT}
```

- 要添加特定域而不使用通配符，请为集群的每个当前路由创建条目：

```
$ oc get --all-namespaces -o jsonpath='{range .items[*]}{range .status.ingress[*]}{.host}
{"\n"}{end}{end}' routes
```

输出示例

```
oauth-openshift.apps.your.cluster.domain.example.com
console-openshift-console.apps.your.cluster.domain.example.com
downloads-openshift-console.apps.your.cluster.domain.example.com
alertmanager-main-openshift-monitoring.apps.your.cluster.domain.example.com
prometheus-k8s-openshift-monitoring.apps.your.cluster.domain.example.com
```

9.11.23. 添加入口防火墙规则

集群需要几个防火墙规则。如果不使用共享 VPC，则由 Ingress 控制器通过 GCP 云供应商创建这些规则。使用共享 VPC 时，您可以立即为所有服务创建集群范围的防火墙规则，或者在集群请求访问时根据事件创建每个规则。当集群请求访问时，通过创建每个规则，您需要准确了解哪些防火墙规则。通过创建集群范围的防火墙规则，您可以在多个集群中应用相同的规则。

如果您选择基于事件创建每个规则，则必须在置备集群后创建防火墙规则，并在控制台通知您缺少规则时在集群生命周期中创建防火墙规则。此时会显示类似以下事件的事件，您必须添加所需的防火墙规则：

```
$ oc get events -n openshift-ingress --field-selector="reason=LoadBalancerManualChange"
```

输出示例

```
Firewall change required by security admin: `gcloud compute firewall-rules create k8s-fw-
a26e631036a3f46cba28f8df67266d55 --network example-network --description "
{\"kubernetes.io/service-name\":\"openshift-ingress/router-default\", \"kubernetes.io/service-
ip\":\"35.237.236.234\"}\" --allow tcp:443,tcp:80 --source-ranges 0.0.0.0/0 --target-tags exampl-fqzq7-
master,exampl-fqzq7-worker --project example-project`
```

如果您在创建这些基于规则的事件时遇到问题，您可以在集群运行时配置集群范围的防火墙规则。

9.11.23.1. 在 GCP 中为共享 VPC 创建集群范围的防火墙规则

您可以创建集群范围的防火墙规则，以允许 OpenShift Container Platform 集群所需的访问。

警告

如果您没有选择基于集群事件创建防火墙规则，您必须创建集群范围的防火墙规则。

先决条件

- 您导出了部署集群所需的 Deployment Manager 模板所需的变量。
- 您在集群所需的 GCP 中创建了网络和负载均衡组件。

流程

1. 添加单个防火墙规则，以允许 Google Cloud Engine 健康检查访问所有服务。此规则可让入口负载均衡器决定其实例的健康状况。

```
$ gcloud compute firewall-rules create --allow='tcp:30000-32767,udp:30000-32767' --
network="${CLUSTER_NETWORK}" --source-
ranges='130.211.0.0/22,35.191.0.0/16,209.85.152.0/22,209.85.204.0/22' --target-
tags="${INFRA_ID}-master,${INFRA_ID}-worker" ${INFRA_ID}-ingress-hc --
account=${HOST_PROJECT_ACCOUNT} --project=${HOST_PROJECT}
```

2. 添加单个防火墙规则以允许访问所有群集服务：

   - 对于外部集群：

```
$ gcloud compute firewall-rules create --allow='tcp:80,tcp:443' --
network="${CLUSTER_NETWORK}" --source-ranges="0.0.0.0/0" --target-
tags="${INFRA_ID}-master,${INFRA_ID}-worker" ${INFRA_ID}-ingress --
account=${HOST_PROJECT_ACCOUNT} --project=${HOST_PROJECT}
```

   - 对于私有集群：

```
$ gcloud compute firewall-rules create --allow='tcp:80,tcp:443' --
network="${CLUSTER_NETWORK}" --source-ranges=${NETWORK_CIDR} --target-
tags="${INFRA_ID}-master,${INFRA_ID}-worker" ${INFRA_ID}-ingress --
account=${HOST_PROJECT_ACCOUNT} --project=${HOST_PROJECT}
```

   由于此规则只允许 TCP 端口 **80** 和 **443** 上的流量，因此请确保添加了服务使用的所有端口。

9.11.24. 在用户置备的基础架构上完成 GCP 安装

在 Google Cloud Platform(GCP)用户置备的基础架构上启动 OpenShift Container Platform 安装后，您可以监控集群事件，直到集群就绪。

先决条件

- 在用户置备的 GCP 基础架构上为 OpenShift Container Platform 集群部署 bootstrap 机器。
- 安装 **oc** CLI 并登录。

流程

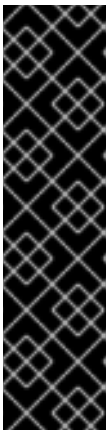
1. 完成集群安装：

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

输出示例

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。



重要

- 安装程序生成的 Ignition 配置文件包含 24 小时后过期的证书，然后在该时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

2. 观察集群的运行状态。

a. 运行以下命令查看当前的集群版本和状态：

```
$ oc get clusterversion
```

输出示例

```
NAME     VERSION   AVAILABLE   PROGRESSING   SINCE   STATUS
version          False       True          24m     Working towards 4.5.4: 99% complete
```

b. 运行以下命令，查看 control plane 上由 Cluster Version Operator(CVO)管理的 Operator：

```
$ oc get clusteroperators
```

输出示例

```
NAME                                 VERSION   AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                       4.5.4     True        False         False     7m56s
cloud-credential                     4.5.4     True        False         False     31m
cluster-autoscaler                   4.5.4     True        False         False     16m
```

```
console                                    4.5.4     True        False         False      10m
csi-snapshot-controller                    4.5.4     True        False         False      16m
dns                                        4.5.4     True        False         False      22m
etcd                                       4.5.4     False       False         False      25s
image-registry                             4.5.4     True        False         False      16m
ingress                                    4.5.4     True        False         False      16m
insights                                   4.5.4     True        False         False      17m
kube-apiserver                             4.5.4     True        False         False      19m
kube-controller-manager                    4.5.4     True        False         False      20m
kube-scheduler                             4.5.4     True        False         False      20m
kube-storage-version-migrator              4.5.4     True        False         False      16m
machine-api                                4.5.4     True        False         False      22m
machine-config                             4.5.4     True        False         False      22m
marketplace                                4.5.4     True        False         False      16m
monitoring                                 4.5.4     True        False         False      10m
network                                    4.5.4     True        False         False      23m
node-tuning                                4.5.4     True        False         False      23m
openshift-apiserver                        4.5.4     True        False         False      17m
openshift-controller-manager               4.5.4     True        False         False      15m
openshift-samples                          4.5.4     True        False         False      16m
operator-lifecycle-manager                 4.5.4     True        False         False      22m
operator-lifecycle-manager-catalog         4.5.4     True        False         False      22m
operator-lifecycle-manager-packageserver   4.5.4     True        False         False      18m
service-ca                                 4.5.4     True        False         False      23m
service-catalog-apiserver                  4.5.4     True        False         False      23m
service-catalog-controller-manager         4.5.4     True        False         False      23m
storage                                    4.5.4     True        False         False      17m
```

c. 运行以下命令来查看您的集群 pod：

```
$ oc get pods --all-namespaces
```

输出示例

```
NAMESPACE                                       NAME
READY     STATUS      RESTARTS   AGE
kube-system                                     etcd-member-ip-10-0-3-111.us-east-
2.compute.internal              1/1       Running     0          35m
kube-system                                     etcd-member-ip-10-0-3-239.us-east-
2.compute.internal              1/1       Running     0          37m
kube-system                                     etcd-member-ip-10-0-3-24.us-east-
2.compute.internal              1/1       Running     0          35m
openshift-apiserver-operator                    openshift-apiserver-operator-6d6674f4f4-
h7t2t                           1/1       Running     1          37m
openshift-apiserver                             apiserver-fm48r
1/1       Running     0          30m
openshift-apiserver                             apiserver-fxkvv
1/1       Running     0          29m
openshift-apiserver                             apiserver-q85nm
1/1       Running     0          29m
...
openshift-service-ca-operator                   openshift-service-ca-operator-66ff6dc6cd-
9r257                           1/1       Running     0          37m
openshift-service-ca                            apiservice-cabundle-injector-695b6bcbc-cl5hm
1/1       Running     0          35m
```

```
openshift-service-ca                                    configmap-cabundle-injector-8498544d7-
25qn6                                   1/1     Running     0          35m
openshift-service-ca                                    service-serving-cert-signer-6445fc9c6-wqdqn
1/1     Running     0          35m
openshift-service-catalog-apiserver-operator            openshift-service-catalog-apiserver-
operator-549f44668b-b5q2w       1/1     Running     0          32m
openshift-service-catalog-controller-manager-operator   openshift-service-catalog-
controller-manager-operator-b78cr2lnm   1/1     Running     0          31m
```

当当前集群版本是 **AVAILABLE** 时，安装已完成。

9.11.25. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

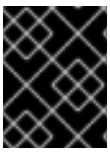
- 有关 Telemetry 服务的更多信息，请参阅关于 远程健康监控

9.11.26. 后续步骤

- 自定义集群。
- 如果需要，您可以选择 不使用远程健康报告。

9.12. 在使用用户置备的受限网络中的 GCP 上安装集群

在 OpenShift Container Platform 版本 4.16 中，您可以使用您提供的基础架构和安装发行内容的内部镜像在 Google Cloud Platform(GCP)上安装集群。



重要

虽然您可以使用镜像安装发行内容安装 OpenShift Container Platform 集群，但您的集群仍需要访问互联网才能使用 GCP API。

此处概述了进行用户提供基础架构安装的步骤。提供的几个 Deployment Manager 模板可帮助完成这些步骤，或者帮助您自行建模。您还可以自由选择通过其他方法创建所需的资源。

重要

执行用户置备的基础架构安装的步骤仅作为示例。使用您提供的基础架构安装集群需要了解云供应商和 OpenShift Container Platform 的安装过程。提供的几个 Deployment Manager 模板可帮助完成这些步骤，或者帮助您自行建模。您还可以自由选择通过其他方法创建所需的资源；模板仅作示例之用。

9.12.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读有关 选择集群安装方法的文档，并为用户准备它。
- 您在镜像主机上创建 registry，并获取您的 OpenShift Container Platform 版本的 **imageContentSources** 数据。

重要

由于安装介质位于镜像主机上，因此您可以使用该计算机完成所有安装步骤。

- 如果使用防火墙，则会 将其配置为允许集群需要访问的站点。虽然您可能需要授予更多站点的访问权限，但您必须授予对 ***.googleapis.com** 和 **accounts.google.com** 的访问权限。
- 如果环境中无法访问云身份和访问管理(IAM)API，或者不想将管理员级别的凭证 secret 存储在 **kube-system** 命名空间中，您可以 手动创建和维护 IAM 凭证。

9.12.2. 关于在受限网络中安装

在 OpenShift Container Platform 4.16 中，可以执行不需要有效的互联网连接来获取软件组件的安装。受限网络安装可以使用安装程序置备的基础架构或用户置备的基础架构完成，具体取决于您要安装集群的云平台。

如果您选择在云平台中执行受限网络安装，您仍需要访问其云 API。有些云功能，比如 Amazon Web Service 的 Route 53 DNS 和 IAM 服务，需要访问互联网。根据您的网络，在裸机硬件、Nutanix 或 VMware vSphere 上安装可能需要较少的互联网访问。

要完成受限网络安装，您必须创建一个 registry，以镜像 OpenShift 镜像 registry 的内容并包含安装介质。您可以在镜像主机上创建此 registry，该主机可同时访问互联网和您的封闭网络，也可以使用满足您的限制条件的其他方法。

重要

由于用户置备安装配置的复杂性，在尝试使用用户置备的基础架构受限网络安装前，请考虑完成标准用户置备的基础架构安装。完成此测试安装后，您可以更轻松地隔离和排除在受限网络中安装过程中可能出现的任何问题。

9.12.2.1. 其他限制

受限网络中的集群有以下额外限制和限制：

- **ClusterVersion** 状态包含一个 **Unable to retrieve available updates** 错误。
- 默认情况下，您无法使用 Developer Catalog 的内容，因为您无法访问所需的镜像流标签。

9.12.3. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来获得用来安装集群的镜像。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。

- 获取执行集群更新所需的软件包。

9.12.4. 配置 GCP 项目

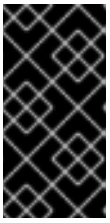
在安装 OpenShift Container Platform 之前，您必须配置 Google Cloud Platform(GCP)项目来托管它。

9.12.4.1. 创建 GCP 项目

要安装 OpenShift Container Platform，您必须在 Google Cloud Platform(GCP)帐户中创建项目来托管集群。

流程

- 创建一个项目来托管 OpenShift Container Platform 集群。请参阅 GCP 文档中的创建和管理 项目。



重要

如果您使用安装程序置备的基础架构，您的 GCP 项目必须使用 Premium Network Service Tier。使用安装程序安装的集群不支持 Standard Network Service Tier。安装程序为 **api-int.<cluster_name>.<base_domain>** URL 配置内部负载均衡；内部负载均衡需要 Premium Tier。

9.12.4.2. 在 GCP 中启用 API 服务

Google Cloud Platform(GCP)项目需要访问多个 API 服务来完成 OpenShift Container Platform 安装。

先决条件

- 已创建一个项目来托管集群。

流程

- 在托管集群的项目中启用以下所需的 API 服务。您还可以启用安装不需要的可选 API 服务。请参阅 GCP 文档中的 启用服务。

表 9.39. 所需的 API 服务

API 服务	控制台服务名称
Compute Engine API	**compute.googleapis.com**
Cloud Resource Manager API	**cloudresourcemanager.googleapis.com**
Google DNS API	**dns.googleapis.com**
IAM Service Account Credentials API	**iamcredentials.googleapis.com**
Identity and Access Management(IAM)API	**iam.googleapis.com**

API 服务	控制台服务名称
Service Usage API	**serviceusage.googleapis.com**

表 9.40. 可选 API 服务

API 服务	控制台服务名称
Google Cloud API	**cloudapis.googleapis.com**
服务管理 API	**servicemanagement.googleapis.com**
Google Cloud Storage JSON API	**storage-api.googleapis.com**
Cloud Storage	**storage-component.googleapis.com**

9.12.4.3. 为 GCP 配置 DNS

要安装 OpenShift Container Platform，您使用的 Google Cloud Platform(GCP)帐户必须在托管 OpenShift Container Platform 集群的同一项目中有一个专用的公共托管区。此区域必须对域具有权威。DNS 服务为集群外部连接提供集群 DNS 解析和名称查询。

流程

1. 确定您的域或子域，以及注册商。您可以转移现有的域和注册商，或通过 GCP 或其他来源获取新的域和注册商。

注意

如果您购买了新的域，则需要时间来传播相关的 DNS 更改。有关通过 Google 购买域的更多信息，请参阅 Google Domains。

2. 在 GCP 项目中为您的域或子域创建一个公共托管区。请参阅 GCP 文档中的 创建公共区。
使用适当的根域，如 **openshiftcorp.com 或** 子域，如 cluster **.openshiftcorp.com**。

3. 从托管区域记录中提取新的权威名称服务器。请参阅 GCP 文档中的 查找您的云 DNS 名称服务器。
您通常有四个名称服务器。

4. 更新域所用名称服务器的注册商记录。例如，如果您将域注册到 Google Domains，请参阅 Google Domains 帮助中的以下主题： 如何切换到自定义名称服务器。

5. 如果您将根域迁移到 Google Cloud DNS，请迁移您的 DNS 记录。请参阅 GCP 文档中的 Migrating to Cloud DNS。

6. 如果您使用子域，请按照贵公司的步骤将其委派记录添加到父域。这个过程可能包括对您公司的 IT 部门或控制您公司的根域和 DNS 服务的部门发出的请求。

9.12.4.4. GCP 帐户限值

OpenShift Container Platform 集群使用许多 Google Cloud Platform(GCP)组件，但默认的 配额 不会影响您安装默认 OpenShift Container Platform 集群的能力。

默认集群包含三台计算和三台 control plane 机器，它使用以下资源。请注意，有些资源只在 bootstrap 过程中需要，并在集群部署后删除。

表 9.41. 默认集群中使用的 GCP 资源

service	组件	位置	所需的资源总数	bootstrap 后删除的资源
服务帐户	IAM	全局	6	1
防火墙规则	Networking	全局	11	1
转发规则	Compute	全局	2	0
健康检查	Compute	全局	2	0
镜像	Compute	全局	1	0
网络	Networking	全局	1	0
路由器	Networking	全局	1	0
Routes	Networking	全局	2	0
子网	Compute	全局	2	0
目标池	Networking	全局	2	0

注意

如果在安装过程中任何配额不足，安装程序会显示一个错误信息，包括超过哪个配额，以及显示区域。

请考虑您的集群的实际大小、预定的集群增长，以及来自与您的帐户关联的其他集群的使用情况。CPU、静态 IP 地址和持久磁盘 SSD（存储）配额是最可能不足的。

如果您计划在以下区域之一部署集群，您将超过最大存储配额，并可能会超过 CPU 配额限制：

- **asia-east2**
- **asia-northeast2**
- **asia-south1**
- **australia-southeast1**
- **europe-north1**

- **europe-west2**
- **europe-west3**
- **europe-west6**
- **northamerica-northeast1**
- **southamerica-east1**
- **us-west2**

您可以从 GCP 控制台 增加资源配额，但可能需要提交一个支持问题单。务必提前规划集群大小，以便在安装 OpenShift Container Platform 集群前有足够的时间来等待支持问题单被处理。

9.12.4.5. 在 GCP 中创建服务帐户

OpenShift Container Platform 需要一个 Google Cloud Platform(GCP)服务帐户，它提供访问 Google API 中数据的验证和授权。如果您没有包含项目中所需角色的现有 IAM 服务帐户，您必须创建一个。

先决条件

- 已创建一个项目来托管集群。

流程

1. 在用于托管 OpenShift Container Platform 集群的项目中创建一个服务帐户。请参阅 GCP 文档中的 创建服务帐户。
2. 为服务帐户授予适当的权限。您可以逐一授予权限，也可以为其分配 **Owner** 角色。请参阅将角色转换到特定资源的服务帐户。

注意

将服务帐户设置为项目的所有者是获取所需权限的最简单方法，这意味着该服务帐户对项目有完全的控制权。您必须确定提供这种能力所带来的风险是否可以接受。

3. 您可以使用 JSON 格式创建服务帐户密钥，或将服务帐户附加到 GCP 虚拟机。请参阅 GCP 文档中的创建服务帐户密钥以及为实例创建并启用服务帐户。
   您必须有一个服务帐户密钥或带有附加服务帐户的虚拟机来创建集群。

注意

如果您使用附加服务帐户的虚拟机来创建集群，则必须在安装前在 **install-config.yaml** 文件中设置 **credentialsMode: Manual**。

9.12.4.6. 所需的 GCP 角色

将 **Owner** 角色附加到您创建的服务帐户时，您可以为该服务帐户授予所有权限，包括安装 OpenShift Container Platform 所需的权限。如果机构的安全策略需要更严格的权限集，您可以创建具有以下权限的服务帐户：如果您将集群部署到现有的虚拟私有云 (VPC) 中，则服务帐户不需要某些网络权限，如以下列表中记录：

安装程序所需的角色

- Compute Admin
- Role Administrator
- Security Admin
- Service Account Admin
- Service Account Key Admin
- Service Account User
- Storage Admin

安装过程中创建网络资源所需的角色

- DNS Administrator

在 passthrough 模式中使用 Cloud Credential Operator 所需的角色

- Compute Load Balancer Admin

用户置备的 GCP 基础架构所需的角色

- Deployment Manager Editor

以下角色应用到 control plane 和计算机器使用的服务帐户：

表 9.42. GCP 服务帐户角色

帐户	角色
Control Plane	**roles/compute.instanceAdmin**
	roles/compute.networkAdmin
	roles/compute.securityAdmin
	roles/storage.admin
	roles/iam.serviceAccountUser
Compute	**roles/compute.viewer**
	roles/storage.admin

9.12.4.7. 用户置备的基础架构所需的 GCP 权限

将 **Owner** 角色附加到您创建的服务帐户时，您可以为该服务帐户授予所有权限，包括安装 OpenShift Container Platform 所需的权限。

如果机构的安全策略需要更严格的权限集，您可以创建具有所需权限的自定义角色。用户置备的基础架构需要以下权限来创建和删除 OpenShift Container Platform 集群。

例 9.94. 创建网络资源所需的权限

- **compute.addresses.create**
- **compute.addresses.createInternal**
- **compute.addresses.delete**
- **compute.addresses.get**
- **compute.addresses.list**
- **compute.addresses.use**
- **compute.addresses.useInternal**
- **compute.firewalls.create**
- **compute.firewalls.delete**
- **compute.firewalls.get**
- **compute.firewalls.list**
- **compute.forwardingRules.create**
- **compute.forwardingRules.get**
- **compute.forwardingRules.list**
- **compute.forwardingRules.setLabels**
- **compute.networks.create**
- **compute.networks.get**
- **compute.networks.list**
- **compute.networks.updatePolicy**
- **compute.routers.create**
- **compute.routers.get**
- **compute.routers.list**
- **compute.routers.update**
- **compute.routes.list**
- **compute.subnetworks.create**
- **compute.subnetworks.get**
- **compute.subnetworks.list**

- **compute.subnetworks.use**
- **compute.subnetworks.useExternalIp**

例 9.95. 创建负载均衡器资源所需的权限

- **compute.regionBackendServices.create**
- **compute.regionBackendServices.get**
- **compute.regionBackendServices.list**
- **compute.regionBackendServices.update**
- **compute.regionBackendServices.use**
- **compute.targetPools.addInstance**
- **compute.targetPools.create**
- **compute.targetPools.get**
- **compute.targetPools.list**
- **compute.targetPools.removeInstance**
- **compute.targetPools.use**

例 9.96. 创建 DNS 资源所需的权限

- **dns.changes.create**
- **dns.changes.get**
- **dns.managedZones.create**
- **dns.managedZones.get**
- **dns.managedZones.list**
- **dns.networks.bindPrivateDNSZone**
- **dns.resourceRecordSets.create**
- **dns.resourceRecordSets.list**
- **dns.resourceRecordSets.update**

例 9.97. 创建服务帐户资源所需的权限

- **iam.serviceAccountKeys.create**
- **iam.serviceAccountKeys.delete**

- **iam.serviceAccountKeys.get**
- **iam.serviceAccountKeys.list**
- **iam.serviceAccounts.actAs**
- **iam.serviceAccounts.create**
- **iam.serviceAccounts.delete**
- **iam.serviceAccounts.get**
- **iam.serviceAccounts.list**
- **resourcemanager.projects.get**
- **resourcemanager.projects.getIamPolicy**
- **resourcemanager.projects.setIamPolicy**

例 9.98. 创建计算资源所需的权限

- **compute.disks.create**
- **compute.disks.get**
- **compute.disks.list**
- **compute.instanceGroups.create**
- **compute.instanceGroups.delete**
- **compute.instanceGroups.get**
- **compute.instanceGroups.list**
- **compute.instanceGroups.update**
- **compute.instanceGroups.use**
- **compute.instances.create**
- **compute.instances.delete**
- **compute.instances.get**
- **compute.instances.list**
- **compute.instances.setLabels**
- **compute.instances.setMetadata**
- **compute.instances.setServiceAccount**
- **compute.instances.setTags**
- **compute.instances.use**

- **compute.machineTypes.get**
- **compute.machineTypes.list**

例 9.99. 创建存储资源需要

- **storage.buckets.create**
- **storage.buckets.delete**
- **storage.buckets.get**
- **storage.buckets.list**
- **storage.objects.create**
- **storage.objects.delete**
- **storage.objects.get**
- **storage.objects.list**

例 9.100. 创建健康检查资源所需的权限

- **compute.healthChecks.create**
- **compute.healthChecks.get**
- **compute.healthChecks.list**
- **compute.healthChecks.useReadOnly**
- **compute.httpHealthChecks.create**
- **compute.httpHealthChecks.get**
- **compute.httpHealthChecks.list**
- **compute.httpHealthChecks.useReadOnly**

例 9.101. 获取 GCP 区域和区域相关信息所需的权限

- **compute.globalOperations.get**
- **compute.regionOperations.get**
- **compute.regions.list**
- **compute.zoneOperations.get**
- **compute.zones.get**
- **compute.zones.list**

例 9.102. 检查服务和配额所需的权限

- monitoring.timeSeries.list
- serviceusage.quotas.get
- serviceusage.services.list

例 9.103. 安装所需的 IAM 权限

- iam.roles.get

例 9.104. 安装所需的镜像权限

- compute.images.create
- compute.images.delete
- compute.images.get
- compute.images.list

例 9.105. 运行收集 bootstrap 的可选权限

- compute.instances.getSerialPortOutput

例 9.106. 删除网络资源所需的权限

- compute.addresses.delete
- compute.addresses.deleteInternal
- compute.addresses.list
- compute.firewalls.delete
- compute.firewalls.list
- compute.forwardingRules.delete
- compute.forwardingRules.list
- compute.networks.delete
- compute.networks.list
- compute.networks.updatePolicy
- compute.routers.delete
- compute.routers.list

- compute.routes.list
- compute.subnetworks.delete
- compute.subnetworks.list

例 9.107. 删除负载均衡器资源所需的权限

- compute.regionBackendServices.delete
- compute.regionBackendServices.list
- compute.targetPools.delete
- compute.targetPools.list

例 9.108. 删除 DNS 资源所需的权限

- dns.changes.create
- dns.managedZones.delete
- dns.managedZones.get
- dns.managedZones.list
- dns.resourceRecordSets.delete
- dns.resourceRecordSets.list

例 9.109. 删除服务帐户资源所需的权限

- iam.serviceAccounts.delete
- iam.serviceAccounts.get
- iam.serviceAccounts.list
- resourcemanager.projects.getIamPolicy
- resourcemanager.projects.setIamPolicy

例 9.110. 删除计算资源所需的权限

- compute.disks.delete
- compute.disks.list
- compute.instanceGroups.delete
- compute.instanceGroups.list

- **compute.instances.delete**
- **compute.instances.list**
- **compute.instances.stop**
- **compute.machineTypes.list**

例 9.111. 删除存储资源需要

- **storage.buckets.delete**
- **storage.buckets.getIamPolicy**
- **storage.buckets.list**
- **storage.objects.delete**
- **storage.objects.list**

例 9.112. 删除健康检查资源所需的权限

- **compute.healthChecks.delete**
- **compute.healthChecks.list**
- **compute.httpHealthChecks.delete**
- **compute.httpHealthChecks.list**

例 9.113. 删除所需的镜像权限

- **compute.images.delete**
- **compute.images.list**

例 9.114. 获取区域相关信息所需的权限

- **compute.regions.get**

例 9.115. 所需的 Deployment Manager 权限

- **deploymentmanager.deployments.create**
- **deploymentmanager.deployments.delete**
- **deploymentmanager.deployments.get**
- **deploymentmanager.deployments.list**

- **deploymentmanager.manifests.get**
- **deploymentmanager.operations.get**
- **deploymentmanager.resources.list**

其他资源

- 优化存储

9.12.4.8. 支持的 GCP 区域

您可以将 OpenShift Container Platform 集群部署到以下 Google Cloud Platform(GCP)区域：

- **africa-south1** (Johannesburg, South Africa)
- **asia-east1** (Changhua County, Taiwan)
- **asia-east2** (Hong Kong)
- **asia-northeast1** (Tokyo, Japan)
- **asia-northeast2** (Osaka, Japan)
- **asia-northeast3** (Seoul, South Korea)
- **asia-south1** (Mumbai, India)
- **asia-south2** (Delhi, India)
- **asia-southeast1** (Jurong West, Singapore)
- **asia-southeast2** (Jakarta, Indonesia)
- **aia-southeast1(** Sydney, Australia)
- **australia-southeast2** (Melbourne, Australia)
- **europe-central2** (Warsaw, USA)
- **europe-north1** (Hamina, Finland)
- **europe-southwest1** (Madrid, Spain)
- **europe-west1** (St. Ghislain, Belgium)
- **europe-west2** (London, England, UK)
- **europe-west3** (Frankfurt, Germany)
- **europe-west4** (Eemshaven, Netherlands)
- **europe-west6** (Zürich, Switzerland)
- **europe-west8** (Milan, Italy)

- **europe-west9** (Paris, France)
- **europe-west12** (Turin, Italy)
- **me-central1** (Doha, Qatar, Middle East)
- **me-central2** (Dammam, Saudi Arabia, Middle East)
- **me-west1** (Tel Aviv, Israel)
- **northamerica-northeast1** (Montréal, Québec, Canada)
- **northamerica-northeast2** (Toronto, Ontario, Canada)
- **southamerica-east1** (São Paulo, Brazil)
- **southamerica-west1** (Santiago, Chile)
- **us-central1** (Council Bluffs, Iowa, USA)
- **us-east1** (Moncks Corner, South Carolina, USA)
- **us-east4** (Ashburn, North Virginia, USA)
- **us-east5** (Columbus, Ohio)
- **us-south1** (Dallas, Texas)
- **us-west1** (The Dalles, Oregon, USA)
- **us-west2** (Los Angeles, California, USA)
- **us-west3** (Salt Lake City, Utah, USA)
- **us-west4** (Las Vegas, Nevada, USA)

注意

要确定地区和区域中可以使用哪些机器类型实例，请参阅 Google 文档。

9.12.4.9. 为 GCP 安装和配置 CLI 工具

要使用用户置备的基础架构在 Google Cloud Platform(GCP)上安装 OpenShift Container Platform，您必须为 GCP 安装和配置 CLI 工具。

先决条件

- 已创建一个项目来托管集群。
- 您创建了服务帐户并授予其所需的权限。

流程

1. 在 **$PATH** 中安装以下二进制文件：
   - **gcloud**

- **gsutil**

请参阅 GCP 文档中的 安装最新的 Cloud SDK 版本。

2. 使用 **gcloud** 工具及您配置的服务帐户进行身份验证。
请参阅 GCP 文档中的 使用服务帐户授权。

9.12.5. 具有用户置备基础架构的集群的要求

对于包含用户置备的基础架构的集群，您必须部署所有所需的机器。

本节论述了在用户置备的基础架构上部署 OpenShift Container Platform 的要求。

9.12.5.1. 集群安装所需的机器

最小的 OpenShift Container Platform 集群需要以下主机：

表 9.43. 最低所需的主机

主机	描述
一个临时 bootstrap 机器	集群需要 bootstrap 机器在三台 control plane 机器上部署 OpenShift Container Platform 集群。您可在安装集群后删除 bootstrap 机器。
三台 control plane 机器	control plane 机器运行组成 control plane 的 Kubernetes 和 OpenShift Container Platform 服务。
至少两台计算机器，也称为 worker 机器。	OpenShift Container Platform 用户请求的工作负载在计算机器上运行。

重要

要保持集群的高可用性，请将独立的物理主机用于这些集群机器。

bootstrap 和 control plane 机器必须使用 Red Hat Enterprise Linux CoreOS(RHCOS)作为操作系统。但是，计算机器可以在 Red Hat Enterprise Linux CoreOS(RHCOS)、Red Hat Enterprise Linux(RHEL) 8.6 和更高的版本。

请注意，RHCOS 基于 Red Hat Enterprise Linux(RHEL) 9.2，并继承其所有硬件认证和要求。查看 红帽企业 Linux 技术功能和限制。

9.12.5.2. 集群安装的最低资源要求

每台集群机器都必须满足以下最低要求：

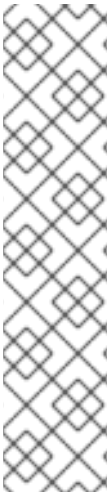
表 9.44. 最低资源要求

机器	操作系统	vCPU [1]	虚拟内存	Storage	每秒输入/输出 (IOPS) [2]
bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane（控制平面）	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS、RHEL 8.6 及更新版本 [3]	2	8 GB	100 GB	300

1. 当未启用并发多线程(SMT)或超线程时，一个 vCPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比例：（每个内核数的线程）× sockets = vCPU。

2. OpenShift Container Platform 和 Kubernetes 对磁盘性能非常敏感，建议使用更快的存储速度，特别是 control plane 节点上需要 10 ms p99 fsync 持续时间的 etcd。请注意，在许多云平台上，存储大小和 IOPS 可一起扩展，因此您可能需要过度分配存储卷来获取足够的性能。

3. 与所有用户置备的安装一样，如果您选择在集群中使用 RHEL 计算机器，则负责所有操作系统生命周期管理和维护，包括执行系统更新、应用补丁和完成所有其他必要的任务。RHEL 7 计算机器的使用已弃用，并已在 OpenShift Container Platform 4.10 及更新的版本中删除。



注意

从 OpenShift Container Platform 版本 4.13 开始，RHCOS 基于 RHEL 版本 9.2，它更新了微架构要求。以下列表包含每个架构需要的最小指令集架构 (ISA)：

- x86-64 体系结构需要 x86-64-v2 ISA
- ARM64 架构需要 ARMv8.0-A ISA
- IBM Power 架构需要 Power 9 ISA
- s390x 架构需要 z14 ISA

如需更多信息，请参阅 RHEL 架构。

如果平台的实例类型满足集群机器的最低要求，则 OpenShift Container Platform 支持使用它。

9.12.5.3. 为 GCP 测试的实例类型

以下 Google Cloud Platform 实例类型已使用 OpenShift Container Platform 测试。

例 9.116. 机器系列

- **A2**
- **A3**
- **C2**

- **C2D**
- **C3**
- **C3D**
- **E2**
- **M1**
- **N1**
- **N2**
- **N2D**
- **Tau T2D**

9.12.5.4. 使用自定义机器类型

支持使用自定义机器类型来安装 OpenShift Container Platform 集群。

使用自定义机器类型时请考虑以下几点：

- 与预定义的实例类型类似，自定义机器类型必须满足 control plane 和计算机器的最低资源要求。如需更多信息，请参阅"集群安装的资源要求"。
- 自定义机器类型的名称必须遵循以下语法：
  custom-<number_of_cpus>-<amount_of_memory_in_mb>

  例如，**custom-6-20480**。

9.12.6. 为 GCP 创建安装文件

要使用用户置备的基础架构在 Google Cloud Platform(GCP)上安装 OpenShift Container Platform，您必须生成安装程序部署集群所需的文件，并进行修改，以便集群只创建要使用的机器。您可以生成并自定义 **install-config.yaml** 文件、Kubernetes 清单和 Ignition 配置文件。您还可以选择在安装准备阶段首先设置独立 **var** 分区。

9.12.6.1. 可选：创建独立 /var 分区

建议安装程序将 OpenShift Container Platform 的磁盘分区保留给安装程序。然而，在有些情况下您可能需要在文件系统的一部分中创建独立分区。

OpenShift Container Platform 支持添加单个分区来将存储附加到 **/var** 分区或 **/var** 的子目录中。例如：

- **/var/lib/containers** ：保存随着系统中添加更多镜像和容器而增长的容器相关内容。
- **/var/lib/etcd** ：保存您可能希望独立保留的数据，比如 etcd 存储的性能优化。
- **/var** ：保存您可能希望独立保留的数据，以满足审计等目的。

通过单独存储 **/var** 目录的内容，可以更轻松地根据需要为区域扩展存储，并在以后重新安装 OpenShift Container Platform，并保持该数据的完整性。使用这个方法，您不必再次拉取所有容器，在更新系统时也不必复制大量日志文件。

因为 **/var** 在进行一个全新的 Red Hat Enterprise Linux CoreOS（RHCOS）安装前必需存在，所以这个流程会在 OpenShift Container Platform 安装过程的 **openshift-install** 准备阶段插入一个创建的机器配置清单的机器配置来设置独立的 **/var** 分区。

重要

如果按照以下步骤在此流程中创建独立 **/var** 分区，则不需要再次创建 Kubernetes 清单和 Ignition 配置文件，如本节所述。

流程

1. 创建存放 OpenShift Container Platform 安装文件的目录：

```
$ mkdir $HOME/clusterconfig
```

2. 运行 **openshift-install**，以在 **manifest** 和 **openshift** 子目录中创建一组文件。在系统提示时回答系统问题：

```
$ openshift-install create manifests --dir $HOME/clusterconfig
```

输出示例

```
? SSH Public Key ...
INFO Credentials loaded from the "myprofile" profile in file "/home/myuser/.aws/credentials"
INFO Consuming Install Config from target directory
INFO Manifests created in: $HOME/clusterconfig/manifests and
$HOME/clusterconfig/openshift
```

3. 可选：确认安装程序在 **clusterconfig/openshift** 目录中创建了清单：

```
$ ls $HOME/clusterconfig/openshift/
```

输出示例

```
99_kubeadmin-password-secret.yaml
99_openshift-cluster-api_master-machines-0.yaml
99_openshift-cluster-api_master-machines-1.yaml
99_openshift-cluster-api_master-machines-2.yaml
...
```

4. 创建用于配置额外分区的 Butane 配置。例如，将文件命名为 **$HOME/clusterconfig/98-var-partition.bu**，将磁盘设备名称改为 **worker** 系统上存储设备的名称，并根据情况设置存储大小。这个示例将 **/var** 目录放在一个单独的分区中：

```
variant: openshift
version: 4.16.0
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
storage:
  disks:
  - device: /dev/disk/by-id/<device_name> 1
```

```
    partitions:
    - label: var
      start_mib: <partition_start_offset> 2
      size_mib: <partition_size> 3
      number: 5
  filesystems:
    - device: /dev/disk/by-partlabel/var
      path: /var
      format: xfs
      mount_options: [defaults, prjquota] 4
      with_mount_unit: true
```

1. 要分区的磁盘的存储设备名称。
2. 在引导磁盘中添加数据分区时，推荐最少使用 25000 MiB(Mebibytes)。root 文件系统会自动调整大小以填充所有可用空间（最多到指定的偏移值）。如果没有指定值，或者指定的值小于推荐的最小值，则生成的 root 文件系统会太小，而在以后进行的 RHCOS 重新安装可能会覆盖数据分区的开始部分。
3. 以兆字节为单位的数据分区大小。
4. 对于用于容器存储的文件系统，必须启用 **prjquota** 挂载选项。

注意

当创建单独的 **/var** 分区时，如果不同的实例类型没有相同的设备名称，则无法为 worker 节点使用不同的实例类型。

5. 从 Butane 配置创建一个清单，并将它保存到 **clusterconfig/openshift** 目录中。例如，运行以下命令：

```
$ butane $HOME/clusterconfig/98-var-partition.bu -o $HOME/clusterconfig/openshift/98-var-partition.yaml
```

6. 再次运行 **openshift-install**，从 **manifest** 和 **openshift** 子目录中的一组文件创建 Ignition 配置：

```
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
auth  bootstrap.ign  master.ign  metadata.json  worker.ign
```

现在，您可以使用 Ignition 配置文件作为安装程序的输入来安装 Red Hat Enterprise Linux CoreOS(RHCOS)系统。

9.12.6.2. 创建安装配置文件

您可以自定义在 Google Cloud Platform(GCP)上安装的 OpenShift Container Platform 集群。

先决条件

- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。对于受限网络安装，这些文件位于您的镜像主机上。
- 您有创建镜像 registry 期间生成的 **imageContentSources** 值。

- 您已获取了镜像 registry 的证书内容。

流程

1. 创建 **install-config.yaml** 文件。

   a. 进入包含安装程序的目录并运行以下命令：

   ```
   $ ./openshift-install create install-config --dir <installation_directory> 1
   ```

   1 对于 **<installation_directory>**，请指定要存储安装程序创建的文件的目录名称。

   在指定目录时：

   - 验证该目录是否具有**执行**权限。在安装目录中运行 Terraform 二进制文件需要这个权限。
   - 使用空目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

   b. 在提示符处，提供云的配置详情：

      i. 可选： 选择用于访问集群机器的 SSH 密钥。

      注意

      对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

      ii. 选择 **gcp** 作为目标平台。

      iii. 如果您没有为计算机上的 GCP 帐户配置服务帐户密钥，则必须从 GCP 获取它，并粘贴文件的内容或输入文件的绝对路径。

      iv. 选择要在其中置备集群的项目 ID。默认值由您配置的服务帐户指定。

      v. 选择要将集群部署到的区域。

      vi. 选择集群要部署到的基域。基域与您为集群创建的公共 DNS 区对应。

      vii. 为集群输入描述性名称。

2. 编辑 **install-config.yaml** 文件，以提供在受限网络中安装所需的额外信息。

   a. 更新 **pullSecret** 值，使其包含 registry 的身份验证信息：

   ```
   pullSecret: '{"auths":{"<mirror_host_name>:5000": {"auth": "<credentials>","email":
   "you@example.com"}}}'
   ```

   对于 **<mirror_host_name>**，请指定 您在镜像 registry 证书中指定的 registry 域名 ；对于 **<credentials>**， 请指定您的镜像 registry 的 base64 编码用户名和密码。

   b. 添加 **additionalTrustBundle** 参数和值。

```
additionalTrustBundle: |
  -----BEGIN CERTIFICATE-----

ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
  -----END CERTIFICATE-----
```

该值必须是您用于镜像 registry 的证书文件内容。证书文件可以是现有的可信证书颁发机构，也可以是您为镜像 registry 生成的自签名证书。

c. 定义 VPC 的网络和子网，以便在父 **platform.gcp** 字段下安装集群：

```
network: <existing_vpc>
controlPlaneSubnet: <control_plane_subnet>
computeSubnet: <compute_subnet>
```

对于 **platform.gcp.network**，请指定现有 Google VPC 的名称。对于 **platform.gcp.controlPlaneSubnet** 和 **platform.gcp.computeSubnet**，请分别指定部署 control plane 机器和计算机器的现有子网。

d. 添加镜像内容资源，类似于以下 YAML 摘录：

```
imageContentSources:
- mirrors:
  - <mirror_host_name>:5000/<repo_name>/release
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <mirror_host_name>:5000/<repo_name>/release
  source: registry.redhat.io/ocp/release
```

对于这些值，请使用您在创建镜像 registry 时记录的 **imageContentSources**。

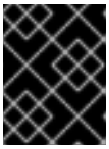
e. 可选：将发布策略设置为 **Internal**：

```
publish: Internal
```

通过设置这个选项，您可以创建一个内部 Ingress Controller 和一个私有负载均衡器。

3. 对您需要的 **install-config.yaml** 文件进行任何其他修改。
有关参数的更多信息，请参阅"安装配置参数"。

4. 备份 **install-config.yaml** 文件，以便您可以使用它安装多个集群。



重要

install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用此文件，必须现在备份。

其他资源

- GCP 的安装配置参数

9.12.6.3. 启用屏蔽虚拟机

您可在安装集群时使用 Shielded 虚拟机。Shielded 虚拟机具有额外的安全功能，包括安全引导、固件和完整性监控和 rootkit 检测。如需更多信息，请参阅 Google 文档中有关 Shielded 虚拟机 的文档。

注意

目前，在具有 64 位 ARM 基础架构的集群中不支持 Shielded 虚拟机。

先决条件

- 您已创建了 **install-config.yaml** 文件。

流程

- 在部署集群前，使用文本编辑器编辑 **install-config.yaml** 文件并添加以下部分之一：

  a. 仅将屏蔽的虚拟机用于 control plane 机器：

```
controlPlane:
  platform:
    gcp:
      secureBoot: Enabled
```

  b. 仅将屏蔽的虚拟机用于计算机器：

```
compute:
- platform:
    gcp:
      secureBoot: Enabled
```

  c. 将屏蔽的虚拟机用于所有机器：

```
platform:
  gcp:
    defaultMachinePlatform:
      secureBoot: Enabled
```

9.12.6.4. 启用机密虚拟机

您可在安装集群时使用机密虚拟机。机密虚拟机在处理数据时加密数据。如需更多信息，请参阅 Google 文档中有关 机密计算的内容。您可以同时启用机密虚拟机和 Shielded 虚拟机，虽然它们不相互依赖。

注意

64 位 ARM 架构目前不支持机密虚拟机。

先决条件

- 您已创建了 **install-config.yaml** 文件。

流程

- 在部署集群前，使用文本编辑器编辑 **install-config.yaml** 文件并添加以下部分之一：

a. 仅将机密虚拟机用于 control plane 机器：

```
controlPlane:
  platform:
    gcp:
      confidentialCompute: Enabled 1
      type: n2d-standard-8 2
      onHostMaintenance: Terminate 3
```

1 启用机密虚拟机。

2 指定支持机密虚拟机的机器类型。机密虚拟机需要 N2D 或 C2D 系列机器类型。有关支持的机器类型的更多信息，请参阅支持的操作系统和机器类型。

3 指定主机维护事件期间虚拟机的行为，如硬件或软件更新。对于使用机密虚拟机的机器，此值必须设置为 **Terminate**，这会停止虚拟机。机密虚拟机不支持实时迁移。

b. 仅将机密虚拟机用于计算机器：

```
compute:
- platform:
    gcp:
      confidentialCompute: Enabled
      type: n2d-standard-8
      onHostMaintenance: Terminate
```

c. 将机密虚拟机用于所有机器：

```
platform:
  gcp:
    defaultMachinePlatform:
      confidentialCompute: Enabled
      type: n2d-standard-8
      onHostMaintenance: Terminate
```

9.12.6.5. 在安装过程中配置集群范围的代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 **Proxy** 对象的 **spec.noProxy** 字段中添加站点来绕过代理。

注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

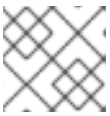
对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，**Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 **.** 以仅匹配子域。例如，**.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代理。
4. 如果提供，安装程序会在 **openshift-config** 命名空间中生成名为 **user-ca-bundle** 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 **trusted-ca-bundle** 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， **Proxy** 对象的 **trustedCA** 字段中也会引用此配置映射。**additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。
5. 可选：决定 **Proxy** 对象的配置以引用 **trustedCA** 字段中 **user-ca-bundle** 配置映射的策略。允许的值是 **Proxyonly** 和 **Always**。仅在配置了 **http/https** 代理时，使用 **Proxyonly** 引用 **user-ca-bundle** 配置映射。使用 **Always** 始终引用 **user-ca-bundle** 配置映射。默认值为 **Proxyonly**。



注意

安装程序不支持代理的 **readinessEndpoints** 字段。

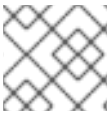
注意

如果安装程序超时，重启并使用安装程序的 **wait-for** 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. 保存该文件并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 cluster 的集群范围代理，该代理 **使用** 提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。



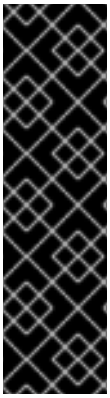
注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

9.12.6.6. 创建 Kubernetes 清单和 Ignition 配置文件

由于您必须修改一些集群定义文件并手动启动集群机器，因此您必须生成 Kubernetes 清单和 Ignition 配置文件来配置机器。

安装配置文件转换为 Kubernetes 清单。清单嵌套到 Ignition 配置文件中，稍后用于配置集群机器。



重要

- OpenShift Container Platform 安装程序生成的 Ignition 配置文件包含 24 小时后过期的证书，然后在该时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

先决条件

- 已获得 OpenShift Container Platform 安装程序。对于受限网络安装，这些文件位于您的镜像主机上。
- 已创建 **install-config.yaml** 安装配置文件。

流程

1. 进入包含 OpenShift Container Platform 安装程序的目录，并为集群生成 Kubernetes 清单：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定包含您创建的 **install-config.yaml** 文件的安装目录。

2. 删除定义 control plane 机器的 Kubernetes 清单文件：

```
$ rm -f <installation_directory>/openshift/99_openshift-cluster-api_master-machines-*.yaml
```

通过删除这些文件，您可以防止集群自动生成 control plane 机器。

3. 删除定义 control plane 机器集的 Kubernetes 清单文件：

```
$ rm -f <installation_directory>/openshift/99_openshift-machine-api_master-control-plane-machine-set.yaml
```

4. 可选：如果您不希望集群置备计算机器，请删除定义 worker 机器的 Kubernetes 清单文件：

```
$ rm -f <installation_directory>/openshift/99_openshift-cluster-api_worker-machineset-*.yaml
```

重要

如果在用户置备的基础架构上安装集群时禁用了 **MachineAPI** 功能，则必须删除定义 worker 机器的 Kubernetes 清单文件。否则，集群将无法安装。

由于您要自行创建和管理 worker 机器，因此不需要初始化这些机器。

5. 检查 **<installation_directory>/manifests/cluster-scheduler-02-config.yml Kubernetes 清单文件中的 mastersSchedulable** 参数是否已设置为 **false**。此设置可防止在 control plane 机器上调度 pod：

   a. 打开 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** 文件。

   b. 找到 **mastersSchedulable** 参数，并确保它被设置为 **false**。

   c. 保存并退出 文件。

6. 可选：如果您不希望 Ingress Operator 代表您创建 DNS 记录，请删除 **<installation_directory>/manifests/cluster-dns-02-config.yml** DNS 配置文件中的 **privateZone 和 publicZone** 部分：

```
apiVersion: config.openshift.io/v1
kind: DNS
metadata:
  creationTimestamp: null
  name: cluster
spec:
  baseDomain: example.openshift.com
  privateZone: 1
    id: mycluster-100419-private-zone
  publicZone: 2
    id: example.openshift.com
status: {}
```

1 2 完全删除此部分。

如果这样做，您必须在后续步骤中手动添加入口 DNS 记录。

7. 要创建 Ignition 配置文件，请从包含安装程序的目录运行以下命令：

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定相同的安装目录。

为安装目录中的 bootstrap、control plane 和计算节点创建 Ignition 配置文件。**kubeadmin-password** 和 **kubeconfig** 文件在 **./<installation_directory>/auth** 目录中创建：

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

其他资源

- 可选：添加入口 DNS 记录

9.12.7. 导出常用变量

9.12.7.1. 提取基础架构名称

Ignition 配置文件包含一个唯一集群标识符，您可以使用它在 Google Cloud Platform(GCP)中唯一地标识您的集群。基础架构名称还用于在 OpenShift Container Platform 安装过程中定位适当的 GCP 资源。提供的 Deployment Manager 模板包含对此基础架构名称的引用，因此您必须提取它。

先决条件

- 已获取 OpenShift Container Platform 安装程序和集群的 pull secret。
- 已为集群生成 Ignition 配置文件。
- 已安装 **jq** 软件包。

流程

- 要从 Ignition 配置文件元数据中提取和查看基础架构名称，请运行以下命令：

```
$ jq -r .infraID <installation_directory>/metadata.json 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

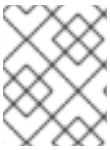
输出示例

```
openshift-vw9j6 1
```

1 此命令的输出是您的集群名称和随机字符串。

9.12.7.2. 为 Deployment Manager 模板导出常用变量

您必须导出与提供的 Deployment Manager 模板一起使用的一组常用变量，它们有助于在 Google Cloud Platform(GCP)上完成用户提供的基础架构安装。



注意

特定的 Deployment Manager 模板可能还需要其他导出变量，这些变量在相关程序中详细介绍。

先决条件

- 获取 OpenShift Container Platform 安装程序和集群的 pull secret。
- 为集群生成 Ignition 配置文件。
- 安装 **jq** 软件包。

流程

1. 导出由提供的 Deployment Manager 模板使用的以下常用变量：

```
$ export BASE_DOMAIN='<base_domain>'
$ export BASE_DOMAIN_ZONE_NAME='<base_domain_zone_name>'
$ export NETWORK_CIDR='10.0.0.0/16'
$ export MASTER_SUBNET_CIDR='10.0.0.0/17'
$ export WORKER_SUBNET_CIDR='10.0.128.0/17'

$ export KUBECONFIG=<installation_directory>/auth/kubeconfig (1)
$ export CLUSTER_NAME=`jq -r .clusterName <installation_directory>/metadata.json`
$ export INFRA_ID=`jq -r .infraID <installation_directory>/metadata.json`
$ export PROJECT_NAME=`jq -r .gcp.projectID <installation_directory>/metadata.json`
$ export REGION=`jq -r .gcp.region <installation_directory>/metadata.json`
```

(1) 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

9.12.8. 在 GCP 中创建 VPC

您必须在 Google Cloud Platform(GCP)中创建一个 VPC，供您的 OpenShift Container Platform 集群使用。您可以自定义 VPC 来满足您的要求。创建 VPC 的一种方法是修改提供的 Deployment Manager 模板。

注意

如果不使用提供的 Deployment Manager 模板来创建 GCP 基础架构，您必须检查提供的信息并手动创建基础架构。如果您的集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 GCP 帐户。
- 为集群生成 Ignition 配置文件。

流程

1. 复制 **VPC 的 Deployment Manager 模板一节中的模板，** 并将它以 **01_vpc.py** 形式保存到计算机上。此模板描述了集群所需的 VPC。

2. 创建 **01_vpc.yaml** 资源定义文件：

```
$ cat <<EOF >01_vpc.yaml
imports:
- path: 01_vpc.py

resources:
- name: cluster-vpc
  type: 01_vpc.py
  properties:
    infra_id: '${INFRA_ID}' 1
    region: '${REGION}' 2
    master_subnet_cidr: '${MASTER_SUBNET_CIDR}' 3
    worker_subnet_cidr: '${WORKER_SUBNET_CIDR}' 4
EOF
```

1. INFRA _ID 是 提取步骤 中的 **INFRA_ID** 基础架构名称。
2. **region** 是集群要部署到的区域，如 **us-central1**。
3. **master_subnet_cidr** 是 master 子网的 CIDR，如 **10.0.0.0/17**。
4. **worker_subnet_cidr** 是 worker 子网的 CIDR，例如 **10.0.128.0/17**。

3. 使用 **gcloud** CLI 创建部署：

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-vpc --config 01_vpc.yaml
```

9.12.8.1. VPC 的 Deployment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的 VPC：

例 9.117. 01_VPC.py Deployment Manager 模板

```
def GenerateConfig(context):

  resources = [{
    'name': context.properties['infra_id'] + '-network',
    'type': 'compute.v1.network',
    'properties': {
        'region': context.properties['region'],
        'autoCreateSubnetworks': False
    }
  }, {
    'name': context.properties['infra_id'] + '-master-subnet',
    'type': 'compute.v1.subnetwork',
    'properties': {
        'region': context.properties['region'],
        'network': '$(ref.' + context.properties['infra_id'] + '-network.selfLink)',
        'ipCidrRange': context.properties['master_subnet_cidr']
```

```
        }
    }, {
        'name': context.properties['infra_id'] + '-worker-subnet',
        'type': 'compute.v1.subnetwork',
        'properties': {
            'region': context.properties['region'],
            'network': '$(ref.' + context.properties['infra_id'] + '-network.selfLink)',
            'ipCidrRange': context.properties['worker_subnet_cidr']
        }
    }, {
        'name': context.properties['infra_id'] + '-router',
        'type': 'compute.v1.router',
        'properties': {
            'region': context.properties['region'],
            'network': '$(ref.' + context.properties['infra_id'] + '-network.selfLink)',
            'nats': [{
                'name': context.properties['infra_id'] + '-nat-master',
                'natIpAllocateOption': 'AUTO_ONLY',
                'minPortsPerVm': 7168,
                'sourceSubnetworkIpRangesToNat': 'LIST_OF_SUBNETWORKS',
                'subnetworks': [{
                    'name': '$(ref.' + context.properties['infra_id'] + '-master-subnet.selfLink)',
                    'sourceIpRangesToNat': ['ALL_IP_RANGES']
                }]
            }, {
                'name': context.properties['infra_id'] + '-nat-worker',
                'natIpAllocateOption': 'AUTO_ONLY',
                'minPortsPerVm': 512,
                'sourceSubnetworkIpRangesToNat': 'LIST_OF_SUBNETWORKS',
                'subnetworks': [{
                    'name': '$(ref.' + context.properties['infra_id'] + '-worker-subnet.selfLink)',
                    'sourceIpRangesToNat': ['ALL_IP_RANGES']
                }]
            }]
        }
    }]

    return {'resources': resources}
```

9.12.9. 用户置备的基础架构对网络的要求

所有 Red Hat Enterprise Linux CoreOS(RHCOS)机器都需要在启动时在 **initramfs** 中配置联网，以获取它们的 Ignition 配置文件。

9.12.9.1. 通过 DHCP 设置集群节点主机名

在 Red Hat Enterprise Linux CoreOS(RHCOS)机器上，主机名是通过 NetworkManager 设置的。默认情况下，机器通过 DHCP 获取其主机名。如果主机名不是由 DHCP 提供，请通过内核参数或者其它方法进行静态设置，请通过反向 DNS 查找获取。反向 DNS 查找在网络初始化后进行，可能需要一些时间来解决。其他系统服务可以在此之前启动，并将主机名检测为 **localhost** 或类似的内容。您可以使用 DHCP 为每个集群节点提供主机名来避免这种情况。

另外，通过 DHCP 设置主机名可以绕过实施 DNS split-horizon 的环境中的手动 DNS 记录名称配置错误。

9.12.9.2. 网络连接要求

您必须配置机器之间的网络连接，以允许 OpenShift Container Platform 集群组件进行通信。每台机器都必须能够解析集群中所有其他机器的主机名。

本节详细介绍了所需的端口。

表 9.45. 用于全机器到所有机器通信的端口

协议	port	描述
ICMP	N/A	网络可访问性测试
TCP	**1936**	指标
	9000-**9999**	主机级别的服务，包括端口 9**100**-**9101** 上的节点导出器，以及端口 **9099** 上的 Cluster Version Operator。
	10250-**10259**	Kubernetes 保留的默认端口
UDP	**4789**	VXLAN
	6081	Geneve
	9000-**9999**	主机级别的服务，包括端口 **9100**到**9101** 上的节点导出器。
	500	IPsec IKE 数据包
	4500	IPsec NAT-T 数据包
	123	UDP 端口 **123** 上的网络时间协议 (NTP) 如果配置了外部 NTP 时间服务器，需要打开 UDP 端口 **123**。
TCP/UDP	**30000**-**32767**	Kubernetes 节点端口
ESP	N/A	IPsec Encapsulating Security Payload(ESP)

表 9.46. 用于所有机器控制平面通信的端口

协议	port	描述
TCP	**6443**	Kubernetes API

表 9.47. control plane 机器用于 control plane 机器通信的端口

协议	port	描述
TCP	**2379-2380**	etcd 服务器和对等端口

9.12.10. 在 GCP 中创建负载均衡器

您必须在 Google Cloud Platform(GCP)中配置负载均衡器，供您的 OpenShift Container Platform 集群使用。创建这些组件的一种方法是修改提供的 Deployment Manager 模板。

注意

如果不使用提供的 Deployment Manager 模板来创建 GCP 基础架构，您必须检查提供的信息并手动创建基础架构。如果您的集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 GCP 帐户。
- 为集群生成 Ignition 配置文件。
- 在 GCP 中创建和配置 VPC 及相关子网。

流程

1. 复制 **内部负载均衡器的 Deployment Manager 模板一节中的模板**，并将它以 **02_lb_int.py** 形式保存到计算机上。此模板描述了集群所需的内部负载均衡对象。
2. 对于外部集群，还要复制本主题 **的外部负载均衡器的 Deployment Manager 模板** 一节中的模板，并将它以 **02_lb_ext.py** 形式保存到计算机上。此模板描述了集群所需的外部负载均衡对象。
3. 导出部署模板使用的变量：
   a. 导出集群网络位置：

```
$ export CLUSTER_NETWORK=(`gcloud compute networks describe ${INFRA_ID}-network --format json | jq -r .selfLink`)
```

   b. 导出 control plane 子网位置：

```
$ export CONTROL_SUBNET=(`gcloud compute networks subnets describe ${INFRA_ID}-master-subnet --region=${REGION} --format json | jq -r .selfLink`)
```

   c. 导出集群使用的三个区：

```
$ export ZONE_0=(`gcloud compute regions describe ${REGION} --format=json | jq -r .zones[0] | cut -d "/" -f9`)
```

```
$ export ZONE_1=(`gcloud compute regions describe ${REGION} --format=json | jq -r .zones[1] | cut -d "/" -f9`)
```

```
$ export ZONE_2=(`gcloud compute regions describe ${REGION} --format=json | jq -r
.zones[2] | cut -d "/" -f9`)
```

4. 创建 **02_infra.yaml** 资源定义文件：

```
$ cat <<EOF >02_infra.yaml
imports:
- path: 02_lb_ext.py
- path: 02_lb_int.py (1)
resources:
- name: cluster-lb-ext (2)
  type: 02_lb_ext.py
  properties:
    infra_id: '${INFRA_ID}' (3)
    region: '${REGION}' (4)
- name: cluster-lb-int
  type: 02_lb_int.py
  properties:
    cluster_network: '${CLUSTER_NETWORK}'
    control_subnet: '${CONTROL_SUBNET}' (5)
    infra_id: '${INFRA_ID}'
    region: '${REGION}'
    zones: (6)
    - '${ZONE_0}'
    - '${ZONE_1}'
    - '${ZONE_2}'
EOF
```

(1) (2) 仅在部署外部集群时需要。

(3) INFRA **_ID 是** 提取步骤 **中的 INFRA_ID** 基础架构名称。

(4) **region** 是集群要部署到的区域，如 **us-central1**。

(5) **control_subnet** 是控制子网的 URI。

(6) **zones** 是 control plane 实例要部署到的区域，如 **us-east1-b**、**us-east1-c** 和 **us-east1-d**。

5. 使用 **gcloud** CLI 创建部署：

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-infra --config 02_infra.yaml
```

6. 导出集群 IP 地址：

```
$ export CLUSTER_IP=(`gcloud compute addresses describe ${INFRA_ID}-cluster-ip --
region=${REGION} --format json | jq -r .address`)
```

7. 对于外部集群，还要导出集群公共 IP 地址：

```
$ export CLUSTER_PUBLIC_IP=(`gcloud compute addresses describe ${INFRA_ID}-cluster-
public-ip --region=${REGION} --format json | jq -r .address`)
```

9.12.10.1. 外部负载均衡器的 Deployment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的外部负载均衡器：

例 9.118. 02_lb_ext.py Deployment Manager 模板

```
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-cluster-public-ip',
        'type': 'compute.v1.address',
        'properties': {
            'region': context.properties['region']
        }
    }, {
        # Refer to docs/dev/kube-apiserver-health-check.md on how to correctly setup health check probe for kube-apiserver
        'name': context.properties['infra_id'] + '-api-http-health-check',
        'type': 'compute.v1.httpHealthCheck',
        'properties': {
            'port': 6080,
            'requestPath': '/readyz'
        }
    }, {
        'name': context.properties['infra_id'] + '-api-target-pool',
        'type': 'compute.v1.targetPool',
        'properties': {
            'region': context.properties['region'],
            'healthChecks': ['$(ref.' + context.properties['infra_id'] + '-api-http-health-check.selfLink)'],
            'instances': []
        }
    }, {
        'name': context.properties['infra_id'] + '-api-forwarding-rule',
        'type': 'compute.v1.forwardingRule',
        'properties': {
            'region': context.properties['region'],
            'IPAddress': '$(ref.' + context.properties['infra_id'] + '-cluster-public-ip.selfLink)',
            'target': '$(ref.' + context.properties['infra_id'] + '-api-target-pool.selfLink)',
            'portRange': '6443'
        }
    }]

    return {'resources': resources}
```

9.12.10.2. 内部负载均衡器的 Deployment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的内部负载均衡器：

例 9.119. 02_lb_int.py Deployment Manager 模板

```
def GenerateConfig(context):
```

```
    backends = []
    for zone in context.properties['zones']:
        backends.append({
            'group': '$(ref.' + context.properties['infra_id'] + '-master-' + zone + '-ig' + '.selfLink)'
        })

    resources = [{
        'name': context.properties['infra_id'] + '-cluster-ip',
        'type': 'compute.v1.address',
        'properties': {
            'addressType': 'INTERNAL',
            'region': context.properties['region'],
            'subnetwork': context.properties['control_subnet']
        }
    }, {
        # Refer to docs/dev/kube-apiserver-health-check.md on how to correctly setup health check
probe for kube-apiserver
        'name': context.properties['infra_id'] + '-api-internal-health-check',
        'type': 'compute.v1.healthCheck',
        'properties': {
            'httpsHealthCheck': {
                'port': 6443,
                'requestPath': '/readyz'
            },
            'type': "HTTPS"
        }
    }, {
        'name': context.properties['infra_id'] + '-api-internal-backend-service',
        'type': 'compute.v1.regionBackendService',
        'properties': {
            'backends': backends,
            'healthChecks': ['$(ref.' + context.properties['infra_id'] + '-api-internal-health-
check.selfLink)'],
            'loadBalancingScheme': 'INTERNAL',
            'region': context.properties['region'],
            'protocol': 'TCP',
            'timeoutSec': 120
        }
    }, {
        'name': context.properties['infra_id'] + '-api-internal-forwarding-rule',
        'type': 'compute.v1.forwardingRule',
        'properties': {
            'backendService': '$(ref.' + context.properties['infra_id'] + '-api-internal-backend-
service.selfLink)',
            'IPAddress': '$(ref.' + context.properties['infra_id'] + '-cluster-ip.selfLink)',
            'loadBalancingScheme': 'INTERNAL',
            'ports': ['6443','22623'],
            'region': context.properties['region'],
            'subnetwork': context.properties['control_subnet']
        }
    }]

    for zone in context.properties['zones']:
        resources.append({
            'name': context.properties['infra_id'] + '-master-' + zone + '-ig',
```

```
            'type': 'compute.v1.instanceGroup',
            'properties': {
                'namedPorts': [
                    {
                        'name': 'ignition',
                        'port': 22623
                    }, {
                        'name': 'https',
                        'port': 6443
                    }
                ],
                'network': context.properties['cluster_network'],
                'zone': zone
            }
        })

    return {'resources': resources}
```

创建外部集群时，除了 **02_lb_ext.py** 模板外，还需要此模板。

9.12.11. 在 GCP 中创建私有 DNS 区域

您必须在 Google Cloud Platform(GCP)中配置私有 DNS 区，供您的 OpenShift Container Platform 集群使用。创建此组件的一种方法是修改提供的 Deployment Manager 模板。

注意

如果不使用提供的 Deployment Manager 模板来创建 GCP 基础架构，您必须检查提供的信息并手动创建基础架构。如果您的集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 GCP 帐户。
- 为集群生成 Ignition 配置文件。
- 在 GCP 中创建和配置 VPC 及相关子网。

流程

1. 复制 **私有 DNS 的 Deployment Manager 模板** 一节中的模板，并将它以 **02_dns.py** 形式保存到计算机上。此模板描述了集群所需的私有 DNS 对象。
2. 创建 **02_dns.yaml** 资源定义文件：

```
$ cat <<EOF >02_dns.yaml
imports:
- path: 02_dns.py

resources:
- name: cluster-dns
  type: 02_dns.py
  properties:
```

```
  infra_id: '${INFRA_ID}' (1)
  cluster_domain: '${CLUSTER_NAME}.${BASE_DOMAIN}' (2)
  cluster_network: '${CLUSTER_NETWORK}' (3)
EOF
```

1. INFRA _ID 是 提取步骤 中的 **INFRA_ID** 基础架构名称。
2. **cluster_domain** 是集群的域，如 **openshift.example.com**。
3. **cluster_network** 是集群网络的 **selfLink** URL。

3. 使用 **gcloud** CLI 创建部署：

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-dns --config 02_dns.yaml
```

4. 由于 Deployment Manager 的限制，模板不会创建 DNS 条目，因此您必须手动创建它们：
   a. 添加内部 DNS 条目：

```
$ if [ -f transaction.yaml ]; then rm transaction.yaml; fi
$ gcloud dns record-sets transaction start --zone ${INFRA_ID}-private-zone
$ gcloud dns record-sets transaction add ${CLUSTER_IP} --name
api.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 60 --type A --zone ${INFRA_ID}-
private-zone
$ gcloud dns record-sets transaction add ${CLUSTER_IP} --name api-
int.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 60 --type A --zone ${INFRA_ID}-
private-zone
$ gcloud dns record-sets transaction execute --zone ${INFRA_ID}-private-zone
```

   b. 对于外部集群，还要添加外部 DNS 条目：

```
$ if [ -f transaction.yaml ]; then rm transaction.yaml; fi
$ gcloud dns record-sets transaction start --zone ${BASE_DOMAIN_ZONE_NAME}
$ gcloud dns record-sets transaction add ${CLUSTER_PUBLIC_IP} --name
api.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 60 --type A --zone
${BASE_DOMAIN_ZONE_NAME}
$ gcloud dns record-sets transaction execute --zone ${BASE_DOMAIN_ZONE_NAME}
```

9.12.11.1. 私有 DNS 的 Deployment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的私有 DNS：

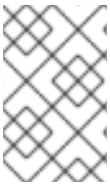
例 9.120. 02_DNS.py Deployment Manager 模板

```
def GenerateConfig(context):

  resources = [{
    'name': context.properties['infra_id'] + '-private-zone',
    'type': 'dns.v1.managedZone',
    'properties': {
      'description': '',
      'dnsName': context.properties['cluster_domain'] + '.',
```

```
            'visibility': 'private',
            'privateVisibilityConfig': {
                'networks': [{
                    'networkUrl': context.properties['cluster_network']
                }]
            }
        }
    }]

    return {'resources': resources}
```

9.12.12. 在 GCP 中创建防火墙规则

您必须在 Google Cloud Platform(GCP)中创建防火墙规则，供您的 OpenShift Container Platform 集群使用。创建这些组件的一种方法是修改提供的 Deployment Manager 模板。



注意

如果不使用提供的 Deployment Manager 模板来创建 GCP 基础架构，您必须检查提供的信息并手动创建基础架构。如果您的集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 GCP 帐户。
- 为集群生成 Ignition 配置文件。
- 在 GCP 中创建和配置 VPC 及相关子网。

流程

1. 复制 **防火墙规则的 Deployment Manager 模板一节中的模板**，并将它以 **03_firewall.py** 形式保存到计算机上。此模板描述了集群所需的安全组。
2. 创建 **03_firewall.yaml** 资源定义文件：

```
$ cat <<EOF >03_firewall.yaml
imports:
- path: 03_firewall.py

resources:
- name: cluster-firewall
  type: 03_firewall.py
  properties:
    allowed_external_cidr: '0.0.0.0/0' (1)
    infra_id: '${INFRA_ID}' (2)
    cluster_network: '${CLUSTER_NETWORK}' (3)
    network_cidr: '${NETWORK_CIDR}' (4)
EOF
```

(1) **allowed_external_cidr** 是 CIDR 范围，可访问集群 API 和 SSH 到 bootstrap 主机。对于内部集群，将此值设置为 **${NETWORK_CIDR}**。

2. INFRA _ID 是 提取步骤 中的 **INFRA_ID** 基础架构名称。
3. **cluster_network** 是集群网络的 **selfLink** URL。
4. **NETWORK_CIDR 是** VPC 网络的 CIDR，如 **10.0.0.0/16**。

3. 使用 **gcloud** CLI 创建部署：

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-firewall --config
03_firewall.yaml
```

9.12.12.1. 防火墙规则的 Deployment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的防火墙破坏：

例 9.121. 03_firewall.py Deployment Manager 模板

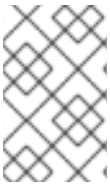
```
def GenerateConfig(context):

  resources = [{
    'name': context.properties['infra_id'] + '-bootstrap-in-ssh',
    'type': 'compute.v1.firewall',
    'properties': {
      'network': context.properties['cluster_network'],
      'allowed': [{
        'IPProtocol': 'tcp',
        'ports': ['22']
      }],
      'sourceRanges': [context.properties['allowed_external_cidr']],
      'targetTags': [context.properties['infra_id'] + '-bootstrap']
    }
  }, {
    'name': context.properties['infra_id'] + '-api',
    'type': 'compute.v1.firewall',
    'properties': {
      'network': context.properties['cluster_network'],
      'allowed': [{
        'IPProtocol': 'tcp',
        'ports': ['6443']
      }],
      'sourceRanges': [context.properties['allowed_external_cidr']],
      'targetTags': [context.properties['infra_id'] + '-master']
    }
  }, {
    'name': context.properties['infra_id'] + '-health-checks',
    'type': 'compute.v1.firewall',
    'properties': {
      'network': context.properties['cluster_network'],
      'allowed': [{
        'IPProtocol': 'tcp',
        'ports': ['6080', '6443', '22624']
      }],
      'sourceRanges': ['35.191.0.0/16', '130.211.0.0/22', '209.85.152.0/22', '209.85.204.0/22'],
```

```
            'targetTags': [context.properties['infra_id'] + '-master']
        }
    }, {
        'name': context.properties['infra_id'] + '-etcd',
        'type': 'compute.v1.firewall',
        'properties': {
            'network': context.properties['cluster_network'],
            'allowed': [{
                'IPProtocol': 'tcp',
                'ports': ['2379-2380']
            }],
            'sourceTags': [context.properties['infra_id'] + '-master'],
            'targetTags': [context.properties['infra_id'] + '-master']
        }
    }, {
        'name': context.properties['infra_id'] + '-control-plane',
        'type': 'compute.v1.firewall',
        'properties': {
            'network': context.properties['cluster_network'],
            'allowed': [{
                'IPProtocol': 'tcp',
                'ports': ['10257']
            },{
                'IPProtocol': 'tcp',
                'ports': ['10259']
            },{
                'IPProtocol': 'tcp',
                'ports': ['22623']
            }],
            'sourceTags': [
                context.properties['infra_id'] + '-master',
                context.properties['infra_id'] + '-worker'
            ],
            'targetTags': [context.properties['infra_id'] + '-master']
        }
    }, {
        'name': context.properties['infra_id'] + '-internal-network',
        'type': 'compute.v1.firewall',
        'properties': {
            'network': context.properties['cluster_network'],
            'allowed': [{
                'IPProtocol': 'icmp'
            },{
                'IPProtocol': 'tcp',
                'ports': ['22']
            }],
            'sourceRanges': [context.properties['network_cidr']],
            'targetTags': [
                context.properties['infra_id'] + '-master',
                context.properties['infra_id'] + '-worker'
            ]
        }
    }, {
        'name': context.properties['infra_id'] + '-internal-cluster',
        'type': 'compute.v1.firewall',
        'properties': {
```

```
            'network': context.properties['cluster_network'],
            'allowed': [{
                'IPProtocol': 'udp',
                'ports': ['4789', '6081']
            },{
                'IPProtocol': 'udp',
                'ports': ['500', '4500']
            },{
                'IPProtocol': 'esp',
            },{
                'IPProtocol': 'tcp',
                'ports': ['9000-9999']
            },{
                'IPProtocol': 'udp',
                'ports': ['9000-9999']
            },{
                'IPProtocol': 'tcp',
                'ports': ['10250']
            },{
                'IPProtocol': 'tcp',
                'ports': ['30000-32767']
            },{
                'IPProtocol': 'udp',
                'ports': ['30000-32767']
            }],
            'sourceTags': [
                context.properties['infra_id'] + '-master',
                context.properties['infra_id'] + '-worker'
            ],
            'targetTags': [
                context.properties['infra_id'] + '-master',
                context.properties['infra_id'] + '-worker'
            ]
        }
    }]

    return {'resources': resources}
```

9.12.13. 在 GCP 中创建 IAM 角色

您必须在 Google Cloud Platform(GCP)中创建 IAM 角色，供您的 OpenShift Container Platform 集群使用。创建这些组件的一种方法是修改提供的 Deployment Manager 模板。



注意

如果不使用提供的 Deployment Manager 模板来创建 GCP 基础架构，您必须检查提供的信息并手动创建基础架构。如果您的集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 GCP 帐户。
- 为集群生成 Ignition 配置文件。

- 在 GCP 中创建和配置 VPC 及相关子网。

流程

1. 复制 **IAM 角色的 Deployment Manager 模板一节中的模板**，并将它以 **03_iam.py** 形式保存到计算机上。此模板描述了集群所需的 IAM 角色。
2. 创建 **03_iam.yaml** 资源定义文件：

```
$ cat <<EOF >03_iam.yaml
imports:
- path: 03_iam.py
resources:
- name: cluster-iam
  type: 03_iam.py
  properties:
    infra_id: '${INFRA_ID}' (1)
EOF
```

(1) INFRA **_ID 是** 提取步骤 **中的 INFRA_ID** 基础架构名称。

3. 使用 **gcloud** CLI 创建部署：

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-iam --config 03_iam.yaml
```

4. 导出 master 服务帐户的变量：

```
$ export MASTER_SERVICE_ACCOUNT=(`gcloud iam service-accounts list --filter
"email~^${INFRA_ID}-m@${PROJECT_NAME}." --format json | jq -r '.[0].email'`)
```

5. 导出 worker 服务帐户的变量：

```
$ export WORKER_SERVICE_ACCOUNT=(`gcloud iam service-accounts list --filter
"email~^${INFRA_ID}-w@${PROJECT_NAME}." --format json | jq -r '.[0].email'`)
```

6. 导出托管计算机器的子网的变量：

```
$ export COMPUTE_SUBNET=(`gcloud compute networks subnets describe ${INFRA_ID}-
worker-subnet --region=${REGION} --format json | jq -r .selfLink`)
```

7. 由于 Deployment Manager 的限制，模板不会创建策略绑定，因此您必须手动创建它们：

```
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/compute.instanceAdmin"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/compute.networkAdmin"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/compute.securityAdmin"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/iam.serviceAccountUser"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${MASTER_SERVICE_ACCOUNT}" --role "roles/storage.admin"
```

```
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${WORKER_SERVICE_ACCOUNT}" --role "roles/compute.viewer"
$ gcloud projects add-iam-policy-binding ${PROJECT_NAME} --member
"serviceAccount:${WORKER_SERVICE_ACCOUNT}" --role "roles/storage.admin"
```

8. 创建服务帐户密钥并将其存储在本地以供以后使用：

```
$ gcloud iam service-accounts keys create service-account-key.json --iam-
account=${MASTER_SERVICE_ACCOUNT}
```

9.12.13.1. IAM 角色的 Deployment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的 IAM 角色：

例 9.122. 03_IAM.py Deployment Manager 模板

```
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-master-node-sa',
        'type': 'iam.v1.serviceAccount',
        'properties': {
            'accountId': context.properties['infra_id'] + '-m',
            'displayName': context.properties['infra_id'] + '-master-node'
        }
    }, {
        'name': context.properties['infra_id'] + '-worker-node-sa',
        'type': 'iam.v1.serviceAccount',
        'properties': {
            'accountId': context.properties['infra_id'] + '-w',
            'displayName': context.properties['infra_id'] + '-worker-node'
        }
    }]

    return {'resources': resources}
```

9.12.14. 为 GCP 基础架构创建 RHCOS 集群镜像

您必须对 OpenShift Container Platform 节点的 Google Cloud Platform(GCP)使用有效的 Red Hat Enterprise Linux CoreOS(RHCOS)镜像。

流程

1. 从 RHCOS 镜像镜像页面获取 RHCOS 镜像。

重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每个发行版本而改变。您必须下载最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。如果可用，请使用与 OpenShift Container Platform 版本匹配的镜像版本。

文件名包含 OpenShift Container Platform 版本号，格式为 **rhcos-<version>-<arch>-gcp.<arch>.tar.gz**。

2. 创建 Google 存储桶：

```
$ gsutil mb gs://<bucket_name>
```

3. 将 RHCOS 镜像上传到 Google 存储桶：

```
$ gsutil cp <downloaded_image_file_path>/rhcos-<version>-x86_64-gcp.x86_64.tar.gz
gs://<bucket_name>
```

4. 将上传的 RHCOS 镜像位置导出为变量：

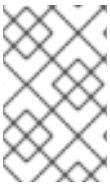
```
$ export IMAGE_SOURCE=gs://<bucket_name>/rhcos-<version>-x86_64-gcp.x86_64.tar.gz
```

5. 创建集群镜像：

```
$ gcloud compute images create "${INFRA_ID}-rhcos-image" \
    --source-uri="${IMAGE_SOURCE}"
```

9.12.15. 在 GCP 中创建 bootstrap 机器

您必须在 Google Cloud Platform(GCP)中创建 bootstrap 机器，以便在 OpenShift Container Platform 集群初始化过程中使用。创建此机器的一种方法是修改提供的 Deployment Manager 模板。



注意

如果不使用提供的 Deployment Manager 模板来创建 bootstrap 机器，您必须检查提供的信息并手动创建基础架构。如果您的集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 GCP 帐户。
- 为集群生成 Ignition 配置文件。
- 在 GCP 中创建和配置 VPC 及相关子网。
- 在 GCP 中创建和配置联网与负载均衡器。
- 创建 control plane 和计算角色。
- 确保已安装 pyOpenSSL。

流程

1. 复制 **bootstrap 机器的 Deployment Manager 模板一节中的模板**，并将它以 **04_bootstrap.py** 形式保存到计算机上。此模板描述了集群所需的 bootstrap 机器。
2. 导出安装程序所需的 Red Hat Enterprise Linux CoreOS(RHCOS)镜像的位置：

```
$ export CLUSTER_IMAGE=(`gcloud compute images describe ${INFRA_ID}-rhcos-image --
format json | jq -r .selfLink`)
```

3. 创建存储桶并上传 **bootstrap.ign** 文件：

```
$ gsutil mb gs://${INFRA_ID}-bootstrap-ignition
```

```
$ gsutil cp <installation_directory>/bootstrap.ign gs://${INFRA_ID}-bootstrap-ignition/
```

4. 为 bootstrap 实例创建一个签名的 URL，用于访问 Ignition 配置。将输出中的 URL 导出为变量：

```
$ export BOOTSTRAP_IGN=`gsutil signurl -d 1h service-account-key.json gs://${INFRA_ID}-
bootstrap-ignition/bootstrap.ign | grep "^gs:" | awk '{print $5}'`
```

5. 创建 **04_bootstrap.yaml** 资源定义文件：

```
$ cat <<EOF >04_bootstrap.yaml
imports:
- path: 04_bootstrap.py

resources:
- name: cluster-bootstrap
  type: 04_bootstrap.py
  properties:
    infra_id: '${INFRA_ID}' (1)
    region: '${REGION}' (2)
    zone: '${ZONE_0}' (3)

    cluster_network: '${CLUSTER_NETWORK}' (4)
    control_subnet: '${CONTROL_SUBNET}' (5)
    image: '${CLUSTER_IMAGE}' (6)
    machine_type: 'n1-standard-4' (7)
    root_volume_size: '128' (8)

    bootstrap_ign: '${BOOTSTRAP_IGN}' (9)
EOF
```

(1) INFRA _ID 是 提取步骤 中的 **INFRA_ID** 基础架构名称。

(2) **region** 是集群要部署到的区域，如 **us-central1**。

(3) **zone** 是 bootstrap 实例要部署到的区域，如 **us-central1-b**。

(4) **cluster_network** 是集群网络的 **selfLink** URL。

(5) **control_subnet** 是控制子网的 **selfLink** URL。

(6) **image** 是 RHCOS 镜像的 **selfLink** URL。

(7) **machine_type** 是实例的机器类型，如 **n1-standard-4**。

(8) **root_volume_size** 是 bootstrap 计算机的引导磁盘大小。

9 **bootstrap_ign** 是创建签名 URL 时的 URL 输出。

6. 使用 **gcloud** CLI 创建部署：

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-bootstrap --config
04_bootstrap.yaml
```

7. 由于 Deployment Manager 的限制，模板无法管理负载均衡器成员资格，因此您必须手动添加 bootstrap 机器。

   a. 将 bootstrap 实例添加到内部负载均衡器实例组中：

```
$ gcloud compute instance-groups unmanaged add-instances \
    ${INFRA_ID}-bootstrap-ig --zone=${ZONE_0} --instances=${INFRA_ID}-bootstrap
```

   b. 将 bootstrap 实例组添加到内部负载均衡器后端服务中：

```
$ gcloud compute backend-services add-backend \
    ${INFRA_ID}-api-internal-backend-service --region=${REGION} --instance-
group=${INFRA_ID}-bootstrap-ig --instance-group-zone=${ZONE_0}
```

9.12.15.1. bootstrap 机器的 Deployment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的 bootstrap 机器：

例 9.123. 04_bootstrap.py Deployment Manager 模板

```
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-bootstrap-public-ip',
        'type': 'compute.v1.address',
        'properties': {
            'region': context.properties['region']
        }
    }, {
        'name': context.properties['infra_id'] + '-bootstrap',
        'type': 'compute.v1.instance',
        'properties': {
            'disks': [{
                'autoDelete': True,
                'boot': True,
                'initializeParams': {
                    'diskSizeGb': context.properties['root_volume_size'],
                    'sourceImage': context.properties['image']
                }
            }],
            'machineType': 'zones/' + context.properties['zone'] + '/machineTypes/' +
context.properties['machine_type'],
            'metadata': {
                'items': [{
                    'key': 'user-data',
```

```
                'value': '{"ignition":{"config":{"replace":{"source":"' + context.properties['bootstrap_ign']
+ '"}},"version":"3.2.0"}}',
            }]
        },
        'networkInterfaces': [{
            'subnetwork': context.properties['control_subnet'],
            'accessConfigs': [{
                'natIP': '$(ref.' + context.properties['infra_id'] + '-bootstrap-public-ip.address)'
            }]
        }],
        'tags': {
            'items': [
                context.properties['infra_id'] + '-master',
                context.properties['infra_id'] + '-bootstrap'
            ]
        },
        'zone': context.properties['zone']
    }
}, {
    'name': context.properties['infra_id'] + '-bootstrap-ig',
    'type': 'compute.v1.instanceGroup',
    'properties': {
        'namedPorts': [
            {
                'name': 'ignition',
                'port': 22623
            }, {
                'name': 'https',
                'port': 6443
            }
        ],
        'network': context.properties['cluster_network'],
        'zone': context.properties['zone']
    }
}]

return {'resources': resources}
```

9.12.16. 在 GCP 中创建 control plane 机器

您必须在 Google Cloud Platform(GCP)中创建 control plane 机器，供您的集群使用。创建这些机器的一种方法是修改提供的 Deployment Manager 模板。

注意

如果不使用提供的 Deployment Manager 模板来创建 control plane 机器，您必须检查提供的信息并手动创建基础架构。如果您的集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 GCP 帐户。
- 为集群生成 Ignition 配置文件。

- 在 GCP 中创建和配置 VPC 及相关子网。
- 在 GCP 中创建和配置联网与负载均衡器。
- 创建 control plane 和计算角色。
- 创建 bootstrap 机器。

流程

1. 复制 **control plane 机器的 Deployment Manager 模板一节中的模板**，并将它以 **05_control_plane.py** 形式保存到计算机上。此模板描述了集群所需的 control plane 机器。
2. 导出资源定义所需的以下变量：

```
$ export MASTER_IGNITION=`cat <installation_directory>/master.ign`
```

3. 创建 **05_control_plane.yaml** 资源定义文件：

```
$ cat <<EOF >05_control_plane.yaml
imports:
- path: 05_control_plane.py

resources:
- name: cluster-control-plane
  type: 05_control_plane.py
  properties:
    infra_id: '${INFRA_ID}' (1)
    zones: (2)
    - '${ZONE_0}'
    - '${ZONE_1}'
    - '${ZONE_2}'

    control_subnet: '${CONTROL_SUBNET}' (3)
    image: '${CLUSTER_IMAGE}' (4)
    machine_type: 'n1-standard-4' (5)
    root_volume_size: '128'
    service_account_email: '${MASTER_SERVICE_ACCOUNT}' (6)

    ignition: '${MASTER_IGNITION}' (7)
EOF
```

(1) INFRA **_ID 是** 提取步骤 **中的 INFRA_ID** 基础架构名称。

(2) **zones** 是 control plane 实例要部署到的区域，如 **us-central1-a**、**us-central1-b** 和 **us-central1-c**。

(3) **control_subnet** 是控制子网的 **selfLink** URL。

(4) **image** 是 RHCOS 镜像的 **selfLink** URL。

(5) **machine_type** 是实例的机器类型，如 **n1-standard-4**。

(6) **service_account_email** 是您创建的 master 服务帐户的电子邮件地址。

7 **Ignition** 是 **master.ign** 文件的内容。

4. 使用 **gcloud** CLI 创建部署：

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-control-plane --config
05_control_plane.yaml
```

5. 由于 Deployment Manager 的限制，模板无法管理负载均衡器成员资格，因此您必须手动添加 control plane 机器。

   - 运行以下命令，将 control plane 机器添加到适当的实例组中：

```
$ gcloud compute instance-groups unmanaged add-instances ${INFRA_ID}-master-
${ZONE_0}-ig --zone=${ZONE_0} --instances=${INFRA_ID}-master-0
```

```
$ gcloud compute instance-groups unmanaged add-instances ${INFRA_ID}-master-
${ZONE_1}-ig --zone=${ZONE_1} --instances=${INFRA_ID}-master-1
```

```
$ gcloud compute instance-groups unmanaged add-instances ${INFRA_ID}-master-
${ZONE_2}-ig --zone=${ZONE_2} --instances=${INFRA_ID}-master-2
```

   - 对于外部集群，还必须运行以下命令将 control plane 机器添加到目标池中：

```
$ gcloud compute target-pools add-instances ${INFRA_ID}-api-target-pool --instances-
zone="${ZONE_0}" --instances=${INFRA_ID}-master-0
```

```
$ gcloud compute target-pools add-instances ${INFRA_ID}-api-target-pool --instances-
zone="${ZONE_1}" --instances=${INFRA_ID}-master-1
```

```
$ gcloud compute target-pools add-instances ${INFRA_ID}-api-target-pool --instances-
zone="${ZONE_2}" --instances=${INFRA_ID}-master-2
```

9.12.16.1. control plane 机器的 Deployment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的 control plane 机器：

例 9.124. 05_control_plane.py Deployment Manager 模板

```
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-master-0',
        'type': 'compute.v1.instance',
        'properties': {
            'disks': [{
                'autoDelete': True,
                'boot': True,
                'initializeParams': {
                    'diskSizeGb': context.properties['root_volume_size'],
                    'diskType': 'zones/' + context.properties['zones'][0] + '/diskTypes/pd-ssd',
```

```
                'sourceImage': context.properties['image']
            }
        }],
        'machineType': 'zones/' + context.properties['zones'][0] + '/machineTypes/' +
context.properties['machine_type'],
        'metadata': {
            'items': [{
                'key': 'user-data',
                'value': context.properties['ignition']
            }]
        },
        'networkInterfaces': [{
            'subnetwork': context.properties['control_subnet']
        }],
        'serviceAccounts': [{
            'email': context.properties['service_account_email'],
            'scopes': ['https://www.googleapis.com/auth/cloud-platform']
        }],
        'tags': {
            'items': [
                context.properties['infra_id'] + '-master',
            ]
        },
        'zone': context.properties['zones'][0]
    }
}, {
    'name': context.properties['infra_id'] + '-master-1',
    'type': 'compute.v1.instance',
    'properties': {
        'disks': [{
            'autoDelete': True,
            'boot': True,
            'initializeParams': {
                'diskSizeGb': context.properties['root_volume_size'],
                'diskType': 'zones/' + context.properties['zones'][1] + '/diskTypes/pd-ssd',
                'sourceImage': context.properties['image']
            }
        }],
        'machineType': 'zones/' + context.properties['zones'][1] + '/machineTypes/' +
context.properties['machine_type'],
        'metadata': {
            'items': [{
                'key': 'user-data',
                'value': context.properties['ignition']
            }]
        },
        'networkInterfaces': [{
            'subnetwork': context.properties['control_subnet']
        }],
        'serviceAccounts': [{
            'email': context.properties['service_account_email'],
            'scopes': ['https://www.googleapis.com/auth/cloud-platform']
        }],
        'tags': {
            'items': [
                context.properties['infra_id'] + '-master',
```

```
                ]
            },
            'zone': context.properties['zones'][1]
        }
    }, {
        'name': context.properties['infra_id'] + '-master-2',
        'type': 'compute.v1.instance',
        'properties': {
            'disks': [{
                'autoDelete': True,
                'boot': True,
                'initializeParams': {
                    'diskSizeGb': context.properties['root_volume_size'],
                    'diskType': 'zones/' + context.properties['zones'][2] + '/diskTypes/pd-ssd',
                    'sourceImage': context.properties['image']
                }
            }],
            'machineType': 'zones/' + context.properties['zones'][2] + '/machineTypes/' +
context.properties['machine_type'],
            'metadata': {
                'items': [{
                    'key': 'user-data',
                    'value': context.properties['ignition']
                }]
            },
            'networkInterfaces': [{
                'subnetwork': context.properties['control_subnet']
            }],
            'serviceAccounts': [{
                'email': context.properties['service_account_email'],
                'scopes': ['https://www.googleapis.com/auth/cloud-platform']
            }],
            'tags': {
                'items': [
                    context.properties['infra_id'] + '-master',
                ]
            },
            'zone': context.properties['zones'][2]
        }
    }]

    return {'resources': resources}
```

9.12.17. 等待 bootstrap 完成并删除 GCP 中的 bootstrap 资源

在 Google Cloud Platform(GCP)中创建所有所需的基础架构后，请等待您通过安装程序生成的 Ignition 配置文件所置备的机器上完成 bootstrap 过程。

先决条件

- 配置 GCP 帐户。
- 为集群生成 Ignition 配置文件。

- 在 GCP 中创建和配置 VPC 及相关子网。
- 在 GCP 中创建和配置联网与负载均衡器。
- 创建 control plane 和计算角色。
- 创建 bootstrap 机器。
- 创建 control plane 机器。

流程

1. 进入包含安装程序的目录并运行以下命令：

```
$ ./openshift-install wait-for bootstrap-complete --dir <installation_directory> \ 1
    --log-level info 2
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不是 **info**。

如果命令退出时没有 **FATAL** 警告，则您的生产环境 control plane 已被初始化。

2. 删除 bootstrap 资源：

```
$ gcloud compute backend-services remove-backend ${INFRA_ID}-api-internal-backend-
service --region=${REGION} --instance-group=${INFRA_ID}-bootstrap-ig --instance-group-
zone=${ZONE_0}
```

```
$ gsutil rm gs://${INFRA_ID}-bootstrap-ignition/bootstrap.ign
```

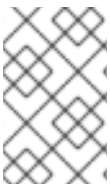
```
$ gsutil rb gs://${INFRA_ID}-bootstrap-ignition
```

```
$ gcloud deployment-manager deployments delete ${INFRA_ID}-bootstrap
```

9.12.18. 在 GCP 中创建额外的 worker 机器

您可以通过分散启动各个实例或利用集群外自动化流程（如自动扩展组），在 Google Cloud Platform(GCP)中为您的集群创建 worker 机器。您还可以利用 OpenShift Container Platform 中的内置集群扩展机制和机器 API。

在本例中，您要使用 Deployment Manager 模板手动启动一个实例。通过在 文件中包括类型为 **06_worker.py** 的其他资源，即可启动其他实例。



注意

如果不使用提供的 Deployment Manager 模板来创建 worker 机器，您必须检查提供的信息并手动创建基础架构。如果您的集群没有正确初始化，您可能需要联系红帽支持并提供您的安装日志。

先决条件

- 配置 GCP 帐户。
- 为集群生成 Ignition 配置文件。
- 在 GCP 中创建和配置 VPC 及相关子网。
- 在 GCP 中创建和配置联网与负载均衡器。
- 创建 control plane 和计算角色。
- 创建 bootstrap 机器。
- 创建 control plane 机器。

流程

1. 复制 **worker 机器的 Deployment Manager 模板一节中的模板**，并将它以 **06_worker.py** 形式保存到计算机上。此模板描述了集群所需的 worker 机器。
2. 导出资源定义使用的变量。
   a. 导出托管计算机器的子网：

   ```
   $ export COMPUTE_SUBNET=(`gcloud compute networks subnets describe
   ${INFRA_ID}-worker-subnet --region=${REGION} --format json | jq -r .selfLink`)
   ```

   b. 为您的服务帐户导出电子邮件地址：

   ```
   $ export WORKER_SERVICE_ACCOUNT=(`gcloud iam service-accounts list --filter
   "email~^${INFRA_ID}-w@${PROJECT_NAME}." --format json | jq -r '.[0].email'`)
   ```

   c. 导出计算机器 Ignition 配置文件的位置：

   ```
   $ export WORKER_IGNITION=`cat <installation_directory>/worker.ign`
   ```

3. 创建 **06_worker.yaml** 资源定义文件：

```
$ cat <<EOF >06_worker.yaml
imports:
- path: 06_worker.py

resources:
- name: 'worker-0' (1)
  type: 06_worker.py
  properties:
    infra_id: '${INFRA_ID}' (2)
    zone: '${ZONE_0}' (3)
    compute_subnet: '${COMPUTE_SUBNET}' (4)
    image: '${CLUSTER_IMAGE}' (5)
    machine_type: 'n1-standard-4' (6)
    root_volume_size: '128'
    service_account_email: '${WORKER_SERVICE_ACCOUNT}' (7)
    ignition: '${WORKER_IGNITION}' (8)
- name: 'worker-1'
```

```
    type: 06_worker.py
    properties:
      infra_id: '${INFRA_ID}' 9
      zone: '${ZONE_1}' 10
      compute_subnet: '${COMPUTE_SUBNET}' 11
      image: '${CLUSTER_IMAGE}' 12
      machine_type: 'n1-standard-4' 13
      root_volume_size: '128'
      service_account_email: '${WORKER_SERVICE_ACCOUNT}' 14
      ignition: '${WORKER_IGNITION}' 15
EOF
```

1 **name** 是 worker 机器的名称，如 **worker-0**。

2 9 INFRA _**ID 是** 提取步骤 **中的 INFRA_ID** 基础架构名称。

3 10 **zone** 是 worker 机器要部署到的区域，如 **us-central1-a**。

4 11 **compute_subnet** 是计算子网的 **selfLink** URL。

5 12 **image** 是 RHCOS 镜像的 **selfLink** URL。[1]

6 13 **machine_type** 是实例的机器类型，如 **n1-standard-4**。

7 14 **service_account_email** 是您创建的 worker 服务帐户的电子邮件地址。

8 15 **ignition** 是 **worker.ign** 文件的内容。

4. 可选：如果要启动更多实例，请在 **06_worker.yaml** 资源定义文件中包含类型为 **06_worker.py** 的其他资源。

5. 使用 **gcloud** CLI 创建部署：

```
$ gcloud deployment-manager deployments create ${INFRA_ID}-worker --config
06_worker.yaml
```

1. 要使用 GCP Marketplace 镜像，请指定要使用的功能：

   - OpenShift Container Platform: **https://www.googleapis.com/compute/v1/projects/redhat-marketplace-public/global/images/redhat-coreos-ocp-413-x86-64-202305021736**
   - OpenShift Platform Plus: **https://www.googleapis.com/compute/v1/projects/redhat-marketplace-public/global/images/redhat-coreos-opp-413-x86-64-202305021736**
   - OpenShift Kubernetes Engine: **https://www.googleapis.com/compute/v1/projects/redhat-marketplace-public/global/images/redhat-coreos-oke-413-x86-64-202305021736**

9.12.18.1. worker 机器的 Deployment Manager 模板

您可以使用以下 Deployment Manager 模板来部署 OpenShift Container Platform 集群所需的 worker 机器：

例 9.125. 06_worker.py Deployment Manager 模板

```
def GenerateConfig(context):

    resources = [{
        'name': context.properties['infra_id'] + '-' + context.env['name'],
        'type': 'compute.v1.instance',
        'properties': {
            'disks': [{
                'autoDelete': True,
                'boot': True,
                'initializeParams': {
                    'diskSizeGb': context.properties['root_volume_size'],
                    'sourceImage': context.properties['image']
                }
            }],
            'machineType': 'zones/' + context.properties['zone'] + '/machineTypes/' +
context.properties['machine_type'],
            'metadata': {
                'items': [{
                    'key': 'user-data',
                    'value': context.properties['ignition']
                }]
            },
            'networkInterfaces': [{
                'subnetwork': context.properties['compute_subnet']
            }],
            'serviceAccounts': [{
                'email': context.properties['service_account_email'],
                'scopes': ['https://www.googleapis.com/auth/cloud-platform']
            }],
            'tags': {
                'items': [
                    context.properties['infra_id'] + '-worker',
                ]
            },
            'zone': context.properties['zone']
        }
    }]

    return {'resources': resources}
```

9.12.19. 使用 CLI 登录集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

9.12.20. 禁用默认的 OperatorHub 目录源

在 OpenShift Container Platform 安装过程中，默认为 OperatorHub 配置由红帽和社区项目提供的源内容的 operator 目录。在受限网络环境中，必须以集群管理员身份禁用默认目录。

流程

- 通过在 **OperatorHub** 对象中添加 **disableAllDefaultSources: true** 来禁用默认目录的源：

```
$ oc patch OperatorHub cluster --type json \
    -p '[{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]'
```

提示

或者，您可以使用 Web 控制台管理目录源。在 **Administration → Cluster Settings → Configuration → OperatorHub** 页面中，点 **Sources** 选项卡，您可以在其中创建、更新、删除、禁用和启用单独的源。

9.12.21. 批准机器的证书签名请求

当您将机器添加到集群时，会为您添加的每台机器生成两个待处理证书签名请求(CSR)。您必须确认这些 CSR 已获得批准，或根据需要自行批准。必须首先批准客户端请求，然后批准服务器请求。

先决条件

- 您已将机器添加到集群中。

流程

1. 确认集群可以识别这些机器：

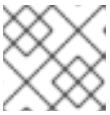
```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
```

```
master-0  Ready    master  63m  v1.29.4
master-1  Ready    master  63m  v1.29.4
master-2  Ready    master  64m  v1.29.4
```

输出中列出了您创建的所有机器。



注意

在有些 CSR 被批准前，前面的输出可能不包括计算节点（也称为 worker 节点）。

2. 检查待处理的 CSR，并确保添加到集群中的每台机器都有 **Pending** 或 **Approved** 状态的客户端请求：

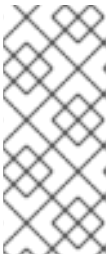
```
$ oc get csr
```

输出示例

```
NAME        AGE    REQUESTOR                                                   CONDITION
csr-8b2br   15m    system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper   Pending
csr-8vnps   15m    system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper   Pending
...
```

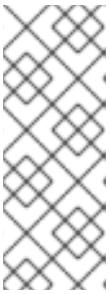
在本例中，两台机器加入集群。您可能会在列表中看到更多已批准的 CSR。

3. 如果 CSR 没有获得批准，在您添加的机器的所有待处理 CSR 都处于 **Pending 状态** 后，请批准集群机器的 CSR：



注意

由于 CSR 会自动轮转，因此请在将机器添加到集群后一小时内批准您的 CSR。如果没有在一小时内批准它们，证书将会轮转，每个节点会存在多个证书。您必须批准所有这些证书。批准客户端 CSR 后，Kubelet 为服务证书创建一个二级 CSR，这需要手动批准。然后，如果 Kubelet 请求具有相同参数的新证书，则后续提供证书续订请求由 **machine-approver** 自动批准。



注意

对于在未启用机器 API 的平台上运行的集群，如裸机和其他用户置备的基础架构，您必须实施一种方法来自动批准 kubelet 提供证书请求(CSR)。如果没有批准请求，则 **oc exec**、oc**rsh** 和 **oc logs** 命令将无法成功，因为 API 服务器连接到 kubelet 时需要服务证书。与 Kubelet 端点联系的任何操作都需要此证书批准。该方法必须监视新的 CSR，确认 CSR 由 system: **node** 或 **system:admin** 组中的 **node-bootstrapper** 服务帐户提交，并确认节点的身份。

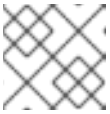
- 要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```



注意

在有些 CSR 被批准前，一些 Operator 可能无法使用。

4. 现在，您的客户端请求已被批准，您必须查看添加到集群中的每台机器的服务器请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE     REQUESTOR                                                   CONDITION
csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

5. 如果剩余的 CSR 没有被批准，且处于 **Pending** 状态，请批准集群机器的 CSR：

- 要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs oc adm certificate approve
```

6. 批准所有客户端和服务器 CSR 后，机器将 处于 **Ready 状态**。运行以下命令验证：

```
$ oc get nodes
```

输出示例

```
NAME      STATUS    ROLES   AGE  VERSION
master-0  Ready     master  73m  v1.29.4
master-1  Ready     master  73m  v1.29.4
master-2  Ready     master  74m  v1.29.4
worker-0  Ready     worker  11m  v1.29.4
worker-1  Ready     worker  11m  v1.29.4
```

注意

批准服务器 CSR 后可能需要几分钟时间让机器过渡到 **Ready 状态**。

其他信息

- 如需有关 CSR 的更多信息，请参阅 证书签名请求。

9.12.22. 可选：添加入口 DNS 记录

如果在创建 Kubernetes 清单并生成 Ignition 配置时删除了 DNS 区配置，您必须手动创建指向入口负载均衡器的 DNS 记录。您可以创建一个通配符 ***.apps.{baseDomain}.** 或特定的记录。您可以根据要求使用 A、CNAME 和其他记录。

先决条件

- 配置 GCP 帐户。
- 在创建 Kubernetes 清单并生成 Ignition 配置时，删除 DNS 区配置。
- 在 GCP 中创建和配置 VPC 及相关子网。
- 在 GCP 中创建和配置联网与负载均衡器。
- 创建 control plane 和计算角色。
- 创建 bootstrap 机器。
- 创建 control plane 机器。
- 创建 worker 机器。

流程

1. 等待入口路由器创建负载均衡器并填充 **EXTERNAL-IP** 字段：

```
$ oc -n openshift-ingress get service router-default
```

输出示例

```
NAME            TYPE           CLUSTER-IP     EXTERNAL-IP     PORT(S)                      AGE
router-default  LoadBalancer  172.30.18.154  35.233.157.184
80:32288/TCP,443:31215/TCP   98
```

2. 在您的区中添加 A 记录：
   - 使用 A 记录：
     i. 导出路由器 IP 地址的变量：

```
$ export ROUTER_IP=`oc -n openshift-ingress get service router-default --no-
headers | awk '{print $4}'`
```

     ii. 在私有区中添加 A 记录：

```
$ if [ -f transaction.yaml ]; then rm transaction.yaml; fi
$ gcloud dns record-sets transaction start --zone ${INFRA_ID}-private-zone
$ gcloud dns record-sets transaction add ${ROUTER_IP} --name
```

```
\*.apps.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 300 --type A --zone
${INFRA_ID}-private-zone
$ gcloud dns record-sets transaction execute --zone ${INFRA_ID}-private-zone
```

iii. 对于外部集群，还要在公共区中添加 A 记录：

```
$ if [ -f transaction.yaml ]; then rm transaction.yaml; fi
$ gcloud dns record-sets transaction start --zone ${BASE_DOMAIN_ZONE_NAME}
$ gcloud dns record-sets transaction add ${ROUTER_IP} --name
\*.apps.${CLUSTER_NAME}.${BASE_DOMAIN}. --ttl 300 --type A --zone
${BASE_DOMAIN_ZONE_NAME}
$ gcloud dns record-sets transaction execute --zone
${BASE_DOMAIN_ZONE_NAME}
```

- 要添加特定域而不使用通配符，请为集群的每个当前路由创建条目：

```
$ oc get --all-namespaces -o jsonpath='{range .items[*]}{range .status.ingress[*]}{.host}
{"\n"}{end}{end}' routes
```

输出示例

```
oauth-openshift.apps.your.cluster.domain.example.com
console-openshift-console.apps.your.cluster.domain.example.com
downloads-openshift-console.apps.your.cluster.domain.example.com
alertmanager-main-openshift-monitoring.apps.your.cluster.domain.example.com
prometheus-k8s-openshift-monitoring.apps.your.cluster.domain.example.com
```

9.12.23. 在用户置备的基础架构上完成 GCP 安装

在 Google Cloud Platform(GCP)用户置备的基础架构上启动 OpenShift Container Platform 安装后，您可以监控集群事件，直到集群就绪。

先决条件

- 在用户置备的 GCP 基础架构上为 OpenShift Container Platform 集群部署 bootstrap 机器。
- 安装 **oc** CLI 并登录。

流程

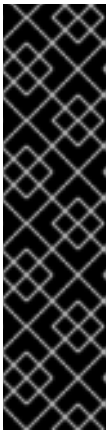
1. 完成集群安装：

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

输出示例

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。



重要

- 安装程序生成的 Ignition 配置文件包含 24 小时后过期的证书，然后在该时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

2. 观察集群的运行状态。

   a. 运行以下命令查看当前的集群版本和状态：

   ```
   $ oc get clusterversion
   ```

   输出示例

   ```
   NAME     VERSION   AVAILABLE   PROGRESSING   SINCE   STATUS
   version            False       True          24m     Working towards 4.5.4: 99% complete
   ```

   b. 运行以下命令，查看 control plane 上由 Cluster Version Operator(CVO)管理的 Operator：

   ```
   $ oc get clusteroperators
   ```

   输出示例

   ```
   NAME                                       VERSION   AVAILABLE   PROGRESSING   DEGRADED
   SINCE
   authentication                             4.5.4     True        False         False      7m56s
   cloud-credential                           4.5.4     True        False         False      31m
   cluster-autoscaler                         4.5.4     True        False         False      16m
   console                                    4.5.4     True        False         False      10m
   csi-snapshot-controller                    4.5.4     True        False         False      16m
   dns                                        4.5.4     True        False         False      22m
   etcd                                       4.5.4     False       False         False      25s
   image-registry                             4.5.4     True        False         False      16m
   ingress                                    4.5.4     True        False         False      16m
   insights                                   4.5.4     True        False         False      17m
   kube-apiserver                             4.5.4     True        False         False      19m
   kube-controller-manager                    4.5.4     True        False         False      20m
   kube-scheduler                             4.5.4     True        False         False      20m
   kube-storage-version-migrator              4.5.4     True        False         False      16m
   machine-api                                4.5.4     True        False         False      22m
   machine-config                             4.5.4     True        False         False      22m
   marketplace                                4.5.4     True        False         False      16m
   monitoring                                 4.5.4     True        False         False      10m
   network                                    4.5.4     True        False         False      23m
   node-tuning                                4.5.4     True        False         False      23m
   openshift-apiserver                        4.5.4     True        False         False      17m
   openshift-controller-manager               4.5.4     True        False         False      15m
   ```

```
openshift-samples                          4.5.4     True        False         False      16m
operator-lifecycle-manager                 4.5.4     True        False         False      22m
operator-lifecycle-manager-catalog         4.5.4     True        False         False      22m
operator-lifecycle-manager-packageserver   4.5.4     True        False         False      18m
service-ca                                 4.5.4     True        False         False      23m
service-catalog-apiserver                  4.5.4     True        False         False      23m
service-catalog-controller-manager         4.5.4     True        False         False      23m
storage                                    4.5.4     True        False         False      17m
```

c. 运行以下命令来查看您的集群 pod：

```
$ oc get pods --all-namespaces
```

输出示例

```
NAMESPACE                                          NAME
READY     STATUS      RESTARTS   AGE
kube-system                                        etcd-member-ip-10-0-3-111.us-east-
2.compute.internal              1/1       Running     0          35m
kube-system                                        etcd-member-ip-10-0-3-239.us-east-
2.compute.internal              1/1       Running     0          37m
kube-system                                        etcd-member-ip-10-0-3-24.us-east-
2.compute.internal              1/1       Running     0          35m
openshift-apiserver-operator                       openshift-apiserver-operator-6d6674f4f4-
h7t2t                           1/1       Running     1          37m
openshift-apiserver                                apiserver-fm48r
1/1       Running     0          30m
openshift-apiserver                                apiserver-fxkvv
1/1       Running     0          29m
openshift-apiserver                                apiserver-q85nm
1/1       Running     0          29m
...
openshift-service-ca-operator                      openshift-service-ca-operator-66ff6dc6cd-
9r257                           1/1       Running     0          37m
openshift-service-ca                               apiservice-cabundle-injector-695b6bcbc-cl5hm
1/1       Running     0          35m
openshift-service-ca                               configmap-cabundle-injector-8498544d7-
25qn6                           1/1       Running     0          35m
openshift-service-ca                               service-serving-cert-signer-6445fc9c6-wqdqn
1/1       Running     0          35m
openshift-service-catalog-apiserver-operator       openshift-service-catalog-apiserver-
operator-549f44668b-b5q2w       1/1       Running     0          32m
openshift-service-catalog-controller-manager-operator   openshift-service-catalog-
controller-manager-operator-b78cr2lnm     1/1       Running     0          31m
```

当当前集群版本是 **AVAILABLE** 时，安装已完成。

9.12.24. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于 远程健康监控

9.12.25. 后续步骤

- 自定义集群。
- 为 Cluster Samples Operator 和 **must-gather** 工具 配置镜像流。
- 了解如何在 受限网络中使用 Operator Lifecycle Manager(OLM)。
- 如果您用来安装集群的镜像 registry 具有可信任的 CA，请通过 配置额外的信任存储将其添加到集群中。
- 如果需要，您可以选择 不使用远程健康报告。
- 如果需要，请参阅 注册断开连接的集群

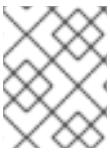
9.13. 在 GCP 上安装三节点集群

在 OpenShift Container Platform 版本 4.16 中，您可以在 Google Cloud Platform (GCP) 上安装三节点集群。三节点集群包含三个 control plane 机器，它们也可以充当计算机器。这种类型的集群提供了一个较小的、效率更高的集群，供集群管理员和开发人员用于测试、开发和生产。

您可以使用安装程序置备或用户置备的基础架构安装三节点集群。

9.13.1. 配置三节点集群

在部署集群前，您可以通过将 **install-config.yaml** 文件中的 worker 节点数量设置为 **0** 来配置三节点集群。将 worker 节点数量设置为 **0** 可确保 control plane 机器可以调度。这允许调度应用程序工作负载从 control plane 节点运行。



注意

因为应用程序工作负载从 control plane 节点运行，所以需要额外的订阅，因为 control plane 节点被视为计算节点。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。

流程

1. 将 **install-config.yaml** 文件中的计算副本数量设置为 **0**，如以下 **compute** 小节中所示：

   三节点集群的 install-config.yaml 文件示例

   ```
   apiVersion: v1
   baseDomain: example.com
   ```

```
compute:
- name: worker
  platform: {}
  replicas: 0
# ...
```

2. 如果您使用用户置备的基础架构部署集群：

   - 创建 Kubernetes 清单文件后，请确保在 **cluster-scheduler-02-config.yml** 文件中将 **spec.mastersSchedulable** 参数设置为 **true**。您可以在 **<installation_directory>/manifests** 中找到此文件。如需更多信息，请参阅使用 Deployment Manager 模板在 GCP 中的"在用户置备的基础架构上安装集群"中的"创建 Kubernetes 清单和 Ignition 配置文件"。

   - 不要创建额外的 worker 节点。

三节点集群的 cluster-scheduler-02-config.yml 文件示例

```
apiVersion: config.openshift.io/v1
kind: Scheduler
metadata:
  creationTimestamp: null
  name: cluster
spec:
  mastersSchedulable: true
  policy:
    name: ""
status: {}
```

9.13.2. 后续步骤

- 使用自定义在 GCP 上安装集群
- 使用 Deployment Manager 模板在 GCP 中的用户置备的基础架构上安装集群

9.14. GCP 的安装配置参数

在 Google Cloud Platform (GCP)上部署 OpenShift Container Platform 集群前，您可以提供参数来自定义集群及其托管平台。在创建 **install-config.yaml** 文件时，您可以通过命令行为所需参数提供值。然后，您可以修改 **install-config.yaml** 文件以进一步自定义集群。

9.14.1. GCP 可用的安装配置参数

下表指定在安装过程中设置所需的、可选和特定于 GCP 的安装配置参数。

注意

安装后，您无法在 **install-config.yaml** 文件中修改这些参数。

9.14.1.1. 所需的配置参数

下表描述了所需的安装配置参数：

表 9.48. 所需的参数

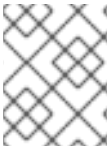
参数	描述	值
apiVersion:	**install-config.yaml** 内容的 API 版本。当前版本为 **v1**。安装程序可能还支持旧的 API 版本。	字符串
baseDomain:	云供应商的基域。基域用于创建到 OpenShift Container Platform 集群组件的路由。集群的完整 DNS 名称是 **baseDomain** 和 **metadata.name** 参数值的组合，其格式为 **<metadata.name>.<baseDomain>**。	完全限定域名或子域名，如 **example.com**。
metadata:	Kubernetes 资源 **ObjectMeta**，其中只消耗 **name** 参数。	对象
metadata: name:	集群的名称。集群的 DNS 记录是 **{{.metadata.name}}.{{.baseDomain}}** 的子域。	小写字母、连字符(**-**)和句点(**.**)字符串，如 **dev**。
platform:	对于特定平台的配置取决于执行安装的环境： **aws**, **baremetal**, **azure**, **gcp**, **ibmcloud**, **nutanix**, **openstack**, **powervs**, **vsphere**, 或 **{}**.有关 **platform.<platform>** 参数的更多信息，请参考下表中您的特定平台。	对象

pullSecret:	从 Red Hat OpenShift Cluster Manager 获取 pull secret，验证从 Quay.io 等服务中下载 OpenShift Container Platform 组件的容器镜像。	<pre>{
		  "auths":{
		  "cloud.openshift.com":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  },
		  "quay.io":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  }
		  }
		}</pre>

9.14.1.2. 网络配置参数

您可以根据现有网络基础架构的要求自定义安装配置。例如，您可以扩展集群网络的 IP 地址块，或者提供不同于默认值的不同 IP 地址块。

仅支持 IPv4 地址。



注意

Red Hat OpenShift Data Foundation 灾难恢复解决方案不支持 Globalnet。对于区域灾难恢复场景，请确保为每个集群中的集群和服务网络使用非重叠的专用 IP 地址。

表 9.49. 网络参数

参数	描述	值
networking:	集群网络的配置。	对象 **注意** 您无法在安装后修改 **网络** 对象指定的参数。
networking: networkType:	要安装的 Red Hat OpenShift Networking 网络插件。	**OVNKubernetes**。**OVNKubernetes** 是 Linux 网络和包含 Linux 和 Windows 服务器的混合网络的 CNI 插件。默认值为 **OVNKubernetes**。
networking: clusterNetwork:	pod 的 IP 地址块。 默认值为 **10.128.0.0/14**，主机前缀为 **/23**。 如果您指定了多个 IP 地址块，块不得重叠。	对象数组。例如： networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23
networking: clusterNetwork: cidr:	使用 **networking.clusterNetwork** 时需要此项。IP 地址块。 IPv4 网络。	无类别域间路由(CIDR)表示法中的 IP 地址块。IPv4 块的前缀长度介于 **0 到 32** 之间。
networking: clusterNetwork: hostPrefix:	分配给每个节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从 given **cidr** 中分配 a/**23** 子网。**hostPrefix** 值 **23** 提供 510（2^(32 - 23)- 2）pod IP 地址。	子网前缀。 默认值为 **23**。

参数	描述	值
networking: serviceNetwork:	服务的 IP 地址块。默认值为 **172.30.0.0/16**。 OVN-Kubernetes 网络插件只支持服务网络的一个 IP 地址块。	CIDR 格式具有 IP 地址块的数组。例如： networking: serviceNetwork: - 172.30.0.0/16
networking: machineNetwork:	机器的 IP 地址块。 如果您指定了多个 IP 地址块，块不得重叠。	对象数组。例如： networking: machineNetwork: - cidr: 10.0.0.0/16
networking: machineNetwork: cidr:	使用 **networking.machineNetwork** 时需要此项。IP 地址块。对于 libvirt 和 IBM Power® Virtual Server 以外的所有平台，默认值为 **10.0.0.0/16**。对于 libvirt，默认值 为 **192.168.126.0/24**。对于 IBM Power® Virtual Server，默认值为 **192.168.0.0/24**。	CIDR 表示法中的 IP 网络块。 例如： **10.0.0.0/16**。 **注意** 将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

9.14.1.3. 可选的配置参数

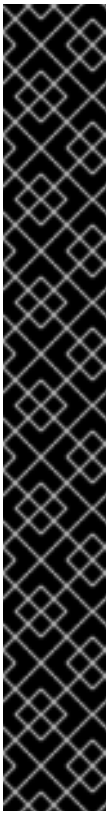
下表描述了可选的安装配置参数：

表 9.50. 可选参数

参数	描述	值
additionalTrustBundle:	添加到节点可信证书存储中的 PEM 编码 X.509 证书捆绑包。配置了代理时，也可以使用此信任捆绑包。	字符串
capabilities:	控制可选核心组件的安装。您可以通过禁用可选组件来减少 OpenShift Container Platform 集群的空间。如需更多信息，请参阅 *安装*中的"集群功能"页面。	字符串数组

参数	描述	值
capabilities: baselineCapabilitySet:	选择要启用的一组初始可选功能。有效值为 **None**、**v4.11**、**v4.12** 和 **vCurrent**。默认值为 **vCurrent**。	字符串
capabilities: additionalEnabledCapabilities:	将可选功能集合扩展到您在 **baselineCapabilitySet** 中指定的范围。您可以在此参数中指定多个功能。	字符串数组
cpuPartitioningMode:	启用工作负载分区，它会隔离 OpenShift Container Platform 服务、集群管理工作负载和基础架构 pod，以便在保留的一组 CPU 上运行。工作负载分区只能在安装过程中启用，且在安装后无法禁用。虽然此字段启用工作负载分区，但它不会将工作负载配置为使用特定的 CPU。如需更多信息，请参阅 *Scalability and Performance* 部分中的 *Workload partitioning* 页面。	**None** 或 **AllNodes**.**None** 是默认值。
compute:	组成计算节点的机器的配置。	**MachinePool** 对象的数组。
compute: architecture:	决定池中机器的指令集合架构。目前，不支持具有不同架构的集群。所有池都必须指定相同的架构。有效值为 **amd64** 和 **arm64**。	字符串
compute: hyperthreading:	是否在计算机器上启用或禁用并发多**线程或超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。	**enabled** 或 **Disabled**
compute: name:	使用 **compute** 时需要此项。机器池的名称。	**worker**

参数	描述	值
compute: platform:	使用 **compute** 时需要此项。使用此参数指定托管 worker 机器的云供应商。此参数值必须与 **controlPlane.platform** 参数值匹配。	**aws**, **azure**, **gcp**, **ibmcloud**, **nutanix**, **openstack**, **powervs**, **vsphere**, 或 **{}**
compute: replicas:	要置备的计算机器数量，也称为 worker 机器。	大于或等于 **2** 的正整数。默认值为 **3**。
featureSet:	为功能集启用集群。功能集是 OpenShift Container Platform 功能的集合，默认情况下不启用。有关在安装过程中启用功能集的更多信息，请参阅"使用功能门启用功能"。	字符串.要启用的功能集的名称，如 **TechPreviewNoUpgrade**。
controlPlane:	组成 control plane 的机器的配置。	**MachinePool** 对象的数组。
controlPlane: architecture:	决定池中机器的指令集合架构。目前，不支持具有不同架构的集群。所有池都必须指定相同的架构。有效值为 **amd64** 和 **arm64**。	字符串
controlPlane: hyperthreading:	是否在 control plane 机器上启用或禁用并发多 **线程或超线程**。默认情况下，启用并发多线程以提高机器内核的性能。 **重要** 如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。	**enabled** 或 **Disabled**
controlPlane: name:	使用 **controlPlane** 时需要此项。机器池的名称。	**master**
controlPlane: platform:	使用 **controlPlane** 时需要此项。使用此参数指定托管 control plane 机器的云供应商。此参数值必须与 **compute.platform** 参数值匹配。	**aws**, **azure**, **gcp**, **ibmcloud**, **nutanix**, **openstack**, **powervs**, **vsphere**, 或 **{}**

参数	描述	值
controlPlane: replicas:	要置备的 control plane 机器数量。	部署单节点 OpenShift 时支持的值为 **3** 或 **1**。
credentialsMode:	Cloud Credential Operator(CCO)模式。如果没有指定模式，CCO 会动态尝试决定提供的凭证的功能，在支持多个模式的平台上首选 mint 模式。	**Mint**、**Passthrough**、**Manual** 或空字符串(**""**)。[1]
fips:	启用或禁用 FIPS 模式。默认值为 **false**（禁用）。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。  **重要** 要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 Red Hat Enterprise Linux (RHEL) 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅在 FIPS 模式中安装该系统。 当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS) 时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。  **注意** 如果使用 Azure File 存储，则无法启用 FIPS 模式。	**false** 或 **true**
imageContentSources:	release-image 内容的源和存储库。	对象数组。包括一个 **source** 以及可选的 **mirrors**，如本表的以下行所述。

参数	描述	值
imageContentSources: source:	使用 **imageContentSources** 时需要此项。指定用户在镜像拉取规格中引用的存储库。	字符串
imageContentSources: mirrors:	指定可能还包含同一镜像的一个或多个仓库。	字符串数组
publish:	如何发布或公开集群的面向用户的端点，如 Kubernetes API、OpenShift 路由。	**内部或外部 .**要部署无法从互联网访问的私有集群，请将 **publish** 设置为 **Internal**。默认值为 **External**。
sshKey:	用于验证对集群机器的访问的 SSH 密钥。 **注意** 对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。	例如，**sshKey: ssh-ed25519 AAAA..**。

1. 不是所有 CCO 模式都支持所有云供应商。有关 CCO 模式的更多信息，请参阅*身份验证和授权*内容中的"管理云供应商凭证"条目。

注意

如果要在 GCP 上安装到共享虚拟私有云 (VPC) 中，则 **credentialsMode** 必须设置为 **Passthrough** 或 **Manual**。

重要

将此参数设置为 **Manual** 可启用在 **kube-system** 项目中存储管理员级别的 secret 的替代方案，这需要额外的配置步骤。如需更多信息，请参阅"在 kube-system 项目中存储管理员级别的 secret"。

9.14.1.4. 其他 Google Cloud Platform(GCP)配置参数

下表描述了其他 GCP 配置参数：

表 9.51. 其他 GCP 参数

参数	描述	值
controlPlane: platform: gcp: osImage: project:	可选。默认情况下，安装程序会下载并安装用于引导 control plane 机器的 Red Hat Enterprise Linux CoreOS (RHCOS)镜像。您可以通过指定安装程序仅用于 control plane 机器的自定义 RHCOS 镜像的位置来覆盖默认行为。	字符串.镜像所在 GCP 项目的名称。
controlPlane: platform: gcp: osImage: name:	安装程序用来引导 control plane 机器的自定义 RHCOS 镜像的名称。如果使用 **controlPlane.platform.gcp.osImage.project**，则需要此字段。	字符串.RHCOS 镜像的名称。

参数	描述	值
compute: platform: gcp: osImage: project:	可选。默认情况下，安装程序会下载并安装用于引导计算机器的 RHCOS 镜像。您可以通过指定安装程序仅用于计算机器的自定义 RHCOS 镜像的位置来覆盖默认行为。	字符串.镜像所在 GCP 项目的名称。
compute: platform: gcp: osImage: name:	安装程序用来引导计算机器的自定义 RHCOS 镜像的名称。如果使用 **compute.platform.gcp.osImage.project**，则需要此字段。	字符串.RHCOS 镜像的名称。
platform: gcp: network:	要部署集群的现有 Virtual Private Cloud (VPC) 的名称。如果要将集群部署到共享 VPC 中，则必须使用包含共享 VPC 的 GCP 项目的名称设置 **platform.gcp.networkProjectID**。	字符串.

参数	描述	值
platform: gcp: networkProjectID:	可选。包含要部署集群的共享 VPC 的 GCP 项目的名称。	字符串.
platform: gcp: projectID:	安装程序安装集群的 GCP 项目的名称。	字符串.
platform: gcp: region:	托管集群的 GCP 区域的名称。	任何有效的区域名称，如 **us-central1**。

参数	描述	值
platform: gcp: controlPlaneSubnet:	要部署 control plane 机器的现有子网的名称。	子网名称。
platform: gcp: computeSubnet:	要部署计算机器的现有子网的名称。	子网名称。

参数	描述	值
platform: gcp: defaultMachinePlatform: zones:	安装程序在其中创建机器的可用区。	有效 GCP 可用区 列表，如 **us-central1-a**，在一个 YAML 序列 中。 **重要** 在 GCP 64 位 ARM 基础架构上运行集群时，请确保使用 Ampere Altra Arm CPU 可用的区域。您可以在 "GCP Availability zones" 链接中找到哪些区与 64 位 ARM 处理器兼容。
platform: gcp: defaultMachinePlatform: osDisk: diskSizeGB:	以 GB(GB)为单位的磁盘大小。	16 GB 到 65536 GB 之间的任何大小。

参数	描述	值
platform: gcp: defaultMachinePlatform: osDisk: diskType:	GCP 磁盘类型。	所有机器的默认磁盘类型。control plane 节点必须使用 **pd-ssd** 磁盘类型。Compute 节点可以使用 **pd-ssd**、**pd-balanced** 或 **pd-standard** 磁盘类型。
platform: gcp: defaultMachinePlatform: osImage: project:	可选。默认情况下，安装程序会下载并安装用于引导 control plane 和计算机器的 RHCOS 镜像。您可以通过为安装程序为两种类型的机器指定自定义 RHCOS 镜像的位置来覆盖默认行为。	字符串.镜像所在 GCP 项目的名称。

参数	描述	值
platform: gcp: defaultMachinePlatform: osImage: name:	安装程序用来引导 control plane 和计算机器的自定义 RHCOS 镜像的名称。如果使用 **platform.gcp.defaultMachinePlatform.osImage.project**，则需要此字段。	字符串.RHCOS 镜像的名称。
platform: gcp: defaultMachinePlatform: tags:	可选。要添加到 control plane 和计算机器的额外网络标签。	一个或多个字符串，如 **network-tag1**。

参数	描述	值
platform: gcp: defaultMachinePlatform: type:	control plane 和计算机器的 GCP 机器类型。	GCP 机器类型，如 **n1-standard-4**。

参数	描述	值
platform: gcp: defaultMachinePlatform: osDisk: encryptionKey: kmsKey: name:	用于机器磁盘加密的客户管理的加密密钥名称。	加密密钥名称。

参数	描述	值
platform: gcp: defaultMachinePlatform: osDisk: encryptionKey: kmsKey: keyRing:	KMS 密钥所属的密钥管理服务 (KMS) 密钥环的名称。	KMS 密钥环名称。

参数	描述	值
platform: gcp: defaultMachinePlatform: osDisk: encryptionKey: kmsKey: location:	KMS 密钥环存在的 GCP 位置。	GCP 位置。

参数	描述	值
platform: gcp: defaultMachinePlatform: osDisk: encryptionKey: kmsKey: projectID:	存在 KMS 密钥环的项目 ID。如果没有设置，则默认值为 **platform.gcp.projectID** 参数的值。	GCP 项目 ID。

参数	描述	值
platform: gcp: defaultMachinePlatform: osDisk: encryptionKey: kmsKeyServiceAccount:	用于 control plane 和计算机器加密请求的 GCP 服务帐户。如果没有，则使用 Compute Engine 默认服务帐户。如需有关 GCP 服务帐户的更多信息，请参阅 Google 文档中的服务帐户。	GCP 服务帐户电子邮件，如 **<service_account_name>@<project_id>.iam.gserviceaccount.com**。

参数	描述	值
platform: gcp: defaultMachinePlatform: secureBoot:	是否为集群中的所有机器启用 Shielded VM 安全引导。Shielded 虚拟机具有额外的安全协议，如安全引导、固件和完整性监控和 rootkit 保护。有关 Shielded 虚拟机的更多信息，请参阅 Google 文档中有关 Shielded 虚拟机 的文档。	**Enabled** 或 **Disabled**。默认值为 **Disabled**。
platform: gcp: defaultMachinePlatform: confidentialCompute:	是否对集群中的所有机器使用机密虚拟机。机密虚拟机在处理过程中为数据提供加密。有关机密计算的更多信息，请参阅 Google 文档中有关机密计算的内容。	**Enabled** 或 **Disabled**。默认值为 **Disabled**。

参数	描述	值
platform: gcp: defaultMachinePlatform: onHostMaintenance:	指定主机维护事件期间所有虚拟机的行为，如软件或硬件更新。对于机密虚拟机，此参数必须设置为 **Terminate**。机密虚拟机不支持实时迁移。	**Terminate** 或 **Migrate**.默认值为 **Migrate**。

参数	描述	值
controlPlane: platform: gcp: osDisk: encryptionKey: kmsKey: name:	用于 control plane 机器磁盘加密的客户管理的加密密钥名称。	加密密钥名称。

参数	描述	值
controlPlane: platform: gcp: osDisk: encryptionKey: kmsKey: keyRing:	对于 control plane 机器，KMS 密钥环的名称。	KMS 密钥环名称。

参数	描述	值
controlPlane: platform: gcp: osDisk: encryptionKey: kmsKey: location:	对于 control plane 机器，存在密钥环的 GCP 位置。有关 KMS 位置的更多信息，请参阅 Google 文档中的Cloud KMS 位置。	密钥环的 GCP 位置。

参数	描述	值
controlPlane: platform: gcp: osDisk: encryptionKey: kmsKey: projectID:	对于 control plane 机器，存在 KMS 密钥环的项目 ID。如果没有设置，则默认值是 VM 项目 ID。	GCP 项目 ID。

参数	描述	值
controlPlane: platform: gcp: osDisk: encryptionKey: kmsKeyServiceAccount:	用于 control plane 机器的加密请求的 GCP 服务帐户。如果没有，则使用 Compute Engine 默认服务帐户。如需有关 GCP 服务帐户的更多信息，请参阅 Google 文档中的服务帐户。	GCP 服务帐户电子邮件，如 **<service_account_name>@<project_id>.iam.gserviceaccount.com**。

参数	描述	值
controlPlane: platform: gcp: osDisk: diskSizeGB:	以 GB(GB)为单位的磁盘大小。这个值适用于 control plane 机器。	16 到 65536 之间的任何整数。
controlPlane: platform: gcp: osDisk: diskType:	control plane 机器的 GCP 磁盘类型。	control plane 机器必须使用 **pd-ssd** 磁盘类型，这是默认设置。

参数	描述	值
controlPlane: platform: gcp: tags:	可选。要添加到 control plane 机器的额外网络标签。如果设置，此参数会覆盖 control plane 机器的 **platform.gcp.defaultMachinePlatform.tags** 参数。	一个或多个字符串，如 **control-plane-tag1**。
controlPlane: platform: gcp: type:	control plane 机器的 GCP 机器类型。如果设置，此参数会覆盖 **platform.gcp.defaultMachinePlatform.type** 参数。	GCP 机器类型，如 **n1-standard-4**。
controlPlane: platform: gcp: zones:	安装程序在其中创建 control plane 机器的可用区。	有效 GCP 可用区 列表，如 **us-central1-a**，在一个 YAML 序列 中。  **重要** 在 GCP 64 位 ARM 基础架构上运行集群时，请确保使用 Ampere Altra Arm CPU 可用的区域。您可以在 "GCP Availability zones" 链接中找到哪些区与 64 位 ARM 处理器兼容。

参数	描述	值
controlPlane: platform: gcp: secureBoot:	是否为 control plane 机器启用 Shielded VM 安全引导。Shielded 虚拟机具有额外的安全协议，如安全引导、固件和完整性监控和 rootkit 保护。有关 Shielded 虚拟机的更多信息，请参阅 Google 文档中有关 Shielded 虚拟机 的文档。	**Enabled** 或 **Disabled**。默认值为 **Disabled**。

参数	描述	值
controlPlane: platform: gcp: confidentialCompute:	是否为 control plane 机器启用机密虚拟机。机密虚拟机在处理数据时为数据提供加密。有关机密虚拟机的更多信息，请参阅 Google 文档中有关机密计算的内容。	**Enabled** 或 **Disabled**。默认值为 **Disabled**。
controlPlane: platform: gcp: onHostMaintenance:	指定在主机维护事件期间 control plane 虚拟机的行为，如软件或硬件更新。对于机密虚拟机，此参数必须设置为 **Terminate**。机密虚拟机不支持实时迁移。	**Terminate** 或 **Migrate**.默认值为 **Migrate**。

参数	描述	值
compute: platform: gcp: osDisk: encryptionKey: kmsKey: name:	用于计算机器磁盘加密的客户管理的加密密钥名称。	加密密钥名称。

参数	描述	值
compute: platform: gcp: osDisk: encryptionKey: kmsKey: keyRing:	对于计算机器，KMS 密钥环的名称。	KMS 密钥环名称。

参数	描述	值
compute: platform: gcp: osDisk: encryptionKey: kmsKey: location:	对于计算机器，存在密钥环的 GCP 位置。有关 KMS 位置的更多信息，请参阅 Google 文档中的Cloud KMS 位置。	密钥环的 GCP 位置。

参数	描述	值
compute: platform: gcp: osDisk: encryptionKey: kmsKey: projectID:	对于计算机器，存在 KMS 密钥环的项目 ID。如果没有设置，则默认值是 VM 项目 ID。	GCP 项目 ID。

参数	描述	值
compute: platform: gcp: osDisk: encryptionKey: kmsKeyServiceAccount:	用于计算机器加密请求的 GCP 服务帐户。如果没有设置这个值，则使用 Compute Engine 默认服务帐户。如需有关 GCP 服务帐户的更多信息，请参阅 Google 文档中的服务帐户。	GCP 服务帐户电子邮件，如 **<service_account_name>@<project_id>.iam.gserviceaccount.com**。

参数	描述	值
compute: platform: gcp: osDisk: diskSizeGB:	以 GB(GB)为单位的磁盘大小。这个值适用于计算机器。	16 到 65536 之间的任何整数。
compute: platform: gcp: osDisk: diskType:	计算机器的 GCP 磁盘类型。	**pd-ssd**、**pd-standard** 或 **pd-balanced**。默认为 **pd-ssd**。

参数	描述	值
compute: platform: gcp: tags:	可选。要添加到计算机器的额外网络标签。如果设置，此参数会覆盖计算机器的 **platform.gcp.defaultMachinePlatform.tags** 参数。	一个或多个字符串，如 **compute-network-tag1**。
compute: platform: gcp: type:	计算机器的 GCP 机器类型。如果设置，此参数会覆盖 **platform.gcp.defaultMachinePlatform.type** 参数。	GCP 机器类型，如 **n1-standard-4**。
compute: platform: gcp: zones:	安装程序在其中创建计算机器的可用区。	有效 GCP 可用区 列表，如 **us-central1-a**，在一个 YAML 序列 中。 **重要** 在 GCP 64 位 ARM 基础架构上运行集群时，请确保使用 Ampere Altra Arm CPU 可用的区域。您可以在 "GCP Availability zones" 链接中找到哪些区与 64 位 ARM 处理器兼容。

参数	描述	值
compute: platform: gcp: secureBoot:	是否为计算机器启用 Shielded VM 安全引导。Shielded 虚拟机具有额外的安全协议，如安全引导、固件和完整性监控和 rootkit 保护。有关 Shielded 虚拟机的更多信息，请参阅 Google 文档中有关 Shielded 虚拟机 的文档。	**Enabled** 或 **Disabled**。默认值为 **Disabled**。
compute: platform: gcp: confidentialCompute:	是否为计算机器启用机密虚拟机。机密虚拟机在处理数据时为数据提供加密。有关机密虚拟机的更多信息，请参阅 Google 文档中有关机密计算的内容。	**Enabled** 或 **Disabled**。默认值为 **Disabled**。

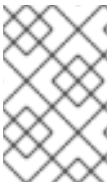
参数	描述	值
compute: platform: gcp: onHostMaintenance:	指定在主机维护事件期间计算虚拟机的行为，如软件或硬件更新。对于机密虚拟机，此参数必须设置为 **Terminate**。机密虚拟机不支持实时迁移。	**Terminate** 或 **Migrate**.默认值为 **Migrate**。

9.15. 在 GCP 上卸载集群

您可以删除部署到 Google Cloud Platform(GCP)的集群。

9.15.1. 删除使用安装程序置备的基础架构的集群

您可以从云中删除使用安装程序置备的基础架构的集群。



注意

卸载后，检查云供应商是否有未正确删除的资源，特别是在用户置备基础架构(UPI)集群中。可能存在安装程序未创建或安装程序无法访问的资源。例如，一些 Google Cloud 资源需要在共享 VPC 主机项目中具有 IAM 权限，或者可能有未使用的 健康检查必须被删除。

先决条件

- 有用于部署集群的安装程序副本。
- 有创建集群时安装程序生成的文件。

流程

1. 在用来安装集群的计算机中包含安装程序的目录中，运行以下命令：

```
$ ./openshift-install destroy cluster \
--dir <installation_directory> --log-level info 1 2
```

1. 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。
2. 要查看不同的详情，请指定 **warn**、**debug** 或 **error**，而不是 **info**。

注意

您必须为集群指定包含集群定义文件的目录。安装程序需要此目录中的 **metadata.json** 文件来删除集群。

2. 可选：删除 **<installation_directory>** 目录和 OpenShift Container Platform 安装程序。

9.15.2. 使用 Cloud Credential Operator 实用程序删除 Google Cloud Platform 资源

卸载使用在集群外管理的短期凭证的 OpenShift Container Platform 集群后，您可以使用 CCO 实用程序 (**ccoctl**)删除 **ccoctl** 在安装过程中创建的 Google Cloud Platform (GCP)资源。

先决条件

- 提取并准备 **ccoctl** 二进制文件。
- 卸载使用短期凭证的 GCP 上的 OpenShift Container Platform 集群。

流程

1. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 自定义资源 (CR) 列表：

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
  --included \ 1
  --to=<path_to_directory_for_credentials_requests> 2
```

1. **--included** 参数仅包含特定集群配置所需的清单。
2. 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

3. 运行以下命令，删除 **ccoctl** 创建的 GCP 资源：

```
$ ccoctl gcp delete \
  --name=<name> \ 1
  --project=<gcp_project_id> \ 2
  --credentials-requests-dir=<path_to_credentials_requests_directory> \
  --force-delete-custom-roles 3
```

1. **<name>** 与最初用于创建和标记云资源的名称匹配。
2. **<gcp_project_id>** 是要在其中删除云资源的 GCP 项目 ID。
3. 可选：此参数删除 **ccoctl** 工具在安装过程中创建的自定义角色。GCP 不会立即永久删除自定义角色。如需更多信息，请参阅GCP 文档中有关删除自定义角色的部分。

验证

- 要验证资源是否已被删除，请查询 GCP。如需更多信息，请参阅 GCP 文档。

第 10 章 在 IBM CLOUD 上安装

10.1. 准备在 IBM CLOUD 上安装

本节中介绍的安装工作流用于 IBM Cloud® 基础架构环境。目前不支持 IBM Cloud® Classic。有关 Classic 和 VPC 基础架构之间的区别的更多信息，请参阅 IBM® 文档。

10.1.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读选择集群安装方法并为用户准备它的文档。

10.1.2. 在 IBM Cloud 上安装 OpenShift Container Platform 的要求

在 IBM Cloud® 上安装 OpenShift Container Platform 前，您必须创建一个服务帐户并配置 IBM Cloud® 帐户。有关创建帐户、启用 API 服务、配置 DNS、IBM Cloud® 帐户限值和支持的 IBM Cloud® 区域的详细信息，请参阅配置 IBM Cloud® 帐户 。

在将集群安装到 IBM Cloud® 时，您必须手动管理云凭证。在安装集群前，请为手动模式配置 Cloud Credential Operator (CCO)如需更多信息，请参阅为 IBM Cloud® 配置 IAM 。

10.1.3. 选择在 IBM Cloud 上安装 OpenShift Container Platform 的方法

您可以使用安装程序置备的基础架构在 IBM Cloud® 上安装 OpenShift Container Platform。此过程涉及使用安装程序为集群置备底层基础架构。目前，不支持使用用户置备的基础架构在 IBM Cloud® 上安装 OpenShift Container Platform。

有关安装程序置备安装过程的更多信息，请参阅安装过程。

10.1.3.1. 在安装程序置备的基础架构上安装集群

您可以使用以下方法之一在 OpenShift Container Platform 安装程序置备的 IBM Cloud® 基础架构上安装集群：

- **在 IBM Cloud® 上安装自定义集群**：您可以在安装程序置备的 IBM Cloud® 基础架构上安装自定义集群。安装程序允许在安装阶段应用一些自定义。其它自定义选项可在安装后使用。
- **使用自定义网络在 IBM Cloud® 上安装集群** ：您可以在安装过程中自定义 OpenShift Container Platform 网络配置，以便集群可以与现有 IP 地址分配共存并遵循您的网络要求。
- **在 IBM Cloud® 上安装集群到现有的 VPC 中**：您可以在现有的 IBM Cloud® 上安装 OpenShift Container Platform。如果您按照公司的说明设置了限制，可以使用这个安装方法，例如在创建新帐户或基础架构时的限制。
- **在现有 VPC 上安装私有集群** ：您可以在现有 Virtual Private Cloud (VPC) 上安装私有集群。您可以使用此方法在互联网不可见的内部网络中部署 OpenShift Container Platform。
- **在受限网络中的 IBM Cloud VPC 上安装集群**：您可以使用安装发行内容的内部镜像在 IBM Cloud VPC 上安装 OpenShift Container Platform。您可以使用此方法安装不需要活跃互联网连接的集群来获取软件组件。

10.1.4. 后续步骤

- 配置 IBM Cloud® 帐户

10.2. 配置 IBM CLOUD 帐户

在安装 OpenShift Container Platform 之前，您必须配置 IBM Cloud® 帐户。

10.2.1. 先决条件

- 您有一个带有订阅的 IBM Cloud® 帐户。您不能在免费或试用 IBM Cloud® 帐户上安装 OpenShift Container Platform。

10.2.2. IBM Cloud 的配额和限值

OpenShift Container Platform 集群使用多个 IBM Cloud® 组件，默认配额和限值会影响您安装 OpenShift Container Platform 集群的能力。如果您使用特定的集群配置，在某些区域部署集群，或者从您的帐户运行多个集群，您可能需要为 IBM Cloud® 帐户请求其他资源。

有关默认 IBM Cloud® 配额和服务限制的完整列表，请参阅 IBM Cloud® 的配额和服务限制 文档。

虚拟私有云(VPC)
每个 OpenShift Container Platform 集群都会创建自己的 VPC。每个区域的默认 VPC 配额是 10，并将允许 10 个集群。在单个区域中有 10 个集群，您必须增加此配额。

应用程序负载均衡器
默认情况下，每个集群创建三个应用程序负载均衡器(ALBs)：

- master API 服务器的内部负载均衡器
- 主 API 服务器的外部负载均衡器
- 路由器的负载均衡器

您可以创建额外的 **LoadBalancer** 服务对象创建额外的 ALBs。VPC 的默认配额是每个区域 50 个。要获得超过 50 个 ALB，您必须提高此配额。

支持 VPC ALB。IBM Cloud® 不支持经典的 ALB。

浮动 IP 地址
默认情况下，安装程序会在区域中的所有可用区间分发 control plane 和计算机器，以便在高可用性配置中置备集群。在每个可用区中，创建一个公共网关，并需要一个单独的浮动 IP 地址。

浮动 IP 地址的默认配额是每个可用区 20 个地址。默认集群配置生成三个浮动 IP 地址：

- **us-east-1** 主区域中的两个浮动 IP 地址。安装后会删除与 bootstrap 节点关联的 IP 地址。
- **us-east-2** secondary 区域中的一个浮动 IP 地址。
- **us-east-3** secondary 区域中的一个浮动 IP 地址。

IBM Cloud® 可以为每个帐户中的每个区域支持最多 19 个集群。如果计划有超过 19 个默认集群，您必须提高此配额。

虚拟服务器实例(VSI)
默认情况下，集群使用 **bx2-4x16** 配置集创建 VSIs，默认包括以下资源：

- 4 个 vCPU

- 16 GB RAM

创建以下节点：

- 一个 **bx2-4x16** bootstrap 机器，它会在安装完成后删除
- 三个 **bx2-4x16** control plane 节点
- 三个 **bx2-4x16** 计算节点

如需更多信息，请参阅 IBM Cloud® 有关支持的配置集 的文档。

表 10.1. VSI 组件配额和限值

VSI 组件	默认 IBM Cloud® 配额	默认集群配置	集群的最大数量
vCPU	每个区域 200 个 vCPU	28 个 vCPU，或 bootstrap 被删除后 24 个 vCPU	每个区域 8 个
RAM	每个区域 1600 GB	在移除后 112 GB 或 96 GB	每个区域 16 个
Storage	每个区域 18 TB	移除后 1050 GB 或 900 GB	每个区域 19 个

如果您计划超过表中声明的资源，您必须提高 IBM Cloud® 帐户配额。

块存储卷
对于每个 VPC 机器，会为其引导卷附加一个块存储设备。默认集群配置创建七 VPC 机器，生成 7 个块存储卷。IBM Cloud® 存储类的额外 Kubernetes 持久性卷声明(PVC)会创建额外的块存储卷。VPC 块存储卷的默认配额是每个区域 300。要获得超过 300 个卷，您必须提高此配额。

10.2.3. 配置 DNS 解析

如何配置 DNS 解析取决于您安装的 OpenShift Container Platform 集群的类型：

- 如果要安装公共集群，请使用 IBM Cloud Internet Services (CIS)。
- 如果要安装私有集群，请使用 IBM Cloud® DNS Services (DNS Services)

10.2.3.1. 使用 IBM Cloud Internet 服务进行 DNS 解析

安装程序使用 IBM Cloud® Internet Services (CIS) 来配置集群 DNS 解析，并为公共集群提供名称查找。

注意

此产品不支持 IPv6，因此无法实现双堆栈或 IPv6 环境。

您必须在与集群相同的帐户的 CIS 中创建域区。您还必须确保该区域对域具有权威。您可以使用根域或子域进行此操作。

先决条件

- 已安装 IBM Cloud® CLI。

- 您有一个现有的域和注册商。如需更多信息，请参阅 IBM® 文档。

流程

1. 创建用于集群的 CIS 实例：
   a. 安装 CIS 插件：

```
$ ibmcloud plugin install cis
```

   b. 创建 CIS 实例：

```
$ ibmcloud cis instance-create <instance_name> standard 1
```

   1 CIS 至少需要一个 **标准** 计划来管理集群子域及其 DNS 记录。

2. 将现有域连接到您的 CIS 实例：
   a. 为 CIS 设置上下文实例：

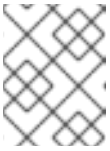
```
$ ibmcloud cis instance-set <instance_name> 1
```

   1 实例云资源名称。

   b. 为 CIS 添加域：

```
$ ibmcloud cis domain-add <domain_name> 1
```

   1 完全限定域名。您可以根据计划配置，使用根域或子域值作为域名。



注意

根域使用格式 **openshiftcorp.com**。子域使用格式为 **cluster.openshiftcorp.com**。

3. 打开 CIS Web 控制台，进入 **Overview** 页面，并记录您的 CIS 名称服务器。这些名称服务器将在下一步中使用。
4. 在域的注册商或 DNS 供应商中为您的域或子域配置名称服务器。如需更多信息，请参阅 IBM Cloud® 文档。

10.2.3.2. 使用 IBM Cloud DNS 服务进行 DNS 解析

安装程序使用 IBM Cloud® DNS 服务来配置集群 DNS 解析，并为私有集群提供名称查找。

您可以通过为集群创建 DNS 服务实例来配置 DNS 解析，然后将 DNS 区添加到 DNS Services 实例。确保该区域对域具有权威。您可以使用根域或子域进行此操作。

注意

IBM Cloud® 不支持 IPv6，因此无法进行双栈或 IPv6 环境。

先决条件

- 已安装 IBM Cloud® CLI。
- 您有一个现有的域和注册商。如需更多信息，请参阅 IBM® 文档。

流程

1. 创建用于集群的 DNS Services 实例：
   a. 运行以下命令来安装 DNS 服务插件：

```
$ ibmcloud plugin install cloud-dns-services
```

   b. 运行以下命令来创建 DNS Services 实例：

```
$ ibmcloud dns instance-create <instance-name> standard-dns 1
```

   1 DNS 服务至少需要 **Standard** 计划来管理集群子域及其 DNS 记录。

2. 为 DNS Services 实例创建 DNS 区域：
   a. 运行以下命令设置目标操作 DNS Services 实例：

```
$ ibmcloud dns instance-target <instance-name>
```

   b. 运行以下命令，将 DNS 区域添加到 DNS Services 实例：

```
$ ibmcloud dns zone-create <zone-name> 1
```

   1 完全限定区域名称。您可以根据计划配置，使用根域或子域值作为区域名称。根域使用格式 **openshiftcorp.com**。子域使用格式为 **cluster.openshiftcorp.com**。

3. 记录您创建的 DNS 区域的名称。作为安装过程的一部分，您必须在部署集群前更新 **install-config.yaml** 文件。使用 DNS 区域的名称作为 **baseDomain** 参数的值。

注意

您不必管理允许的网络或配置"A"DNS 资源记录。根据需要，安装程序会自动配置这些资源。

10.2.4. IBM Cloud IAM 策略和 API 密钥

要将 OpenShift Container Platform 安装到 IBM Cloud® 帐户中，安装程序需要一个 IAM API 密钥，它提供访问 IBM Cloud® 服务 API 的身份验证和授权。您可以使用包含所需策略的现有 IAM API 密钥或创建新策略。

有关 IBM Cloud® IAM 概述，请参阅 IBM Cloud® 文档。

10.2.4.1. 所需的访问策略

您必须为 IBM Cloud® 帐户分配所需的访问策略。

表 10.2. 所需的访问策略

服务类型	service	访问策略范围	平台访问	服务访问
帐户管理	IAM Identity Service	所有资源或资源子集 [1]	Editor, Operator, Viewer, Administrator	服务 ID 创建者
帐户管理 [2]	身份和访问权限管理	所有资源	Editor, Operator, Viewer, Administrator	
帐户管理	仅限资源组	帐户中的所有资源组	Administrator	
IAM 服务	云对象存储	所有资源或资源子集 [1]	Editor, Operator, Viewer, Administrator	Reader, Writer, Manager, Content Reader, Object Reader, Object Writer
IAM 服务	Internet 服务	所有资源或资源子集 [1]	Editor, Operator, Viewer, Administrator	Reader、Writer、Manager
IAM 服务	DNS 服务	所有资源或资源子集 [1]	Editor, Operator, Viewer, Administrator	Reader、Writer、Manager
IAM 服务	VPC 基础架构服务	所有资源或资源子集 [1]	Editor, Operator, Viewer, Administrator	Reader、Writer、Manager

1. 应根据您要分配访问权限的粒度来设置策略访问范围。范围可以设置为 **All resources** 或 **基于所选属性的资源**。
2. 可选：只有安装程序创建资源组时才需要此访问策略。有关资源组的更多信息，请参阅 IBM® 文档。

10.2.4.2. 访问策略分配

在 IBM Cloud® IAM 中，可以将访问策略附加到不同的主题：

- 访问组（推荐）
- 服务 ID
- 用户

建议的方法是在访问组中定义 IAM 访问策略。这有助于组织 OpenShift Container Platform 所需的所有访问权限，并可让您向这个组注册用户和服务 ID。如果需要，您还可以为 用户和服务 ID 分配访问权限。

10.2.4.3. 创建 API 密钥

您必须为 IBM Cloud® 帐户创建用户 API 密钥或服务 ID API 密钥。

先决条件

- 您已为 IBM Cloud® 帐户分配了所需的访问策略。
- 您已将 IAM 访问策略附加到访问组或其他适当的资源。

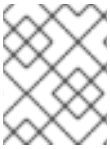
流程

- 根据您定义的 IAM 访问策略，创建一个 API 密钥。
  例如，如果您为用户分配了访问策略，您必须创建一个 用户 API 密钥。如果您将访问策略分配给服务 ID，您必须创建一个服务 ID API 密钥。如果您的访问策略分配给一个访问组，您可以使用任一 API 密钥类型。有关 IBM Cloud® API 密钥的更多信息，请参阅了解 API 密钥。

10.2.5. 支持的 IBM Cloud 区域

您可以将 OpenShift Container Platform 集群部署到以下区域：

- **au-syd** (Sydney, Australia)
- **br-sao** (Sao Paulo, Brazil)
- **ca-tor** (Toronto, Canada)
- **eu-de** (Frankfurt, Germany)
- **eu-gb** (London, United Kingdom)
- **eu-es** (Madrid, Spain)
- **jp-osa** (Osaka, Japan)
- **jp-tok** (Tokyo, Japan)
- **us-east** (Washington DC, United States)
- **us-south** (Dallas, United States)



注意

OpenShift Container Platform 4.14.6 及更早的版本中不支持在 **eu-es** (Madrid, Spain)区域中部署集群。

10.2.6. 后续步骤

- 为 IBM Cloud® 配置 IAM。

10.3. 为 IBM CLOUD 配置 IAM

在无法访问云身份和访问管理 (IAM) API 的环境中，您必须在安装集群前将 Cloud Credential Operator (CCO) 置于手动模式。

10.3.1. 在 kube-system 项目中存储管理员级别的 secret 的替代方案

Cloud Credential Operator(CCO)将云供应商凭证作为 Kubernetes 自定义资源定义(CRD)进行管理。您可以通过在 **install-config.yaml** 文件中为 **credentialsMode** 参数设置不同的值，将 CCO 配置为满足机构的安全要求。

IBM Cloud® 不支持在集群 **kube-system** 项目中存储管理员级别的凭证 secret；因此，您必须在安装 OpenShift Container Platform 时将 CCO 的 **credentialsMode** 参数设置为 **Manual**，并手动管理云凭证。

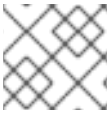
使用手动模式可允许每个集群组件只拥有所需的权限，而无需在集群中存储管理员级别的凭证。如果您的环境没有连接到云供应商公共 IAM 端点，您还可以使用此模式。但是，每次升级都必须手动将权限与新发行镜像协调。您还必须手动为每个请求它们的组件提供凭证。

其他资源

- 关于 Cloud Credential Operator

10.3.2. 配置 Cloud Credential Operator 工具

当 Cloud Credential Operator(CCO)以手动模式运行时，要从集群外部创建和管理云凭证，提取并准备 CCO 实用程序(**ccoctl**)二进制文件。



注意

ccoctl 工具是在 Linux 环境中运行的 Linux 二进制文件。

先决条件

- 您可以访问具有集群管理员权限的 OpenShift Container Platform 帐户。
- 已安装 OpenShift CLI(**oc**)。

流程

1. 运行以下命令，为 OpenShift Container Platform 发行镜像设置变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. 运行以下命令，从 OpenShift Container Platform 发行镜像获取 CCO 容器镜像：

```
$ CCO_IMAGE=$(oc adm release info --image-for='cloud-credential-operator'
$RELEASE_IMAGE -a ~/.pull-secret)
```

注意

确保 **$RELEASE_IMAGE** 的架构与将使用 **ccoctl** 工具的环境架构相匹配。

3. 运行以下命令，将 CCO 容器镜像中的 **ccoctl** 二进制文件提取到 OpenShift Container Platform 发行镜像中：

```
$ oc image extract $CCO_IMAGE \
  --file="/usr/bin/ccoctl.<rhel_version>" \ 1
  -a ~/.pull-secret
```

1 对于 **<rhel_version>**，请指定与主机使用的 Red Hat Enterprise Linux (RHEL) 版本对应的值。如果没有指定值，则默认使用 **ccoctl.rhel8**。以下值有效：

- **rhel8**: 为使用 RHEL 8 的主机指定这个值。
- **rhel9** ：为使用 RHEL 9 的主机指定这个值。

4. 运行以下命令更改权限以使 **ccoctl** 可执行：

```
$ chmod 775 ccoctl.<rhel_version>
```

验证

- 要验证 **ccoctl** 是否可使用，请运行以下命令来显示帮助文件：

```
$ ccoctl --help
```

ccoctl --help 的输出

```
OpenShift credentials provisioning tool

Usage:
  ccoctl [command]

Available Commands:
  aws         Manage credentials objects for AWS cloud
  azure       Manage credentials objects for Azure
  gcp         Manage credentials objects for Google cloud
  help        Help about any command
  ibmcloud    Manage credentials objects for IBM Cloud
  nutanix     Manage credentials objects for Nutanix

Flags:
  -h, --help   help for ccoctl

Use "ccoctl [command] --help" for more information about a command.
```

其他资源

- 为 IBM Cloud® 轮转 API 密钥。

10.3.3. 后续步骤

- 使用自定义在 IBM Cloud® 上安装集群

10.3.4. 其他资源

- 准备使用手动维护的凭证更新集群

10.4. 用户管理的 IBM CLOUD 加密

默认情况下，在部署 OpenShift Container Platform 集群时，使用供应商管理的加密来保护以下内容：

- control plane 和计算机器的根（boot）卷
- 部署集群后置备的持久性卷（data 卷）

在安装过程中，您可以指定一个 IBM® Key Protect for IBM Cloud® (Key Protect) root 覆盖默认行为。

如果使用我们自己的 root 密钥时，您可以使用 **encryptionKey** 参数修改安装配置文件 (**install-config.yaml**) 以指定 root 密钥的 Cloud Resource Name (CRN)。

您可以指定：

- 所有集群机器都使用相同的 root 密钥。要实现这一点，将密钥指定为集群默认机器配置的一部分。
  当作为默认机器配置的一部分指定时，所有受管存储类都会使用此密钥更新。因此，安装后置备的数据卷也会使用此密钥加密。
- 单独的 root 密钥用于 control plane 和计算机器池。

有关 **encryptionKey** 参数的更多信息，请参阅 其他 IBM Cloud 配置参数。

注意

确保您已有与您的 IBM Cloud Block Storage 服务集成的 Key Protect。如需更多信息，请参阅 Key Protect 文档。

10.4.1. 后续步骤

安装 OpenShift Container Platform 集群：

- 使用自定义在 IBM Cloud 上安装集群
- 使用网络自定义在 IBM Cloud 上安装集群
- 在 IBM Cloud 上安装集群到现有的 VPC 中
- 在 IBM Cloud 上安装私有集群

10.5. 使用自定义在 IBM CLOUD 上安装集群

在 OpenShift Container Platform 版本 4.16 中，您可以在安装程序在 IBM Cloud® 上置备的基础架构上安装自定义的集群。要自定义安装，请在安装集群前修改 **install-config.yaml** 文件中的参数。

10.5.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读有关 选择集群安装方法的文档，并为用户准备它。
- 已将 IBM Cloud® 帐户配置为托管集群。
- 如果使用防火墙，则会 将其配置为允许集群需要访问的站点。
- 在安装集群前已经配置了 **ccoctl** 工具。如需更多信息，请参阅为 IBM Cloud® 配置 IAM。

10.5.2. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类型的基础架构上执行受限网络安装。在此过程中，您可以下载所需的内容，并使用它为镜像 registry 填充安装软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群前，您要更新镜像 registry 的内容。

10.5.3. 为集群节点 SSH 访问生成密钥对

在 OpenShift Container Platform 安装过程中，您可以为安装程序提供 SSH 公钥。密钥通过它们的 Ignition 配置文件传递给 Red Hat Enterprise Linux CoreOS(RHCOS)节点，用于验证对节点的 SSH 访问。密钥添加到每个节点上 **core** 用户的 **~/.ssh/authorized_keys** 列表中，这将启用免密码身份验证。

将密钥传递给节点后，您可以使用密钥对作为用户 **核心** 通过 SSH 连接到 RHCOS 节点。若要通过 SSH 访问节点，必须由 SSH 为您的本地用户管理私钥身份。

如果要通过 SSH 连接到集群节点来执行安装调试或灾难恢复，则必须在安装过程中提供 SSH 公钥。**./openshift-install gather** 命令还需要在集群节点上设置 SSH 公钥。

重要

不要在生产环境中跳过这个过程，在生产环境中需要灾难恢复和调试。

注意

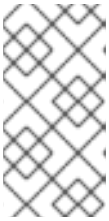
您必须使用本地密钥，而不是使用特定平台方法配置 的密钥，如 AWS 密钥对。

流程

1. 如果您在本地计算机上没有可用于在集群节点上进行身份验证的现有 SSH 密钥对，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name>
```
(1)

(1) 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_ed25519**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。



注意

如果您计划在 **x86_64**、**ppc64le** 和 **s390x** 架构上安装使用 RHEL 加密库（这些加密库已提交给 NIST 用于 FIPS 140-2/140-3 验证）的 OpenShift Container Platform 集群，则不要创建使用 **ed25519** 算法的密钥。相反，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

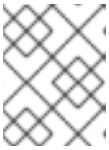
2. 查看公共 SSH 密钥：

```
$ cat <path>/<file_name>.pub
```

例如，运行以下命令来查看 **~/.ssh/id_ed25519.pub** 公钥：

```
$ cat ~/.ssh/id_ed25519.pub
```

3. 将 SSH 私钥身份添加到本地用户的 SSH 代理（如果尚未添加）。在集群节点上，或者要使用 **./openshift-install gather** 命令，需要对该密钥进行 SSH 代理管理，才能在集群节点上进行免密码 SSH 身份验证。



注意

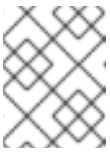
在某些发行版中，自动管理默认 SSH 私钥身份，如 **~/.ssh/id_rsa** 和 **~/.ssh/id_dsa**。

a. 如果 **ssh-agent** 进程尚未为您的本地用户运行，请将其作为后台任务启动：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```



注意

如果集群处于 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

4. 将 SSH 私钥添加到 **ssh-agent** ：

```
$ ssh-add <path>/<file_name> (1)
```

(1) 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_ed25519.pub**

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

后续步骤

- 安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

10.5.4. 获取安装程序

在安装 OpenShift Container Platform 前，将安装文件下载到您用于安装的主机上。

先决条件

- 您有一台运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB。

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐户，请使用您的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入到安装类型的页面，下载与您的主机操作系统和架构对应的安装程序，并将该文件放在您要存储安装配置文件的目录中。

重要

安装程序会在用来安装集群的计算机上创建几个文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。

重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，请为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。此 pull secret 允许您与所含授权机构提供的服务进行身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

10.5.5. 导出 API 密钥

您必须将您创建的 API 密钥设置为全局变量；安装程序会在启动期间设置 API 密钥。

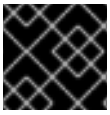
先决条件

- 您已为 IBM Cloud® 帐户创建了用户 API 密钥或服务 ID API 密钥。

流程

- 将帐户的 API 密钥导出为全局变量：

```
$ export IC_API_KEY=<api_key>
```



重要

您必须按照指定方式设置变量名称，安装程序需要在启动期间存在变量名称。

10.5.6. 创建安装配置文件

您可以自定义在 IBM Cloud® 上安装的 OpenShift Container Platform 集群。

先决条件

- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。

流程

1. 创建 **install-config.yaml** 文件。
   a. 进入包含安装程序的目录并运行以下命令：

   ```
   $ ./openshift-install create install-config --dir <installation_directory> 1
   ```

   1 对于 **<installation_directory>**，请指定要存储安装程序创建的文件的目录名称。

   在指定目录时：

   - 验证该目录是否具有**执行**权限。在安装目录中运行 Terraform 二进制文件需要这个权限。
   - 使用空目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

   b. 在提示符处，提供云的配置详情：
      i. 可选： 选择用于访问集群机器的 SSH 密钥。

      注意

      对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

      ii. 选择 **ibmcloud** 作为目标平台。
      iii. 选择要将集群部署到的区域。
      iv. 选择集群要部署到的基域。基域与您为集群创建的公共 DNS 区对应。
      v. 为集群输入描述性名称。
2. 修改 **install-config.yaml** 文件。您可以在"安装配置参数"部分找到有关可用参数的更多信息。
3. 备份 **install-config.yaml** 文件，以便您可以使用它安装多个集群。

重要

install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用此文件，必须现在备份。

其他资源

- IBM Cloud® 的安装配置参数。

10.5.6.1. 集群安装的最低资源要求

每台集群机器都必须满足以下最低要求：

表 10.3. 最低资源要求

机器	操作系统	vCPU	虚拟内存	Storage	每秒输入/输出 (IOPS)
bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane（控制平面）	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS	2	8 GB	100 GB	300

注意

从 OpenShift Container Platform 版本 4.13 开始，RHCOS 基于 RHEL 版本 9.2，它更新了微架构要求。以下列表包含每个架构需要的最小指令集架构 (ISA)：

- x86-64 体系结构需要 x86-64-v2 ISA
- ARM64 架构需要 ARMv8.0-A ISA
- IBM Power 架构需要 Power 9 ISA
- s390x 架构需要 z14 ISA

如需更多信息，请参阅 RHEL 架构。

如果平台的实例类型满足集群机器的最低要求，则 OpenShift Container Platform 支持使用它。

其他资源

- 优化存储

10.5.6.2. 为 IBM Cloud 测试的实例类型

以下 IBM Cloud® 实例类型已使用 OpenShift Container Platform 测试。

例 10.1. 机器系列

- **c4.***
- **c5.***
- **c5a.***
- **i3.***
- **m4.***
- **m5.***
- **m5a.***
- **m6a.***
- **m6i.***
- **r4.***
- **r5.***
- **r5a.***
- **r6i.***
- **t3.***
- **t3a.***

10.5.6.3. IBM Cloud 的自定义 install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多详情，或修改所需参数的值。

重要

此示例 YAML 文件仅供参考。您必须使用安装程序来获取 **install-config.yaml** 文件，然后修改该文件。

```
apiVersion: v1
baseDomain: example.com (1)
controlPlane: (2) (3)
  hyperthreading: Enabled (4)
  name: master
  platform:
    ibmcloud: {}
  replicas: 3
compute: (5) (6)
- hyperthreading: Enabled (7)
  name: worker
  platform:
    ibmcloud: {}
  replicas: 3
```

```
metadata:
  name: test-cluster 8
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 9
  serviceNetwork:
  - 172.30.0.0/16
platform:
  ibmcloud:
    region: us-south 10
credentialsMode: Manual
publish: External
pullSecret: '{"auths": ...}' 11
fips: false 12
sshKey: ssh-ed25519 AAAA... 13
```

1 8 10 11 必需。安装程序会提示您输入这个值。

2 5 如果没有提供这些参数和值，安装程序会提供默认值。

3 6 **controlPlane** 部分是一个单个映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，controlPlane **部分** 的第一行则不以连字符开头。仅使用一个 control plane 池。

4 7 启用或禁用并发多线程，也称为 Hyper-Threading。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将 参数值设置为 **Disabled** 来禁用它。如果在某些集群机器中禁用并发多线程，则必须在所有集群机器中禁用它。

重要

如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。如果您禁用并发多线程，请为您的机器使用较大的类型，如 **n1-standard-8**。

9 要安装的集群网络插件。默认值 **OVNKubernetes** 是唯一支持的值。

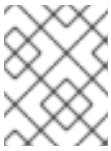
12 启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 Red Hat Enterprise Linux (RHEL) 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅在 FIPS 模式中安装该系统。

当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS)时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。

13 可选：提供用于访问集群中机器的 **sshKey** 值。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

10.5.6.4. 在安装过程中配置集群范围的代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 **Proxy** 对象的 **spec.noProxy** 字段中添加站点来绕过代理。

注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，**Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 **.** 以仅匹配子域。例如，**.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代理。
4. 如果提供，安装程序会在 **openshift-config** 命名空间中生成名为 **user-ca-bundle** 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network

射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 **trusted-ca-bundle** 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， **Proxy** 对象的 **trustedCA** 字段中也会引用此配置映射。**additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

5 可选：决定 **Proxy** 对象的配置以引用 **trustedCA** 字段中 **user-ca-bundle** 配置映射的策略。允许的值是 **Proxyonly** 和 **Always**。仅在配置了 **http/https** 代理时，使用 **Proxyonly** 引用 **user-ca-bundle** 配置映射。使用 **Always** 始终引用 **user-ca-bundle** 配置映射。默认值为 **Proxyonly**。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。

注意

如果安装程序超时，重启并使用安装程序的 **wait-for** 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. 保存该文件并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 cluster 的集群范围代理，该代理 **使用** 提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

10.5.7. 手动创建 IAM

安装集群需要 Cloud Credential Operator(CCO)以手动模式运行。虽然安装程序为手动模式配置 CCO，但您必须为云供应商指定身份和访问管理 secret。

您可以使用 Cloud Credential Operator (CCO)实用程序(**ccoctl**)创建所需的 IBM Cloud® 资源。

先决条件

- 您已配置了 **ccoctl** 二进制文件。
- 您有一个现有的 **install-config.yaml** 文件。

流程

1. 编辑 **install-config.yaml** 配置文件，使其包含将 **credentialsMode** 参数设置为 **Manual**。

   install-config.yaml 配置文件示例

```
apiVersion: v1
baseDomain: cluster1.example.com
credentialsMode: Manual 1
```

```
compute:
- architecture: amd64
  hyperthreading: Enabled
```

1 添加这一行将 **credentialsMode** 参数设置为 **Manual**。

2. 要生成清单，请在包含安装程序的目录中运行以下命令：

```
$ openshift-install create manifests --dir <installation_directory>
```

3. 在包含安装程序的目录中，运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

4. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 自定义资源 (CR) 列表：

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
  --included \ 1
  --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
  --to=<path_to_directory_for_credentials_requests> 3
```

1 **--included** 参数仅包含特定集群配置所需的清单。

2 指定 **install-config.yaml** 文件的位置。

3 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

此命令为每个 **CredentialsRequest** 对象创建一个 YAML 文件。

CredentialsRequest 对象示例

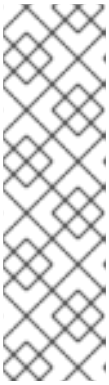
```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  labels:
    controller-tools.k8s.io: "1.0"
  name: openshift-image-registry-ibmcos
  namespace: openshift-cloud-credential-operator
spec:
  secretRef:
    name: installer-cloud-credentials
    namespace: openshift-image-registry
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: IBMCloudProviderSpec
    policies:
    - attributes:
      - name: serviceName
```

```
    value: cloud-object-storage
  roles:
  - crn:v1:bluemix:public:iam::::role:Viewer
  - crn:v1:bluemix:public:iam::::role:Operator
  - crn:v1:bluemix:public:iam::::role:Editor
  - crn:v1:bluemix:public:iam::::serviceRole:Reader
  - crn:v1:bluemix:public:iam::::serviceRole:Writer
- attributes:
  - name: resourceType
    value: resource-group
  roles:
  - crn:v1:bluemix:public:iam::::role:Viewer
```

5. 为每个凭证请求创建服务 ID，分配定义的策略、创建 API 密钥并生成 secret：

```
$ ccoctl ibmcloud create-service-id \
  --credentials-requests-dir=<path_to_credential_requests_directory> \ 1
  --name=<cluster_name> \ 2
  --output-dir=<installation_directory> \ 3
  --resource-group-name=<resource_group_name> 4
```

1. 指定包含组件 **CredentialsRequest** 对象文件的目录。
2. 指定 OpenShift Container Platform 集群的名称。
3. 可选：指定您希望 **ccoctl** 实用程序在其中创建对象的目录。默认情况下，实用程序在运行命令的目录中创建对象。
4. 可选：指定用于限制访问策略的资源组的名称。



注意

如果您的集群使用 **TechPreviewNoUpgrade** 功能集启用的技术预览功能，则必须包含 **--enable-tech-preview** 参数。

如果提供了不正确的资源组名称，安装会在 bootstrap 阶段失败。要查找正确的资源组名称，请运行以下命令：

```
$ grep resourceGroupName <installation_directory>/manifests/cluster-
infrastructure-02-config.yml
```

验证

- 确保在集群的 **manifests** 目录中生成了适当的 secret。

10.5.8. 部署集群

您可以在兼容云平台上安装 OpenShift Container Platform。

重要

在初始安装过程中，您只能运行安装程序的 **create cluster** 命令一次。

先决条件

- 您已使用托管集群的云平台配置了帐户。
- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。
- 已确认主机上的云供应商帐户具有部署集群的正确权限。权限不正确的帐户会导致安装过程失败，并显示包括缺失权限的错误消息。

流程

- 进入包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1. 对于 **<installation_directory>**，请指定自定义 **./install-config.yaml** 文件的位置。
2. 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不是 **info**。

验证

当集群部署成功完成时：

- 终端会显示用于访问集群的说明，包括指向 Web 控制台和 **kubeadmin** 用户的凭证的链接。
- 凭证信息还会输出到 **<installation_directory>/.openshift_install.log**.

重要

不要删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

输出示例

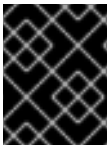
```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

10.5.9. 安装 OpenShift CLI

您可以安装 OpenShift CLI(**oc**)来使用命令行界面与 OpenShift Container Platform 进行交互。您可以在 Linux、Windows 或 macOS 上安装 **oc**。



重要

如果安装了旧版本的 **oc**，则可能无法使用 OpenShift Container Platform 4.16 中的所有命令。下载并安装新版本的 **oc**。

在 Linux 上安装 OpenShift CLI
您可以按照以下流程在 Linux 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **产品变体** 下拉列表中选择架构。
3. 从 **版本** 下拉列表中选择适当的版本。
4. 点 **OpenShift v4.16 Linux Client** 条目旁的 **Download Now** 来保存文件。
5. 解包存档：

```
$ tar xvf <file>
```

6. 将 **oc** 二进制文件放到 **PATH 中的目录中**。
要查看您的 **PATH**，请执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

在 Windows 上安装 OpenShift CLI
您可以按照以下流程在 Windows 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 Windows Client** 条目旁的 **Download Now** 来保存文件。
4. 使用 ZIP 程序解压存档。
5. 将 **oc** 二进制文件移到 **PATH 中的目录中**。
要查看您的 **PATH**，请打开命令提示并执行以下命令：

```
C:\> path
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

在 macOS 上安装 OpenShift CLI
您可以按照以下流程在 macOS 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 macOS Client** 条目旁的 **Download Now** 来保存文件。

注意

对于 macOS arm64，请选择 **OpenShift v4.16 macOS arm64 Client** 条目。

4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
要查看您的 **PATH**，请打开终端并执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

10.5.10. 使用 CLI 登录集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

其他资源

- 访问Web控制台

10.5.11. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 关于远程健康监控

10.5.12. 后续步骤

- 自定义集群。
- 如果需要，您可以选择 不使用远程健康报告。

10.6. 使用网络自定义在 IBM CLOUD 上安装集群

在 OpenShift Container Platform 版本 4.16 中，您可以使用自定义的网络配置在安装程序在 IBM Cloud® 上置备的基础架构上安装集群。通过自定义网络配置，您的集群可以与环境中现有的 IP 地址分配共存，并与现有的 MTU 和 VXLAN 配置集成。要自定义安装，请在安装集群前修改 **install-config.yaml** 文件中的参数。

您必须在安装过程中设置大多数网络配置参数，且您只能在正在运行的集群中修改 **kubeProxy** 配置参数。

10.6.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读有关 选择集群安装方法的文档，并为用户准备它。

- 已将 IBM Cloud® 帐户配置为托管集群。
- 如果使用防火墙，则会 将其配置为允许集群需要访问的站点。
- 在安装集群前已经配置了 **ccoctl** 工具。如需更多信息，请参阅为 IBM Cloud® 配置 IAM。

10.6.2. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类型的基础架构上执行受限网络安装。在此过程中，您可以下载所需的内容，并使用它为镜像 registry 填充安装软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群前，您要更新镜像 registry 的内容。

10.6.3. 为集群节点 SSH 访问生成密钥对

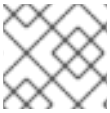
在 OpenShift Container Platform 安装过程中，您可以为安装程序提供 SSH 公钥。密钥通过它们的 Ignition 配置文件传递给 Red Hat Enterprise Linux CoreOS(RHCOS)节点，用于验证对节点的 SSH 访问。密钥添加到每个节点上 **core** 用户的 **~/.ssh/authorized_keys** 列表中，这将启用免密码身份验证。

将密钥传递给节点后，您可以使用密钥对作为用户 **核心** 通过 SSH 连接到 RHCOS 节点。若要通过 SSH 访问节点，必须由 SSH 为您的本地用户管理私钥身份。

如果要通过 SSH 连接到集群节点来执行安装调试或灾难恢复，则必须在安装过程中提供 SSH 公钥。**./openshift-install gather** 命令还需要在集群节点上设置 SSH 公钥。

重要

不要在生产环境中跳过这个过程，在生产环境中需要灾难恢复和调试。



注意

您必须使用本地密钥，而不是使用特定平台方法配置 的密钥，如 AWS 密钥对。

流程

1. 如果您在本地计算机上没有可用于在集群节点上进行身份验证的现有 SSH 密钥对，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_ed25519**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

注意

如果您计划在 **x86_64**、**ppc64le** 和 **s390x** 架构上安装使用 RHEL 加密库（这些加密库已提交给 NIST 用于 FIPS 140-2/140-3 验证）的 OpenShift Container Platform 集群，则不要创建使用 **ed25519** 算法的密钥。相反，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

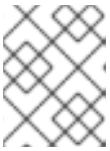
2. 查看公共 SSH 密钥：

```
$ cat <path>/<file_name>.pub
```

例如，运行以下命令来查看 **~/.ssh/id_ed25519.pub** 公钥：

```
$ cat ~/.ssh/id_ed25519.pub
```

3. 将 SSH 私钥身份添加到本地用户的 SSH 代理（如果尚未添加）。在集群节点上，或者要使用 **./openshift-install gather** 命令，需要对该密钥进行 SSH 代理管理，才能在集群节点上进行免密码 SSH 身份验证。



注意

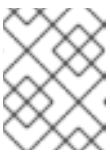
在某些发行版中，自动管理默认 SSH 私钥身份，如 **~/.ssh/id_rsa** 和 **~/.ssh/id_dsa**。

a. 如果 **ssh-agent** 进程尚未为您的本地用户运行，请将其作为后台任务启动：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```



注意

如果集群处于 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

4. 将 SSH 私钥添加到 **ssh-agent** ：

```
$ ssh-add <path>/<file_name> 1
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_ed25519.pub**

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

后续步骤

- 安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

10.6.4. 获取安装程序

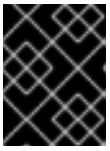
在安装 OpenShift Container Platform 前，将安装文件下载到您用于安装的主机上。

先决条件

- 您有一台运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB。

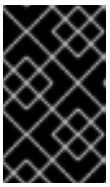
流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐户，请使用您的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入到安装类型的页面，下载与您的主机操作系统和架构对应的安装程序，并将该文件放在您要存储安装配置文件的目录中。



重要

安装程序会在用来安装集群的计算机上创建几个文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。



重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，请为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。此 pull secret 允许您与所含授权机构提供的服务进行身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

10.6.5. 导出 API 密钥

您必须将您创建的 API 密钥设置为全局变量；安装程序会在启动期间设置 API 密钥。

先决条件

- 您已为 IBM Cloud® 帐户创建了用户 API 密钥或服务 ID API 密钥。

流程

- 将帐户的 API 密钥导出为全局变量：

```
$ export IC_API_KEY=<api_key>
```

重要

您必须按照指定方式设置变量名称，安装程序需要在启动期间存在变量名称。

10.6.6. 创建安装配置文件

您可以自定义在 IBM Cloud® 上安装的 OpenShift Container Platform 集群。

先决条件

- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。

流程

1. 创建 **install-config.yaml** 文件。
   a. 进入包含安装程序的目录并运行以下命令：

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

   1 对于 **<installation_directory>**，请指定要存储安装程序创建的文件的目录名称。

   在指定目录时：

   - 验证该目录是否具有**执行**权限。在安装目录中运行 Terraform 二进制文件需要这个权限。
   - 使用空目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

   b. 在提示符处，提供云的配置详情：
      i. 可选： 选择用于访问集群机器的 SSH 密钥。

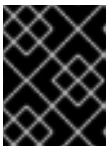
注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

      ii. 选择 **ibmcloud** 作为目标平台。
      iii. 选择要将集群部署到的区域。
      iv. 选择集群要部署到的基域。基域与您为集群创建的公共 DNS 区对应。
      v. 为集群输入描述性名称。

2. 修改 **install-config.yaml** 文件。您可以在"安装配置参数"部分找到有关可用参数的更多信息。

3. 备份 **install-config.yaml** 文件，以便您可以使用它安装多个集群。



重要

install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用此文件，必须现在备份。

其他资源

- IBM Cloud® 的安装配置参数。

10.6.6.1. 集群安装的最低资源要求

每台集群机器都必须满足以下最低要求：

表 10.4. 最低资源要求

机器	操作系统	vCPU	虚拟内存	Storage	每秒输入/输出 (IOPS)
bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane（控制平面）	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS	2	8 GB	100 GB	300

注意

从 OpenShift Container Platform 版本 4.13 开始，RHCOS 基于 RHEL 版本 9.2，它更新了微架构要求。以下列表包含每个架构需要的最小指令集架构 (ISA)：

- x86-64 体系结构需要 x86-64-v2 ISA
- ARM64 架构需要 ARMv8.0-A ISA
- IBM Power 架构需要 Power 9 ISA
- s390x 架构需要 z14 ISA

如需更多信息，请参阅 RHEL 架构。

如果平台的实例类型满足集群机器的最低要求，则 OpenShift Container Platform 支持使用它。

10.6.6.2. 为 IBM Cloud 测试的实例类型

以下 IBM Cloud® 实例类型已使用 OpenShift Container Platform 测试。

例 10.2. 机器系列

- **c4.***

- **c5.***
- **c5a.***
- **i3.***
- **m4.***
- **m5.***
- **m5a.***
- **m6a.***
- **m6i.***
- **r4.***
- **r5.***
- **r5a.***
- **r6i.***
- **t3.***
- **t3a.***

其他资源

- 优化存储

10.6.6.3. IBM Cloud 的自定义 install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多详情，或修改所需参数的值。

重要

此示例 YAML 文件仅供参考。您必须使用安装程序来获取 **install-config.yaml** 文件，然后修改该文件。

```
apiVersion: v1
baseDomain: example.com (1)
controlPlane: (2) (3)
  hyperthreading: Enabled (4)
  name: master
  platform:
    ibmcloud: {}
  replicas: 3
compute: (5) (6)
- hyperthreading: Enabled (7)
  name: worker
```

```
  platform:
    ibmcloud: {}
  replicas: 3
metadata:
  name: test-cluster 8
networking: 9
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 10
  serviceNetwork:
  - 172.30.0.0/16
platform:
  ibmcloud:
    region: us-south 11
credentialsMode: Manual
publish: External
pullSecret: '{"auths": ...}' 12
fips: false 13
sshKey: ssh-ed25519 AAAA... 14
```

1 8 11 12 必需。安装程序会提示您输入这个值。

2 5 9 如果没有提供这些参数和值，安装程序会提供默认值。

3 6 **controlPlane** 部分是一个单个映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，controlPlane **部分** 的第一行则不以连字符开头。仅使用一个 control plane 池。

4 7 启用或禁用并发多线程，也称为 Hyper-Threading。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将 参数值设置为 **Disabled** 来禁用它。如果在某些集群机器中禁用并发多线程，则必须在所有集群机器中禁用它。

重要

如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。如果您禁用并发多线程，请为您的机器使用较大的类型，如 **n1-standard-8**。

10 要安装的集群网络插件。默认值 **OVNKubernetes** 是唯一支持的值。

13 启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 Red Hat Enterprise Linux (RHEL) 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅在 FIPS 模式中安装该系统。

当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS)时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。

14 可选：提供用于访问集群中机器的 **sshKey** 值。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

10.6.6.4. 在安装过程中配置集群范围的代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 **Proxy** 对象的 **spec.noProxy** 字段中添加站点来绕过代理。

注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，**Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
```

```
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 **.** 以仅匹配子域。例如，**.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代理。
4. 如果提供，安装程序会在 **openshift-config** 命名空间中生成名为 **user-ca-bundle** 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 **trusted-ca-bundle** 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， **Proxy** 对象的 **trustedCA** 字段中也会引用此配置映射。**additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。
5. 可选：决定 **Proxy** 对象的配置以引用 **trustedCA** 字段中 **user-ca-bundle** 配置映射的策略。允许的值是 **Proxyonly** 和 **Always**。仅在配置了 **http/https** 代理时，使用 **Proxyonly** 引用 **user-ca-bundle** 配置映射。使用 **Always** 始终引用 **user-ca-bundle** 配置映射。默认值为 **Proxyonly**。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。

注意

如果安装程序超时，重启并使用安装程序的 **wait-for** 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. 保存该文件并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 cluster 的集群范围代理，该代理 **使用** 提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

10.6.7. 手动创建 IAM

安装集群需要 Cloud Credential Operator(CCO)以手动模式运行。虽然安装程序为手动模式配置 CCO，但您必须为云供应商指定身份和访问管理 secret。

您可以使用 Cloud Credential Operator (CCO)实用程序(**ccoctl**)创建所需的 IBM Cloud® 资源。

先决条件

- 您已配置了 **ccoctl** 二进制文件。

- 您有一个现有的 **install-config.yaml** 文件。

流程

1. 编辑 **install-config.yaml** 配置文件，使其包含将 **credentialsMode** 参数设置为 **Manual**。

   install-config.yaml 配置文件示例

   ```
   apiVersion: v1
   baseDomain: cluster1.example.com
   credentialsMode: Manual (1)
   compute:
   - architecture: amd64
     hyperthreading: Enabled
   ```

   (1) 添加这一行将 **credentialsMode** 参数设置为 **Manual**。

2. 要生成清单，请在包含安装程序的目录中运行以下命令：

   ```
   $ openshift-install create manifests --dir <installation_directory>
   ```

3. 在包含安装程序的目录中，运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

   ```
   $ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
   ```

4. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 自定义资源 (CR) 列表：

   ```
   $ oc adm release extract \
     --from=$RELEASE_IMAGE \
     --credentials-requests \
     --included \(1)
     --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \(2)
     --to=<path_to_directory_for_credentials_requests> (3)
   ```

   (1) **--included** 参数仅包含特定集群配置所需的清单。

   (2) 指定 **install-config.yaml** 文件的位置。

   (3) 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

   此命令为每个 **CredentialsRequest** 对象创建一个 YAML 文件。

   CredentialsRequest 对象示例

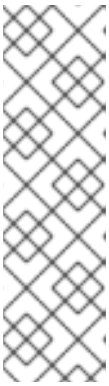
   ```
   apiVersion: cloudcredential.openshift.io/v1
   kind: CredentialsRequest
   metadata:
     labels:
       controller-tools.k8s.io: "1.0"
   ```

```
    name: openshift-image-registry-ibmcos
    namespace: openshift-cloud-credential-operator
  spec:
    secretRef:
      name: installer-cloud-credentials
      namespace: openshift-image-registry
    providerSpec:
      apiVersion: cloudcredential.openshift.io/v1
      kind: IBMCloudProviderSpec
      policies:
      - attributes:
        - name: serviceName
          value: cloud-object-storage
        roles:
        - crn:v1:bluemix:public:iam::::role:Viewer
        - crn:v1:bluemix:public:iam::::role:Operator
        - crn:v1:bluemix:public:iam::::role:Editor
        - crn:v1:bluemix:public:iam::::serviceRole:Reader
        - crn:v1:bluemix:public:iam::::serviceRole:Writer
      - attributes:
        - name: resourceType
          value: resource-group
        roles:
        - crn:v1:bluemix:public:iam::::role:Viewer
```

5. 为每个凭证请求创建服务 ID，分配定义的策略、创建 API 密钥并生成 secret：

```
$ ccoctl ibmcloud create-service-id \
  --credentials-requests-dir=<path_to_credential_requests_directory> \ 1
  --name=<cluster_name> \ 2
  --output-dir=<installation_directory> \ 3
  --resource-group-name=<resource_group_name> 4
```

1. 指定包含组件 **CredentialsRequest** 对象文件的目录。
2. 指定 OpenShift Container Platform 集群的名称。
3. 可选：指定您希望 **ccoctl** 实用程序在其中创建对象的目录。默认情况下，实用程序在运行命令的目录中创建对象。
4. 可选：指定用于限制访问策略的资源组的名称。



注意

如果您的集群使用 **TechPreviewNoUpgrade** 功能集启用的技术预览功能，则必须包含 **--enable-tech-preview** 参数。

如果提供了不正确的资源组名称，安装会在 bootstrap 阶段失败。要查找正确的资源组名称，请运行以下命令：

```
$ grep resourceGroupName <installation_directory>/manifests/cluster-
infrastructure-02-config.yml
```

验证

- 确保在集群的 **manifests** 目录中生成了适当的 secret。

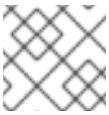
10.6.8. 网络配置阶段

OpenShift Container Platform 安装前有两个阶段，您可以在其中自定义网络配置。

第 1 阶段

在创建清单文件前，您可以自定义 **install-config.yaml** 文件中的以下与网络相关的字段：

- **networking.networkType**
- **networking.clusterNetwork**
- **networking.serviceNetwork**
- **networking.machineNetwork**
  有关这些字段的更多信息，请参阅 *安装配置参数*。



注意

将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

重要

CIDR 范围 **172.17.0.0/16** 由 libVirt 保留。对于集群中的任何网络，您无法使用此范围或与这个范围重叠的范围。

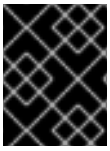
第 2 阶段

运行 **openshift-install create 清单创建** 清单文件后，您可以只使用您要修改的字段定义自定义 Cluster Network Operator 清单。您可以使用 清单指定高级网络配置。

您不能覆盖在 stage 2 阶段 1 中在 **install-config.yaml** 文件中指定的值。但是，您可以在第 2 阶段进一步自定义网络插件。

10.6.9. 指定高级网络配置

您可以使用网络插件的高级网络配置将集群集成到现有网络环境中。您只能在安装集群前指定高级网络配置。



重要

不支持通过修改安装程序创建的 OpenShift Container Platform 清单文件来自定义网络配置。支持应用您创建的清单文件，如以下流程中所示。

先决条件

- 您已创建 **install-config.yaml** 文件并完成对其所做的任何修改。

流程

1. 进入包含安装程序的目录并创建清单：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 **<installation_directory>** 指定包含集群的 **install-config.yaml** 文件的目录名称。

2. 在 **<installation_directory>/manifests/** 目录中 为**高级网络配置创建一个名为 cluster-network-03-config.yml** 的 stub 清单文件：

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
```

3. 在 **cluster-network-03-config.yml** 文件中指定集群的高级网络配置，如下例所示：

   为 OVN-Kubernetes 网络供应商启用 IPsec

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  defaultNetwork:
    ovnKubernetesConfig:
      ipsecConfig:
        mode: Full
```

4. 可选：备份 **manifests/cluster-network-03-config.yml** 文件。创建 Ignition 配置文件时，安装程序会使用 **manifests/** 目录。

10.6.10. Cluster Network Operator 配置

集群网络的配置作为 Cluster Network Operator(CNO)配置的一部分指定，并存储在名为 **cluster** 的自定义资源(CR)对象中。CR 指定 **operator.openshift.io** API 组中的 **Network** API 的字段。

CNO 配置在集群安装过程中从 **Network.config.openshift.io** API 组中的 **Network** API 继承以下字段：

clusterNetwork

从中分配 Pod IP 地址的 IP 地址池。

serviceNetwork

服务的 IP 地址池.

defaultNetwork.type

集群网络插件。**OVNKubernetes** 是安装期间唯一支持的插件。

您可以通过在名为 **cluster** 的 CNO 对象中设置 **defaultNetwork** 对象的字段来为集群指定集群网络插件配置。

10.6.10.1. Cluster Network Operator 配置对象

下表中描述了 Cluster Network Operator(CNO)的字段：

表 10.5. Cluster Network Operator 配置对象

字段	类型	描述
metadata.name	**字符串**	CNO 对象的名称。这个名称始终是 **集群**。
spec.clusterNetwork	**array**	用于指定从哪些 IP 地址块分配 Pod IP 地址以及集群中每个节点的子网前缀长度的列表。例如： spec: clusterNetwork: - cidr: 10.128.0.0/19 hostPrefix: 23 - cidr: 10.128.32.0/19 hostPrefix: 23
spec.serviceNetwork	**array**	服务的 IP 地址块。OpenShift SDN 和 OVN-Kubernetes 网络插件只支持服务网络的一个 IP 地址块。例如： spec: serviceNetwork: - 172.30.0.0/14 您只能在创建清单前在 **install-config.yaml** 文件中自定义此字段。该值在清单文件中是只读的。
spec.defaultNetwork	**object**	为集群网络配置网络插件。
spec.kubeProxyConfig	**object**	此对象的字段指定 kube-proxy 配置。如果使用 OVN-Kubernetes 集群网络供应商，则 kube-proxy 配置不会起作用。

defaultNetwork 对象配置

下表列出了 **defaultNetwork** 对象的值：

表 10.6. defaultNetwork 对象

字段	类型	描述
type	**字符串**	**OVNKubernetes**。Red Hat OpenShift Networking 网络插件在安装过程中被选择。此值在集群安装后无法更改。 **注意** OpenShift Container Platform 默认使用 OVN-Kubernetes 网络插件。OpenShift SDN 不再作为新集群的安装选择提供。
ovnKubernetesConfig	**object**	此对象仅对 OVN-Kubernetes 网络插件有效。

配置 OVN-Kubernetes 网络插件
下表描述了 OVN-Kubernetes 网络插件的配置字段：

表 10.7. **ovnKubernetesConfig** object

字段	类型	描述
mtu	**integer**	Geneve（通用网络虚拟化封装）覆盖网络的最大传输单元 (MTU)。这根据主网络接口的 MTU 自动探测。您通常不需要覆盖检测到的 MTU。 如果自动探测的值不是您期望的值，请确认节点上主网络接口上的 MTU 是否正确。您不能使用这个选项更改节点上主网络接口的 MTU 值。 如果集群中不同节点需要不同的 MTU 值，则必须将此值设置为 **比** 集群中的最低 MTU 值小 100。例如，如果集群中的某些节点的 MTU 为 **9001**，而某些节点的 MTU 为 **1500**，则必须将此值设置为 **1400**。
genevePort	**integer**	用于所有 Geneve 数据包的端口。默认值为 **6081**。此值在集群安装后无法更改。
ipsecConfig	**object**	指定用于自定义 IPsec 配置的配置对象。
ipv4	**object**	为 IPv4 设置指定配置对象。
ipv6	**object**	为 IPv6 设置指定配置对象。
policyAuditConfig	**object**	指定用于自定义网络策略审计日志的配置对象。如果未设置，则使用默认的审计日志设置。
gatewayConfig	**object**	可选：指定一个配置对象来自定义如何将出口流量发送到节点网关。  **注意** 在迁移出口流量时，工作负载和服务流量会受到一定影响，直到 Cluster Network Operator (CNO) 成功推出更改。

表 10.8. **ovnKubernetesConfig.ipv4** object

字段	类型	描述

字段	类型	描述
internalTransitSwitchSubnet	字符串	如果您的现有网络基础架构与 **100.88.0.0/16** IPv4 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。启用东西流量的分布式传输交换机的子网。此子网不能与 OVN-Kubernetes 或主机本身使用的任何其他子网重叠。必须足够大，以适应集群中的每个节点一个 IP 地址。 默认值为 **100.88.0.0/16**。
internalJoinSubnet	字符串	如果您的现有网络基础架构与 **100.64.0.0/16** IPv4 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。您必须确保 IP 地址范围没有与 OpenShift Container Platform 安装使用的任何其他子网重叠。IP 地址范围必须大于可添加到集群的最大节点数。例如，如果 **clusterNetwork.cidr** 值为 **10.128.0.0/14**，并且 **clusterNetwork.hostPrefix** 值为 **/23**，则最大节点数量为 **2^(23-14)=512**。 默认值为 **100.64.0.0/16**。

表 10.9. **ovnKubernetesConfig.ipv6** object

字段	类型	描述
internalTransitSwitchSubnet	字符串	如果您的现有网络基础架构与 **fd97::/64** IPv6 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。启用东西流量的分布式传输交换机的子网。此子网不能与 OVN-Kubernetes 或主机本身使用的任何其他子网重叠。必须足够大，以适应集群中的每个节点一个 IP 地址。 默认值为 **fd97::/64**。
internalJoinSubnet	字符串	如果您的现有网络基础架构与 **fd98::/64** IPv6 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。您必须确保 IP 地址范围没有与 OpenShift Container Platform 安装使用的任何其他子网重叠。IP 地址范围必须大于可添加到集群的最大节点数。 默认值为 **fd98::/64**。

表 10.10. **policyAuditConfig** object

字段	类型	描述
rateLimit	整数	每个节点每秒生成一次的消息数量上限。默认值为每秒 **20** 条消息。
maxFileSize	整数	审计日志的最大大小，以字节为单位。默认值为 **50000000** 或 50 MB。
maxLogFiles	整数	保留的日志文件的最大数量。

字段	类型	描述
目的地	字符串	以下附加审计日志目标之一： **libc** 主机上的 journald 进程的 libc **syslog（）** 函数。 **UDP:<host>:<port>** 一个 syslog 服务器。将 **<host>:<port>** 替换为 **syslog 服务器的主机** 和端口。 **Unix:<file>** 由 **<file>** 指定的 Unix 域套接字文件。 **null** 不要将审计日志发送到任何其他目标。
syslogFacility	字符串	syslog 工具，如 as **kern**，如 RFC5424 定义。默认值为 **local0。**

表 10.11. **gatewayConfig** object

字段	类型	描述
routingViaHost	**布尔值**	将此字段设置为 **true**，将来自 pod 的出口流量发送到主机网络堆栈。对于依赖于在内核路由表中手动配置路由的高级别安装和应用程序，您可能需要将出口流量路由到主机网络堆栈。默认情况下，出口流量在 OVN 中进行处理以退出集群，不受内核路由表中的特殊路由的影响。默认值为 **false**。 此字段与 Open vSwitch 硬件卸载功能有交互。如果将此字段设置为 **true**，则不会获得卸载的性能优势，因为主机网络堆栈会处理出口流量。
ipForwarding	**object**	您可以使用 **Network** 资源中的 **ipForwarding** 规格来控制 OVN-Kubernetes 管理接口上所有流量的 IP 转发。指定 **Restricted** 只允许 Kubernetes 相关流量的 IP 转发。指定 **Global** 以允许转发所有 IP 流量。对于新安装，默认值为 **Restricted**。对于到 OpenShift Container Platform 4.14 或更高版本的更新，默认值为 **Global**。
ipv4	**object**	可选：指定一个对象来为主机配置内部 OVN-Kubernetes 伪装地址，以服务 IPv4 地址的流量。
ipv6	**object**	可选：指定一个对象来为主机配置内部 OVN-Kubernetes 伪装地址，以服务 IPv6 地址的流量。

表 10.12. **gatewayConfig.ipv4** 对象

字段	类型	描述
internalMasqueradeSubnet	**字符串**	内部使用的伪装 IPv4 地址，以启用主机服务流量。主机配置了这些 IP 地址和共享网关网桥接口。默认值为 **169.254.169.0/29**。

表 10.13. **gatewayConfig.ipv6** 对象

字段	类型	描述
internalMasqueradeSubnet	**字符串**	内部使用的伪装 IPv6 地址，以启用主机服务流量。主机配置了这些 IP 地址和共享网关网桥接口。默认值为 **fd69::/125**。

表 10.14. **ipsecConfig** 对象

字段	类型	描述
模式	**字符串**	指定 IPsec 实现的行为。必须是以下值之一： • **Disabled**: 在集群节点上不启用 IPsec。 • **External** ：对于带有外部主机的网络流量，启用 IPsec。 • **Full**: IPsec 为带有外部主机的 pod 流量和网络流量启用 IPsec。

启用 IPSec 的 OVN-Kubernetes 配置示例

```
defaultNetwork:
  type: OVNKubernetes
  ovnKubernetesConfig:
    mtu: 1400
    genevePort: 6081
      ipsecConfig:
        mode: Full
```

重要

使用 OVNKubernetes 可能会导致 IBM Power® 上的堆栈耗尽问题。

kubeProxyConfig 对象配置（仅限 OpenShiftSDN 容器网络接口）
kubeProxyConfig 对象的值在下表中定义：

表 10.15. **kubeProxyConfig** object

字段	类型	描述

字段	类型	描述
iptablesSyncPeriod	**字符串**	**iptables** 规则的刷新周期。默认值为 **30s**。有效的后缀包括 **s**、**m** 和 **h**，具体参见 Go 时间 包 文档。 **注意** 由于 OpenShift Container Platform 4.3 及更高版本中引进了性能改进，不再需要调整 **iptablesSyncPeriod** 参数。
proxyArguments.iptables-min-sync-period	**array**	刷新 **iptables** 规则前的最短持续时间。此字段确保刷新的频率不会过于频繁。有效的后缀包括 **s**、**m** 和 **h**，具体参见 Go **time** 软件包。默认值为： kubeProxyConfig: proxyArguments: iptables-min-sync-period: - 0s

10.6.11. 部署集群

您可以在兼容云平台上安装 OpenShift Container Platform。

重要

在初始安装过程中，您只能运行安装程序的 **create cluster** 命令一次。

先决条件

- 您已使用托管集群的云平台配置了帐户。
- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。
- 已确认主机上的云供应商帐户具有部署集群的正确权限。权限不正确的帐户会导致安装过程失败，并显示包括缺失权限的错误消息。

流程

- 进入包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1. 对于 **<installation_directory>**，请指定自定义 **./install-config.yaml** 文件的位置。
2. 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不是 **info**。

验证

当集群部署成功完成时：

- 终端会显示用于访问集群的说明，包括指向 Web 控制台和 **kubeadmin** 用户的凭证的链接。
- 凭证信息还会输出到 **<installation_directory>/.openshift_install.log**.

重要

不要删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

10.6.12. 安装 OpenShift CLI

您可以安装 OpenShift CLI(**oc**)来使用命令行界面与 OpenShift Container Platform 进行交互。您可以在 Linux、Windows 或 macOS 上安装 **oc**。

重要

如果安装了旧版本的 **oc**，则可能无法使用 OpenShift Container Platform 4.16 中的所有命令。下载并安装新版本的 **oc**。

在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **产品变体** 下拉列表中选择架构。
3. 从 **版本** 下拉列表中选择适当的版本。

4. 点 **OpenShift v4.16 Linux Client** 条目旁的 **Download Now** 来保存文件。
5. 解包存档：

```
$ tar xvf <file>
```

6. 将 **oc** 二进制文件放到 **PATH 中的目录中**。
要查看您的 **PATH**，请执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

在 Windows 上安装 OpenShift CLI
您可以按照以下流程在 Windows 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 Windows Client** 条目旁的 **Download Now** 来保存文件。
4. 使用 ZIP 程序解压存档。
5. 将 **oc** 二进制文件移到 **PATH 中的目录中**。
要查看您的 **PATH**，请打开命令提示并执行以下命令：

```
C:\> path
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

在 macOS 上安装 OpenShift CLI
您可以按照以下流程在 macOS 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 macOS Client** 条目旁的 **Download Now** 来保存文件。

注意

对于 macOS arm64，请选择 **OpenShift v4.16 macOS arm64 Client** 条目。

4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
   要查看您的 **PATH**，请打开终端并执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

10.6.13. 使用 CLI 登录集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

其他资源

- 访问Web控制台

10.6.14. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 关于远程健康监控

10.6.15. 后续步骤

- 自定义集群。
- 如果需要，您可以选择 不使用远程健康报告。

10.7. 在 IBM CLOUD 上安装集群到现有的 VPC 中

在 OpenShift Container Platform 版本 4.16 中，您可以在 IBM Cloud® 上将集群安装到现有的 Virtual Private Cloud (VPC)中。安装程序会置备所需基础架构的其余部分，您可以进一步自定义这些基础架构。要自定义安装，请在安装集群前修改 **install-config.yaml** 文件中的参数。

10.7.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读有关 选择集群安装方法的文档，并为用户准备它。
- 已将 IBM Cloud® 帐户配置为托管集群 。
- 如果使用防火墙，则会 将其配置为允许集群需要访问的站点。
- 在安装集群前已经配置了 **ccoctl** 工具。如需更多信息，请参阅为 IBM Cloud® 配置 IAM 。

10.7.2. 关于使用自定义 VPC

在 OpenShift Container Platform 4.16 中，您可以将集群部署到现有 IBM® Virtual Private Cloud (VPC)的子网。将 OpenShift Container Platform 部署到现有的 VPC 中可帮助您避免新帐户中的限制，或者更容易地利用公司所设置的操作限制。如果您无法获得您自己创建 VPC 所需的基础架构创建权限，请使用这个安装选项。

因为安装程序不知道现有子网中的其他组件，所以无法选择子网 CIDR 等。您必须为安装集群的子网配置网络。

10.7.2.1. 使用 VPC 的要求

您必须在安装集群前正确配置现有的 VPC 及其子网。安装程序不会创建以下组件：

- NAT 网关
- 子网
- 路由表

- VPC 网络

安装程序无法：

- 从属网络范围供集群使用
- 为子网设置路由表
- 设置 VPC 选项，如 DHCP

注意

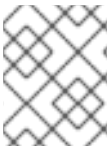
安装程序要求您使用由云提供的 DNS 服务器。不支持使用自定义 DNS 服务器，并导致安装失败。

10.7.2.2. VPC 验证

VPC 和所有子网都必须位于现有资源组中。集群部署到现有的 VPC 中。

作为安装的一部分，在 **install-config.yaml** 文件中指定以下内容：

- 包含 VPC 和子网的现有资源组的名称 (**networkResourceGroupName**)
- 现有 VPC 的名称(**vpcName**)
- 为 control plane 机器和计算机器创建的子网 (**controlPlaneSubnets** 和 **computeSubnets**)

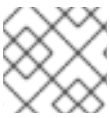


注意

额外的安装程序置备的集群资源被部署到单独的资源组 (**resourceGroupName**) 中。您可在安装集群前指定此资源组。如果未定义，则会为集群创建新的资源组。

要确保您提供的子网适合，安装程序会确认以下内容：

- 您指定的所有子网都存在。
- 对于区域中的每个可用区，您可以指定：
  - 一个用于 control plane 机器的子网。
  - 计算机器的一个子网。
- 您指定的机器 CIDR 包含计算机器和 control plane 机器的子网。



注意

不支持子网 ID。

10.7.2.3. 集群间隔离

如果您将 OpenShift Container Platform 部署到现有网络中，集群服务的隔离将在以下方面减少：

- 您可以在同一 VPC 中安装多个 OpenShift Container Platform 集群。
- 整个网络允许 ICMP 入站流量。

- 整个网络都允许 TCP 端口 22 入站流量 (SSH)。
- 整个网络都允许 control plane TCP 6443 入站流量 (Kubernetes API)。
- 整个网络都允许 control plane TCP 22623 入站流量 (MCS) 。

10.7.3. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类型的基础架构上执行受限网络安装。在此过程中，您可以下载所需的内容，并使用它为镜像 registry 填充安装软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群前，您要更新镜像 registry 的内容。

10.7.4. 为集群节点 SSH 访问生成密钥对

在 OpenShift Container Platform 安装过程中，您可以为安装程序提供 SSH 公钥。密钥通过它们的 Ignition 配置文件传递给 Red Hat Enterprise Linux CoreOS(RHCOS)节点，用于验证对节点的 SSH 访问。密钥添加到每个节点上 **core** 用户的 **~/.ssh/authorized_keys** 列表中，这将启用免密码身份验证。

将密钥传递给节点后，您可以使用密钥对作为用户 **核心** 通过 SSH 连接到 RHCOS 节点。若要通过 SSH 访问节点，必须由 SSH 为您的本地用户管理私钥身份。

如果要通过 SSH 连接到集群节点来执行安装调试或灾难恢复，则必须在安装过程中提供 SSH 公钥。**./openshift-install gather** 命令还需要在集群节点上设置 SSH 公钥。

重要

不要在生产环境中跳过这个过程，在生产环境中需要灾难恢复和调试。

注意

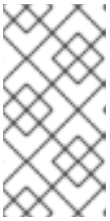
您必须使用本地密钥，而不是使用特定平台方法配置 的密钥，如 AWS 密钥对。

流程

1. 如果您在本地计算机上没有可用于在集群节点上进行身份验证的现有 SSH 密钥对，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_ed25519**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。



注意

如果您计划在 **x86_64**、**ppc64le** 和 **s390x** 架构上安装使用 RHEL 加密库（这些加密库已提交给 NIST 用于 FIPS 140-2/140-3 验证）的 OpenShift Container Platform 集群，则不要创建使用 **ed25519** 算法的密钥。相反，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

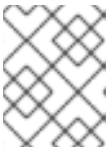
2. 查看公共 SSH 密钥：

```
$ cat <path>/<file_name>.pub
```

例如，运行以下命令来查看 **~/.ssh/id_ed25519.pub** 公钥：

```
$ cat ~/.ssh/id_ed25519.pub
```

3. 将 SSH 私钥身份添加到本地用户的 SSH 代理（如果尚未添加）。在集群节点上，或者要使用 **./openshift-install gather** 命令，需要对该密钥进行 SSH 代理管理，才能在集群节点上进行免密码 SSH 身份验证。



注意

在某些发行版中，自动管理默认 SSH 私钥身份，如 **~/.ssh/id_rsa** 和 **~/.ssh/id_dsa**。

a. 如果 **ssh-agent** 进程尚未为您的本地用户运行，请将其作为后台任务启动：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果集群处于 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

4. 将 SSH 私钥添加到 **ssh-agent** ：

```
$ ssh-add <path>/<file_name> 1
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_ed25519.pub**

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

后续步骤

- 安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

10.7.5. 获取安装程序

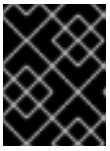
在安装 OpenShift Container Platform 前，将安装文件下载到您用于安装的主机上。

先决条件

- 您有一台运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB。

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐户，请使用您的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入到安装类型的页面，下载与您的主机操作系统和架构对应的安装程序，并将该文件放在您要存储安装配置文件的目录中。



重要

安装程序会在用来安装集群的计算机上创建几个文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。

重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，请为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。此 pull secret 允许您与所含授权机构提供的服务进行身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

10.7.6. 导出 API 密钥

您必须将您创建的 API 密钥设置为全局变量；安装程序会在启动期间设置 API 密钥。

先决条件

- 您已为 IBM Cloud® 帐户创建了用户 API 密钥或服务 ID API 密钥。

流程

- 将帐户的 API 密钥导出为全局变量：

```
$ export IC_API_KEY=<api_key>
```

重要

您必须按照指定方式设置变量名称，安装程序需要在启动期间存在变量名称。

10.7.7. 创建安装配置文件

您可以自定义在 IBM Cloud® 上安装的 OpenShift Container Platform 集群。

先决条件

- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。

流程

1. 创建 **install-config.yaml** 文件。
   a. 进入包含安装程序的目录并运行以下命令：

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

   1 对于 **<installation_directory>**，请指定要存储安装程序创建的文件的目录名称。

   在指定目录时：

   - 验证该目录是否具有**执行**权限。在安装目录中运行 Terraform 二进制文件需要这个权限。
   - 使用空目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

   b. 在提示符处，提供云的配置详情：
      i. 可选： 选择用于访问集群机器的 SSH 密钥。

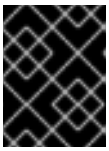
注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

      ii. 选择 **ibmcloud** 作为目标平台。
      iii. 选择要将集群部署到的区域。
      iv. 选择集群要部署到的基域。基域与您为集群创建的公共 DNS 区对应。
      v. 为集群输入描述性名称。

2. 修改 **install-config.yaml** 文件。您可以在"安装配置参数"部分找到有关可用参数的更多信息。

3. 备份 **install-config.yaml** 文件，以便您可以使用它安装多个集群。



重要

install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用此文件，必须现在备份。

其他资源

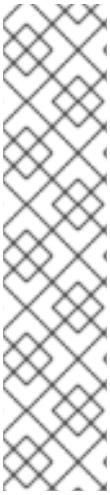
- IBM Cloud® 的安装配置参数。

10.7.7.1. 集群安装的最低资源要求

每台集群机器都必须满足以下最低要求：

表 10.16. 最低资源要求

机器	操作系统	vCPU	虚拟内存	Storage	每秒输入/输出 (IOPS)
bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane（控制平面）	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS	2	8 GB	100 GB	300



注意

从 OpenShift Container Platform 版本 4.13 开始，RHCOS 基于 RHEL 版本 9.2，它更新了微架构要求。以下列表包含每个架构需要的最小指令集架构 (ISA)：

- x86-64 体系结构需要 x86-64-v2 ISA
- ARM64 架构需要 ARMv8.0-A ISA
- IBM Power 架构需要 Power 9 ISA
- s390x 架构需要 z14 ISA

如需更多信息，请参阅 RHEL 架构。

如果平台的实例类型满足集群机器的最低要求，则 OpenShift Container Platform 支持使用它。

10.7.7.2. 为 IBM Cloud 测试的实例类型

以下 IBM Cloud® 实例类型已使用 OpenShift Container Platform 测试。

例 10.3. 机器系列

- **c4.***

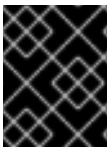
- **c5.***
- **c5a.***
- **i3.***
- **m4.***
- **m5.***
- **m5a.***
- **m6a.***
- **m6i.***
- **r4.***
- **r5.***
- **r5a.***
- **r6i.***
- **t3.***
- **t3a.***

其他资源

- 优化存储

10.7.7.3. IBM Cloud 的自定义 install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多详情，或修改所需参数的值。



重要

此示例 YAML 文件仅供参考。您必须使用安装程序来获取 **install-config.yaml** 文件，然后修改该文件。

```
apiVersion: v1
baseDomain: example.com (1)
controlPlane: (2) (3)
  hyperthreading: Enabled (4)
  name: master
  platform:
    ibmcloud: {}
  replicas: 3
compute: (5) (6)
- hyperthreading: Enabled (7)
  name: worker
```

```
  platform:
    ibmcloud: {}
  replicas: 3
metadata:
  name: test-cluster 8
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14 9
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 10
  serviceNetwork:
  - 172.30.0.0/16
platform:
  ibmcloud:
    region: eu-gb 11
    resourceGroupName: eu-gb-example-cluster-rg 12
    networkResourceGroupName: eu-gb-example-existing-network-rg 13
    vpcName: eu-gb-example-network-1 14
    controlPlaneSubnets: 15
      - eu-gb-example-network-1-cp-eu-gb-1
      - eu-gb-example-network-1-cp-eu-gb-2
      - eu-gb-example-network-1-cp-eu-gb-3
    computeSubnets: 16
      - eu-gb-example-network-1-compute-eu-gb-1
      - eu-gb-example-network-1-compute-eu-gb-2
      - eu-gb-example-network-1-compute-eu-gb-3
credentialsMode: Manual
publish: External
pullSecret: '{"auths": ...}' 17
fips: false 18
sshKey: ssh-ed25519 AAAA... 19
```

1 8 11 17 必需。安装程序会提示您输入这个值。

2 5 如果没有提供这些参数和值，安装程序会提供默认值。

3 6 **controlPlane** 部分是一个单个映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，controlPlane **部分** 的第一行则不以连字符开头。仅使用一个 control plane 池。

4 7 启用或禁用并发多线程，也称为 Hyper-Threading。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将 参数值设置为 **Disabled** 来禁用它。如果在某些集群机器中禁用并发多线程，则必须在所有集群机器中禁用它。

重要

如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。如果您禁用并发多线程，请为您的机器使用较大的类型，如 **n1-standard-8**。

9 机器 CIDR 必须包含计算机器和 control plane 机器的子网。

10 要安装的集群网络插件。默认值 **OVNKubernetes** 是唯一支持的值。

12 现有资源组的名称。所有安装程序置备的集群资源都部署到此资源组中。如果未定义，则会为集群创建新的资源组。

13 指定包含现有虚拟私有云 (VPC) 的资源组名称。现有的 VPC 和子网应该位于此资源组中。集群将安装到此 VPC 中。

14 指定现有 VPC 的名称。

15 指定要部署 control plane 机器的现有子网的名称。子网必须属于您指定的 VPC。为区域中的每个可用区指定一个子网。

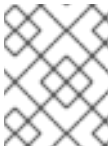
16 指定要部署计算机器的现有子网的名称。子网必须属于您指定的 VPC。为区域中的每个可用区指定一个子网。

18 启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS)时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。

19 可选：提供用于访问集群中机器的 **sshKey** 值。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

10.7.7.4. 在安装过程中配置集群范围的代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 **Proxy** 对象的 **spec.noProxy** 字段中添加站点来绕过代理。

注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，**Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

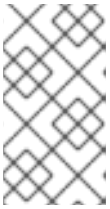
1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 **.** 以仅匹配子域。例如，**.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代理。
4. 如果提供，安装程序会在 **openshift-config** 命名空间中生成名为 **user-ca-bundle** 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 **trusted-ca-bundle** 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， **Proxy** 对象的 **trustedCA** 字段中也会引用此配置映射。**additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。
5. 可选：决定 **Proxy** 对象的配置以引用 **trustedCA** 字段中 **user-ca-bundle** 配置映射的策略。允许的值是 **Proxyonly** 和 **Always**。仅在配置了 **http/https** 代理时，使用 **Proxyonly** 引用 **user-ca-bundle** 配置映射。使用 **Always** 始终引用 **user-ca-bundle** 配置映射。默认值为 **Proxyonly**。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。



注意

如果安装程序超时，重启并使用安装程序的 **wait-for** 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. 保存该文件并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 cluster 的集群范围代理，该代理 **使用** 提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

10.7.8. 手动创建 IAM

安装集群需要 Cloud Credential Operator(CCO)以手动模式运行。虽然安装程序为手动模式配置 CCO，但您必须为云供应商指定身份和访问管理 secret。

您可以使用 Cloud Credential Operator (CCO)实用程序(**ccoctl**)创建所需的 IBM Cloud® 资源。

先决条件

- 您已配置了 **ccoctl** 二进制文件。
- 您有一个现有的 **install-config.yaml** 文件。

流程

1. 编辑 **install-config.yaml** 配置文件，使其包含将 **credentialsMode** 参数设置为 **Manual**。

   install-config.yaml 配置文件示例

   ```
   apiVersion: v1
   baseDomain: cluster1.example.com
   credentialsMode: Manual 1
   compute:
   - architecture: amd64
     hyperthreading: Enabled
   ```

   1 添加这一行将 **credentialsMode** 参数设置为 **Manual**。

2. 要生成清单，请在包含安装程序的目录中运行以下命令：

   ```
   $ openshift-install create manifests --dir <installation_directory>
   ```

3. 在包含安装程序的目录中，运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

   ```
   $ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
   ```

4. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 自定义资源 (CR) 列表：

   ```
   $ oc adm release extract \
     --from=$RELEASE_IMAGE \
     --credentials-requests \
     --included \ 1
     --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
     --to=<path_to_directory_for_credentials_requests> 3
   ```

   1 **--included** 参数仅包含特定集群配置所需的清单。

2 指定 **install-config.yaml** 文件的位置。

3 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

此命令为每个 **CredentialsRequest** 对象创建一个 YAML 文件。

CredentialsRequest 对象示例

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  labels:
    controller-tools.k8s.io: "1.0"
  name: openshift-image-registry-ibmcos
  namespace: openshift-cloud-credential-operator
spec:
  secretRef:
    name: installer-cloud-credentials
    namespace: openshift-image-registry
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: IBMCloudProviderSpec
    policies:
    - attributes:
      - name: serviceName
        value: cloud-object-storage
      roles:
      - crn:v1:bluemix:public:iam::::role:Viewer
      - crn:v1:bluemix:public:iam::::role:Operator
      - crn:v1:bluemix:public:iam::::role:Editor
      - crn:v1:bluemix:public:iam::::serviceRole:Reader
      - crn:v1:bluemix:public:iam::::serviceRole:Writer
    - attributes:
      - name: resourceType
        value: resource-group
      roles:
      - crn:v1:bluemix:public:iam::::role:Viewer
```

5. 为每个凭证请求创建服务 ID，分配定义的策略、创建 API 密钥并生成 secret：

```
$ ccoctl ibmcloud create-service-id \
  --credentials-requests-dir=<path_to_credential_requests_directory> \1
  --name=<cluster_name> \2
  --output-dir=<installation_directory> \3
  --resource-group-name=<resource_group_name> 4
```

1 指定包含组件 **CredentialsRequest** 对象文件的目录。

2 指定 OpenShift Container Platform 集群的名称。

3 可选：指定您希望 **ccoctl** 实用程序在其中创建对象的目录。默认情况下，实用程序在运行命令的目录中创建对象。

4 可选：指定用于限制访问策略的资源组的名称。

注意

如果您的集群使用 **TechPreviewNoUpgrade** 功能集启用的技术预览功能，则必须包含 **--enable-tech-preview** 参数。

如果提供了不正确的资源组名称，安装会在 bootstrap 阶段失败。要查找正确的资源组名称，请运行以下命令：

```
$ grep resourceGroupName <installation_directory>/manifests/cluster-infrastructure-02-config.yml
```

验证

- 确保在集群的 **manifests** 目录中生成了适当的 secret。

10.7.9. 部署集群

您可以在兼容云平台上安装 OpenShift Container Platform。

重要

在初始安装过程中，您只能运行安装程序的 **create cluster** 命令一次。

先决条件

- 您已使用托管集群的云平台配置了帐户。
- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。
- 已确认主机上的云供应商帐户具有部署集群的正确权限。权限不正确的帐户会导致安装过程失败，并显示包括缺失权限的错误消息。

流程

- 进入包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定自定义 **./install-config.yaml** 文件的位置。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不是 **info**。

验证

当集群部署成功完成时：

- 终端会显示用于访问集群的说明，包括指向 Web 控制台和 **kubeadmin** 用户的凭证的链接。
- 凭证信息还会输出到 **<installation_directory>/.openshift_install.log**.

重要

不要删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

10.7.10. 安装 OpenShift CLI

您可以安装 OpenShift CLI(**oc**)来使用命令行界面与 OpenShift Container Platform 进行交互。您可以在 Linux、Windows 或 macOS 上安装 **oc**。

重要

如果安装了旧版本的 **oc**，则可能无法使用 OpenShift Container Platform 4.16 中的所有命令。下载并安装新版本的 **oc**。

在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **产品变体** 下拉列表中选择架构。
3. 从 **版本** 下拉列表中选择适当的版本。
4. 点 **OpenShift v4.16 Linux Client** 条目旁的 **Download Now** 来保存文件。
5. 解包存档：

```
$ tar xvf <file>
```

6. 将 **oc** 二进制文件放到 **PATH 中的目录中**。
   要查看您的 **PATH**，请执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

在 Windows 上安装 OpenShift CLI
您可以按照以下流程在 Windows 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 Windows Client** 条目旁的 **Download Now** 来保存文件。
4. 使用 ZIP 程序解压存档。
5. 将 **oc** 二进制文件移到 **PATH 中的目录中**。
   要查看您的 **PATH**，请打开命令提示并执行以下命令：

```
C:\> path
```

验证

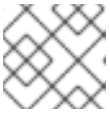
- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

在 macOS 上安装 OpenShift CLI
您可以按照以下流程在 macOS 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 macOS Client** 条目旁的 **Download Now** 来保存文件。



注意

对于 macOS arm64，请选择 **OpenShift v4.16 macOS arm64 Client** 条目。

4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。

要查看您的 **PATH**，请打开终端并执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

10.7.11. 使用 CLI 登录集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

其他资源

- 访问Web控制台

10.7.12. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 关于远程健康监控

10.7.13. 后续步骤

- 自定义集群。
- 可选： 不使用远程健康报告。

10.8. 在 IBM CLOUD 上安装私有集群

在 OpenShift Container Platform 版本 4.16 中，您可以将私有集群安装到现有的 VPC 中。安装程序会置备所需基础架构的其余部分，您可以进一步自定义这些基础架构。要自定义安装，请在安装集群前修改 **install-config.yaml** 文件中的参数。

10.8.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读有关 选择集群安装方法的文档，并为用户准备它。
- 已将 IBM Cloud® 帐户配置为托管集群 。
- 如果使用防火墙，则会 将其配置为允许集群需要访问的站点。
- 在安装集群前已经配置了 **ccoctl** 工具。如需更多信息，请参阅为 IBM Cloud® 配置 IAM 。

10.8.2. 私有集群

您可以部署不公开外部端点的私有 OpenShift Container Platform 集群。私有集群只能从内部网络访问，且无法在互联网中看到。

默认情况下，OpenShift Container Platform 被置备为使用可公开访问的 DNS 和端点。在部署集群时，私有集群会将 DNS、Ingress Controller 和 API 服务器设置为私有。这意味着集群资源只能从您的内部网络访问，且不能在互联网中看到。

重要

如果集群有任何公共子网，管理员创建的负载均衡器服务可能会公开访问。为确保集群安全性，请验证这些服务是否已明确标注为私有。

要部署私有集群，您必须：

- 使用满足您的要求的现有网络。集群资源可能会在网络上的其他集群间共享。
- 使用 IBM Cloud® DNS Services 创建 DNS 区域，并将其指定为集群的基域。如需更多信息，请参阅"使用 IBM Cloud® DNS 服务来配置 DNS 解析"。
- 从有权访问的机器中部署：
  - 您置备的云的 API 服务。
  - 您调配的网络上的主机。

- 用于获取安装介质的互联网。

您可以使用符合这些访问要求的机器，并按照您的公司规定进行操作。例如，此机器可以是云网络上的堡垒主机，也可以是可通过 VPN 访问网络的机器。

10.8.3. IBM Cloud 中的私有集群

要在 IBM Cloud® 上创建私有集群，您必须提供一个现有的私有 VPC 和子网来托管集群。安装程序还必须能够解析集群所需的 DNS 记录。安装程序只为内部流量配置 Ingress Operator 和 API 服务器。

集群仍然需要访问互联网来访问 IBM Cloud® API。

安装私有集群时不需要或创建以下项目：

- 公共子网
- 支持公共入口的公共网络负载均衡器
- 与集群的 **baseDomain** 匹配的公共 DNS 区域

安装程序会使用您指定的 **baseDomain** 来创建私有 DNS 区域以及集群所需的记录。集群被配置，以便 Operator 不会为集群创建公共记录，并将所有集群机器放置在您指定的私有子网中。

10.8.3.1. 限制

IBM Cloud® 上的私有集群只受到与集群部署的现有 VPC 相关的限制。

10.8.4. 关于使用自定义 VPC

在 OpenShift Container Platform 4.16 中，您可以将集群部署到现有 IBM® Virtual Private Cloud (VPC)的子网。将 OpenShift Container Platform 部署到现有的 VPC 中可帮助您避免新帐户中的限制，或者更容易地利用公司所设置的操作限制。如果您无法获得您自己创建 VPC 所需的基础架构创建权限，请使用这个安装选项。

因为安装程序不知道现有子网中的其他组件，所以无法选择子网 CIDR 等。您必须为安装集群的子网配置网络。

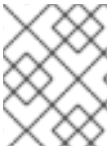
10.8.4.1. 使用 VPC 的要求

您必须在安装集群前正确配置现有的 VPC 及其子网。安装程序不会创建以下组件：

- NAT 网关
- 子网
- 路由表
- VPC 网络

安装程序无法：

- 从属网络范围供集群使用
- 为子网设置路由表
- 设置 VPC 选项，如 DHCP



注意

安装程序要求您使用由云提供的 DNS 服务器。不支持使用自定义 DNS 服务器，并导致安装失败。

10.8.4.2. VPC 验证

VPC 和所有子网都必须位于现有资源组中。集群部署到现有的 VPC 中。

作为安装的一部分，在 **install-config.yaml** 文件中指定以下内容：

- 包含 VPC 和子网的现有资源组的名称 (**networkResourceGroupName**)
- 现有 VPC 的名称(**vpcName**)
- 为 control plane 机器和计算机器创建的子网 (**controlPlaneSubnets** 和 **computeSubnets**)

注意

额外的安装程序置备的集群资源被部署到单独的资源组 (**resourceGroupName**) 中。您可在安装集群前指定此资源组。如果未定义，则会为集群创建新的资源组。

要确保您提供的子网适合，安装程序会确认以下内容：

- 您指定的所有子网都存在。
- 对于区域中的每个可用区，您可以指定：
  - 一个用于 control plane 机器的子网。
  - 计算机器的一个子网。
- 您指定的机器 CIDR 包含计算机器和 control plane 机器的子网。

注意

不支持子网 ID。

10.8.4.3. 集群间隔离

如果您将 OpenShift Container Platform 部署到现有网络中，集群服务的隔离将在以下方面减少：

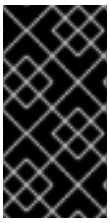
- 您可以在同一 VPC 中安装多个 OpenShift Container Platform 集群。
- 整个网络允许 ICMP 入站流量。
- 整个网络都允许 TCP 端口 22 入站流量 (SSH)。
- 整个网络都允许 control plane TCP 6443 入站流量 (Kubernetes API)。
- 整个网络都允许 control plane TCP 22623 入站流量 (MCS) 。

10.8.5. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。



重要

如果您的集群无法直接访问互联网，则可以在置备的某些类型的基础架构上执行受限网络安装。在此过程中，您可以下载所需的内容，并使用它为镜像 registry 填充安装软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群前，您要更新镜像 registry 的内容。

10.8.6. 为集群节点 SSH 访问生成密钥对

在 OpenShift Container Platform 安装过程中，您可以为安装程序提供 SSH 公钥。密钥通过它们的 Ignition 配置文件传递给 Red Hat Enterprise Linux CoreOS(RHCOS)节点，用于验证对节点的 SSH 访问。密钥添加到每个节点上 **core** 用户的 **~/.ssh/authorized_keys** 列表中，这将启用免密码身份验证。

将密钥传递给节点后，您可以使用密钥对作为用户 **核心** 通过 SSH 连接到 RHCOS 节点。若要通过 SSH 访问节点，必须由 SSH 为您的本地用户管理私钥身份。

如果要通过 SSH 连接到集群节点来执行安装调试或灾难恢复，则必须在安装过程中提供 SSH 公钥。**./openshift-install gather** 命令还需要在集群节点上设置 SSH 公钥。

重要

不要在生产环境中跳过这个过程，在生产环境中需要灾难恢复和调试。

注意

您必须使用本地密钥，而不是使用特定平台方法配置 的密钥，如 AWS 密钥对。

流程

1. 如果您在本地计算机上没有可用于在集群节点上进行身份验证的现有 SSH 密钥对，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_ed25519**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

注意

如果您计划在 **x86_64**、**ppc64le** 和 **s390x** 架构上安装使用 RHEL 加密库（这些加密库已提交给 NIST 用于 FIPS 140-2/140-3 验证）的 OpenShift Container Platform 集群，则不要创建使用 **ed25519** 算法的密钥。相反，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

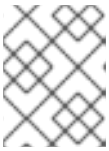
2. 查看公共 SSH 密钥：

```
$ cat <path>/<file_name>.pub
```

例如，运行以下命令来查看 **~/.ssh/id_ed25519.pub** 公钥：

```
$ cat ~/.ssh/id_ed25519.pub
```

3. 将 SSH 私钥身份添加到本地用户的 SSH 代理（如果尚未添加）。在集群节点上，或者要使用 **./openshift-install gather** 命令，需要对该密钥进行 SSH 代理管理，才能在集群节点上进行免密码 SSH 身份验证。



注意

在某些发行版中，自动管理默认 SSH 私钥身份，如 **~/.ssh/id_rsa** 和 **~/.ssh/id_dsa**。

a. 如果 **ssh-agent** 进程尚未为您的本地用户运行，请将其作为后台任务启动：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果集群处于 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

4. 将 SSH 私钥添加到 **ssh-agent** ：

```
$ ssh-add <path>/<file_name> 1
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_ed25519.pub**

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

后续步骤

- 安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

10.8.7. 获取安装程序

在安装 OpenShift Container Platform 前，将安装文件下载到云网络上的堡垒主机或可通过 VPN 访问网络的机器中。

有关私有集群安装要求的更多信息，请参阅 "Private cluster"。

先决条件

- 您有一个运行 Linux 的机器，如 Red Hat Enterprise Linux 8，本地磁盘空间为 500 MB。

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐户，请使用您的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入到安装类型的页面，下载与您的主机操作系统和架构对应的安装程序，并将该文件放在您要存储安装配置文件的目录中。

重要

安装程序会在用来安装集群的计算机上创建几个文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。

重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，请为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。此 pull secret 允许您与所含授权机构提供的服务进行身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

10.8.8. 导出 API 密钥

您必须将您创建的 API 密钥设置为全局变量；安装程序会在启动期间设置 API 密钥。

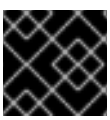
先决条件

- 您已为 IBM Cloud® 帐户创建了用户 API 密钥或服务 ID API 密钥。

流程

- 将帐户的 API 密钥导出为全局变量：

```
$ export IC_API_KEY=<api_key>
```



重要

您必须按照指定方式设置变量名称，安装程序需要在启动期间存在变量名称。

10.8.9. 手动创建安装配置文件

安装集群要求您手动创建安装配置文件。

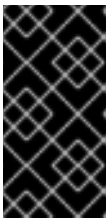
先决条件

- 您在本地机器上有一个 SSH 公钥来提供给安装程序。该密钥将用于在集群节点上进行 SSH 身份验证，以进行调试和灾难恢复。
- 已获取 OpenShift Container Platform 安装程序和集群的 pull secret。

流程

1. 创建一个安装目录来存储所需的安装资产：

```
$ mkdir <installation_directory>
```



重要

您必须创建一个目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

2. 自定义提供的 **install-config.yaml** 文件模板示例，并将其保存在 **<installation_directory>** 中。

注意

此配置文件必须命名为 **install-config.yaml**。

3. 备份 **install-config.yaml** 文件，以便您可以使用它安装多个集群。

重要

install-config.yaml 文件会在安装过程的下一步中使用。现在必须备份它。

其他资源

- IBM Cloud® 的安装配置参数。

10.8.9.1. 集群安装的最低资源要求

每台集群机器都必须满足以下最低要求：

表 10.17. 最低资源要求

机器	操作系统	vCPU	虚拟内存	Storage	每秒输入/输出 (IOPS)
bootstrap	RHCOS	4	16 GB	100 GB	300

机器	操作系统	vCPU	虚拟内存	Storage	每秒输入/输出 (IOPS)
Control plane（控制平面）	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS	2	8 GB	100 GB	300

注意

从 OpenShift Container Platform 版本 4.13 开始，RHCOS 基于 RHEL 版本 9.2，它更新了微架构要求。以下列表包含每个架构需要的最小指令集架构 (ISA)：

- x86-64 体系结构需要 x86-64-v2 ISA
- ARM64 架构需要 ARMv8.0-A ISA
- IBM Power 架构需要 Power 9 ISA
- s390x 架构需要 z14 ISA

如需更多信息，请参阅 RHEL 架构。

如果平台的实例类型满足集群机器的最低要求，则 OpenShift Container Platform 支持使用它。

其他资源

- 优化存储

10.8.9.2. 为 IBM Cloud 测试的实例类型

以下 IBM Cloud® 实例类型已使用 OpenShift Container Platform 测试。

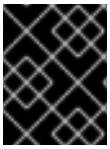
例 10.4. 机器系列

- **c4.***
- **c5.***
- **c5a.***
- **i3.***
- **m4.***
- **m5.***
- **m5a.***
- **m6a.***
- **m6i.***

- **r4.***
- **r5.***
- **r5a.***
- **r6i.***
- **t3.***
- **t3a.***

10.8.9.3. IBM Cloud 的自定义 install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多详情，或修改所需参数的值。



重要

此示例 YAML 文件仅供参考。您必须使用安装程序来获取 **install-config.yaml** 文件，然后修改该文件。

```
apiVersion: v1
baseDomain: example.com (1)
controlPlane: (2) (3)
  hyperthreading: Enabled (4)
  name: master
  platform:
    ibmcloud: {}
  replicas: 3
compute: (5) (6)
- hyperthreading: Enabled (7)
  name: worker
  platform:
    ibmcloud: {}
  replicas: 3
metadata:
  name: test-cluster (8)
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14 (9)
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16 (10)
  networkType: OVNKubernetes (11)
  serviceNetwork:
  - 172.30.0.0/16
platform:
  ibmcloud:
    region: eu-gb (12)
    resourceGroupName: eu-gb-example-cluster-rg (13)
    networkResourceGroupName: eu-gb-example-existing-network-rg (14)
```

```
    vpcName: eu-gb-example-network-1 15
    controlPlaneSubnets: 16
      - eu-gb-example-network-1-cp-eu-gb-1
      - eu-gb-example-network-1-cp-eu-gb-2
      - eu-gb-example-network-1-cp-eu-gb-3
    computeSubnets: 17
      - eu-gb-example-network-1-compute-eu-gb-1
      - eu-gb-example-network-1-compute-eu-gb-2
      - eu-gb-example-network-1-compute-eu-gb-3
credentialsMode: Manual
publish: Internal 18
pullSecret: '{"auths": ...}' 19
fips: false 20
sshKey: ssh-ed25519 AAAA... 21
```

1 8 12 19 必需。

2 5 如果没有提供这些参数和值，安装程序会提供默认值。

3 6 **controlPlane** 部分是一个单个映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，controlPlane **部分** 的第一行则不以连字符开头。仅使用一个 control plane 池。

4 7 启用或禁用并发多线程，也称为 Hyper-Threading。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将 参数值设置为 **Disabled** 来禁用它。如果在某些集群机器中禁用并发多线程，则必须在所有集群机器中禁用它。

重要

如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。如果您禁用并发多线程，请为您的机器使用较大的类型，如 **n1-standard-8**。

9 机器 CIDR 必须包含计算机器和 control plane 机器的子网。

10 CIDR 必须包含 **platform.ibmcloud.controlPlaneSubnets** 和 **platform.ibmcloud.computeSubnets** 中定义的子网。

11 要安装的集群网络插件。默认值 **OVNKubernetes** 是唯一支持的值。

13 现有资源组的名称。所有安装程序置备的集群资源都部署到此资源组中。如果未定义，则会为集群创建新的资源组。

14 指定包含现有虚拟私有云 (VPC) 的资源组名称。现有的 VPC 和子网应该位于此资源组中。集群将安装到此 VPC 中。

15 指定现有 VPC 的名称。

16 指定要部署 control plane 机器的现有子网的名称。子网必须属于您指定的 VPC。为区域中的每个可用区指定一个子网。

17 指定要部署计算机器的现有子网的名称。子网必须属于您指定的 VPC。为区域中的每个可用区指定一个子网。

18 如何发布集群的面向用户的端点。将 **publish** 设置为 **Internal** 以部署私有集群。默认值为 **External**。

20 启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes

重要

当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS)时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。

21 可选：提供用于访问集群中机器的 **sshKey** 值。

注意

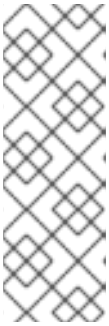
对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

10.8.9.4. 在安装过程中配置集群范围的代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 **Proxy** 对象的 **spec.noProxy** 字段中添加站点来绕过代理。



注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，**Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
```

```
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle>
```
(5)

1. 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 **.** 以仅匹配子域。例如，**.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代理。
4. 如果提供，安装程序会在 **openshift-config** 命名空间中生成名为 **user-ca-bundle** 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 **trusted-ca-bundle** 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， **Proxy** 对象的 **trustedCA** 字段中也会引用此配置映射。**additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。
5. 可选：决定 **Proxy** 对象的配置以引用 **trustedCA** 字段中 **user-ca-bundle** 配置映射的策略。允许的值是 **Proxyonly** 和 **Always**。仅在配置了 **http/https** 代理时，使用 **Proxyonly** 引用 **user-ca-bundle** 配置映射。使用 **Always** 始终引用 **user-ca-bundle** 配置映射。默认值为 **Proxyonly**。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。

注意

如果安装程序超时，重启并使用安装程序的 **wait-for** 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. 保存该文件并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 cluster 的集群范围代理，该代理 **使用** 提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

10.8.10. 手动创建 IAM

安装集群需要 Cloud Credential Operator(CCO)以手动模式运行。虽然安装程序为手动模式配置 CCO，但您必须为云供应商指定身份和访问管理 secret。

您可以使用 Cloud Credential Operator (CCO)实用程序(**ccoctl**)创建所需的 IBM Cloud® 资源。

先决条件

- 您已配置了 **ccoctl** 二进制文件。

- 您有一个现有的 **install-config.yaml** 文件。

流程

1. 编辑 **install-config.yaml** 配置文件，使其包含将 **credentialsMode** 参数设置为 **Manual**。

   install-config.yaml 配置文件示例

   ```
   apiVersion: v1
   baseDomain: cluster1.example.com
   credentialsMode: Manual (1)
   compute:
   - architecture: amd64
     hyperthreading: Enabled
   ```

   (1) 添加这一行将 **credentialsMode** 参数设置为 **Manual**。

2. 要生成清单，请在包含安装程序的目录中运行以下命令：

   ```
   $ openshift-install create manifests --dir <installation_directory>
   ```

3. 在包含安装程序的目录中，运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

   ```
   $ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
   ```

4. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 自定义资源 (CR) 列表：

   ```
   $ oc adm release extract \
     --from=$RELEASE_IMAGE \
     --credentials-requests \
     --included \(1)
     --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \(2)
     --to=<path_to_directory_for_credentials_requests> (3)
   ```

   (1) **--included** 参数仅包含特定集群配置所需的清单。

   (2) 指定 **install-config.yaml** 文件的位置。

   (3) 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

   此命令为每个 **CredentialsRequest** 对象创建一个 YAML 文件。

   CredentialsRequest 对象示例

   ```
   apiVersion: cloudcredential.openshift.io/v1
   kind: CredentialsRequest
   metadata:
     labels:
       controller-tools.k8s.io: "1.0"
   ```

```
  name: openshift-image-registry-ibmcos
  namespace: openshift-cloud-credential-operator
spec:
  secretRef:
    name: installer-cloud-credentials
    namespace: openshift-image-registry
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: IBMCloudProviderSpec
    policies:
    - attributes:
      - name: serviceName
        value: cloud-object-storage
      roles:
      - crn:v1:bluemix:public:iam::::role:Viewer
      - crn:v1:bluemix:public:iam::::role:Operator
      - crn:v1:bluemix:public:iam::::role:Editor
      - crn:v1:bluemix:public:iam::::serviceRole:Reader
      - crn:v1:bluemix:public:iam::::serviceRole:Writer
    - attributes:
      - name: resourceType
        value: resource-group
      roles:
      - crn:v1:bluemix:public:iam::::role:Viewer
```

5. 为每个凭证请求创建服务 ID，分配定义的策略、创建 API 密钥并生成 secret：

```
$ ccoctl ibmcloud create-service-id \
  --credentials-requests-dir=<path_to_credential_requests_directory> \ 1
  --name=<cluster_name> \ 2
  --output-dir=<installation_directory> \ 3
  --resource-group-name=<resource_group_name> 4
```

1. 指定包含组件 **CredentialsRequest** 对象文件的目录。
2. 指定 OpenShift Container Platform 集群的名称。
3. 可选：指定您希望 **ccoctl** 实用程序在其中创建对象的目录。默认情况下，实用程序在运行命令的目录中创建对象。
4. 可选：指定用于限制访问策略的资源组的名称。

注意

如果您的集群使用 **TechPreviewNoUpgrade** 功能集启用的技术预览功能，则必须包含 **--enable-tech-preview** 参数。

如果提供了不正确的资源组名称，安装会在 bootstrap 阶段失败。要查找正确的资源组名称，请运行以下命令：

```
$ grep resourceGroupName <installation_directory>/manifests/cluster-
infrastructure-02-config.yml
```

验证

- 确保在集群的 **manifests** 目录中生成了适当的 secret。

10.8.11. 部署集群

您可以在兼容云平台上安装 OpenShift Container Platform。

重要

在初始安装过程中，您只能运行安装程序的 **create cluster** 命令一次。

先决条件

- 您已使用托管集群的云平台配置了帐户。
- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。
- 已确认主机上的云供应商帐户具有部署集群的正确权限。权限不正确的帐户会导致安装过程失败，并显示包括缺失权限的错误消息。

流程

- 进入包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定自定义 **./install-config.yaml** 文件的位置。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不是 **info**。

验证

当集群部署成功完成时：

- 终端会显示用于访问集群的说明，包括指向 Web 控制台和 **kubeadmin** 用户的凭证的链接。
- 凭证信息还会输出到 **<installation_directory>/.openshift_install.log**.

重要

不要删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
```

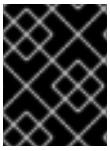
```
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

10.8.12. 安装 OpenShift CLI

您可以安装 OpenShift CLI(**oc**)来使用命令行界面与 OpenShift Container Platform 进行交互。您可以在 Linux、Windows 或 macOS 上安装 **oc**。



重要

如果安装了旧版本的 **oc**，则可能无法使用 OpenShift Container Platform 4.16 中的所有命令。下载并安装新版本的 **oc**。

在 Linux 上安装 OpenShift CLI
您可以按照以下流程在 Linux 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **产品变体** 下拉列表中选择架构。
3. 从 **版本** 下拉列表中选择适当的版本。
4. 点 **OpenShift v4.16 Linux Client** 条目旁的 **Download Now** 来保存文件。
5. 解包存档：

```
$ tar xvf <file>
```

6. 将 **oc** 二进制文件放到 **PATH** 中的目录中。
   要查看您的 **PATH**，请执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 Windows Client** 条目旁的 **Download Now** 来保存文件。
4. 使用 ZIP 程序解压存档。
5. 将 **oc** 二进制文件移到 **PATH 中的目录中**。
   要查看您的 **PATH**，请打开命令提示并执行以下命令：

```
C:\> path
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 macOS Client** 条目旁的 **Download Now** 来保存文件。

注意

对于 macOS arm64，请选择 **OpenShift v4.16 macOS arm64 Client** 条目。

4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
   要查看您的 **PATH**，请打开终端并执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

10.8.13. 使用 CLI 登录集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

其他资源

- 访问Web控制台

10.8.14. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 关于远程健康监控

10.8.15. 后续步骤

- 自定义集群。
- 如果需要，您可以选择 不使用远程健康报告。

10.9. 在受限网络中的 IBM CLOUD 上安装集群

在 OpenShift Container Platform 4.16 中，您可以通过创建安装发行内容的内部镜像在受限网络中的集群，方法是创建一个可访问 IBM Cloud® 上现有 Virtual Private Cloud (VPC) 的安装发行内容的内部镜像。

10.9.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 已将 IBM Cloud 帐户配置为托管集群。
- 您有一个容器镜像 registry，可供互联网和受限网络访问。容器镜像 registry 应该镜像 OpenShift 镜像 registry 的内容，并包含安装介质。如需更多信息，请参阅使用 oc-mirror 插件为断开连接的安装镜像镜像。
- 在 IBM Cloud® 上有一个满足以下要求的 VPC：
  - VPC 包含镜像 registry 或具有防火墙规则或对等连接来访问其他位置托管的镜像 registry。
  - VPC 可以使用公共端点访问 IBM Cloud® 服务端点。如果网络限制限制对公共服务端点的访问，请评估这些服务中可能可用的备用端点。如需更多信息，请参阅访问 IBM 服务端点。

  默认情况下，您无法使用安装程序置备的 VPC。
- 如果您计划将端点网关配置为使用 IBM Cloud® Virtual Private Endpoints，请考虑以下要求：
  - 端点网关支持目前仅限于 **us-east** 和 **us-south** 区域。
  - VPC 必须允许到端点网关的流量。您可以使用 VPC 的默认安全组或新安全组来允许端口 443 上的流量。如需更多信息，请参阅允许端点网关流量。
- 如果使用防火墙，则会 将其配置为允许集群需要访问的站点。
- 在安装集群前已经配置了 **ccoctl** 工具。如需更多信息，请参阅为 IBM Cloud VPC 配置 IAM。

10.9.2. 关于在受限网络中安装

在 OpenShift Container Platform 4.16 中，可以执行不需要有效的互联网连接来获取软件组件的安装。受限网络安装可以使用安装程序置备的基础架构或用户置备的基础架构完成，具体取决于您要安装集群的云平台。

10.9.2.1. 所需的互联网访问和安装主机

您可以使用堡垒主机或可移植设备完成安装，该设备可同时访问互联网和您的关闭网络。您必须使用可访问互联网的主机：

- 下载安装程序、OpenShift CLI (**oc**) 和 CCO 实用程序 (**ccoctl**)。
- 使用安装程序查找 Red Hat Enterprise Linux CoreOS (RHCOS)镜像并创建安装配置文件。
- 使用 **oc** 从 CCO 容器镜像中提取 **ccoctl**。
- 使用 **oc** 和 **ccoctl** 为 IBM Cloud® 配置 IAM。

10.9.2.2. 访问镜像 registry

要完成受限网络安装，您必须创建一个 registry，以镜像 OpenShift 镜像 registry 的内容并包含安装介质。

您可以在镜像主机上创建此 registry，该主机可同时访问互联网和受限网络，也可以使用满足您机构安全限制的其他方法。

有关为断开连接的安装镜像镜像的更多信息，请参阅"添加资源"。

10.9.2.3. 访问 IBM 服务端点

安装程序需要访问以下 IBM Cloud® 服务端点：

- 云对象存储
- DNS 服务
- 全局搜索
- 全局标记
- 身份服务
- 资源控制器
- 资源管理器
- VPC

注意

如果您要为 IBM Cloud® root 密钥指定 IBM® 密钥作为安装过程的一部分，还需要密钥保护的服务端点。

默认情况下，公共端点用于访问该服务。如果网络限制限制对公共服务端点的访问，您可以覆盖默认行为。

在部署集群前，您可以更新安装配置文件 (**install-config.yaml**) 来指定备用服务端点的 URI。有关用法的更多信息，请参阅"附加资源"。

10.9.2.4. 其他限制

受限网络中的集群有以下额外限制和限制：

- **ClusterVersion** 状态包含一个 **Unable to retrieve available updates** 错误。
- 默认情况下，您无法使用 Developer Catalog 的内容，因为您无法访问所需的镜像流标签。

其他资源

- 使用 oc-mirror 插件为断开连接的安装镜像镜像
- 其他 IBM Cloud 配置参数

10.9.3. 关于使用自定义 VPC

在 OpenShift Container Platform 4.16 中，您可以将集群部署到现有 IBM® Virtual Private Cloud (VPC)的子网。将 OpenShift Container Platform 部署到现有的 VPC 中可帮助您避免新帐户中的限制，或者更容易地利用公司所设置的操作限制。如果您无法获得您自己创建 VPC 所需的基础架构创建权限，请使用这个安装选项。

因为安装程序不知道现有子网中的其他组件，所以无法选择子网 CIDR 等。您必须为安装集群的子网配置网络。

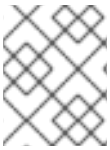
10.9.3.1. 使用 VPC 的要求

您必须在安装集群前正确配置现有的 VPC 及其子网。安装程序不会创建以下组件：

- NAT 网关
- 子网
- 路由表
- VPC 网络

安装程序无法：

- 从属网络范围供集群使用
- 为子网设置路由表
- 设置 VPC 选项，如 DHCP



注意

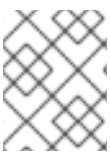
安装程序要求您使用由云提供的 DNS 服务器。不支持使用自定义 DNS 服务器，并导致安装失败。

10.9.3.2. VPC 验证

VPC 和所有子网都必须位于现有资源组中。集群部署到现有的 VPC 中。

作为安装的一部分，在 **install-config.yaml** 文件中指定以下内容：

- 包含 VPC 和子网的现有资源组的名称 (**networkResourceGroupName**)
- 现有 VPC 的名称(**vpcName**)
- 为 control plane 机器和计算机器创建的子网 (**controlPlaneSubnets** 和 **computeSubnets**)



注意

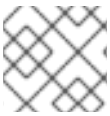
额外的安装程序置备的集群资源被部署到单独的资源组 (**resourceGroupName**) 中。您可在安装集群前指定此资源组。如果未定义，则会为集群创建新的资源组。

要确保您提供的子网适合，安装程序会确认以下内容：

- 您指定的所有子网都存在。
- 对于区域中的每个可用区，您可以指定：

- 一个用于 control plane 机器的子网。
- 计算机器的一个子网。

- 您指定的机器 CIDR 包含计算机器和 control plane 机器的子网。



注意

不支持子网 ID。

10.9.3.3. 集群间隔离

如果您将 OpenShift Container Platform 部署到现有网络中，集群服务的隔离将在以下方面减少：

- 您可以在同一 VPC 中安装多个 OpenShift Container Platform 集群。
- 整个网络允许 ICMP 入站流量。
- 整个网络都允许 TCP 端口 22 入站流量 (SSH)。
- 整个网络都允许 control plane TCP 6443 入站流量 (Kubernetes API)。
- 整个网络都允许 control plane TCP 22623 入站流量 (MCS) 。

10.9.3.4. 允许端点网关流量

如果使用 IBM Cloud® Virtual Private 端点，则必须将您的 Virtual Private Cloud (VPC) 配置为允许到端点网关的流量。

VPC 的默认安全组配置为允许所有出站流量到端点网关。因此，允许 VPC 和端点网关之间的流量的最简单方法是修改默认安全组以允许端口 443 上的入站流量。

注意

如果您选择配置新的安全组，则必须配置安全组以允许入站和出站流量。

先决条件

- 已安装 IBM Cloud® 命令行界面实用程序 (**ibmcloud**)。

流程

1. 运行以下命令，获取默认安全组的标识符：

```
$ DEFAULT_SG=$(ibmcloud is vpc <your_vpc_name> --output JSON | jq -r
'.default_security_group.id')
```

2. 运行以下命令，添加允许端口 443 上入站流量的规则：

```
$ ibmcloud is security-group-rule-add $DEFAULT_SG inbound tcp --remote 0.0.0.0/0 --port-
min 443 --port-max 443
```

注意

确保您的端点网关已配置为使用此安全组。

10.9.4. 为集群节点 SSH 访问生成密钥对

在 OpenShift Container Platform 安装过程中，您可以为安装程序提供 SSH 公钥。密钥通过它们的 Ignition 配置文件传递给 Red Hat Enterprise Linux CoreOS(RHCOS)节点，用于验证对节点的 SSH 访问。密钥添加到每个节点上 **core** 用户的 **~/.ssh/authorized_keys** 列表中，这将启用免密码身份验证。

将密钥传递给节点后，您可以使用密钥对作为用户 **核心** 通过 SSH 连接到 RHCOS 节点。若要通过 SSH 访问节点，必须由 SSH 为您的本地用户管理私钥身份。

如果要通过 SSH 连接到集群节点来执行安装调试或灾难恢复，则必须在安装过程中提供 SSH 公钥。**./openshift-install gather** 命令还需要在集群节点上设置 SSH 公钥。

重要

不要在生产环境中跳过这个过程，在生产环境中需要灾难恢复和调试。

注意

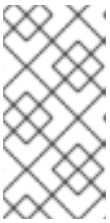
您必须使用本地密钥，而不是使用特定平台方法配置 的密钥，如 AWS 密钥对。

流程

1. 如果您在本地计算机上没有可用于在集群节点上进行身份验证的现有 SSH 密钥对，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_ed25519**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。



注意

如果您计划在 **x86_64**、**ppc64le** 和 **s390x** 架构上安装使用 RHEL 加密库（这些加密库已提交给 NIST 用于 FIPS 140-2/140-3 验证）的 OpenShift Container Platform 集群，则不要创建使用 **ed25519** 算法的密钥。相反，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

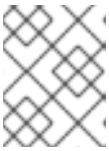
2. 查看公共 SSH 密钥：

```
$ cat <path>/<file_name>.pub
```

例如，运行以下命令来查看 **~/.ssh/id_ed25519.pub** 公钥：

```
$ cat ~/.ssh/id_ed25519.pub
```

3. 将 SSH 私钥身份添加到本地用户的 SSH 代理（如果尚未添加）。在集群节点上，或者要使用 **./openshift-install gather** 命令，需要对该密钥进行 SSH 代理管理，才能在集群节点上进行免密码 SSH 身份验证。



注意

在某些发行版中，自动管理默认 SSH 私钥身份，如 ~/**.ssh/id_rsa** 和 ~/**.ssh/id_dsa**。

a. 如果 **ssh-agent** 进程尚未为您的本地用户运行，请将其作为后台任务启动：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果集群处于 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

4. 将 SSH 私钥添加到 **ssh-agent** ：

```
$ ssh-add <path>/<file_name> 1
```

1 指定 SSH 私钥的路径和文件名，如 ~/**.ssh/id_ed25519.pub**

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

后续步骤

- 安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

10.9.5. 导出 API 密钥

您必须将您创建的 API 密钥设置为全局变量；安装程序会在启动期间设置 API 密钥。

先决条件

- 您已为 IBM Cloud® 帐户创建了用户 API 密钥或服务 ID API 密钥。

流程

- 将帐户的 API 密钥导出为全局变量：

```
$ export IC_API_KEY=<api_key>
```

重要

您必须按照指定方式设置变量名称，安装程序需要在启动期间存在变量名称。

10.9.6. 下载 RHCOS 集群镜像

安装程序需要 Red Hat Enterprise Linux CoreOS (RHCOS) 镜像来安装集群。在可选时，在部署前下载 Red Hat Enterprise Linux CoreOS (RHCOS) 会删除创建集群时访问互联网的需要。

使用安装程序查找并下载 Red Hat Enterprise Linux CoreOS (RHCOS) 镜像。

先决条件

- 运行安装程序的主机可以访问互联网。

流程

1. 进入包含安装程序的目录并运行以下命令：

```
$ ./openshift-install coreos print-stream-json
```

2. 使用命令的输出来查找 IBM Cloud® 镜像的位置。

```
.Example output
----
  "release": "415.92.202311241643-0",
  "formats": {
    "qcow2.gz": {
      "disk": {
        "location": "https://rhcos.mirror.openshift.com/art/storage/prod/streams/4.15-
9.2/builds/415.92.202311241643-0/x86_64/rhcos-415.92.202311241643-0-
ibmcloud.x86_64.qcow2.gz",
        "sha256":
"6b562dee8431bec3b93adeac1cfefcd5e812d41e3b7d78d3e28319870ffc9eae",
        "uncompressed-sha256":
"5a0f9479505e525a30367b6a6a6547c86a8f03136f453c1da035f3aa5daa8bc9"
----
```

3. 下载并提取镜像存档。使镜像在安装程序用来创建集群的主机上可用。

10.9.7. 手动创建安装配置文件

安装集群要求您手动创建安装配置文件。

先决条件

- 已获取 OpenShift Container Platform 安装程序和集群的 pull secret。
- 具有在镜像 registry 时创建的 **imageContentSourcePolicy.yaml** 文件。
- 您已获取了镜像 registry 的证书内容。

流程

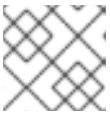
1. 创建一个安装目录来存储所需的安装资产：

```
$ mkdir <installation_directory>
```

重要

您必须创建一个目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

2. 自定义提供的 **install-config.yaml** 文件模板示例，并将其保存在 **<installation_directory>** 中。



注意

此配置文件必须命名为 **install-config.yaml**。

在自定义示例模板时，请确定提供在受限网络中安装所需的信息：

a. 更新 **pullSecret** 值，使其包含 registry 的身份验证信息：

```
pullSecret: '{"auths":{"<mirror_host_name>:5000": {"auth": "<credentials>","email":
"you@example.com"}}}'
```

对于 **<mirror_host_name>**，请指定 您在镜像 registry 证书中指定的 registry 域名 ；对于 **<credentials>**， 请指定您的镜像 registry 的 base64 编码用户名和密码。

b. 添加 **additionalTrustBundle** 参数和值。

```
additionalTrustBundle: |
  -----BEGIN CERTIFICATE-----

ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
  -----END CERTIFICATE-----
```

该值必须是您用于镜像 registry 的证书文件内容。证书文件可以是现有的可信证书颁发机构，也可以是您为镜像 registry 生成的自签名证书。

c. 定义 VPC 的网络和子网，以便在父 **platform.ibmcloud** 字段下安装集群：

```
vpcName: <existing_vpc>
controlPlaneSubnets: <control_plane_subnet>
computeSubnets: <compute_subnet>
```

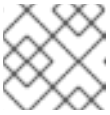
对于 **platform.ibmcloud.vpcName**，请指定现有 IBM Cloud VPC 的名称。对于 **platform.ibmcloud.controlPlaneSubnets** 和 **platform.ibmcloud.computeSubnets**，请分别指定要分别部署 control plane 机器和计算机器的现有子网。

d. 添加镜像内容资源，类似于以下 YAML 摘录：

```
imageContentSources:
- mirrors:
  - <mirror_host_name>:5000/<repo_name>/release
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <mirror_host_name>:5000/<repo_name>/release
  source: registry.redhat.io/ocp/release
```

对于这些值，请使用镜像 registry 时创建的 **imageContentSourcePolicy.yaml** 文件。

e. 如果网络限制使用公共端点访问所需的 IBM Cloud® 服务，请将 **serviceEndpoints** 小节添加到 **platform.ibmcloud** 以指定备用服务端点。



注意

您只能为每个服务指定一个备用服务端点。

使用备用服务端点的示例

```
# ...
serviceEndpoints:
  - name: IAM
    url: <iam_alternate_endpoint_url>
  - name: VPC
    url: <vpc_alternate_endpoint_url>
  - name: ResourceController
    url: <resource_controller_alternate_endpoint_url>
  - name: ResourceManager
    url: <resource_manager_alternate_endpoint_url>
  - name: DNSServices
    url: <dns_services_alternate_endpoint_url>
  - name: COS
    url: <cos_alternate_endpoint_url>
  - name: GlobalSearch
    url: <global_search_alternate_endpoint_url>
  - name: GlobalTagging
    url: <global_tagging_alternate_endpoint_url>
# ...
```

f. 可选：将发布策略设置为 **Internal**：

```
publish: Internal
```

通过设置这个选项，您可以创建一个内部 Ingress Controller 和一个私有负载均衡器。

注意

如果您使用 **External** 的默认值，您的网络必须能够访问 IBM Cloud® Internet Services (CIS) 的公共端点。对于虚拟私有端点，未启用 CIS。

3. 备份 **install-config.yaml** 文件，以便您可以使用它安装多个集群。

重要

install-config.yaml 文件会在安装过程的下一步中使用。现在必须备份它。

10.9.7.1. 在安装过程中配置集群范围的代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 **Proxy** 对象的 **spec.noProxy** 字段中添加站点来绕过代理。

注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，**Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 **.** 以仅匹配子域。例如，**.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代理。
4. 如果提供，安装程序会在 **openshift-config** 命名空间中生成名为 **user-ca-bundle** 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 **trusted-ca-bundle** 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， **Proxy** 对象的 **trustedCA** 字段中也会引用此配置映射。**additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。
5. 可选：决定 **Proxy** 对象的配置以引用 **trustedCA** 字段中 **user-ca-bundle** 配置映射的策略。允许的值是 **Proxyonly** 和 **Always**。仅在配置了 **http/https** 代理时，使用 **Proxyonly** 引用 **user-ca-bundle** 配置映射。使用 **Always** 始终引用 **user-ca-bundle** 配置映射。默认值为 **Proxyonly**。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。

注意

如果安装程序超时，重启并使用安装程序的 **wait-for** 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. 保存该文件并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 cluster 的集群范围代理，该代理 **使用** 提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

其他资源

- IBM Cloud® 的安装配置参数。

10.9.7.2. 集群安装的最低资源要求

每台集群机器都必须满足以下最低要求：

表 10.18. 最低资源要求

机器	操作系统	vCPU	虚拟内存	Storage	每秒输入/输出 (IOPS)
bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane（控制平面）	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS	2	8 GB	100 GB	300

注意

从 OpenShift Container Platform 版本 4.13 开始，RHCOS 基于 RHEL 版本 9.2，它更新了微架构要求。以下列表包含每个架构需要的最小指令集架构 (ISA)：

- x86-64 体系结构需要 x86-64-v2 ISA
- ARM64 架构需要 ARMv8.0-A ISA
- IBM Power 架构需要 Power 9 ISA
- s390x 架构需要 z14 ISA

如需更多信息，请参阅 RHEL 架构。

如果平台的实例类型满足集群机器的最低要求，则 OpenShift Container Platform 支持使用它。

10.9.7.3. 为 IBM Cloud 测试的实例类型

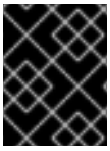
以下 IBM Cloud® 实例类型已使用 OpenShift Container Platform 测试。

例 10.5. 机器系列

- **c4.***
- **c5.***
- **c5a.***
- **i3.***
- **m4.***
- **m5.***
- **m5a.***
- **m6a.***
- **m6i.***
- **r4.***
- **r5.***
- **r5a.***
- **r6i.***
- **t3.***
- **t3a.***

10.9.7.4. IBM Cloud 的自定义 install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多详情，或修改所需参数的值。



重要

此示例 YAML 文件仅供参考。您必须使用安装程序来获取 **install-config.yaml** 文件，然后修改该文件。

```
apiVersion: v1
baseDomain: example.com (1)
controlPlane: (2) (3)
  hyperthreading: Enabled (4)
  name: master
  platform:
    ibm-cloud: {}
  replicas: 3
compute: (5) (6)
- hyperthreading: Enabled (7)
  name: worker
  platform:
    ibmcloud: {}
  replicas: 3
metadata:
  name: test-cluster (8)
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14 (9)
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16 (10)
  networkType: OVNKubernetes (11)
  serviceNetwork:
  - 172.30.0.0/16
platform:
  ibmcloud:
    region: us-east (12)
    resourceGroupName: us-east-example-cluster-rg (13)
    serviceEndpoints: (14)
      - name: IAM
        url: https://private.us-east.iam.cloud.ibm.com
      - name: VPC
        url: https://us-east.private.iaas.cloud.ibm.com/v1
      - name: ResourceController
        url: https://private.us-east.resource-controller.cloud.ibm.com
      - name: ResourceManager
        url: https://private.us-east.resource-controller.cloud.ibm.com
      - name: DNSServices
        url: https://api.private.dns-svcs.cloud.ibm.com/v1
      - name: COS
        url: https://s3.direct.us-east.cloud-object-storage.appdomain.cloud
      - name: GlobalSearch
        url: https://api.private.global-search-tagging.cloud.ibm.com
      - name: GlobalTagging
```

```
      url: https://tags.private.global-search-tagging.cloud.ibm.com
    networkResourceGroupName: us-east-example-existing-network-rg 15
    vpcName: us-east-example-network-1 16
    controlPlaneSubnets: 17
      - us-east-example-network-1-cp-us-east-1
      - us-east-example-network-1-cp-us-east-2
      - us-east-example-network-1-cp-us-east-3
    computeSubnets: 18
      - us-east-example-network-1-compute-us-east-1
      - us-east-example-network-1-compute-us-east-2
      - us-east-example-network-1-compute-us-east-3
credentialsMode: Manual
pullSecret: '{"auths":{"<local_registry>": {"auth": "<credentials>","email": "you@example.com"}}}' 19
fips: false 20
sshKey: ssh-ed25519 AAAA... 21
additionalTrustBundle: | 22
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
imageContentSources: 23
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-v4.0-art-dev
```

1 8 12 必需。

2 5 如果没有提供这些参数和值，安装程序会提供默认值。

3 6 **controlPlane** 部分是一个单个映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，controlPlane **部分** 的第一行则不以连字符开头。仅使用一个 control plane 池。

4 7 启用或禁用并发多线程，也称为 Hyper-Threading。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将 参数值设置为 **Disabled** 来禁用它。如果在某些集群机器中禁用并发多线程，则必须在所有集群机器中禁用它。

重要

如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。如果您禁用并发多线程，请为您的机器使用较大的类型，如 **n1-standard-8**。

9 机器 CIDR 必须包含计算机器和 control plane 机器的子网。

10 CIDR 必须包含 **platform.ibmcloud.controlPlaneSubnets** 和 **platform.ibmcloud.computeSubnets** 中定义的子网。

11 要安装的集群网络插件。默认值 **OVNKubernetes** 是唯一支持的值。

13 现有资源组的名称。所有安装程序置备的集群资源都部署到此资源组中。如果未定义，则会为集群创建新的资源组。

14 根据 VPC 的网络限制，根据需要指定备用服务端点。这会覆盖该服务的默认公共端点。

15 指定包含现有虚拟私有云 (VPC) 的资源组名称。现有的 VPC 和子网应该位于此资源组中。集群将安装到此 VPC 中。

16 指定现有 VPC 的名称。

17 指定要部署 control plane 机器的现有子网的名称。子网必须属于您指定的 VPC。为区域中的每个可用区指定一个子网。

18 指定要部署计算机器的现有子网的名称。子网必须属于您指定的 VPC。为区域中的每个可用区指定一个子网。

19 对于 **<local_registry>**，请指定 registry 域名，以及您的镜像 registry 用来提供内容的可选端口。例如： registry.example.com 或 registry.example.com:5000。对于 **<credentials>**，请为您的镜像 registry 指定 base64 编码的用户名和密码。

20 启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

只有在 **x86_64** 架构的 OpenShift Container Platform 部署中才支持使用 FIPS 验证或Modules in Process 加密库。

21 可选：提供用于访问集群中机器的 **sshKey** 值。

22 提供用于镜像 registry 的证书文件内容。

23 从镜像 registry 时创建的 **imageContentSourcePolicy.yaml** 文件的 **metadata.name: release-0** 部分提供这些值。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

10.9.8. 安装 OpenShift CLI

您可以安装 OpenShift CLI(**oc**)来使用命令行界面与 OpenShift Container Platform 进行交互。您可以在 Linux、Windows 或 macOS 上安装 **oc**。

重要

如果安装了旧版本的 **oc**，则可能无法使用 OpenShift Container Platform 4.16 中的所有命令。下载并安装新版本的 **oc**。

在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **产品变体** 下拉列表中选择架构。
3. 从 **版本** 下拉列表中选择适当的版本。
4. 点 **OpenShift v4.16 Linux Client** 条目旁的 **Download Now** 来保存文件。
5. 解包存档：

```
$ tar xvf <file>
```

6. 将 **oc** 二进制文件放到 **PATH 中的目录中**。
要查看您的 **PATH**，请执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

在 Windows 上安装 OpenShift CLI
您可以按照以下流程在 Windows 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 Windows Client** 条目旁的 **Download Now** 来保存文件。
4. 使用 ZIP 程序解压存档。
5. 将 **oc** 二进制文件移到 **PATH 中的目录中**。
要查看您的 **PATH**，请打开命令提示并执行以下命令：

```
C:\> path
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

在 macOS 上安装 OpenShift CLI
您可以按照以下流程在 macOS 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。

2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 macOS Client** 条目旁的 **Download Now** 来保存文件。

注意

对于 macOS arm64，请选择 **OpenShift v4.16 macOS arm64 Client** 条目。

4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
要查看您的 **PATH**，请打开终端并执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

10.9.9. 手动创建 IAM

安装集群需要 Cloud Credential Operator(CCO)以手动模式运行。虽然安装程序为手动模式配置 CCO，但您必须为云供应商指定身份和访问管理 secret。

您可以使用 Cloud Credential Operator (CCO)实用程序(**ccoctl**)创建所需的 IBM Cloud® 资源。

先决条件

- 您已配置了 **ccoctl** 二进制文件。
- 您有一个现有的 **install-config.yaml** 文件。

流程

1. 编辑 **install-config.yaml** 配置文件，使其包含将 **credentialsMode** 参数设置为 **Manual**。

install-config.yaml 配置文件示例

```
apiVersion: v1
baseDomain: cluster1.example.com
credentialsMode: Manual (1)
compute:
- architecture: amd64
  hyperthreading: Enabled
```

(1) 添加这一行将 **credentialsMode** 参数设置为 **Manual**。

2. 要生成清单，请在包含安装程序的目录中运行以下命令：

```
$ openshift-install create manifests --dir <installation_directory>
```

3. 在包含安装程序的目录中，运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

4. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 自定义资源 (CR) 列表：

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
  --included \ 1
  --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
  --to=<path_to_directory_for_credentials_requests> 3
```

1. **--included** 参数仅包含特定集群配置所需的清单。
2. 指定 **install-config.yaml** 文件的位置。
3. 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

此命令为每个 **CredentialsRequest** 对象创建一个 YAML 文件。

CredentialsRequest 对象示例

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  labels:
    controller-tools.k8s.io: "1.0"
  name: openshift-image-registry-ibmcos
  namespace: openshift-cloud-credential-operator
spec:
  secretRef:
    name: installer-cloud-credentials
    namespace: openshift-image-registry
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: IBMCloudProviderSpec
    policies:
    - attributes:
      - name: serviceName
        value: cloud-object-storage
      roles:
      - crn:v1:bluemix:public:iam::::role:Viewer
      - crn:v1:bluemix:public:iam::::role:Operator
      - crn:v1:bluemix:public:iam::::role:Editor
      - crn:v1:bluemix:public:iam::::serviceRole:Reader
      - crn:v1:bluemix:public:iam::::serviceRole:Writer
    - attributes:
      - name: resourceType
```

```
    value: resource-group
  roles:
  - crn:v1:bluemix:public:iam::::role:Viewer
```

5. 为每个凭证请求创建服务 ID，分配定义的策略、创建 API 密钥并生成 secret：

```
$ ccoctl ibmcloud create-service-id \
  --credentials-requests-dir=<path_to_credential_requests_directory> \1
  --name=<cluster_name> \2
  --output-dir=<installation_directory> \3
  --resource-group-name=<resource_group_name> 4
```

1. 指定包含组件 **CredentialsRequest** 对象文件的目录。
2. 指定 OpenShift Container Platform 集群的名称。
3. 可选：指定您希望 **ccoctl** 实用程序在其中创建对象的目录。默认情况下，实用程序在运行命令的目录中创建对象。
4. 可选：指定用于限制访问策略的资源组的名称。

注意

如果您的集群使用 **TechPreviewNoUpgrade** 功能集启用的技术预览功能，则必须包含 **--enable-tech-preview** 参数。

如果提供了不正确的资源组名称，安装会在 bootstrap 阶段失败。要查找正确的资源组名称，请运行以下命令：

```
$ grep resourceGroupName <installation_directory>/manifests/cluster-infrastructure-02-config.yml
```

验证

- 确保在集群的 **manifests** 目录中生成了适当的 secret。

10.9.10. 部署集群

您可以在兼容云平台上安装 OpenShift Container Platform。

重要

在初始安装过程中，您只能运行安装程序的 **create cluster** 命令一次。

先决条件

- 您已使用托管集群的云平台配置了帐户。
- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。
  如果本地可用的 Red Hat Enterprise Linux CoreOS (RHCOS) 镜像，运行安装程序的主机不需要访问互联网。

- 已确认主机上的云供应商帐户具有部署集群的正确权限。权限不正确的帐户会导致安装过程失败，并显示包括缺失权限的错误消息。

流程

1. 运行以下命令，导出 **OPENSHIFT_INSTALL_OS_IMAGE_OVERRIDE** 变量来指定 Red Hat Enterprise Linux CoreOS (RHCOS) 镜像的位置：

```
$ export OPENSHIFT_INSTALL_OS_IMAGE_OVERRIDE="<path_to_image>/rhcos-<image_version>-ibmcloud.x86_64.qcow2.gz"
```

2. 进入包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1. 对于 **<installation_directory>**，请指定自定义 **./install-config.yaml** 文件的位置。
2. 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不是 **info**。

验证

当集群部署成功完成时：

- 终端会显示用于访问集群的说明，包括指向 Web 控制台和 **kubeadmin** 用户的凭证的链接。
- 凭证信息还会输出到 **<installation_directory>/.openshift_install.log**.

重要

不要删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

10.9.11. 使用 CLI 登录集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

其他资源

- 访问Web控制台

10.9.12. 安装后

完成以下步骤以完成集群的配置。

10.9.12.1. 禁用默认的 OperatorHub 目录源

在 OpenShift Container Platform 安装过程中，默认为 OperatorHub 配置由红帽和社区项目提供的源内容的 operator 目录。在受限网络环境中，必须以集群管理员身份禁用默认目录。

流程

- 通过在 **OperatorHub** 对象中添加 **disableAllDefaultSources: true 来** 禁用默认目录的源：

```
$ oc patch OperatorHub cluster --type json \
    -p '[{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]'
```

提示

或者，您可以使用 Web 控制台管理目录源。在 **Administration → Cluster Settings → Configuration → OperatorHub** 页面中，点 **Sources** 选项卡，您可以在其中创建、更新、删除、禁用和启用单独的源。

10.9.12.2. 将策略资源安装到集群中

使用 oc-mirror OpenShift CLI (oc) 插件镜像 OpenShift Container Platform 内容会创建资源，其中包括 **catalogSource-certified-operator-index.yaml** 和 **imageContentSourcePolicy.yaml**。

- **ImageContentSourcePolicy** 资源将镜像 registry 与源 registry 关联，并将在线 registry 中的镜像拉取请求重定向到镜像 registry。
- Operator Lifecycle Manager (OLM) 使用 **CatalogSource** 资源来检索有关镜像 registry 中可用 Operator 的信息，允许用户发现和安装 Operator。

安装集群后，您必须将这些资源安装到集群中。

先决条件

- 您已将镜像设置为断开连接的环境中的 registry 镜像。
- 您可以使用具有 **cluster-admin** 角色的用户访问集群。

流程

1. 以具有 **cluster-admin** 角色的用户身份登录 OpenShift CLI。
2. 将结果目录中的 YAML 文件应用到集群：

```
$ oc apply -f ./oc-mirror-workspace/results-<id>/
```

验证

1. 验证 **ImageContentSourcePolicy** 资源是否已成功安装：

```
$ oc get imagecontentsourcepolicy
```

2. 验证 **CatalogSource** 资源是否已成功安装：

```
$ oc get catalogsource --all-namespaces
```

10.9.13. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 关于远程健康监控

10.9.14. 后续步骤

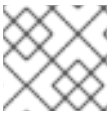
- 自定义集群。
- 可选：不使用远程健康报告。

10.10. IBM CLOUD 的安装配置参数

在 IBM Cloud® 上部署 OpenShift Container Platform 集群前，您可以提供参数来自定义集群以及托管它的平台。在创建 **install-config.yaml** 文件时，您可以通过命令行为所需参数提供值。然后，您可以修改 **install-config.yaml** 文件以进一步自定义集群。

10.10.1. IBM Cloud 可用的安装配置参数

下表指定您可以在安装过程中设置所需的、可选和 IBM Cloud 特定安装配置参数。



注意

安装后，您无法在 **install-config.yaml** 文件中修改这些参数。

10.10.1.1. 所需的配置参数

下表描述了所需的安装配置参数：

表 10.19. 所需的参数

参数	描述	值
apiVersion:	**install-config.yaml** 内容的 API 版本。当前版本为 **v1**。安装程序可能还支持旧的 API 版本。	字符串

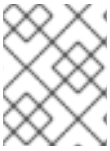
参数	描述	值
baseDomain:	云供应商的基域。基域用于创建到 OpenShift Container Platform 集群组件的路由。集群的完整 DNS 名称是 **baseDomain** 和 **metadata.name** 参数值的组合，其格式为 **<metadata.name>.<baseDomain>**。	完全限定域名或子域名，如 **example.com**。
metadata:	Kubernetes 资源 **ObjectMeta**，其中只消耗 **name** 参数。	对象
metadata: name:	集群的名称。集群的 DNS 记录是 **{{.metadata.name}}.{{.baseDomain}}** 的子域。	小写字母、连字符(**-**)和句点(**.**)字符串，如 **dev**。
platform:	对于特定平台的配置取决于执行安装的环境： **aws**, **baremetal**, **azure**, **gcp**, **ibmcloud**, **nutanix**, **openstack**, **powervs**, **vsphere**, 或 **{}**.有关 **platform.<platform>** 参数的更多信息，请参考下表中您的特定平台。	对象

pullSecret:	从 Red Hat OpenShift Cluster Manager 获取 pull secret，验证从 Quay.io 等服务中下载 OpenShift Container Platform 组件的容器镜像。	{
		"auths":{
		"cloud.openshift.com":{
		"auth":"b3Blb=",
		"email":"you@example.com"
		},
		"quay.io":{
		"auth":"b3Blb=",
		"email":"you@example.com"
		}
		}
		}

10.10.1.2. 网络配置参数

您可以根据现有网络基础架构的要求自定义安装配置。例如，您可以扩展集群网络的 IP 地址块，或者提供不同于默认值的不同 IP 地址块。

仅支持 IPv4 地址。



注意

Red Hat OpenShift Data Foundation 灾难恢复解决方案不支持 Globalnet。对于区域灾难恢复场景，请确保为每个集群中的集群和服务网络使用非重叠的专用 IP 地址。

表 10.20. 网络参数

参数	描述	值
networking:	集群网络的配置。	对象  **注意** 您无法在安装后修改 **网络** 对象指定的参数。
networking: networkType:	要安装的 Red Hat OpenShift Networking 网络插件。	**OVNKubernetes**。**OVNKubernetes** 是 Linux 网络和包含 Linux 和 Windows 服务器的混合网络的 CNI 插件。默认值为 **OVNKubernetes**。

networking:
clusterNetwork:
	pod 的 IP 地址块。
	默认值为 **10.128.0.0/14**，主机前缀为 **/23**。
	如果您指定了多个 IP 地址块，块不得重叠。
		对象数组。例如：
		`networking:`
		`clusterNetwork:`
		`- cidr: 10.128.0.0/14`
		`hostPrefix: 23`

networking: clusterNetwork: cidr:	使用 **networking.clusterNetwork** 时需要此项。IP 地址块。 IPv4 网络。	无类别域间路由(CIDR)表示法中的 IP 地址块。IPv4 块的前缀长度介于 **0 到 32** 之间。
networking: clusterNetwork: hostPrefix:	分配给每个节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从 given **cidr** 中分配 a/**23** 子网。**hostPrefix** 值 **23** 提供 510（2^(32 - 23)- 2）pod IP 地址。	子网前缀。 默认值为 **23**。

networking:
serviceNetwork:
	服务的 IP 地址块。默认值为 **172.30.0.0/16**。
	OVN-Kubernetes 网络插件只支持服务网络的一个 IP 地址块。
		CIDR 格式具有 IP 地址块的数组。例如：
		`networking:`
		`serviceNetwork:`
		`- 172.30.0.0/16`

参数	描述	值
networking: machineNetwork:	机器的 IP 地址块。 如果您指定了多个 IP 地址块，块不得重叠。	对象数组。例如： networking: machineNetwork: - cidr: 10.0.0.0/16
networking: machineNetwork: cidr:	使用 **networking.machineNetwork** 时需要此项。IP 地址块。对于 libvirt 和 IBM Power® Virtual Server 以外的所有平台，默认值为 **10.0.0.0/16**。对于 libvirt，默认值 为 **192.168.126.0/24**。对于 IBM Power® Virtual Server，默认值为 **192.168.0.0/24**。如果您要将集群部署到现有的 Virtual Private Cloud (VPC)，则 CIDR 必须包含在 **platform.ibmcloud.controlPlaneSubnets** and **platform.ibmcloud.computeSubnets** 中定义的子网。	CIDR 表示法中的 IP 网络块。 例如： **10.0.0.0/16**。 **注意** 将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

10.10.1.3. 可选的配置参数

下表描述了可选的安装配置参数：

表 10.21. 可选参数

参数	描述	值
additionalTrustBundle:	添加到节点可信证书存储中的 PEM 编码 X.509 证书捆绑包。配置了代理时，也可以使用此信任捆绑包。	字符串
capabilities:	控制可选核心组件的安装。您可以通过禁用可选组件来减少 OpenShift Container Platform 集群的空间。如需更多信息，请参阅 *安装*中的"集群功能"页面。	字符串数组
capabilities: baselineCapabilitySet:	选择要启用的一组初始可选功能。有效值为 **None**、**v4.11**、**v4.12** 和 **vCurrent**。默认值为 **vCurrent**。	字符串

参数	描述	值
capabilities: additionalEnabledCapabilities:	将可选功能集合扩展到您在 **baselineCapabilitySet** 中指定的范围。您可以在此参数中指定多个功能。	字符串数组
cpuPartitioningMode:	启用工作负载分区，它会隔离 OpenShift Container Platform 服务、集群管理工作负载和基础架构 pod，以便在保留的一组 CPU 上运行。工作负载分区只能在安装过程中启用，且在安装后无法禁用。虽然此字段启用工作负载分区，但它不会将工作负载配置为使用特定的 CPU。如需更多信息，请参阅 *Scalability and Performance* 部分中的 *Workload partitioning* 页面。	**None** 或 **AllNodes**.**None** 是默认值。
compute:	组成计算节点的机器的配置。	**MachinePool** 对象的数组。
compute: architecture:	决定池中机器的指令集合架构。目前，不支持具有不同架构的集群。所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
compute: hyperthreading:	是否在计算机器上启用或禁用并发多**线程或超线程**。默认情况下，启用并发多线程以提高机器内核的性能。 **重要** 如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。	**enabled** 或 **Disabled**
compute: name:	使用 **compute** 时需要此项。机器池的名称。	**worker**
compute: platform:	使用 **compute** 时需要此项。使用此参数指定托管 worker 机器的云供应商。此参数值必须与 **controlPlane.platform** 参数值匹配。	**aws**, **azure**, **gcp**, **ibmcloud**, **nutanix**, **openstack**, **powervs**, **vsphere**, 或 **{}**

参数	描述	值
compute: replicas:	要置备的计算机器数量，也称为 worker 机器。	大于或等于 **2** 的正整数。默认值为 **3**。
featureSet:	为功能集启用集群。功能集是 OpenShift Container Platform 功能的集合，默认情况下不启用。有关在安装过程中启用功能集的更多信息，请参阅"使用功能门启用功能"。	字符串.要启用的功能集的名称，如 **TechPreviewNoUpgrade**。
controlPlane:	组成 control plane 的机器的配置。	**MachinePool** 对象的数组。
controlPlane: architecture:	决定池中机器的指令集合架构。目前，不支持具有不同架构的集群。所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
controlPlane: hyperthreading:	是否在 control plane 机器上启用或禁用并发多 **线程或超线程**。默认情况下，启用并发多线程以提高机器内核的性能。 **重要** 如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。	**enabled** 或 **Disabled**
controlPlane: name:	使用 **controlPlane** 时需要此项。机器池的名称。	**master**
controlPlane: platform:	使用 **controlPlane** 时需要此项。使用此参数指定托管 control plane 机器的云供应商。此参数值必须与 **compute.platform** 参数值匹配。	**aws**, **azure**, **gcp**, **ibmcloud**, **nutanix**, **openstack**, **powervs**, **vsphere**, 或 **{}**
controlPlane: replicas:	要置备的 control plane 机器数量。	部署单节点 OpenShift 时支持的值为 **3** 或 **1**。

参数	描述	值
credentialsMode:	Cloud Credential Operator(CCO)模式。如果没有指定模式，CCO 会动态尝试决定提供的凭证的功能，在支持多个模式的平台上首选 mint 模式。	**Mint**、**Passthrough**、**Manual** 或空字符串(""")。[1]

参数	描述	值
fips:	启用或禁用 FIPS 模式。默认值为 **false**（禁用）。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。  **重要** 要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 Red Hat Enterprise Linux (RHEL) 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅在 FIPS 模式中安装该系统。 当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS) 时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。  **注意** 如果使用 Azure File 存储，则无法启用 FIPS 模式。	**false** 或 **true**
imageContentSources:	release-image 内容的源和存储库。	对象数组。包括一个 **source** 以及可选的 **mirrors**，如本表的以下行所述。

参数	描述	值
imageContentSources: source:	使用 **imageContentSources** 时需要此项。指定用户在镜像拉取规格中引用的存储库。	字符串
imageContentSources: mirrors:	指定可能还包含同一镜像的一个或多个仓库。	字符串数组
publish:	如何发布或公开集群的面向用户的端点，如 Kubernetes API、OpenShift 路由。	**内部或外部 .**要部署无法从互联网访问的私有集群，请将 **publish** 设置为 **Internal**。默认值为 **External**。
sshKey:	用于验证对集群机器的访问的 SSH 密钥。  **注意** 对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。	例如，**sshKey: ssh-ed25519 AAAA..**。

1. 不是所有 CCO 模式都支持所有云供应商。有关 CCO 模式的更多信息，请参阅*身份验证和授权*内容中的"管理云供应商凭证"条目。

10.10.1.4. 其他 IBM Cloud 配置参数

下表描述了其他 IBM Cloud® 配置参数：

表 10.22. 其他 IBM Cloud (R)参数

参数	描述	值
controlPlane: platform: ibmcloud: bootVolume: encryptionKey:	一个 IBM® Key Protect for IBM Cloud® (Key Protect) root 密钥，应只用于加密 control plane 机器的 root (boot) 卷。	root 密钥的 Cloud Resource Name (CRN)。 CRN 必须用引号 ("") 括起。
compute: platform: ibmcloud: bootVolume: encryptionKey:	一个 Key Protect root 密钥，应只用于加密计算机器的 root (boot) 卷。	root 密钥的 CRN。 CRN 必须用引号 ("") 括起。

参数	描述	值
platform: ibmcloud: defaultMachinePlatform: bootvolume: encryptionKey:	一个 Key Protect root 密钥，应用于加密所有集群机器的 root (boot) 卷。 当作为默认机器配置的一部分指定时，所有受管存储类都会使用此密钥更新。因此，安装后置备的数据卷也会使用此密钥加密。	root 密钥的 CRN。 CRN 必须用引号 ("") 括起。
platform: ibmcloud: resourceGroupName:	现有资源组的名称。默认情况下，安装程序置备的 VPC 和集群资源放置在此资源组中。如果没有指定，安装程序会为集群创建资源组。如果您要将集群部署到现有的 VPC 中，安装程序置备的集群资源将放置在此资源组中。如果没有指定，安装程序会为集群创建资源组。您置备的 VPC 资源必须存在于您使用 **networkResourceGroupName** 参数指定的资源组中。在这两种情况下，此资源组必须仅用于单个集群安装，因为集群组件假定资源组中所有资源的所有权 [1]	字符串，如 **existing_resource_group**。

参数	描述	值
platform: ibmcloud: serviceEndpoints: - name: url:	服务端点名称和 URI 列表。 默认情况下，安装程序和集群组件使用公共服务端点访问所需的 IBM Cloud® 服务。 如果网络限制限制对公共服务端点的访问，您可以指定一个备用服务端点来覆盖默认行为。 您只能为每个服务指定一个备用服务端点： • 云对象存储 • DNS 服务 • 全局搜索 • 全局标记 • 身份服务 • 密钥保护 • 资源控制器 • 资源管理器 • VPC	有效服务端点名称和完全限定的 URI。 有效名称包括： • **COS** • **DNSServices** • **GlobalServices** • **GlobalTagging** • **IAM** • **KeyProtect** • **ResourceController** • **ResourceManager** • **VPC**
platform: ibmcloud: networkResourceGroupName:	现有资源组的名称。此资源包含集群要部署到的现有 VPC 和子网。当将集群部署到已置备的 VPC 时，需要此参数。	字符串，如 **existing_network_resource_group**。

参数	描述	值
platform: ibmcloud: dedicatedHosts: profile:	要创建的新专用主机。如果您为 **platform.ibmcloud.dedicatedHosts.name** 指定值，则不需要此参数。	有效的 IBM Cloud® 专用主机配置文件，如 **cx2-host-152x304**. [2]
platform: ibmcloud: dedicatedHosts: name:	存在一个现有的专用主机。如果您为 **platform.ibmcloud.dedicatedHosts.profile** 指定一个值，则不需要此参数。	字符串，如 **my-dedicated-host-name**。
platform: ibmcloud: type:	所有 IBM Cloud® 机器的实例类型。	有效的 IBM Cloud® 实例类型，如 **bx2-8x32**. [2]

参数	描述	值
platform: ibmcloud: vpcName:	要将集群部署到的现有 VPC 的名称。	字符串.
platform: ibmcloud: controlPlaneSubnets:	要将 control plane 机器部署到的 VPC 中现有子网的名称。为每个可用区指定一个子网。	字符串数组
platform: ibmcloud: computeSubnets:	要将计算机器部署到的 VPC 中现有子网的名称。为每个可用区指定一个子网。不支持子网 ID。	字符串数组

1. 不论您定义现有资源组还是安装程序创建了资源组，都决定在卸载资源组时如何对待资源组。如果您定义了资源组，安装程序会删除所有安装程序置备的资源，但只保留资源组；如果资源组是作为安装的一部分创建，安装程序将删除所有安装程序置备的资源和资源组。

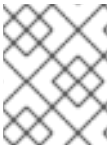
2. 要确定哪个配置集最符合您的需要，请参阅 IBM® 文档中的实例配置集。

10.11. 在 IBM CLOUD 上卸载集群

您可以删除部署到 IBM Cloud® 的集群。

10.11.1. 删除使用安装程序置备的基础架构的集群

您可以从云中删除使用安装程序置备的基础架构的集群。



注意

卸载后，检查云供应商是否有未正确删除的资源，特别是在用户置备基础架构(UPI)集群中。可能存在安装程序未创建或安装程序无法访问的资源。

先决条件

- 有用于部署集群的安装程序副本。
- 有创建集群时安装程序生成的文件。
- 您已配置了 **ccoctl** 二进制文件。
- 已安装 IBM Cloud® CLI 并安装或更新 VPC 基础架构服务插件。如需更多信息，请参阅 IBM Cloud® CLI 文档中的 "先决条件"。

流程

1. 如果满足以下条件，则需要执行此步骤：
   - 安装程序作为安装过程的一部分创建了资源组。
   - 部署集群后，或您的应用程序创建了持久性卷声明(PVC)。

   在这种情况下，在卸载集群时不会删除 PVC，这可以防止资源组被成功删除。要防止失败：

   a. 使用 CLI 登录 IBM Cloud®。

   b. 要列出 PVC，请运行以下命令：

   ```
   $ ibmcloud is volumes --resource-group-name <infrastructure_id>
   ```

   有关列出卷的更多信息，请参阅 IBM Cloud® CLI 文档。

   c. 要删除 PVC，请运行以下命令：

   ```
   $ ibmcloud is volume-delete --force <volume_id>
   ```

   有关删除卷的更多信息，请参阅 IBM Cloud® CLI 文档。

2. 导出在安装过程中创建的 API 密钥。

```
$ export IC_API_KEY=<api_key>
```

注意

您必须按照指定方式设置变量名称。安装程序需要存在变量名称来删除安装集群时所创建的服务 ID。

3. 在用来安装集群的计算机中包含安装程序的目录中，运行以下命令：

```
$ ./openshift-install destroy cluster \
--dir <installation_directory> --log-level info 1 2
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

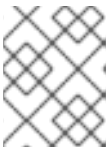
2 要查看不同的详情，请指定 **warn**、**debug** 或 **error**，而不是 **info**。

注意

您必须为集群指定包含集群定义文件的目录。安装程序需要此目录中的 **metadata.json** 文件来删除集群。

4. 删除为集群创建的手动 CCO 凭证：

```
$ ccoctl ibmcloud delete-service-id \
    --credentials-requests-dir <path_to_credential_requests_directory> \
    --name <cluster_name>
```



注意

如果您的集群使用 **TechPreviewNoUpgrade** 功能集启用的技术预览功能，则必须包含 **--enable-tech-preview** 参数。

5. 可选：删除 **<installation_directory>** 目录和 OpenShift Container Platform 安装程序。

第 11 章 在 NUTANIX 上安装

11.1. 准备在 NUTANIX 上安装

在安装 OpenShift Container Platform 集群前，请确定您的 Nutanix 环境满足以下要求。

11.1.1. Nutanix 版本要求

您必须将 OpenShift Container Platform 集群安装到符合以下要求的 Nutanix 环境。

表 11.1. Nutanix 虚拟环境的版本要求

组件	所需的版本
Nutanix AOS	6.5.2.7 或更高版本
Prism Central	pc.2022.6 或更高版本

11.1.2. 环境要求

在安装 OpenShift Container Platform 集群前，请查看以下 Nutanix AOS 环境要求。

11.1.2.1. 所需的帐户权限

安装程序需要访问具有所需权限的 Nutanix 帐户，以部署集群并维护其每日操作。以下选项可供您使用：

- 您可以使用具有管理特权的本地 Prism Central 用户帐户。使用本地帐户是授予对具有所需权限的帐户访问权限的最快的方法。
- 如果机构的安全策略要求您使用更严格的权限集，请使用下表中列出的权限在 Prism Central 中创建自定义 Cloud Native 角色。然后，您可以将角色分配给作为 Prism Central 身份验证目录成员的用户帐户。

管理此用户帐户时请考虑以下几点：

- 在为角色分配实体时，请确保用户只能访问部署虚拟机所需的 Prism Element 和子网。
- 确保该用户是需要为其分配虚拟机的项目的成员。

如需更多信息，请参阅 Nutanix 文档中有关自定义云原生角色, 分配角色, 和为项目添加用户的内容。

例 11.1. 创建自定义云原生角色所需的权限

Nutanix 对象	必要时	Nutanix API 中所需的权限	描述

Nutanix 对象	必要时	Nutanix API 中所需的权限	描述
Categories	Always	**Create_Category_Mapping** **Create_Or_Update_Name_Category** **Create_Or_Update_Value_Category** **Delete_Category_Mapping** **Delete_Name_Category** **Delete_Value_Category** **View_Category_Mapping** **View_Name_Category** **View_Value_Category**	创建、读取和删除分配给 OpenShift Container Platform 机器的类别。
镜像	Always	**Create_Image** **Delete_Image** **View_Image**	创建、读取和删除用于 OpenShift Container Platform 机器的操作系统镜像。
虚拟机	Always	**Create_Virtual_Machine** **Delete_Virtual_Machine** **View_Virtual_Machine**	创建、读取和删除 OpenShift Container Platform 机器。
Clusters	Always	**View_Cluster**	查看托管 OpenShift Container Platform 机器的 Prism Element 集群。
子网	Always	**View_Subnet**	查看托管 OpenShift Container Platform 机器的子网。
项目	如果您将项目与计算机器、control plane 机器或所有机器关联。	**View_Project**	查看 Prism Central 中定义的项目，并允许将项目分配给 OpenShift Container Platform 机器。

11.1.2.2. 集群限制

可用资源因集群而异。Nutanix 环境中可能的集群数量主要受可用的存储空间限制，以及与集群所创建的资源相关的限制，以及部署集群所需的资源（如 IP 地址和网络）。

11.1.2.3. 集群资源

使用标准集群至少需要 800 GB 存储。

当您部署使用安装程序置备的基础架构的 OpenShift Container Platform 集群时，安装程序必须能够在 Nutanix 实例中创建多个资源。虽然这些资源使用 856 GB 存储，但 bootstrap 节点会作为安装过程的一部分被销毁。

标准的 OpenShift Container Platform 安装会创建以下资源：

- 1 个标签
- 虚拟机：
  - 1 个磁盘镜像
  - 1 个临时 bootstrap 节点
  - 3 个 control plane 节点
  - 3 个计算机器

11.1.2.4. 网络要求

您必须对网络使用 AHV IP 地址管理(IPAM)或动态主机配置协议(DHCP)，并确保它被配置为为集群机器提供持久的 IP 地址。另外，在安装 OpenShift Container Platform 集群前创建以下网络资源：

- IP 地址
- DNS 记录

注意

建议集群中的每个 OpenShift Container Platform 节点都可以访问可通过 DHCP 发现的网络时间协议 (NTP) 服务器。没有 NTP 服务器即可安装。但是，NTP 服务器可防止错误通常与异步服务器时钟相关联。

11.1.2.4.1. 所需的 IP 地址

安装程序置备的安装需要两个静态虚拟 IP (VIP) 地址：

- API 的 VIP 地址是必需的。此地址用于访问集群 API。
- 需要一个 ingress 的 VIP 地址。此地址用于集群入口流量。

安装 OpenShift Container Platform 集群时，可以指定这些 IP 地址。

11.1.2.4.2. DNS 记录

您必须在适当的 DNS 服务器中为托管 OpenShift Container Platform 集群的 Nutanix 实例创建两个静态 IP 地址的 DNS 记录。在每个记录中，**<cluster_name>** 是集群名称，**<base_domain> 是您** 在安装集群时指定的集群基域。

如果使用自己的 DNS 或 DHCP 服务器，还必须为每个节点创建记录，包括 bootstrap、control plane 和计算节点。

完整的 DNS 记录采用以下形式： **<component>.<cluster_name>.<base_domain>.**。

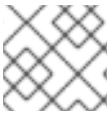
表 11.2. 所需的 DNS 记录

组件	记录	描述
API VIP	**api.<cluster_name>.<base_domain>.**	此 DNS A/AAAA 或 CNAME 记录必须指向 control plane 机器的负载均衡器。此记录必须由集群外的客户端和集群中的所有节点解析。
Ingress VIP	***.apps.<cluster_name>.<base_domain>.**	通配符 DNS A/AAAA 或 CNAME 记录，指向以运行入口路由器 Pod 的机器（默认为 worker 节点）为目标的负载均衡器。此记录必须由集群外的客户端和集群中的所有节点解析。

11.1.3. 配置 Cloud Credential Operator 工具

Cloud Credential Operator(CCO)将云供应商凭证作为 Kubernetes 自定义资源定义(CRD)进行管理。要在 Nutanix 上安装集群，您必须将 CCO 设置为**手动模式**，作为安装过程的一部分。

当 Cloud Credential Operator(CCO)以手动模式运行时，要从集群外部创建和管理云凭证，提取并准备 CCO 实用程序(**ccoctl**)二进制文件。



注意

ccoctl 工具是在 Linux 环境中运行的 Linux 二进制文件。

先决条件

- 您可以访问具有集群管理员权限的 OpenShift Container Platform 帐户。
- 已安装 OpenShift CLI(**oc**)。

流程

1. 运行以下命令，为 OpenShift Container Platform 发行镜像设置变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. 运行以下命令，从 OpenShift Container Platform 发行镜像获取 CCO 容器镜像：

```
$ CCO_IMAGE=$(oc adm release info --image-for='cloud-credential-operator'
$RELEASE_IMAGE -a ~/.pull-secret)
```

注意

确保 **$RELEASE_IMAGE** 的架构与将使用 **ccoctl** 工具的环境架构相匹配。

3. 运行以下命令，将 CCO 容器镜像中的 **ccoctl** 二进制文件提取到 OpenShift Container Platform 发行镜像中：

```
$ oc image extract $CCO_IMAGE \
  --file="/usr/bin/ccoctl.<rhel_version>" \ 1
  -a ~/.pull-secret
```

1 对于 **<rhel_version>**，请指定与主机使用的 Red Hat Enterprise Linux (RHEL) 版本对应的值。如果没有指定值，则默认使用 **ccoctl.rhel8**。以下值有效：

- **rhel8**: 为使用 RHEL 8 的主机指定这个值。
- **rhel9** ：为使用 RHEL 9 的主机指定这个值。

4. 运行以下命令更改权限以使 **ccoctl** 可执行：

```
$ chmod 775 ccoctl.<rhel_version>
```

验证

- 要验证 **ccoctl** 是否可使用，请运行以下命令来显示帮助文件：

```
$ ccoctl --help
```

ccoctl --help 的输出

```
OpenShift credentials provisioning tool

Usage:
  ccoctl [command]

Available Commands:
  aws         Manage credentials objects for AWS cloud
  azure       Manage credentials objects for Azure
  gcp         Manage credentials objects for Google cloud
  help        Help about any command
  ibmcloud    Manage credentials objects for IBM Cloud
  nutanix     Manage credentials objects for Nutanix

Flags:
  -h, --help   help for ccoctl

Use "ccoctl [command] --help" for more information about a command.
```

其他资源

- 准备使用手动维护的凭证更新集群

11.2. 使用多个 PRISM ELEMENT 的容错部署

默认情况下，安装程序会将 control plane 和计算机器安装到单个 Nutanix Prism Element (集群) 中。要改进 OpenShift Container Platform 集群的容错功能，您现在可以通过配置故障域来指定这些机器分布在多个 Nutanix 集群中。

故障域代表额外的 Prism Element 实例，在安装过程中和安装后可用于 OpenShift Container Platform 机器池。

11.2.1. 故障域要求

在计划使用故障域时，请考虑以下要求：

- 所有 Nutanix Prism Element 实例都必须由同一 Prism Central 实例管理。不支持由多个 Prism Central 实例组成的部署。
- 组成 Prism Element 集群的机器必须位于同一以太网网络中，以便故障域能够相互通信。
- 每个 Prism Element 中都需要一个子网，将用作 OpenShift Container Platform 集群中的故障域。在定义这些子网时，它们必须共享相同的 IP 地址前缀 (CIDR)，并且应包含 OpenShift Container Platform 集群使用的虚拟 IP 地址。

11.2.2. 安装方法和故障域配置

OpenShift Container Platform 的安装方法决定了如何以及何时配置故障域：

- 如果使用安装程序置备的基础架构部署，您可以在部署集群前在安装配置文件中配置故障域。如需更多信息，请参阅配置故障域。
  您还可以在部署集群后配置故障域。有关在安装后配置故障域的更多信息，请参阅将 故障域添加到现有 Nutanix 集群。
- 如果使用自己管理的基础架构（用户置备的基础架构）部署，则不需要额外的配置。部署集群后，您可以在故障域间手动分发 control plane 和计算机器。

11.3. 在 NUTANIX 上安装集群

在 OpenShift Container Platform 版本 4.16 中，您可以选择以下选项之一在 Nutanix 实例上安装集群：

使用安装程序置备的基础架构 ：使用以下部分中的步骤使用安装程序置备的基础架构。安装程序置备的基础架构适用于在连接的或断开连接的网络环境中安装。安装程序置备的基础架构包括为集群置备底层基础架构的安装程序。

使用 Assisted Installer: 在 console.redhat.com 中托管的 Assisted Installer。Assisted Installer 无法在断开连接的环境中使用。Assisted Installer 不会为集群置备底层基础架构，因此您必须在运行 Assisted Installer 前置备基础架构。使用 Assisted Installer 安装还提供与 Nutanix 集成，从而启用自动扩展。如需了解更多详细信息，请参阅使用 Assisted Installer 安装内部集群。

使用用户置备的基础架构 ：完成 在任意平台上安装集群文档中概述的相关步骤。

11.3.1. 先决条件

- 您已查看了有关 OpenShift Container Platform 安装和更新流程 的详细信息。
- 安装程序需要访问 Prism Central 和 Prism Element 上的端口 9440。您确认可以访问端口 9440。

- 如果使用防火墙，已满足以下先决条件：
  - 已确认可以访问端口 9440。control plane 节点必须能够访问端口 9440 上的 Prism Central 和 Prism Element 才能成功安装。
  - 您已将防火墙配置为授予访问 OpenShift Container Platform 所需站点的访问权限。这包括使用 Telemetry。
- 如果您的 Nutanix 环境使用默认自签名 SSL 证书，请将它替换为 CA 签名的证书。安装程序需要一个有效的 CA 签名证书来访问 Prism Central API。有关替换自签名证书的更多信息，请参阅 Nutanix AOS 安全指南。
  如果您的 Nutanix 环境使用内部 CA 来发布证书，您必须将集群范围代理配置为安装过程的一部分。如需更多信息，请参阅 配置自定义 PKI。

重要

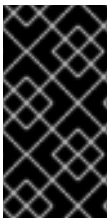
使用 2048 位证书。如果您使用带有 Prism Central 2022.x 的 4096 位证书，则安装会失败。

11.3.2. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。



重要

如果您的集群无法直接访问互联网，则可以在置备的某些类型的基础架构上执行受限网络安装。在此过程中，您可以下载所需的内容，并使用它为镜像 registry 填充安装软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群前，您要更新镜像 registry 的内容。

11.3.3. Prism Central 的互联网访问

Prism Central 需要访问互联网来获得用来安装集群的 Red Hat Enterprise Linux CoreOS (RHCOS) 镜像。Nutanix 的 RHCOS 镜像位于 **rhcos.mirror.openshift.com**。

11.3.4. 为集群节点 SSH 访问生成密钥对

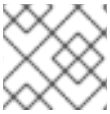
在 OpenShift Container Platform 安装过程中，您可以为安装程序提供 SSH 公钥。密钥通过它们的 Ignition 配置文件传递给 Red Hat Enterprise Linux CoreOS(RHCOS)节点，用于验证对节点的 SSH 访问。密钥添加到每个节点上 **core** 用户的 **~/.ssh/authorized_keys** 列表中，这将启用免密码身份验证。

将密钥传递给节点后，您可以使用密钥对作为用户 **核心** 通过 SSH 连接到 RHCOS 节点。若要通过 SSH 访问节点，必须由 SSH 为您的本地用户管理私钥身份。

如果要通过 SSH 连接到集群节点来执行安装调试或灾难恢复，则必须在安装过程中提供 SSH 公钥。**./openshift-install gather** 命令还需要在集群节点上设置 SSH 公钥。

重要

不要在生产环境中跳过这个过程，在生产环境中需要灾难恢复和调试。



注意

您必须使用本地密钥，而不是使用特定平台方法配置 的密钥，如 AWS 密钥对。

流程

1. 如果您在本地计算机上没有可用于在集群节点上进行身份验证的现有 SSH 密钥对，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_ed25519**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

注意

如果您计划在 **x86_64**、**ppc64le** 和 **s390x** 架构上安装使用 RHEL 加密库（这些加密库已提交给 NIST 用于 FIPS 140-2/140-3 验证）的 OpenShift Container Platform 集群，则不要创建使用 **ed25519** 算法的密钥。相反，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 查看公共 SSH 密钥：

```
$ cat <path>/<file_name>.pub
```

例如，运行以下命令来查看 **~/.ssh/id_ed25519.pub** 公钥：

```
$ cat ~/.ssh/id_ed25519.pub
```

3. 将 SSH 私钥身份添加到本地用户的 SSH 代理（如果尚未添加）。在集群节点上，或者要使用 **./openshift-install gather** 命令，需要对该密钥进行 SSH 代理管理，才能在集群节点上进行免密码 SSH 身份验证。

注意

在某些发行版中，自动管理默认 SSH 私钥身份，如 **~/.ssh/id_rsa** 和 **~/.ssh/id_dsa**。

a. 如果 **ssh-agent** 进程尚未为您的本地用户运行，请将其作为后台任务启动：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果集群处于 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

4. 将 SSH 私钥添加到 **ssh-agent** ：

```
$ ssh-add <path>/<file_name> 1
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_ed25519.pub**

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

后续步骤

- 安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

11.3.5. 获取安装程序

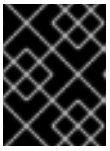
在安装 OpenShift Container Platform 前，将安装文件下载到您用于安装的主机上。

先决条件

- 您有一台运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB。

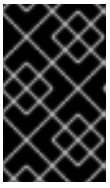
流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐户，请使用您的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入到安装类型的页面，下载与您的主机操作系统和架构对应的安装程序，并将该文件放在您要存储安装配置文件的目录中。



重要

安装程序会在用来安装集群的计算机上创建几个文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。



重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，请为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。此 pull secret 允许您与所含授权机构提供的服务进行身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

11.3.6. 在您的系统信任中添加 Nutanix root CA 证书

因为安装程序需要访问 Prism Central API，所以您必须在安装 OpenShift Container Platform 集群前将 Nutanix 可信根 CA 证书添加到您的系统信任中。

流程

1. 在 Prism Central web 控制台中下载 Nutanix root CA 证书。
2. 提取包含 Nutanix root CA 证书的压缩文件。
3. 将您的操作系统的文件添加到系统信任中。例如，在 Fedora 操作系统中运行以下命令：

```
# cp certs/lin/* /etc/pki/ca-trust/source/anchors
```

4. 更新您的系统信任关系。例如，在 Fedora 操作系统中运行以下命令：

```
# update-ca-trust extract
```

11.3.7. 创建安装配置文件

您可以自定义在 Nutanix 上安装的 OpenShift Container Platform 集群。

先决条件

- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。
- 您已确认满足了 Nutanix 网络要求。如需更多信息，请参阅"准备在 Nutanix 上安装"。

流程

1. 创建 **install-config.yaml** 文件。
   a. 进入包含安装程序的目录并运行以下命令：

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定要存储安装程序创建的文件的目录名称。

在指定目录时：

- 验证该目录是否具有**执行**权限。在安装目录中运行 Terraform 二进制文件需要这个权限。
- 使用空目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

b. 在提示符处，提供云的配置详情：

i. 可选： 选择用于访问集群机器的 SSH 密钥。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

ii. 选择 **nutanix** 作为目标平台。

iii. 输入 Prism Central 域名或 IP 地址。

iv. 输入用于登录 Prism Central 的端口。

v. 输入用于登录 Prism Central 的凭证。
安装程序连接到 Prism Central。

vi. 选择用于管理 OpenShift Container Platform 集群的 Prism Element。

vii. 选择要使用的网络子网。

viii. 输入您为 control plane API 访问配置的虚拟 IP 地址。

ix. 输入您为集群入口配置的虚拟 IP 地址。

x. 输入基域。这个基域必须与您在 DNS 记录中配置的域相同。

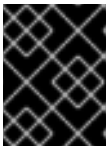
xi. 为集群输入描述性名称。
您输入的集群名称必须与您在配置 DNS 记录时指定的集群名称匹配。

2. 可选：更新 **install.config.yaml** 文件中的一个或多个默认配置参数，以自定义安装。
有关参数的更多信息，请参阅"安装配置参数"。

注意

如果要安装三节点集群，请确保将 **compute.replicas** 参数设置为 **0**。这样可确保集群的 control plane 可以调度。如需更多信息，请参阅"在 Nutanix 上安装三节点集群"。

3. 备份 **install-config.yaml** 文件，以便您可以使用它安装多个集群。



重要

install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用此文件，必须现在备份。

其他资源

- Nutanix 的安装配置参数

11.3.7.1. Nutanix 的自定义 install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多详情，或修改所需参数的值。

重要

此示例 YAML 文件仅供参考。您必须使用安装程序来获取 **install-config.yaml** 文件，并进行修改。

```
apiVersion: v1
baseDomain: example.com (1)
compute: (2)
- hyperthreading: Enabled (3)
  name: worker
  replicas: 3
  platform:
    nutanix: (4)
      cpus: 2
      coresPerSocket: 2
      memoryMiB: 8196
      osDisk:
        diskSizeGiB: 120
      categories: (5)
      - key: <category_key_name>
        value: <category_value>
controlPlane: (6)
  hyperthreading: Enabled (7)
  name: master
  replicas: 3
  platform:
    nutanix: (8)
      cpus: 4
      coresPerSocket: 2
      memoryMiB: 16384
      osDisk:
        diskSizeGiB: 120
      categories: (9)
      - key: <category_key_name>
        value: <category_value>
metadata:
  creationTimestamp: null
  name: test-cluster (10)
networking:
  clusterNetwork:
    - cidr: 10.128.0.0/14
      hostPrefix: 23
  machineNetwork:
    - cidr: 10.0.0.0/16
  networkType: OVNKubernetes (11)
  serviceNetwork:
    - 172.30.0.0/16
platform:
  nutanix:
    apiVIPs:
      - 10.40.142.7 (12)
```

```
    defaultMachinePlatform:
      bootType: Legacy
      categories: 13
      - key: <category_key_name>
        value: <category_value>
      project: 14
        type: name
        name: <project_name>
    ingressVIPs:
      - 10.40.142.8 15
    prismCentral:
      endpoint:
        address: your.prismcentral.domainname 16
        port: 9440 17
      password: <password> 18
      username: <username> 19
    prismElements:
    - endpoint:
        address: your.prismelement.domainname
        port: 9440
      uuid: 0005b0f1-8f43-a0f2-02b7-3cecef193712
    subnetUUIDs:
    - c7938dc6-7659-453e-a688-e26020c68e43
    clusterOSImage: http://example.com/images/rhcos-47.83.202103221318-0-nutanix.x86_64.qcow2
20
  credentialsMode: Manual
  publish: External
  pullSecret: '{"auths": ...}' 21
  fips: false 22
  sshKey: ssh-ed25519 AAAA... 23
```

1 10 12 15 16 17 18 19 21 必需。安装程序会提示您输入这个值。

2 6 **controlPlane** 部分是一个单个映射，但 compute 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，controlPlane **部分** 的第一行则不以连字符开头。虽然这两个部分目前都定义了单一机器池，但未来的 OpenShift Container Platform 版本可能在安装过程中支持定义多个计算池。仅使用一个 control plane 池。

3 7 是否要启用或禁用并发多线程或 **超线程**。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将 参数值设置为 **Disabled** 来禁用它。如果在某些集群机器中禁用并发多线程，则必须在所有集群机器中禁用它。

重要

如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。

4 8 可选：为 compute 和 control plane 机器提供额外的机器池参数配置。

5 9 13 可选：提供一个或多个 prism category 键值对和一个 prism category 值。这些类别键值对必须在 Prism Central 中存在。您可以为计算机器、control plane 机器或所有机器提供单独的类别。

11 要安装的集群网络插件。默认值 **OVNKubernetes** 是唯一支持的值。

14

可选：指定与其关联的项目。为项目类型指定 **name** 或 **uuid**，然后提供对应的 UUID 或项目名称。您可以将项目与计算机器、control plane 机器或所有机器关联。

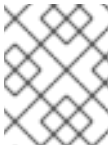
20 可选： 默认情况下，安装程序会下载并安装 Red Hat Enterprise Linux CoreOS (RHCOS) 镜像。如果 Prism Central 没有互联网访问，您可以通过在任何 HTTP 服务器上托管 RHCOS 镜像来覆盖默认行为，并将安装程序指向镜像。

22 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS)时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。

23 可选：您可以提供用于访问集群中机器的 **sshKey** 值。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

11.3.7.2. 配置故障域

故障域通过在多个 Nutanix Prism Elements (clusters) 间分布 control plane 和计算机器来提高 OpenShift Container Platform 集群的容错功能。

提示

建议您配置三个故障域以确保高可用性。

先决条件

- 您有一个安装配置文件 (**install-config.yaml**)。

流程

1. 编辑 **install-config.yaml** 文件并添加以下小节来配置第一个故障域：

```
apiVersion: v1
baseDomain: example.com
compute:
# ...
platform:
  nutanix:
    failureDomains:
    - name: <failure_domain_name>
      prismElement:
        name: <prism_element_name>
        uuid: <prism_element_uuid>
```

```
      subnetUUIDs:
      - <network_uuid>
# ...
```

其中：

<failure_domain_name>

指定故障域的唯一名称。名称的长度不能超过 64 个字符，可以包括小写字母、数字和短划线 (-)。短划线不能是名称的第一个或最后一个。

<prism_element_name>

可选。指定 Prism Element 的名称。

<prism_element_uuid>

指定 Prism Element 的 UUID。

<network_uuid>

指定 Prism Element 子网对象的 UUID。子网的 IP 地址前缀 (CIDR) 应包含 OpenShift Container Platform 集群使用的虚拟 IP 地址。在 OpenShift Container Platform 集群中，只支持每个故障域 (Prism Element) 一个子网。

2. 根据需要，配置额外的故障域。

3. 要在故障域间分发 control plane 和计算机器，请执行以下操作之一：

- 如果 compute 和 control plane 机器可以共享同一组故障域，请在集群的默认机器配置下添加故障域名称。

共享一组故障域的 control plane 和计算机器示例

```
apiVersion: v1
baseDomain: example.com
compute:
# ...
platform:
  nutanix:
    defaultMachinePlatform:
      failureDomains:
        - failure-domain-1
        - failure-domain-2
        - failure-domain-3
# ...
```

- 如果 compute 和 control plane 机器必须使用不同的故障域，请在对应的机器池下添加故障域名称。

使用不同的故障域的 control plane 和计算机器示例

```
apiVersion: v1
baseDomain: example.com
compute:
# ...
controlPlane:
  platform:
    nutanix:
      failureDomains:
```

```
        - failure-domain-1
        - failure-domain-2
        - failure-domain-3
# ...
compute:
  platform:
    nutanix:
      failureDomains:
        - failure-domain-1
        - failure-domain-2
# ...
```

4. 保存该文件。

11.3.7.3. 在安装过程中配置集群范围的代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 **Proxy** 对象的 **spec.noProxy** 字段中添加站点来绕过代理。

注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，**Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 **http**。

2 用于创建集群外 HTTPS 连接的代理 URL。

3 要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 **.** 以仅匹配子域。例如，**.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代理。

4 如果提供，安装程序会在 **openshift-config** 命名空间中生成名为 **user-ca-bundle** 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 **trusted-ca-bundle** 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， **Proxy** 对象的 **trustedCA** 字段中也会引用此配置映射。**additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

5 可选：决定 **Proxy** 对象的配置以引用 **trustedCA** 字段中 **user-ca-bundle** 配置映射的策略。允许的值是 **Proxyonly** 和 **Always**。仅在配置了 **http/https** 代理时，使用 **Proxyonly** 引用 **user-ca-bundle** 配置映射。使用 **Always** 始终引用 **user-ca-bundle** 配置映射。默认值为 **Proxyonly**。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。

注意

如果安装程序超时，重启并使用安装程序的 **wait-for** 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. 保存该文件并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 cluster 的集群范围代理，该代理 **使用** 提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

11.3.8. 安装 OpenShift CLI

您可以安装 OpenShift CLI(**oc**)来使用命令行界面与 OpenShift Container Platform 进行交互。您可以在 Linux、Windows 或 macOS 上安装 **oc**。

重要

如果安装了旧版本的 **oc**，则可能无法使用 OpenShift Container Platform 4.16 中的所有命令。下载并安装新版本的 **oc**。

在 Linux 上安装 OpenShift CLI
您可以按照以下流程在 Linux 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **产品变体** 下拉列表中选择架构。
3. 从 **版本** 下拉列表中选择适当的版本。
4. 点 **OpenShift v4.16 Linux Client** 条目旁的 **Download Now** 来保存文件。
5. 解包存档：

```
$ tar xvf <file>
```

6. 将 **oc** 二进制文件放到 **PATH 中的目录中**。
要查看您的 **PATH**，请执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

在 Windows 上安装 OpenShift CLI
您可以按照以下流程在 Windows 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 Windows Client** 条目旁的 **Download Now** 来保存文件。
4. 使用 ZIP 程序解压存档。
5. 将 **oc** 二进制文件移到 **PATH 中的目录中**。
要查看您的 **PATH**，请打开命令提示并执行以下命令：

```
C:\> path
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

在 macOS 上安装 OpenShift CLI
您可以按照以下流程在 macOS 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。

2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 macOS Client** 条目旁的 **Download Now** 来保存文件。

注意

对于 macOS arm64，请选择 **OpenShift v4.16 macOS arm64 Client** 条目。

4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
要查看您的 **PATH**，请打开终端并执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

11.3.9. 为 Nutanix 配置 IAM

安装集群需要 Cloud Credential Operator(CCO)以手动模式运行。虽然安装程序为手动模式配置 CCO，您必须指定身份和访问管理 secret。

先决条件

- 您已配置了 **ccoctl** 二进制文件。
- 您有一个 **install-config.yaml** 文件。

流程

1. 按照以下格式创建一个包含凭证数据的 YAML 文件：

凭证数据格式

```
credentials:
- type: basic_auth (1)
  data:
    prismCentral: (2)
      username: <username_for_prism_central>
      password: <password_for_prism_central>
    prismElements: (3)
    - name: <name_of_prism_element>
      username: <username_for_prism_element>
      password: <password_for_prism_element>
```

(1) 指定身份验证类型。仅支持基本身份验证。

(2) 指定 Prism Central 凭证。

3 可选：指定 Prism Element 凭证。

2. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

3. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 自定义资源 (CR) 列表：

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
  --included \ 1
  --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
  --to=<path_to_directory_for_credentials_requests> 3
```

1 **--included** 参数仅包含特定集群配置所需的清单。

2 指定 **install-config.yaml** 文件的位置。

3 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

CredentialsRequest 对象示例

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  annotations:
    include.release.openshift.io/self-managed-high-availability: "true"
  labels:
    controller-tools.k8s.io: "1.0"
  name: openshift-machine-api-nutanix
  namespace: openshift-cloud-credential-operator
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: NutanixProviderSpec
  secretRef:
    name: nutanix-credentials
    namespace: openshift-machine-api
```

4. 运行以下命令，使用 **ccoctl** 工具处理所有 **CredentialsRequest** 对象：

```
$ ccoctl nutanix create-shared-secrets \
  --credentials-requests-dir=<path_to_credentials_requests_directory> \ 1
  --output-dir=<ccoctl_output_dir> \ 2
  --credentials-source-filepath=<path_to_credentials_file> 3
```

1 指定包含组件 **CredentialsRequests** 对象文件的目录路径。

2 可选：指定您希望 **ccoctl** 实用程序在其中创建对象的目录。默认情况下，实用程序在运行

3 可选：指定包含凭证数据 YAML 文件的目录。默认情况下，**ccoctl** 预期此文件位于 **<home_directory>/.nutanix/credentials** 中。

5. 编辑 **install-config.yaml** 配置文件，使 **credentialsMode** 参数设置为 **Manual**。

install-config.yaml 配置文件示例

```
apiVersion: v1
baseDomain: cluster1.example.com
credentialsMode: Manual 1
...
```

1 添加这一行，将 **credentialsMode** 参数设置为 **Manual**。

6. 运行以下命令来创建安装清单：

```
$ openshift-install create manifests --dir <installation_directory> 1
```

1 指定包含集群的 **install-config.yaml** 文件的目录路径。

7. 运行以下命令，将生成的凭证文件复制到目标清单目录中：

```
$ cp <ccoctl_output_dir>/manifests/*credentials.yaml ./<installation_directory>/manifests
```

验证

- 确保 **manifests** 目录中存在适当的 secret。

```
$ ls ./<installation_directory>/manifests
```

输出示例

```
cluster-config.yaml
cluster-dns-02-config.yml
cluster-infrastructure-02-config.yml
cluster-ingress-02-config.yml
cluster-network-01-crd.yml
cluster-network-02-config.yml
cluster-proxy-01-config.yaml
cluster-scheduler-02-config.yml
cvo-overrides.yaml
kube-cloud-config.yaml
kube-system-configmap-root-ca.yaml
machine-config-server-tls-secret.yaml
openshift-config-secret-pull-secret.yaml
openshift-cloud-controller-manager-nutanix-credentials-credentials.yaml
openshift-machine-api-nutanix-credentials-credentials.yaml
```

11.3.10. 添加 Nutanix CCM 所需的配置映射和 secret 资源

在 Nutanix 上安装需要额外的 **ConfigMap** 和 **Secret** 资源，才能与 Nutanix Cloud Controller Manager (CCM) 集成。

先决条件

- 您已在安装目录中创建了 **manifests** 目录。

流程

1. 进入 **manifests** 目录：

```
$ cd <path_to_installation_directory>/manifests
```

2. 创建名为 **openshift-cloud-controller-manager-cloud-config.yaml** 的 **cloud-conf ConfigMap** 文件并添加以下信息：

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: cloud-conf
  namespace: openshift-cloud-controller-manager
data:
  cloud.conf: "{
     \"prismCentral\": {
        \"address\": \"<prism_central_FQDN/IP>\", (1)
        \"port\": 9440,
         \"credentialRef\": {
            \"kind\": \"Secret\",
            \"name\": \"nutanix-credentials\",
            \"namespace\": \"openshift-cloud-controller-manager\"
         }
      },
      \"topologyDiscovery\": {
         \"type\": \"Prism\",
         \"topologyCategories\": null
      },
      \"enableCustomLabeling\": true
    }"
```

(1) 指定 Prism Central FQDN/IP。

3. 验证文件 **cluster-infrastructure-02-config.yml** 是否存在，并且具有以下信息：

```
spec:
  cloudConfig:
    key: config
    name: cloud-provider-config
```

11.3.11. 用户管理的负载均衡器的服务

您可以将 OpenShift Container Platform 集群配置为使用用户管理的负载均衡器来代替默认负载均衡器。

重要

配置用户管理的负载均衡器取决于您的厂商的负载均衡器。

本节中的信息和示例仅用于指导目的。有关供应商负载均衡器的更多信息，请参阅供应商文档。

红帽支持用户管理的负载均衡器的以下服务：

- Ingress Controller
- OpenShift API
- OpenShift MachineConfig API

您可以选择是否要为用户管理的负载均衡器配置一个或多个所有服务。仅配置 Ingress Controller 服务是一个通用的配置选项。要更好地了解每个服务，请查看以下图表：

图 11.1. 显示 OpenShift Container Platform 环境中运行的 Ingress Controller 的网络工作流示例

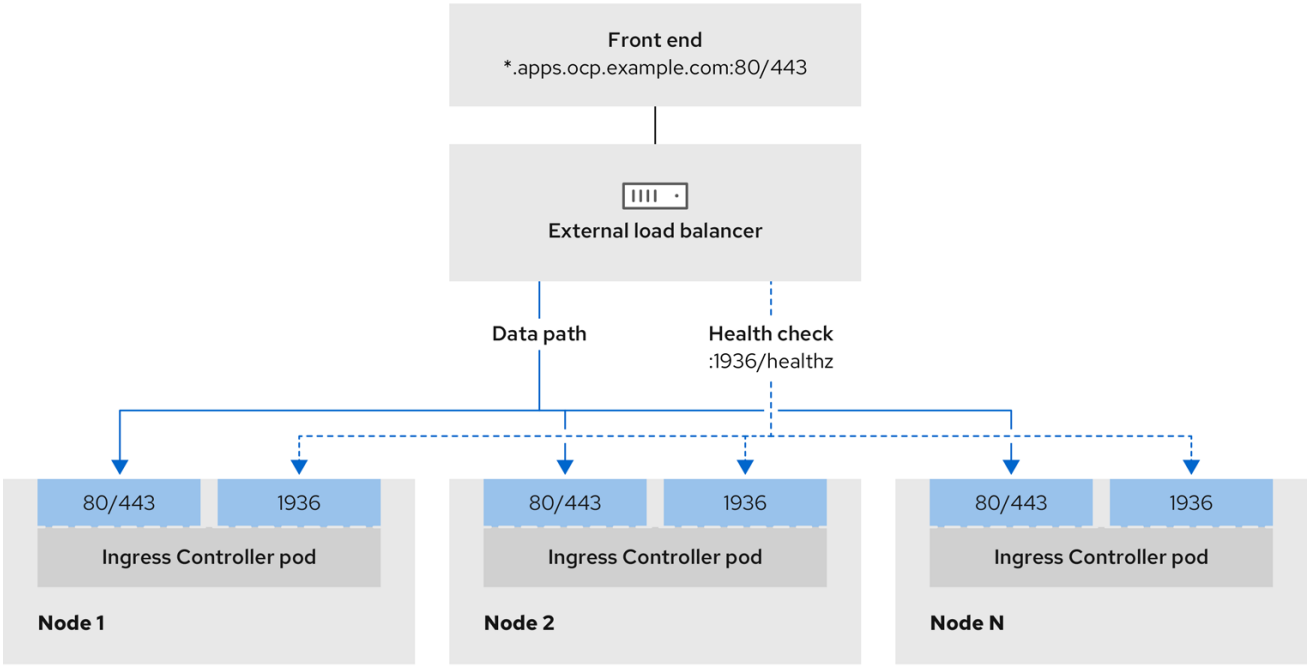


图 11.2. 显示 OpenShift Container Platform 环境中运行的 OpenShift API 的网络工作流示例

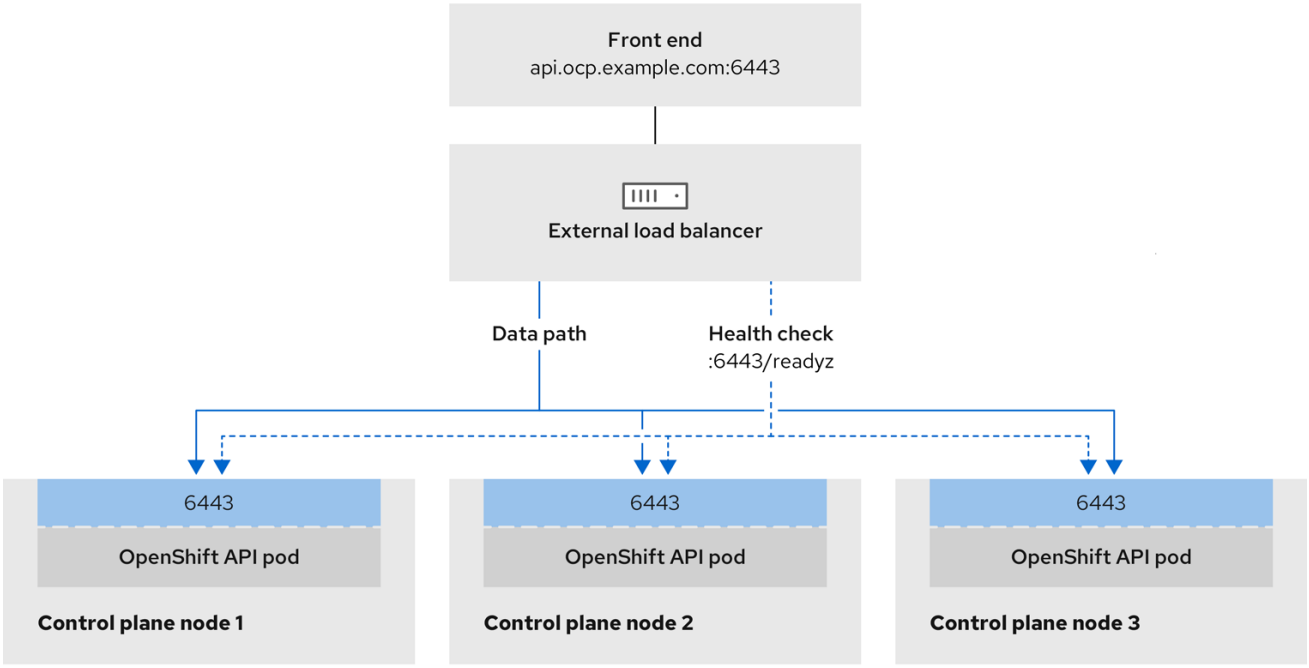
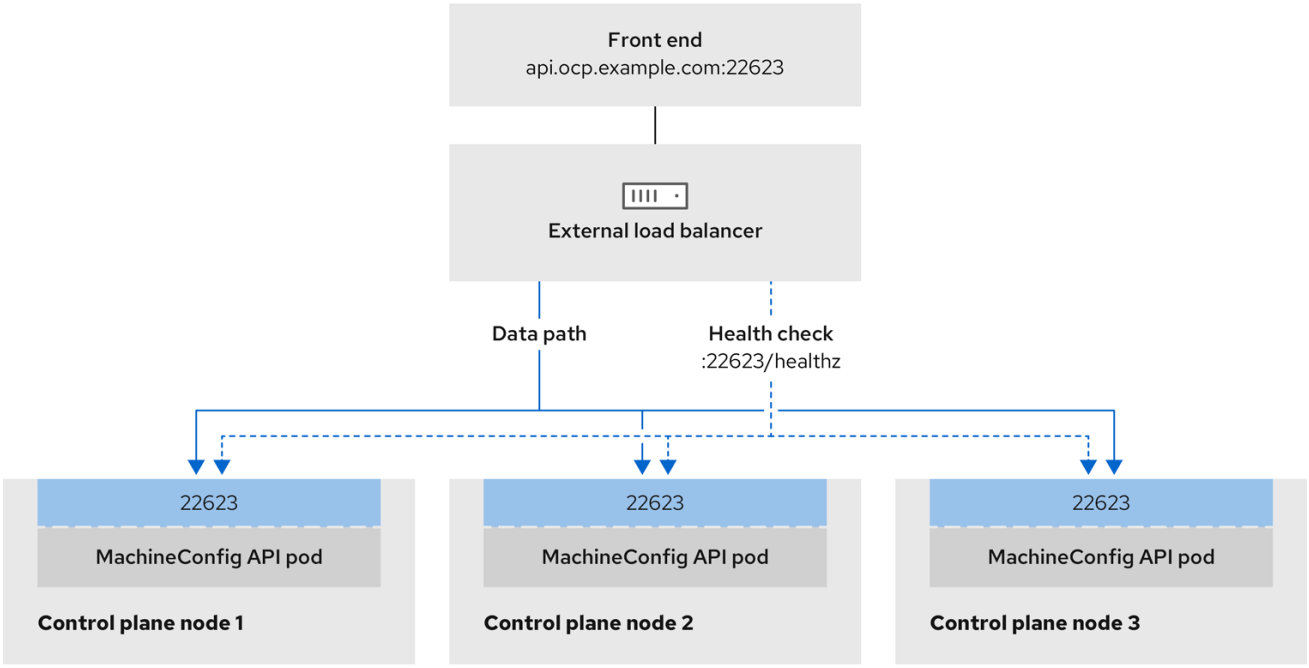


图 11.3. 显示 OpenShift Container Platform 环境中运行的 OpenShift MachineConfig API 的网络工作流示例



用户管理的负载均衡器支持以下配置选项：

- 使用节点选择器将 Ingress Controller 映射到一组特定的节点。您必须为这个集合中的每个节点分配一个静态 IP 地址，或者将每个节点配置为从动态主机配置协议(DHCP)接收相同的 IP 地址。基础架构节点通常接收这种类型的配置。

- 以子网上的所有 IP 地址为目标。此配置可减少维护开销，因为您可以在这些网络中创建和销毁节点，而无需重新配置负载均衡器目标。如果您使用较小的网络上的机器集来部署入口 pod，如 **/27** 或 **/28**，您可以简化负载均衡器目标。

  提示

  您可以通过检查机器配置池的资源来列出网络中存在的所有 IP 地址。

在为 OpenShift Container Platform 集群配置用户管理的负载均衡器前，请考虑以下信息：

- 对于前端 IP 地址，您可以对前端 IP 地址、Ingress Controller 的负载均衡器和 API 负载均衡器使用相同的 IP 地址。查看厂商的文档以获取此功能的相关信息。
- 对于后端 IP 地址，请确保 OpenShift Container Platform control plane 节点的 IP 地址在用户管理的负载均衡器生命周期内不会改变。您可以通过完成以下操作之一来实现此目的：
  - 为每个 control plane 节点分配一个静态 IP 地址。
  - 将每个节点配置为在每次节点请求 DHCP 租期时从 DHCP 接收相同的 IP 地址。根据供应商，DHCP 租期可能采用 IP 保留或静态 DHCP 分配的形式。
- 在 Ingress Controller 后端服务的用户管理的负载均衡器中手动定义运行 Ingress Controller 的每个节点。例如，如果 Ingress Controller 移到未定义节点，则可能会出现连接中断。

11.3.11.1. 配置用户管理的负载均衡器

您可以将 OpenShift Container Platform 集群配置为使用用户管理的负载均衡器来代替默认负载均衡器。

重要

在配置用户管理的负载均衡器前，请确保阅读用户管理的负载均衡器部分。

阅读适用于您要为用户管理的负载均衡器配置的服务的以下先决条件。

注意

MetalLB，在集群中运行，充当用户管理的负载均衡器。

OpenShift API 的先决条件

- 您定义了前端 IP 地址。
- TCP 端口 6443 和 22623 在负载均衡器的前端 IP 地址上公开。检查以下项：
  - 端口 6443 提供对 OpenShift API 服务的访问。
  - 端口 22623 可以为节点提供 ignition 启动配置。
- 前端 IP 地址和端口 6443 可以被您的系统的所有用户访问，其位置为 OpenShift Container Platform 集群外部。
- 前端 IP 地址和端口 22623 只能被 OpenShift Container Platform 节点访问。
- 负载均衡器后端可以在端口 6443 和 22623 上与 OpenShift Container Platform control plane 节点通信。

Ingress Controller 的先决条件

- 您定义了前端 IP 地址。
- TCP 端口 443 和 80 在负载均衡器的前端 IP 地址上公开。
- 前端 IP 地址、端口 80 和端口 443 可以被您的系统所有用户访问，以及 OpenShift Container Platform 集群外部的位置。
- 前端 IP 地址、端口 80 和端口 443 可被 OpenShift Container Platform 集群中运行的所有节点访问。
- 负载均衡器后端可以在端口 80、443 和 1936 上与运行 Ingress Controller 的 OpenShift Container Platform 节点通信。

健康检查 URL 规格的先决条件

您可以通过设置健康检查 URL 来配置大多数负载均衡器，以确定服务是否可用或不可用。OpenShift Container Platform 为 OpenShift API、Machine Configuration API 和 Ingress Controller 后端服务提供这些健康检查。

以下示例显示了之前列出的后端服务的健康检查规格：

Kubernetes API 健康检查规格示例

```
Path: HTTPS:6443/readyz
Healthy threshold: 2
Unhealthy threshold: 2
Timeout: 10
Interval: 10
```

Machine Config API 健康检查规格示例

```
Path: HTTPS:22623/healthz
Healthy threshold: 2
Unhealthy threshold: 2
Timeout: 10
Interval: 10
```

Ingress Controller 健康检查规格示例

```
Path: HTTP:1936/healthz/ready
Healthy threshold: 2
Unhealthy threshold: 2
Timeout: 5
Interval: 10
```

流程

1. 配置 HAProxy Ingress Controller，以便您可以在端口 6443、22623、443 和 80 上从负载均衡器访问集群。根据您的需要，您可以在 HAProxy 配置中指定来自多个子网的单个子网或 IP 地址的 IP 地址。

   带有列出子网的 HAProxy 配置示例

```
# ...
listen my-cluster-api-6443
    bind 192.168.1.100:6443
    mode tcp
    balance roundrobin
  option httpchk
  http-check connect
  http-check send meth GET uri /readyz
  http-check expect status 200
    server my-cluster-master-2 192.168.1.101:6443 check inter 10s rise 2 fall 2
    server my-cluster-master-0 192.168.1.102:6443 check inter 10s rise 2 fall 2
    server my-cluster-master-1 192.168.1.103:6443 check inter 10s rise 2 fall 2

listen my-cluster-machine-config-api-22623
    bind 192.168.1.100:22623
    mode tcp
    balance roundrobin
  option httpchk
  http-check connect
  http-check send meth GET uri /healthz
  http-check expect status 200
    server my-cluster-master-2 192.168.1.101:22623 check inter 10s rise 2 fall 2
    server my-cluster-master-0 192.168.1.102:22623 check inter 10s rise 2 fall 2
    server my-cluster-master-1 192.168.1.103:22623 check inter 10s rise 2 fall 2

listen my-cluster-apps-443
    bind 192.168.1.100:443
    mode tcp
    balance roundrobin
  option httpchk
  http-check connect
  http-check send meth GET uri /healthz/ready
  http-check expect status 200
    server my-cluster-worker-0 192.168.1.111:443 check port 1936 inter 10s rise 2 fall 2
    server my-cluster-worker-1 192.168.1.112:443 check port 1936 inter 10s rise 2 fall 2
    server my-cluster-worker-2 192.168.1.113:443 check port 1936 inter 10s rise 2 fall 2

listen my-cluster-apps-80
   bind 192.168.1.100:80
   mode tcp
   balance roundrobin
  option httpchk
  http-check connect
  http-check send meth GET uri /healthz/ready
  http-check expect status 200
    server my-cluster-worker-0 192.168.1.111:80 check port 1936 inter 10s rise 2 fall 2
    server my-cluster-worker-1 192.168.1.112:80 check port 1936 inter 10s rise 2 fall 2
    server my-cluster-worker-2 192.168.1.113:80 check port 1936 inter 10s rise 2 fall 2
# ...
```

带有多个列出子网的 HAProxy 配置示例

```
# ...
listen api-server-6443
    bind *:6443
```

```
    mode tcp
      server master-00 192.168.83.89:6443 check inter 1s
      server master-01 192.168.84.90:6443 check inter 1s
      server master-02 192.168.85.99:6443 check inter 1s
      server bootstrap 192.168.80.89:6443 check inter 1s

listen machine-config-server-22623
    bind *:22623
    mode tcp
      server master-00 192.168.83.89:22623 check inter 1s
      server master-01 192.168.84.90:22623 check inter 1s
      server master-02 192.168.85.99:22623 check inter 1s
      server bootstrap 192.168.80.89:22623 check inter 1s

listen ingress-router-80
    bind *:80
    mode tcp
    balance source
      server worker-00 192.168.83.100:80 check inter 1s
      server worker-01 192.168.83.101:80 check inter 1s

listen ingress-router-443
    bind *:443
    mode tcp
    balance source
      server worker-00 192.168.83.100:443 check inter 1s
      server worker-01 192.168.83.101:443 check inter 1s

listen ironic-api-6385
    bind *:6385
    mode tcp
    balance source
      server master-00 192.168.83.89:6385 check inter 1s
      server master-01 192.168.84.90:6385 check inter 1s
      server master-02 192.168.85.99:6385 check inter 1s
      server bootstrap 192.168.80.89:6385 check inter 1s

listen inspector-api-5050
    bind *:5050
    mode tcp
    balance source
      server master-00 192.168.83.89:5050 check inter 1s
      server master-01 192.168.84.90:5050 check inter 1s
      server master-02 192.168.85.99:5050 check inter 1s
      server bootstrap 192.168.80.89:5050 check inter 1s
# ...
```

2. 使用 **curl** CLI 命令验证用户管理的负载均衡器及其资源是否正常运行：

   a. 运行以下命令并查看响应，验证集群机器配置 API 是否可以被 Kubernetes API 服务器资源访问：

      ```
      $ curl https://<loadbalancer_ip_address>:6443/version --insecure
      ```

      如果配置正确，您会收到 JSON 对象的响应：

```
{
  "major": "1",
  "minor": "11+",
  "gitVersion": "v1.11.0+ad103ed",
  "gitCommit": "ad103ed",
  "gitTreeState": "clean",
  "buildDate": "2019-01-09T06:44:10Z",
  "goVersion": "go1.10.3",
  "compiler": "gc",
  "platform": "linux/amd64"
}
```

b. 运行以下命令并观察输出，验证集群机器配置 API 是否可以被 Machine 配置服务器资源访问：

```
$ curl -v https://<loadbalancer_ip_address>:22623/healthz --insecure
```

如果配置正确，命令的输出会显示以下响应：

```
HTTP/1.1 200 OK
Content-Length: 0
```

c. 运行以下命令并观察输出，验证控制器是否可以被端口 80 上的 Ingress Controller 资源访问：

```
$ curl -I -L -H "Host: console-openshift-console.apps.<cluster_name>.<base_domain>"
http://<load_balancer_front_end_IP_address>
```

如果配置正确，命令的输出会显示以下响应：

```
HTTP/1.1 302 Found
content-length: 0
location: https://console-openshift-console.apps.ocp4.private.opequon.net/
cache-control: no-cache
```

d. 运行以下命令并观察输出，验证控制器是否可以被端口 443 上的 Ingress Controller 资源访问：

```
$ curl -I -L --insecure --resolve console-openshift-console.apps.<cluster_name>.
<base_domain>:443:<Load Balancer Front End IP Address> https://console-openshift-
console.apps.<cluster_name>.<base_domain>
```

如果配置正确，命令的输出会显示以下响应：

```
HTTP/1.1 200 OK
referrer-policy: strict-origin-when-cross-origin
set-cookie: csrf-
token=UlYWOyQ62LWjw2h003xtYSKlh1a0Py2hhctw0WmV2YEdhJjFyQwWcGBsja261dG
LgaYO0nxzVErhiXt6QepA7g==; Path=/; Secure; SameSite=Lax
x-content-type-options: nosniff
x-dns-prefetch-control: off
x-frame-options: DENY
x-xss-protection: 1; mode=block
```

```
date: Wed, 04 Oct 2023 16:29:38 GMT
content-type: text/html; charset=utf-8
set-cookie:
1e2670d92730b515ce3a1bb65da45062=1bf5e9573c9a2760c964ed1659cc1673; path=/;
HttpOnly; Secure; SameSite=None
cache-control: private
```

3. 配置集群的 DNS 记录，使其以用户管理的负载均衡器的前端 IP 地址为目标。您必须在负载均衡器上将记录更新为集群 API 和应用程序的 DNS 服务器。

修改 DNS 记录示例

```
<load_balancer_ip_address>  A  api.<cluster_name>.<base_domain>
A record pointing to Load Balancer Front End
```

```
<load_balancer_ip_address>   A apps.<cluster_name>.<base_domain>
A record pointing to Load Balancer Front End
```

重要

DNS 传播可能需要一些时间才能获得每个 DNS 记录。在验证每个记录前，请确保每个 DNS 记录传播。

4. 要使 OpenShift Container Platform 集群使用用户管理的负载均衡器，您必须在集群的 **install-config.yaml** 文件中指定以下配置：

```
# ...
platform:
  nutanix:
    loadBalancer:
      type: UserManaged 1
      apiVIPs:
      - <api_ip> 2
      ingressVIPs:
      - <ingress_ip> 3
# ...
```

1. 为 **type** 参数设置 **UserManaged**，为集群指定用户管理的负载均衡器。参数默认为 **OpenShiftManagedDefault**，它表示默认的内部负载均衡器。对于 **openshift-kni-infra** 命名空间中定义的服务，用户管理的负载均衡器可将 **coredns** 服务部署到集群中的 pod，但忽略 **keepalived** 和 **haproxy** 服务。

2. 指定用户管理的负载均衡器时所需的参数。指定用户管理的负载均衡器的公共 IP 地址，以便 Kubernetes API 可以与用户管理的负载均衡器通信。

3. 指定用户管理的负载均衡器时所需的参数。指定用户管理的负载均衡器的公共 IP 地址，以便用户管理的负载均衡器可以管理集群的入口流量。

验证

1. 使用 **curl** CLI 命令验证用户管理的负载均衡器和 DNS 记录配置是否正常工作：

a. 运行以下命令并查看输出，验证您可以访问集群 API：

```
$ curl https://api.<cluster_name>.<base_domain>:6443/version --insecure
```

如果配置正确，您会收到 JSON 对象的响应：

```
{
  "major": "1",
  "minor": "11+",
  "gitVersion": "v1.11.0+ad103ed",
  "gitCommit": "ad103ed",
  "gitTreeState": "clean",
  "buildDate": "2019-01-09T06:44:10Z",
  "goVersion": "go1.10.3",
  "compiler": "gc",
  "platform": "linux/amd64"
  }
```

b. 运行以下命令并查看输出，验证您可以访问集群机器配置：

```
$ curl -v https://api.<cluster_name>.<base_domain>:22623/healthz --insecure
```

如果配置正确，命令的输出会显示以下响应：

```
HTTP/1.1 200 OK
Content-Length: 0
```

c. 运行以下命令并查看输出，验证您可以在端口上访问每个集群应用程序：

```
$ curl http://console-openshift-console.apps.<cluster_name>.<base_domain> -I -L --
insecure
```

如果配置正确，命令的输出会显示以下响应：

```
HTTP/1.1 302 Found
content-length: 0
location: https://console-openshift-console.apps.<cluster-name>.<base domain>/
cache-control: no-cacheHTTP/1.1 200 OK
referrer-policy: strict-origin-when-cross-origin
set-cookie: csrf-
token=39HoZgztDnzjJkq/JuLJMeoKNXlfiVv2YgZc09c3TBOBU4NI6kDXaJH1LdicNhN1UsQ
Wzon4Dor9GWGfopaTEQ==; Path=/; Secure
x-content-type-options: nosniff
x-dns-prefetch-control: off
x-frame-options: DENY
x-xss-protection: 1; mode=block
date: Tue, 17 Nov 2020 08:42:10 GMT
content-type: text/html; charset=utf-8
set-cookie:
1e2670d92730b515ce3a1bb65da45062=9b714eb87e93cf34853e87a92d6894be; path=/;
HttpOnly; Secure; SameSite=None
cache-control: private
```

d. 运行以下命令并查看输出，验证您可以在端口 443 上访问每个集群应用程序：

```
$ curl https://console-openshift-console.apps.<cluster_name>.<base_domain> -I -L --
insecure
```

如果配置正确，命令的输出会显示以下响应：

```
HTTP/1.1 200 OK
referrer-policy: strict-origin-when-cross-origin
set-cookie: csrf-
token=UlYWOyQ62LWjw2h003xtYSKlh1a0Py2hhctw0WmV2YEdhJjFyQwWcGBsja261dG
LgaYO0nxzVErhiXt6QepA7g==; Path=/; Secure; SameSite=Lax
x-content-type-options: nosniff
x-dns-prefetch-control: off
x-frame-options: DENY
x-xss-protection: 1; mode=block
date: Wed, 04 Oct 2023 16:29:38 GMT
content-type: text/html; charset=utf-8
set-cookie:
1e2670d92730b515ce3a1bb65da45062=1bf5e9573c9a2760c964ed1659cc1673; path=/;
HttpOnly; Secure; SameSite=None
cache-control: private
```

11.3.12. 部署集群

您可以在兼容云平台上安装 OpenShift Container Platform。

重要

在初始安装过程中，您只能运行安装程序的 **create cluster** 命令一次。

先决条件

- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。
- 已确认主机上的云供应商帐户具有部署集群的正确权限。权限不正确的帐户会导致安装过程失败，并显示包括缺失权限的错误消息。

流程

- 进入包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1. 对于 **<installation_directory>**，请指定自定义 **./install-config.yaml** 文件的位置。
2. 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不是 **info**。

验证

当集群部署成功完成时：

- 终端会显示用于访问集群的说明，包括指向 Web 控制台和 **kubeadmin** 用户的凭证的链接。

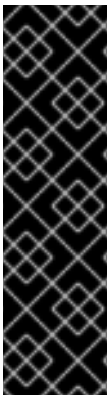
- 凭证信息还会输出到 **<installation_directory>/.openshift_install.log**.

重要

不要删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

11.3.13. 配置默认存储容器

安装集群后，您必须安装 Nutanix CSI Operator 并为集群配置默认存储容器。

如需更多信息，请参阅 Nutanix 文档安装 CSI Operator 和配置 registry 存储。

11.3.14. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

11.3.15. 其他资源

- 关于远程健康监控

11.3.16. 后续步骤

- 选择不使用远程健康报告

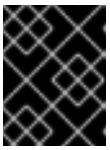
- 自定义集群

11.4. 在受限网络中的 NUTANIX 上安装集群

在 OpenShift Container Platform 4.16 中，您可以通过创建安装发行内容的内部镜像在受限网络中的 Nutanix 基础架构上安装集群。

11.4.1. 先决条件

- 您已查看了有关 OpenShift Container Platform 安装和更新流程 的详细信息。
- 安装程序需要访问 Prism Central 和 Prism Element 上的端口 9440。您确认可以访问端口 9440。
- 如果使用防火墙，已满足以下先决条件：
  - 已确认可以访问端口 9440。control plane 节点必须能够访问端口 9440 上的 Prism Central 和 Prism Element 才能成功安装。
  - 您已将防火墙配置为授予访问 OpenShift Container Platform 所需站点的访问权限。这包括使用 Telemetry。
- 如果您的 Nutanix 环境使用默认自签名 SSL/TLS 证书，请将它替换为 CA 签名的证书。安装程序需要一个有效的 CA 签名证书来访问 Prism Central API。有关替换自签名证书的更多信息，请参阅 Nutanix AOS 安全指南。
  如果您的 Nutanix 环境使用内部 CA 来发布证书，您必须将集群范围代理配置为安装过程的一部分。如需更多信息，请参阅 配置自定义 PKI。



重要

使用 2048 位证书。如果您使用带有 Prism Central 2022.x 的 4096 位证书，则安装会失败。

- 您有一个容器镜像 registry，如 Red Hat Quay。如果您还没有 registry，您可以使用 mirror registry *for Red Hat OpenShift 创建镜像 registry*。
- 您已使用 oc-mirror OpenShift CLI (oc) 插件 将所有所需的 OpenShift Container Platform 内容和其他镜像（包括 Nutanix CSI Operator）镜像到您的 mirror registry。

重要

由于安装介质位于镜像主机上，因此您可以使用该计算机完成所有安装步骤。

11.4.2. 关于在受限网络中安装

在 OpenShift Container Platform 4.16 中，可以执行不需要有效的互联网连接来获取软件组件的安装。受限网络安装可以使用安装程序置备的基础架构或用户置备的基础架构完成，具体取决于您要安装集群的云平台。

如果您选择在云平台中执行受限网络安装，您仍需要访问其云 API。有些云功能，比如 Amazon Web Service 的 Route 53 DNS 和 IAM 服务，需要访问互联网。根据您的网络，在裸机硬件、Nutanix 或 VMware vSphere 上安装可能需要较少的互联网访问。

要完成受限网络安装，您必须创建一个 registry，以镜像 OpenShift 镜像 registry 的内容并包含安装介质。您可以在镜像主机上创建此 registry，该主机可同时访问互联网和您的封闭网络，也可以使用满足您的限制条件的其他方法。

11.4.2.1. 其他限制

受限网络中的集群有以下额外限制和限制：

- **ClusterVersion** 状态包含一个 **Unable to retrieve available updates** 错误。
- 默认情况下，您无法使用 Developer Catalog 的内容，因为您无法访问所需的镜像流标签。

11.4.3. 为集群节点 SSH 访问生成密钥对

在 OpenShift Container Platform 安装过程中，您可以为安装程序提供 SSH 公钥。密钥通过它们的 Ignition 配置文件传递给 Red Hat Enterprise Linux CoreOS(RHCOS)节点，用于验证对节点的 SSH 访问。密钥添加到每个节点上 **core** 用户的 **~/.ssh/authorized_keys** 列表中，这将启用免密码身份验证。

将密钥传递给节点后，您可以使用密钥对作为用户 **核心** 通过 SSH 连接到 RHCOS 节点。若要通过 SSH 访问节点，必须由 SSH 为您的本地用户管理私钥身份。

如果要通过 SSH 连接到集群节点来执行安装调试或灾难恢复，则必须在安装过程中提供 SSH 公钥。**./openshift-install gather** 命令还需要在集群节点上设置 SSH 公钥。

重要

不要在生产环境中跳过这个过程，在生产环境中需要灾难恢复和调试。

注意

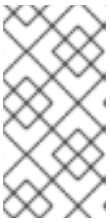
您必须使用本地密钥，而不是使用特定平台方法配置 的密钥，如 AWS 密钥对。

流程

1. 如果您在本地计算机上没有可用于在集群节点上进行身份验证的现有 SSH 密钥对，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_ed25519**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。



注意

如果您计划在 **x86_64**、**ppc64le** 和 **s390x** 架构上安装使用 RHEL 加密库（这些加密库已提交给 NIST 用于 FIPS 140-2/140-3 验证）的 OpenShift Container Platform 集群，则不要创建使用 **ed25519** 算法的密钥。相反，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

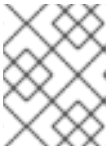
2. 查看公共 SSH 密钥：

```
$ cat <path>/<file_name>.pub
```

例如，运行以下命令来查看 ~/**.ssh/id_ed25519.pub** 公钥：

```
$ cat ~/.ssh/id_ed25519.pub
```

3. 将 SSH 私钥身份添加到本地用户的 SSH 代理（如果尚未添加）。在集群节点上，或者要使用 **./openshift-install gather** 命令，需要对该密钥进行 SSH 代理管理，才能在集群节点上进行免密码 SSH 身份验证。



注意

在某些发行版中，自动管理默认 SSH 私钥身份，如 ~/**.ssh/id_rsa** 和 ~/**.ssh/id_dsa**。

a. 如果 **ssh-agent** 进程尚未为您的本地用户运行，请将其作为后台任务启动：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果集群处于 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

4. 将 SSH 私钥添加到 **ssh-agent** ：

```
$ ssh-add <path>/<file_name> 1
```

1 指定 SSH 私钥的路径和文件名，如 ~/**.ssh/id_ed25519.pub**

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

后续步骤

- 安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

11.4.4. 在您的系统信任中添加 Nutanix root CA 证书

因为安装程序需要访问 Prism Central API，所以您必须在安装 OpenShift Container Platform 集群前将 Nutanix 可信根 CA 证书添加到您的系统信任中。

流程

1. 在 Prism Central web 控制台中下载 Nutanix root CA 证书。
2. 提取包含 Nutanix root CA 证书的压缩文件。

3. 将您的操作系统的文件添加到系统信任中。例如，在 Fedora 操作系统中运行以下命令：

```
# cp certs/lin/* /etc/pki/ca-trust/source/anchors
```

4. 更新您的系统信任关系。例如，在 Fedora 操作系统中运行以下命令：

```
# update-ca-trust extract
```

11.4.5. 下载 RHCOS 集群镜像

Prism Central 需要访问 Red Hat Enterprise Linux CoreOS (RHCOS) 镜像来安装集群。您可以使用安装程序查找并下载 RHCOS 镜像，并通过内部 HTTP 服务器或 Nutanix 对象提供它。

先决条件

- 获取 OpenShift Container Platform 安装程序和集群的 pull secret。对于受限网络安装，这些文件位于您的镜像主机上。

流程

1. 进入包含安装程序的目录并运行以下命令：

```
$ ./openshift-install coreos print-stream-json
```

2. 使用命令的输出来查找 Nutanix 镜像的位置，然后点链接下载它。

输出示例

```
"nutanix": {
  "release": "411.86.202210041459-0",
  "formats": {
    "qcow2": {
      "disk": {
        "location": "https://rhcos.mirror.openshift.com/art/storage/releases/rhcos-
4.11/411.86.202210041459-0/x86_64/rhcos-411.86.202210041459-0-
nutanix.x86_64.qcow2",
        "sha256":
"42e227cac6f11ac37ee8a2f9528bb3665146566890577fd55f9b950949e5a54b"
```

3. 通过内部 HTTP 服务器或 Nutanix 对象提供镜像。

4. 记录下载镜像的位置。在部署集群前，您可以使用镜像的位置更新安装配置文件中的 **platform** 部分 (**install-config.yaml**)。

指定 RHCOS 镜像的 install-config.yaml 文件的片段

```
platform:
  nutanix:
    clusterOSImage: http://example.com/images/rhcos-411.86.202210041459-0-
nutanix.x86_64.qcow2
```

11.4.6. 创建安装配置文件

您可以自定义在 Nutanix 上安装的 OpenShift Container Platform 集群。

先决条件

- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。对于受限网络安装，这些文件位于您的镜像主机上。
- 具有在镜像 registry 时创建的 **imageContentSourcePolicy.yaml** 文件。
- 具有您下载的 Red Hat Enterprise Linux CoreOS (RHCOS) 镜像的位置。
- 您已获取了镜像 registry 的证书内容。
- 您已检索了 Red Hat Enterprise Linux CoreOS (RHCOS) 镜像，并将其上传到可访问的位置。
- 您已确认满足了 Nutanix 网络要求。如需更多信息，请参阅"准备在 Nutanix 上安装"。

流程

1. 创建 **install-config.yaml** 文件。
   a. 进入包含安装程序的目录并运行以下命令：

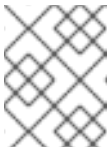
```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

   1 对于 **<installation_directory>**，请指定要存储安装程序创建的文件的目录名称。

   在指定目录时：

   - 验证该目录是否具有**执行**权限。在安装目录中运行 Terraform 二进制文件需要这个权限。
   - 使用空目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

   b. 在提示符处，提供云的配置详情：
      i. 可选： 选择用于访问集群机器的 SSH 密钥。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

      ii. 选择 **nutanix** 作为目标平台。
      iii. 输入 Prism Central 域名或 IP 地址。
      iv. 输入用于登录 Prism Central 的端口。
      v. 输入用于登录 Prism Central 的凭证。
      安装程序连接到 Prism Central。

vi. 选择用于管理 OpenShift Container Platform 集群的 Prism Element。

vii. 选择要使用的网络子网。

viii. 输入您为 control plane API 访问配置的虚拟 IP 地址。

ix. 输入您为集群入口配置的虚拟 IP 地址。

x. 输入基域。这个基域必须与您在 DNS 记录中配置的域相同。

xi. 为集群输入描述性名称。
您输入的集群名称必须与您在配置 DNS 记录时指定的集群名称匹配。

2. 在 **install-config.yaml** 文件中，将 **platform.nutanix.clusterOSImage** 的值设置为镜像位置或名称。例如：

```
platform:
  nutanix:
    clusterOSImage: http://mirror.example.com/images/rhcos-47.83.202103221318-0-
nutanix.x86_64.qcow2
```

3. 编辑 **install-config.yaml** 文件，以提供在受限网络中安装所需的额外信息。

a. 更新 **pullSecret** 值，使其包含 registry 的身份验证信息：

```
pullSecret: '{"auths":{"<mirror_host_name>:5000": {"auth": "<credentials>","email":
"you@example.com"}}}'
```

对于 **<mirror_host_name>**，请指定 您在镜像 registry 证书中指定的 registry 域名 ；对于 **<credentials>**， 请指定您的镜像 registry 的 base64 编码用户名和密码。

b. 添加 **additionalTrustBundle** 参数和值。

```
additionalTrustBundle: |
  -----BEGIN CERTIFICATE-----

ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
  -----END CERTIFICATE-----
```

该值必须是您用于镜像 registry 的证书文件内容。证书文件可以是现有的可信证书颁发机构，也可以是您为镜像 registry 生成的自签名证书。

c. 添加镜像内容资源，类似于以下 YAML 摘录：

```
imageContentSources:
- mirrors:
  - <mirror_host_name>:5000/<repo_name>/release
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <mirror_host_name>:5000/<repo_name>/release
  source: registry.redhat.io/ocp/release
```

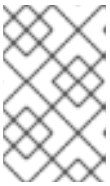
对于这些值，请使用镜像 registry 时创建的 **imageContentSourcePolicy.yaml** 文件。

d. 可选：将发布策略设置为 **Internal**：

```
publish: Internal
```

通过设置这个选项，您可以创建一个内部 Ingress Controller 和一个私有负载均衡器。

4. 可选：更新 **install.config.yaml** 文件中的一个或多个默认配置参数，以自定义安装。
有关参数的更多信息，请参阅"安装配置参数"。



注意

如果要安装三节点集群，请确保将 **compute.replicas** 参数设置为 **0**。这样可确保集群的 control plane 可以调度。如需更多信息，请参阅"在 {platform} 上安装三节点集群"。

5. 备份 **install-config.yaml** 文件，以便您可以使用它安装多个集群。

重要

install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用此文件，必须现在备份。

其他资源

- Nutanix 的安装配置参数

11.4.6.1. Nutanix 的自定义 install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多详情，或修改所需参数的值。

重要

此示例 YAML 文件仅供参考。您必须使用安装程序来获取 **install-config.yaml** 文件，并进行修改。

```
apiVersion: v1
baseDomain: example.com (1)
compute: (2)
- hyperthreading: Enabled (3)
  name: worker
  replicas: 3
  platform:
    nutanix: (4)
      cpus: 2
      coresPerSocket: 2
      memoryMiB: 8196
      osDisk:
        diskSizeGiB: 120
      categories: (5)
      - key: <category_key_name>
        value: <category_value>
controlPlane: (6)
  hyperthreading: Enabled (7)
```

```
  name: master
  replicas: 3
  platform:
    nutanix: 8
      cpus: 4
      coresPerSocket: 2
      memoryMiB: 16384
      osDisk:
        diskSizeGiB: 120
      categories: 9
      - key: <category_key_name>
        value: <category_value>
metadata:
  creationTimestamp: null
  name: test-cluster 10
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 11
  serviceNetwork:
  - 172.30.0.0/16
platform:
  nutanix:
    apiVIP: 10.40.142.7 12
    ingressVIP: 10.40.142.8 13
    defaultMachinePlatform:
      bootType: Legacy
      categories: 14
      - key: <category_key_name>
        value: <category_value>
      project: 15
        type: name
        name: <project_name>
    prismCentral:
      endpoint:
        address: your.prismcentral.domainname 16
        port: 9440 17
      password: <password> 18
      username: <username> 19
    prismElements:
    - endpoint:
        address: your.prismelement.domainname
        port: 9440
      uuid: 0005b0f1-8f43-a0f2-02b7-3cecef193712
    subnetUUIDs:
    - c7938dc6-7659-453e-a688-e26020c68e43
    clusterOSImage: http://example.com/images/rhcos-47.83.202103221318-0-nutanix.x86_64.qcow2
20
credentialsMode: Manual
publish: External
pullSecret: '{"auths":{"<local_registry>": {"auth": "<credentials>","email": "you@example.com"}}}' 21
```

```
fips: false (22)
sshKey: ssh-ed25519 AAAA... (23)
additionalTrustBundle: | (24)
  -----BEGIN CERTIFICATE-----
  ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
  -----END CERTIFICATE-----
imageContentSources: (25)
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-v4.0-art-dev
```

1 10 12 13 16 17 18 19 必需。安装程序会提示您输入这个值。

2 6 **controlPlane** 部分是一个单个映射，但 compute 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，controlPlane **部分** 的第一行则不以连字符开头。虽然这两个部分目前都定义了单一机器池，但未来的 OpenShift Container Platform 版本可能在安装过程中支持定义多个计算池。仅使用一个 control plane 池。

3 7 是否要启用或禁用并发多线程或 **超线程**。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将 参数值设置为 **Disabled** 来禁用它。如果在某些集群机器中禁用并发多线程，则必须在所有集群机器中禁用它。

重要

如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。

4 8 可选：为 compute 和 control plane 机器提供额外的机器池参数配置。

5 9 14 可选：提供一个或多个 prism category 键值对和一个 prism category 值。这些类别键值对必须在 Prism Central 中存在。您可以为计算机器、control plane 机器或所有机器提供单独的类别。

11 要安装的集群网络插件。默认值 **OVNKubernetes** 是唯一支持的值。

15 可选：指定与其关联的项目。为项目类型指定 **name** 或 **uuid**，然后提供对应的 UUID 或项目名称。您可以将项目与计算机器、control plane 机器或所有机器关联。

20 可选： 默认情况下，安装程序会下载并安装 Red Hat Enterprise Linux CoreOS (RHCOS) 镜像。如果 Prism Central 无法访问互联网，您可以通过在任何 HTTP 服务器或 Nutanix 对象或 Nutanix 对象上托管 RHCOS 镜像来覆盖默认行为，并将安装程序指向镜像。

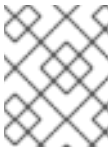
21 对于 **<local_registry>**，请指定 registry 域名，以及您的镜像 registry 用来提供内容的可选端口。例如 **registry.example.com 或 registry.example.com:5000**。对于 **<credentials>**，请为您的镜像 registry 指定 base64 编码的用户名和密码。

22 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS)时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。

23 可选：您可以提供用于访问集群中机器的 **sshKey** 值。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

24 提供用于镜像 registry 的证书文件内容。

25 从镜像 registry 时创建的 **imageContentSourcePolicy.yaml** 文件的 **metadata.name: release-0** 部分提供这些值。

11.4.6.2. 配置故障域

故障域通过在多个 Nutanix Prism Elements (clusters) 间分布 control plane 和计算机器来提高 OpenShift Container Platform 集群的容错功能。

提示

建议您配置三个故障域以确保高可用性。

先决条件

- 您有一个安装配置文件 (**install-config.yaml**)。

流程

1. 编辑 **install-config.yaml** 文件并添加以下小节来配置第一个故障域：

```
apiVersion: v1
baseDomain: example.com
compute:
# ...
platform:
  nutanix:
    failureDomains:
    - name: <failure_domain_name>
      prismElement:
        name: <prism_element_name>
        uuid: <prism_element_uuid>
      subnetUUIDs:
      - <network_uuid>
# ...
```

其中：

<failure_domain_name>

指定故障域的唯一名称。名称的长度不能超过 64 个字符，可以包括小写字母、数字和短划线 (-)。短划线不能是名称的第一个或最后一个。

<prism_element_name>

可选。指定 Prism Element 的名称。

<prism_element_uuid>

指定 Prism Element 的 UUID。

<network_uuid>

指定 Prism Element 子网对象的 UUID。子网的 IP 地址前缀 (CIDR) 应包含 OpenShift Container Platform 集群使用的虚拟 IP 地址。在 OpenShift Container Platform 集群中，只支持每个故障域 (Prism Element) 一个子网。

2. 根据需要，配置额外的故障域。

3. 要在故障域间分发 control plane 和计算机器，请执行以下操作之一：

   - 如果 compute 和 control plane 机器可以共享同一组故障域，请在集群的默认机器配置下添加故障域名称。

     共享一组故障域的 control plane 和计算机器示例

     ```
     apiVersion: v1
     baseDomain: example.com
     compute:
     # ...
     platform:
       nutanix:
         defaultMachinePlatform:
           failureDomains:
             - failure-domain-1
             - failure-domain-2
             - failure-domain-3
     # ...
     ```

   - 如果 compute 和 control plane 机器必须使用不同的故障域，请在对应的机器池下添加故障域名称。

     使用不同的故障域的 control plane 和计算机器示例

     ```
     apiVersion: v1
     baseDomain: example.com
     compute:
     # ...
     controlPlane:
       platform:
         nutanix:
           failureDomains:
             - failure-domain-1
             - failure-domain-2
             - failure-domain-3
     # ...
     compute:
       platform:
         nutanix:
           failureDomains:
     ```

```
        - failure-domain-1
        - failure-domain-2
# ...
```

4. 保存该文件。

11.4.6.3. 在安装过程中配置集群范围的代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 **Proxy** 对象的 **spec.noProxy** 字段中添加站点来绕过代理。

注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，**Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

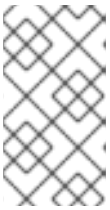
```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 **.** 以仅匹配子域。例如，**.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 * 绕过所有目的地的代理。
4. 如果提供，安装程序会在 **openshift-config** 命名空间中生成名为 **user-ca-bundle** 的配置映

5 可选：决定 **Proxy** 对象的配置以引用 **trustedCA** 字段中 **user-ca-bundle** 配置映射的策略。允许的值是 **Proxyonly** 和 **Always**。仅在配置了 **http/https** 代理时，使用 **Proxyonly**

注意

安装程序不支持代理的 **readinessEndpoints** 字段。



注意

如果安装程序超时，重启并使用安装程序的 **wait-for** 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. 保存该文件并在安装 OpenShift Container Platform 时引用。

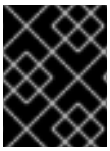
安装程序会创建一个名为 cluster 的集群范围代理，该代理 **使用** 提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

11.4.7. 安装 OpenShift CLI

您可以安装 OpenShift CLI(**oc**)来使用命令行界面与 OpenShift Container Platform 进行交互。您可以在 Linux、Windows 或 macOS 上安装 **oc**。



重要

如果安装了旧版本的 **oc**，则可能无法使用 OpenShift Container Platform 4.16 中的所有命令。下载并安装新版本的 **oc**。

在 Linux 上安装 OpenShift CLI
您可以按照以下流程在 Linux 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **产品变体** 下拉列表中选择架构。
3. 从 **版本** 下拉列表中选择适当的版本。
4. 点 **OpenShift v4.16 Linux Client** 条目旁的 **Download Now** 来保存文件。
5. 解包存档：

```
$ tar xvf <file>
```

6. 将 **oc** 二进制文件放到 **PATH** 中的目录中。
要查看您的 **PATH**，请执行以下命令：

```
```

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

在 Windows 上安装 OpenShift CLI
您可以按照以下流程在 Windows 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 Windows Client** 条目旁的 **Download Now** 来保存文件。
4. 使用 ZIP 程序解压存档。
5. 将 **oc** 二进制文件移到 **PATH 中的目录中**。
   要查看您的 **PATH**，请打开命令提示并执行以下命令：

```
C:\> path
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

在 macOS 上安装 OpenShift CLI
您可以按照以下流程在 macOS 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 macOS Client** 条目旁的 **Download Now** 来保存文件。

注意

对于 macOS arm64，请选择 **OpenShift v4.16 macOS arm64 Client** 条目。

4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
   要查看您的 **PATH**，请打开终端并执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

11.4.8. 为 Nutanix 配置 IAM

安装集群需要 Cloud Credential Operator(CCO)以手动模式运行。虽然安装程序为手动模式配置 CCO，您必须指定身份和访问管理 secret。

先决条件

- 您已配置了 **ccoctl** 二进制文件。
- 您有一个 **install-config.yaml** 文件。

流程

1. 按照以下格式创建一个包含凭证数据的 YAML 文件：

   凭证数据格式

```
credentials:
- type: basic_auth (1)
  data:
    prismCentral: (2)
      username: <username_for_prism_central>
      password: <password_for_prism_central>
    prismElements: (3)
    - name: <name_of_prism_element>
      username: <username_for_prism_element>
      password: <password_for_prism_element>
```

   (1) 指定身份验证类型。仅支持基本身份验证。

   (2) 指定 Prism Central 凭证。

   (3) 可选：指定 Prism Element 凭证。

2. 运行以下命令，使用安装文件中的发行镜像设置 **$RELEASE_IMAGE** 变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

3. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 **CredentialsRequest** 自定义资源 (CR) 列表：

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
```

```
--included \ 1
--install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
--to=<path_to_directory_for_credentials_requests> 3
```

1. **--included** 参数仅包含特定集群配置所需的清单。
2. 指定 **install-config.yaml** 文件的位置。
3. 指定要存储 **CredentialsRequest** 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

CredentialsRequest 对象示例

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  annotations:
    include.release.openshift.io/self-managed-high-availability: "true"
  labels:
    controller-tools.k8s.io: "1.0"
  name: openshift-machine-api-nutanix
  namespace: openshift-cloud-credential-operator
spec:
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: NutanixProviderSpec
  secretRef:
    name: nutanix-credentials
    namespace: openshift-machine-api
```

4. 运行以下命令，使用 **ccoctl** 工具处理所有 **CredentialsRequest** 对象：

```
$ ccoctl nutanix create-shared-secrets \
  --credentials-requests-dir=<path_to_credentials_requests_directory> \ 1
  --output-dir=<ccoctl_output_dir> \ 2
  --credentials-source-filepath=<path_to_credentials_file> 3
```

1. 指定包含组件 **CredentialsRequests** 对象文件的目录路径。
2. 可选：指定您希望 **ccoctl** 实用程序在其中创建对象的目录。默认情况下，实用程序在运行命令的目录中创建对象。
3. 可选：指定包含凭证数据 YAML 文件的目录。默认情况下，**ccoctl** 预期此文件位于 **<home_directory>/.nutanix/credentials** 中。

5. 编辑 **install-config.yaml** 配置文件，使 **credentialsMode** 参数设置为 **Manual**。

install-config.yaml 配置文件示例

```
apiVersion: v1
baseDomain: cluster1.example.com
credentialsMode: Manual 1
```

```
...
```

1 添加这一行，将 **credentialsMode** 参数设置为 **Manual**。

6. 运行以下命令来创建安装清单：

```
$ openshift-install create manifests --dir <installation_directory> 1
```

1 指定包含集群的 **install-config.yaml** 文件的目录路径。

7. 运行以下命令，将生成的凭证文件复制到目标清单目录中：

```
$ cp <ccoctl_output_dir>/manifests/*credentials.yaml ./<installation_directory>/manifests
```

验证

- 确保 **manifests** 目录中存在适当的 secret。

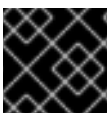
```
$ ls ./<installation_directory>/manifests
```

输出示例

```
cluster-config.yaml
cluster-dns-02-config.yml
cluster-infrastructure-02-config.yml
cluster-ingress-02-config.yml
cluster-network-01-crd.yml
cluster-network-02-config.yml
cluster-proxy-01-config.yaml
cluster-scheduler-02-config.yml
cvo-overrides.yaml
kube-cloud-config.yaml
kube-system-configmap-root-ca.yaml
machine-config-server-tls-secret.yaml
openshift-config-secret-pull-secret.yaml
openshift-cloud-controller-manager-nutanix-credentials-credentials.yaml
openshift-machine-api-nutanix-credentials-credentials.yaml
```

11.4.9. 部署集群

您可以在兼容云平台上安装 OpenShift Container Platform。



重要

在初始安装过程中，您只能运行安装程序的 **create cluster** 命令一次。

先决条件

- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。

- 已确认主机上的云供应商帐户具有部署集群的正确权限。权限不正确的帐户会导致安装过程失败，并显示包括缺失权限的错误消息。

流程

- 进入包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定自定义 **./install-config.yaml** 文件的位置。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不是 **info**。

验证

当集群部署成功完成时：

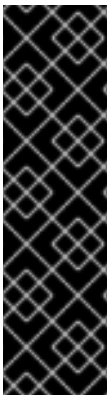
- 终端会显示用于访问集群的说明，包括指向 Web 控制台和 **kubeadmin** 用户的凭证的链接。
- 凭证信息还会输出到 **<installation_directory>/.openshift_install.log**.

重要

不要删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

11.4.10. 安装后

完成以下步骤以完成集群的配置。

11.4.10.1. 禁用默认的 OperatorHub 目录源

在 OpenShift Container Platform 安装过程中，默认为 OperatorHub 配置由红帽和社区项目提供的源内容的 operator 目录。在受限网络环境中，必须以集群管理员身份禁用默认目录。

流程

- 通过在 **OperatorHub** 对象中添加 **disableAllDefaultSources: true 来** 禁用默认目录的源：

```
$ oc patch OperatorHub cluster --type json \
    -p '[{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]'
```

提示

或者，您可以使用 Web 控制台管理目录源。在 **Administration → Cluster Settings → Configuration → OperatorHub** 页面中，点 **Sources** 选项卡，您可以在其中创建、更新、删除、禁用和启用单独的源。

11.4.10.2. 将策略资源安装到集群中

使用 oc-mirror OpenShift CLI (oc) 插件镜像 OpenShift Container Platform 内容会创建资源，其中包括 **catalogSource-certified-operator-index.yaml** 和 **imageContentSourcePolicy.yaml**。

- **ImageContentSourcePolicy** 资源将镜像 registry 与源 registry 关联，并将在线 registry 中的镜像拉取请求重定向到镜像 registry。
- Operator Lifecycle Manager (OLM) 使用 **CatalogSource** 资源来检索有关镜像 registry 中可用 Operator 的信息，允许用户发现和安装 Operator。

安装集群后，您必须将这些资源安装到集群中。

先决条件

- 您已将镜像设置为断开连接的环境中的 registry 镜像。
- 您可以使用具有 **cluster-admin** 角色的用户访问集群。

流程

1. 以具有 **cluster-admin** 角色的用户身份登录 OpenShift CLI。
2. 将结果目录中的 YAML 文件应用到集群：

```
$ oc apply -f ./oc-mirror-workspace/results-<id>/
```

验证

1. 验证 **ImageContentSourcePolicy** 资源是否已成功安装：

```
$ oc get imagecontentsourcepolicy
```

2. 验证 **CatalogSource** 资源是否已成功安装：

```
$ oc get catalogsource --all-namespaces
```

11.4.10.3. 配置默认存储容器

安装集群后，您必须安装 Nutanix CSI Operator 并为集群配置默认存储容器。

如需更多信息，请参阅 Nutanix 文档安装 CSI Operator 和配置 registry 存储。

11.4.11. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

11.4.12. 其他资源

- 关于远程健康监控

11.4.13. 后续步骤

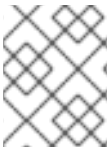
- 如果需要，请参阅 不使用远程健康报告
- 如果需要，请参阅 注册断开连接的集群
- 自定义集群

11.5. 在 NUTANIX 上安装三节点集群

在 OpenShift Container Platform 版本 4.16 中，您可以在 Nutanix 上安装三节点集群。三节点集群包含三个 control plane 机器，它们也可以充当计算机器。这种类型的集群提供了一个较小的、效率更高的集群，供集群管理员和开发人员用于测试、开发和生产。

11.5.1. 配置三节点集群

在部署集群前，您可以通过将 **install-config.yaml** 文件中的 worker 节点数量设置为 **0** 来配置三节点集群。将 worker 节点数量设置为 **0** 可确保 control plane 机器可以调度。这允许调度应用程序工作负载从 control plane 节点运行。



注意

因为应用程序工作负载从 control plane 节点运行，所以需要额外的订阅，因为 control plane 节点被视为计算节点。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。

流程

- 将 **install-config.yaml** 文件中的计算副本数量设置为 **0**，如以下 **compute** 小节中所示：

  三节点集群的 install-config.yaml 文件示例

```
apiVersion: v1
baseDomain: example.com
compute:
- name: worker
  platform: {}
  replicas: 0
# ...
```

11.5.2. 后续步骤

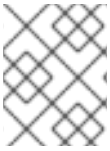
- 在 Nutanix 上安装集群

11.6. 在 NUTANIX 上卸载集群

您可以删除部署到 Nutanix 的集群。

11.6.1. 删除使用安装程序置备的基础架构的集群

您可以从云中删除使用安装程序置备的基础架构的集群。



注意

卸载后，检查云供应商是否有未正确删除的资源，特别是在用户置备基础架构(UPI)集群中。可能存在安装程序未创建或安装程序无法访问的资源。

先决条件

- 有用于部署集群的安装程序副本。
- 有创建集群时安装程序生成的文件。

流程

1. 在用来安装集群的计算机中包含安装程序的目录中，运行以下命令：

```
$ ./openshift-install destroy cluster \
--dir <installation_directory> --log-level info 1 2
```

1. 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。
2. 要查看不同的详情，请指定 **warn**、**debug** 或 **error**，而不是 **info**。

注意

您必须为集群指定包含集群定义文件的目录。安装程序需要此目录中的 **metadata.json** 文件来删除集群。

2. 可选：删除 **<installation_directory>** 目录和 OpenShift Container Platform 安装程序。

11.7. NUTANIX 的安装配置参数

在 Nutanix 上部署 OpenShift Container Platform 集群前，您可以提供参数来自定义集群和托管它的平台。在创建 **install-config.yaml** 文件时，您可以通过命令行为所需参数提供值。然后，您可以修改 **install-config.yaml** 文件以进一步自定义集群。

11.7.1. Nutanix 可用的安装配置参数

下表指定所需的、可选和特定于 Nutanix 的安装配置参数，您可以作为安装过程的一部分设置。

注意

安装后，您无法在 **install-config.yaml** 文件中修改这些参数。

11.7.1.1. 所需的配置参数

下表描述了所需的安装配置参数：

表 11.3. 所需的参数

参数	描述	值
apiVersion:	**install-config.yaml** 内容的 API 版本。当前版本为 **v1**。安装程序可能还支持旧的 API 版本。	字符串
baseDomain:	云供应商的基域。基域用于创建到 OpenShift Container Platform 集群组件的路由。集群的完整 DNS 名称是 **baseDomain** 和 **metadata.name** 参数值的组合，其格式为 **<metadata.name>.<baseDomain>**。	完全限定域名或子域名，如 **example.com**。
metadata:	Kubernetes 资源 **ObjectMeta**，其中只消耗 **name** 参数。	对象
metadata: name:	集群的名称。集群的 DNS 记录是 **{{.metadata.name}}.{{.baseDomain}}** 的子域。	小写字母和连字符 (**-**) 的字符串，如 **dev**。

参数	描述	值
platform:	对于特定平台的配置取决于执行安装的环境： **aws**, **baremetal**, **azure**, **gcp**, **ibmcloud**, **nutanix**, **openstack**, **powervs**, **vsphere**, 或 **{}**.有关 **platform.<platform>** 参数的更多信息，请参考下表中您的特定平台。	对象

pullSecret:	从 Red Hat OpenShift Cluster Manager 获取 pull secret，验证从 Quay.io 等服务中下载 OpenShift Container Platform 组件的容器镜像。	<pre>{
		  "auths":{
		  "cloud.openshift.com":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  },
		  "quay.io":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  }
		  }
		}</pre>

11.7.1.2. 网络配置参数

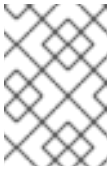
您可以根据现有网络基础架构的要求自定义安装配置。例如，您可以扩展集群网络的 IP 地址块，或者提供不同于默认值的不同 IP 地址块。

仅支持 IPv4 地址。

注意

Red Hat OpenShift Data Foundation 灾难恢复解决方案不支持 Globalnet。对于区域灾难恢复场景，请确保为每个集群中的集群和服务网络使用非重叠的专用 IP 地址。

表 11.4. 网络参数

参数	描述	值
networking:	集群网络的配置。	对象  **注意** 您无法在安装后修改 **网络** 对象指定的参数。

参数	描述	值
networking: networkType:	要安装的 Red Hat OpenShift Networking 网络插件。	**OVNKubernetes**。**OVNKubernetes** 是 Linux 网络和包含 Linux 和 Windows 服务器的混合网络的 CNI 插件。默认值为 **OVNKubernetes**。
networking: clusterNetwork:	pod 的 IP 地址块。 默认值为 **10.128.0.0/14**，主机前缀为 **/23**。 如果您指定了多个 IP 地址块，块不得重叠。	对象数组。例如： networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23
networking: clusterNetwork: cidr:	使用 **networking.clusterNetwork** 时需要此项。IP 地址块。 IPv4 网络。	无类别域间路由(CIDR)表示法中的 IP 地址块。IPv4 块的前缀长度介于 **0 到 32** 之间。
networking: clusterNetwork: hostPrefix:	分配给每个节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从 given **cidr** 中分配 a/**23** 子网。**hostPrefix** 值 **23** 提供 510（2^(32 - 23)- 2）pod IP 地址。	子网前缀。 默认值为 **23**。
networking: serviceNetwork:	服务的 IP 地址块。默认值为 **172.30.0.0/16**。 OVN-Kubernetes 网络插件只支持服务网络的一个 IP 地址块。	CIDR 格式具有 IP 地址块的数组。例如： networking: serviceNetwork: - 172.30.0.0/16
networking: machineNetwork:	机器的 IP 地址块。 如果您指定了多个 IP 地址块，块不得重叠。	对象数组。例如： networking: machineNetwork: - cidr: 10.0.0.0/16
networking: machineNetwork: cidr:	使用 **networking.machineNetwork** 时需要此项。IP 地址块。对于 libvirt 和 IBM Power® Virtual Server 以外的所有平台，默认值为 **10.0.0.0/16**。对于 libvirt，默认值 为 **192.168.126.0/24**。对于 IBM Power® Virtual Server，默认值为 **192.168.0.0/24**。	CIDR 表示法中的 IP 网络块。 例如：**10.0.0.0/16**。 **注意** 将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

11.7.1.3. 可选的配置参数

下表描述了可选的安装配置参数：

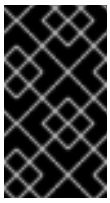
表 11.5. 可选参数

参数	描述	值
additionalTrustBundle:	添加到节点可信证书存储中的 PEM 编码 X.509 证书捆绑包。配置了代理时，也可以使用此信任捆绑包。	字符串
capabilities:	控制可选核心组件的安装。您可以通过禁用可选组件来减少 OpenShift Container Platform 集群的空间。如需更多信息，请参阅 *安装*中的"集群功能"页面。	字符串数组
capabilities: baselineCapabilitySet:	选择要启用的一组初始可选功能。有效值为 **None**、**v4.11**、**v4.12** 和 **vCurrent**。默认值为 **vCurrent**。	字符串
capabilities: additionalEnabledCapabilities:	将可选功能集合扩展到您在 **baselineCapabilitySet** 中指定的范围。您可以在此参数中指定多个功能。	字符串数组
cpuPartitioningMode:	启用工作负载分区，它会隔离 OpenShift Container Platform 服务、集群管理工作负载和基础架构 pod，以便在保留的一组 CPU 上运行。工作负载分区只能在安装过程中启用，且在安装后无法禁用。虽然此字段启用工作负载分区，但它不会将工作负载配置为使用特定的 CPU。如需更多信息，请参阅 *Scalability and Performance* 部分中的 *Workload partitioning* 页面。	**None** 或 **AllNodes**.**None** 是默认值。
compute:	组成计算节点的机器的配置。	**MachinePool** 对象的数组。
compute: architecture:	决定池中机器的指令集合架构。目前，不支持具有不同架构的集群。所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串

参数	描述	值
compute: hyperthreading:	是否在计算机器上启用或禁用并发多**线程或超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。	**enabled** 或 **Disabled**
compute: name:	使用 **compute** 时需要此项。机器池的名称。	**worker**
compute: platform:	使用 **compute** 时需要此项。使用此参数指定托管 worker 机器的云供应商。此参数值必须与 **controlPlane.platform** 参数值匹配。	**aws**, **azure**, **gcp**, **ibmcloud**, **nutanix**, **openstack**, **powervs**, **vsphere**, 或 **{}**
compute: replicas:	要置备的计算机器数量，也称为 worker 机器。	大于或等于 **2** 的正整数。默认值为 **3**。
featureSet:	为功能集启用集群。功能集是 OpenShift Container Platform 功能的集合，默认情况下不启用。有关在安装过程中启用功能集的更多信息，请参阅"使用功能门启用功能"。	字符串.要启用的功能集的名称，如 **TechPreviewNoUpgrade**。
controlPlane:	组成 control plane 的机器的配置。	**MachinePool** 对象的数组。
controlPlane: architecture:	决定池中机器的指令集合架构。目前，不支持具有不同架构的集群。所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串

参数	描述	值
controlPlane: hyperthreading:	是否在 control plane 机器上启用或禁用并发多**线程或超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。	**enabled** 或 **Disabled**
controlPlane: name:	使用 **controlPlane** 时需要此项。机器池的名称。	**master**
controlPlane: platform:	使用 **controlPlane** 时需要此项。使用此参数指定托管 control plane 机器的云供应商。此参数值必须与 **compute.platform** 参数值匹配。	**aws**, **azure**, **gcp**, **ibmcloud**, **nutanix**, **openstack**, **powervs**, **vsphere**, 或 **{}**
controlPlane: replicas:	要置备的 control plane 机器数量。	部署单节点 OpenShift 时支持的值为 **3** 或 **1**。
credentialsMode:	Cloud Credential Operator(CCO)模式。如果没有指定模式，CCO 会动态尝试决定提供的凭证的功能，在支持多个模式的平台上首选 mint 模式。	**Mint**、**Passthrough**、**Manual** 或空字符串(**""**)。[1]

参数	描述	值
fips:	启用或禁用 FIPS 模式。默认值为 **false**（禁用）。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。  **重要** 要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 Red Hat Enterprise Linux (RHEL) 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅在 FIPS 模式中安装该系统。 当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS) 时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。  **注意** 如果使用 Azure File 存储，则无法启用 FIPS 模式。	**false** 或 **true**
imageContentSources:	release-image 内容的源和存储库。	对象数组。包括一个 **source** 以及可选的 **mirrors**，如本表的以下行所述。
imageContentSources: source:	使用 **imageContentSources** 时需要此项。指定用户在镜像拉取规格中引用的存储库。	字符串

参数	描述	值
imageContentSources: mirrors:	指定可能还包含同一镜像的一个或多个仓库。	字符串数组
publish:	如何发布或公开集群的面向用户的端点，如 Kubernetes API、OpenShift 路由。	**内部或外部 .**默认值为 **External**。 在非云平台上不支持将此字段设置为 **Internal**。  **重要** 如果将字段的值设为 **Internal**，集群将无法运行。如需更多信息，请参阅 BZ#1953035。
sshKey:	用于验证对集群机器的访问的 SSH 密钥。  **注意** 对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。	例如，**sshKey: ssh-ed25519 AAAA..**。

1. 不是所有 CCO 模式都支持所有云供应商。有关 CCO 模式的更多信息，请参阅*身份验证和授权*内容中的"管理云供应商凭证"条目。

11.7.1.4. 额外的 Nutanix 配置参数

下表描述了额外的 Nutanix 配置参数：

表 11.6. 额外的 Nutanix 集群参数

参数	描述	值
compute: platform: nutanix: categories: key:	应用到计算虚拟机的 prism category 键的名称。此参数必须由 **value** 参数加加，而 **key** 和 **value** 参数必须在 Prism Central 中存在。有关类别的更多信息，请参阅类别管理。	字符串

参数	描述	值
compute: platform: nutanix: categories: value:	应用到计算虚拟机的 prism category 键值对的值。此参数必须由 **key** 参数加加，而 **key** 和 **value** 参数必须在 Prism Central 中存在。	字符串
compute: platform: nutanix: failureDomains:	仅适用于计算机器的故障域。 故障域在 **platform.nutanix.failureDomains** 中指定。	列表。 一个或多个故障域的名称。
compute: platform: nutanix: project: type:	用于为计算虚拟机选择项目的标识符类型。项目定义用于管理权限、网络和其他参数的用户角色的逻辑组。如需有关项目的更多信息，请参阅项目概述。	**name** 或 **uuid**
compute: platform: nutanix: project: name: or uuid:	与计算机器关联的项目的名称或 UUID。此参数必须由 **type** 参数显示。	字符串
compute: platform: nutanix: bootType:	计算机器使用的引导类型。您必须在 OpenShift Container Platform 4.16 中使用 **Legacy** 引导类型。有关引导类型的更多信息，请参阅 虚拟环境中的了解 UEFI、安全引导和 TPM。	**Legacy**、**SecureBoot** 或 **UEFI**。默认值为 **Legacy**。
controlPlane: platform: nutanix: categories: key:	应用到 control plane 虚拟机的 prism category 键的名称。此参数必须由 **value** 参数加加，而 **key** 和 **value** 参数必须在 Prism Central 中存在。有关类别的更多信息，请参阅类别管理。	字符串

参数	描述	值
controlPlane: platform: nutanix: categories: value:	应用到 control plane 虚拟机的 prism category 键值对的值。此参数必须由 **key** 参数加加，而 **key** 和 **value** 参数必须在 Prism Central 中存在。	字符串
controlPlane: platform: nutanix: failureDomains:	仅适用于 control plane 机器的故障域。 故障域在 **platform.nutanix.failureDomains** 中指定。	列表。 一个或多个故障域的名称。
controlPlane: platform: nutanix: project: type:	用于为 control plane 虚拟机选择项目的标识符类型。项目定义用于管理权限、网络和其他参数的用户角色的逻辑组。如需有关项目的更多信息，请参阅项目概述。	**name** 或 **uuid**
controlPlane: platform: nutanix: project: name: or uuid:	与 control plane 虚拟机关联的项目的名称或 UUID。此参数必须由 **type** 参数显示。	字符串
platform: nutanix: defaultMachinePlatform: categories: key:	适用于所有虚拟机的 prism category 键的名称。此参数必须由 **value** 参数加加，而 **key** 和 **value** 参数必须在 Prism Central 中存在。有关类别的更多信息，请参阅类别管理。	字符串
platform: nutanix: defaultMachinePlatform: categories: value:	应用到所有虚拟机的 prism category 键值对的值。此参数必须由 **key** 参数加加，而 **key** 和 **value** 参数必须在 Prism Central 中存在。	字符串

参数	描述	值
platform: nutanix: defaulatMachinePlatform: failureDomains:	适用于 control plane 和计算机器的故障域。 故障域在 **platform.nutanix.failureDomains** 中指定。	列表。 一个或多个故障域的名称。

platform:
  nutanix:
 	
defaultMachinePlatform:
  project:
  type:
	用于为所有虚拟机选择项目的标识符类型。项目定义用于管理权限、网络和其他参数的用户角色的逻辑组。如需有关项目的更多信息，请参阅项目概述。	**name** 或 **uuid**。

platform:
  nutanix:
 	
defaultMachinePlatform:
  project:
  name: or uuid:
	与所有虚拟机关联的项目的名称或 UUID。此参数必须由 **type** 参数显示。	字符串

platform:
  nutanix:
 	
defaultMachinePlatform:
  bootType:
	所有机器的引导类型。您必须在 OpenShift Container Platform 4.16 中使用 **Legacy** 引导类型。有关引导类型的更多信息，请参阅 虚拟环境中的了解 UEFI、安全引导和 TPM。	**Legacy**、**SecureBoot** 或 **UEFI**。默认值为 **Legacy**。

platform:
  nutanix:
  apiVIP:
	为 control plane API 访问配置的虚拟 IP(VIP)地址。	IP 地址

参数	描述	值
platform: nutanix: failureDomains: - name: prismElement: name: uuid: subnetUUIDs: -	默认情况下，安装程序会将集群机器安装到单个 Prism Element 实例。您可以为容错指定额外的 Prism Element 实例，然后将它们应用到： • 集群的默认机器配置 • 只有 control plane 或计算机器	配置的故障域列表。 有关使用的更多信息，请参阅"在 Nutanix 上安装集群"中的"配置故障域"。
platform: nutanix: ingressVIP:	为集群入口配置的虚拟 IP(VIP)地址。	IP 地址
platform: nutanix: prismCentral: endpoint: address:	Prism Central 域名或 IP 地址。	字符串
platform: nutanix: prismCentral: endpoint: port:	用于登录到 Prism Central 的端口。	字符串
platform: nutanix: prismCentral: password:	Prism Central 用户名的密码。	字符串
platform: nutanix: prismCentral: username:	用于登录到 Prism Central 的用户名。	字符串

参数	描述	值
platform: nutanix: prismElements: endpoint: address:	Prism Element 域名或 IP 地址。[1]	字符串
platform: nutanix: prismElements: endpoint: port:	用于登录到 Prism Element 的端口。	字符串
platform: nutanix: prismElements: uuid:	Prism Element 的通用唯一标识符 (UUID)。	字符串
platform: nutanix: subnetUUIDs:	包含您配置的虚拟 IP 地址和 DNS 记录的 Prism Element 网络的 UUID。[2]	字符串
platform: nutanix: clusterOSImage:	可选：默认情况下，安装程序会下载并安装 Red Hat Enterprise Linux CoreOS (RHCOS) 镜像。如果 Prism Central 没有互联网访问，您可以通过在任何 HTTP 服务器上托管 RHCOS 镜像来覆盖默认行为，并将安装程序指向镜像。	HTTP 或 HTTPS URL，可选使用 SHA-256 校验和。例如： http://example.com/images/rhcos-47.83.202103221318-0-nutanix.x86_64.qcow2

1. **prismElements** 部分包含 Prism Elements (clusters) 的列表。Prism Element 包括了所有 Nutanix 资源，如用于托管 OpenShift Container Platform 集群的虚拟机和子网。

2. 仅支持 OpenShift Container Platform 集群中每个 Prism Element 的一个子网。

第 12 章 使用辅助安装程序安装内部

12.1. 使用 ASSISTED INSTALLER 安装内部集群

您可以使用辅助安装程序在内部硬件或内部虚拟机中安装 OpenShift Container Platform。使用 Assisted Installer 安装 OpenShift Container Platform 支持 **x86_64**、**AArch** 64、**ppc64le** 和 **s390x** CPU 架构。

12.1.1. 使用引导安装程序

Assisted Installer 是一个在 Red Hat Hybrid Cloud Console 上提供的用户友好的安装解决方案。Assisted Installer 支持各种部署平台，专注于裸机，Nutanix 和 vSphere 基础架构。

辅助安装程序提供安装功能作为服务。此软件即服务 (SaaS) 方法具有以下优点：

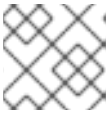
- **Web 用户界面：**在无需手动创建安装配置文件的情况下，web 用户界面会执行集群安装。
- **没有 bootstrap 节点：**使用 Assisted Installer 安装时不需要 bootstrap 节点。bootstrap 过程在集群的节点上执行。
- **Host:** Assisted Installer 主机：
  - Ignition 文件
  - 安装配置
  - 发现 ISO
  - 安装程序
- **简化的安装工作流：** 部署不需要深入了解 OpenShift Container Platform。Assisted Installer 提供了合理的默认值，并将安装程序作为服务提供，它：
  - 消除在本地安装和运行 OpenShift Container Platform 安装程序的需要。
  - 确保安装程序的最新版本，到最新的已测试的 z-stream 版本。旧版本保持可用（如果需要）。
  - 启用使用 API 构建自动化，而无需在本地运行 OpenShift Container Platform 安装程序。
- **高级网络：** 辅助安装程序支持使用 SDN 和 OVN 的 IPv4 网络，以及只使用 OVN 的 IPv6 和双堆栈网络，基于 NMState 的静态 IP 寻址和 HTTP/S 代理。OVN 是 OpenShift Container Platform 4.12 及更新的版本的默认 Container Network Interface (CNI)。SDN 最多支持 OpenShift Container Platform 4.14，但不支持 OpenShift Container Platform 4.15 及更新的版本。
- **预安装验证 ：**辅助安装程序在安装前验证配置，以确保高可能成功。验证过程包括以下检查：
  - 确保网络连接
  - 确保足够的网络带宽
  - 确保与 registry 的连接
  - 确保集群节点之间的时间同步
  - 验证集群节点满足最低硬件要求

- 验证安装配置参数

- **REST API：**辅助安装程序具有 REST API，支持自动化。

Assisted Installer 支持在连接的环境中安装 OpenShift Container Platform，包括使用可选的 HTTP/S 代理。它可安装以下内容：

- 高可用性 OpenShift Container Platform 或单节点 OpenShift (SNO)
- 在具有完整平台集成的裸机、Nutanix 或 vSphere 上的 OpenShift Container Platform，或其他虚拟平台上没有集成
- 可选： OpenShift Virtualization、多集群引擎、逻辑卷管理器 (LVM) 存储和 OpenShift Data Foundation



注意

目前，IBM Z®(**s390x**) 架构不支持 OpenShift Virtualization 和 LVM 存储。

用户界面提供了一个直观的交互式工作流，其中自动化不存在或不需要。用户也可以使用 REST API 自动执行安装。

详情请参阅 OpenShift Container Platform 的辅助安装程序 文档。

12.1.2. 对引导安装程序的 API 支持

在宣布弃用后，Assisted Installer 支持的 API 最少会保持三个月的稳定。

第 13 章 使用基于代理的安装程序安装内部集群

13.1. 准备使用基于代理的安装程序安装

13.1.1. 关于基于代理的安装程序

基于代理的安装方法提供了以任何方式引导内部服务器的灵活性。它将辅助安装服务的使用与离线运行的功能相结合，包括在 air-gapped 环境中。基于代理的安装是 OpenShift Container Platform 安装程序的子命令。它生成一个可引导的 ISO 镜像，其中包含使用可用发行镜像部署 OpenShift Container Platform 集群所需的所有信息。

配置的格式与安装程序置备的基础架构和用户置备的基础架构安装方法相同。基于代理的安装程序也可以选择生成或接受 Zero Touch Provisioning (ZTP) 自定义资源。ZTP 允许您使用裸机设备声明配置来置备新的边缘站点。

表 13.1. 基于代理的安装程序支持的构架

CPU 架构	连接安装	断开连接的安装
64-bit x86	✓	✓
64-bit ARM	✓	✓
ppc64le	✓	✓
s390x	✓	✓

13.1.2. 了解基于代理的安装程序

作为 OpenShift Container Platform 用户，您可以在断开连接的环境中利用 Assisted Installer 托管服务的优势。

基于代理的安装包含一个可引导 ISO，其中包含辅助发现代理和辅助服务。两者都需要执行集群安装，但后者仅在其中一个主机上运行。

openshift-install agent create image 子命令会根据您提供的输入生成一个临时 ISO。您可以选择通过以下清单提供输入：

Preferred:

- **install-config.yaml**
- **agent-config.yaml**

可选：ZTP 清单

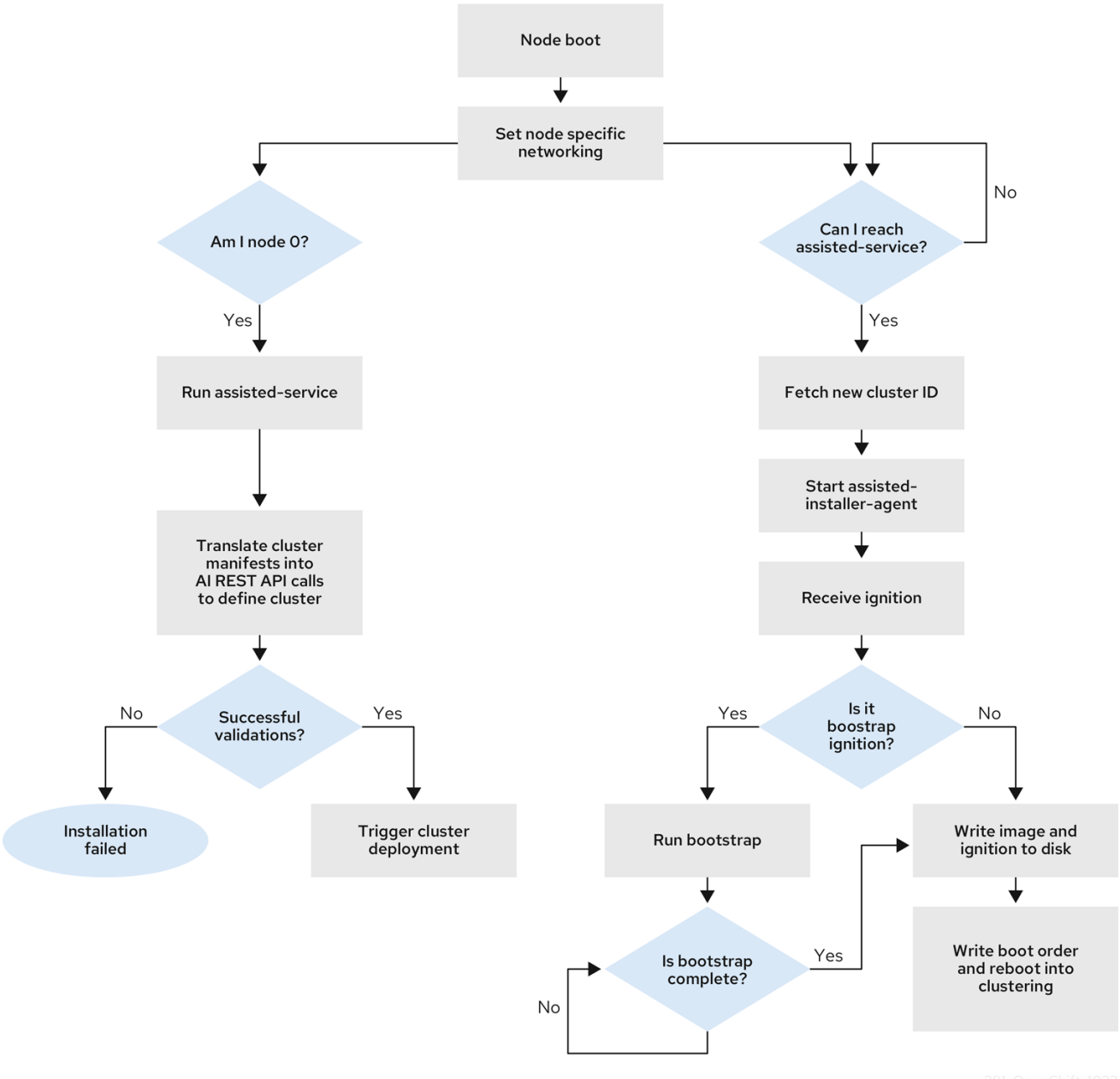
- **cluster-manifests/cluster-deployment.yaml**
- **cluster-manifests/agent-cluster-install.yaml**
- **cluster-manifests/pull-secret.yaml**
- **cluster-manifests/infraenv.yaml**

- **cluster-manifests/cluster-image-set.yaml**
- **cluster-manifests/nmstateconfig.yaml**
- **mirror/registries.conf**
- **mirror/ca-bundle.crt**

13.1.2.1. 基于代理的安装程序工作流

其中一个 control plane 主机在引导过程中运行 Assisted Service，最终成为 bootstrap 主机。此节点称为 **rendezvous 主机** (node 0)。Assisted Service 确保所有主机都满足要求，并触发 OpenShift Container Platform 集群部署。所有节点都将 Red Hat Enterprise Linux CoreOS (RHCOS) 镜像写入磁盘。非引导节点重新引导并启动集群部署。节点重启后，rendezvous 主机会重启并加入集群。bootstrap 已完成，并部署了集群。

图 13.1. 节点安装工作流



您可以通过 **openshift-install agent create image** 子命令为以下拓扑安装断开连接的 OpenShift Container Platform 集群：

- **单节点 OpenShift Container Platform 集群 (SNO)**：一个 master 和 worker 的节点。
- **三节点 OpenShift Container Platform 集群**：一个紧凑集群，它有三个 master 节点，也是 worker 节点。
- **高可用性 OpenShift Container Platform 集群 (HA)**：具有任意数量的 worker 节点的 master 节点。

13.1.2.2. 拓扑的建议资源

为以下拓扑推荐的集群资源：

表 13.2. 推荐的集群资源

Topology	control plane 节点数量	计算节点数量	vCPU	memory	存储
单节点集群	1	0	8 个 vCPU 内核	16 GB RAM	120 GB
紧凑集群	3	0 或 1	8 个 vCPU 内核	16 GB RAM	120 GB
HA 集群	3	2 及更高版本	8 个 vCPU 内核	16 GB RAM	120 GB

在 **install-config.yaml** 中，指定在其上执行安装的平台。支持以下平台：

- **baremetal**
- **vsphere**
- **none**

重要

对于平台，**none**:

- **none** 选项需要在集群中置备 DNS 名称解析和负载平衡基础架构。如需更多信息，请参阅"Additional resources"部分中的*使用平台"none"选项集群的要求*。
- 在尝试在虚拟化或云环境中安装 OpenShift Container Platform 集群前，请参阅有关在未经测试的平台上部署 OpenShift Container Platform 的指南 中的信息。

其他资源

- 使用平台"none"选项对集群的要求
- 增加网络 MTU
- 将 worker 节点添加到单节点 OpenShift 集群

13.1.3. 关于 FIPS 合规性

对于许多 OpenShift Container Platform 客户，在将任何系统投入生产前需要达到一定级别的法规就绪状态或合规性。这种法规就绪状态可通过国家标准、行业标准或机构的企业监管框架来施加。FIPS（Federal Information Processing Standards）合规性是高安全性环境中所需的最重要的组件之一，可确保节点上只允许使用支持的加密技术。

重要

要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 Red Hat Enterprise Linux (RHEL) 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅在 FIPS 模式中安装该系统。

当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS)时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。

13.1.4. 通过基于代理的安装程序配置 FIPS

在集群部署期间，当在集群中部署 Red Hat Enterprise Linux CoreOS (RHCOS) 机器时，会应用联邦信息处理标准 (FIPS) 更改。在 Red Hat Enterprise Linux(RHEL) 机器中，您必须在计划用作 worker 机器的机器上安装操作系统时启用 FIPS 模式。

您可以通过首选使用 **install-config.yaml** 和 **agent-config.yaml** 来启用 FIPS 模式：

1. 您必须在 **install-config.yaml** 文件中将 **fips** 字段的值设置为 **True** ：

   install-config.yaml.file 示例

   ```
   apiVersion: v1
   baseDomain: test.example.com
   metadata:
     name: sno-cluster
   fips: True
   ```

2. 可选：如果使用 GitOps ZTP 清单，您必须在 **agent-cluster-install.yaml** 文件的 **Agent-install.openshift.io/install-config-overrides** 字段中将 **fips** 的值设置为 **True** ：

   agent-cluster-install.yaml 文件示例

   ```
   apiVersion: extensions.hive.openshift.io/v1beta1
   kind: AgentClusterInstall
   metadata:
     annotations:
       agent-install.openshift.io/install-config-overrides: '{"fips": True}'
     name: sno-cluster
     namespace: sno-cluster-test
   ```

其他资源

- OpenShift 安全指南手册
- 支持 FIPS 加密

13.1.5. 主机配置

您可以在 **agent-config.yaml** 文件中为集群中的每个主机创建额外的配置，如网络配置和 root 设备提示。

重要

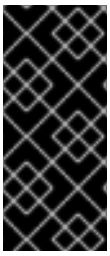
对于您配置的每个主机，您必须提供主机上接口的 MAC 地址，以指定您要配置的主机。

13.1.5.1. 主机角色

集群中的每个主机都被分配了 **master** 或 **worker** 的角色。您可以使用 **role** 参数为 **agent-config.yaml** 文件中的每个主机定义角色。如果您没有为主机分配角色，则会在安装过程中随机分配角色。

建议为您的主机显式定义角色。

rendezvousIP 必须分配给具有 **master** 角色的主机。这可以手动完成，或者通过允许基于代理的安装程序分配角色。



重要

您不需要为 rendezvous 主机显式定义 **master** 角色，但您无法创建与这个分配冲突的配置。

例如，如果您有 4 个主机明确定义为具有 **master** 角色，则在安装过程中自动分配 **worker** 角色的最后一个主机无法配置为 rendezvous 主机。

agent-config.yaml 文件示例

```
apiVersion: v1beta1
kind: AgentConfig
metadata:
  name: example-cluster
rendezvousIP: 192.168.111.80
hosts:
  - hostname: master-1
    role: master
    interfaces:
      - name: eno1
        macAddress: 00:ef:44:21:e6:a5
  - hostname: master-2
    role: master
    interfaces:
      - name: eno1
        macAddress: 00:ef:44:21:e6:a6
  - hostname: master-3
    role: master
    interfaces:
      - name: eno1
        macAddress: 00:ef:44:21:e6:a7
  - hostname: worker-1
    role: worker
    interfaces:
      - name: eno1
        macAddress: 00:ef:44:21:e6:a8
```

-

13.1.5.2. 关于 root 设备提示

rootDeviceHints 参数可让安装程序将 Red Hat Enterprise Linux CoreOS(RHCOS)镜像置备到特定的设备。安装程序会按照发现设备的顺序检查设备，并将发现的值与 hint 值进行比较。安装程序使用第一个与 hint 值匹配的发现设备。配置可以组合多个 hint，但设备必须与所有提示匹配，以便安装程序进行选择。

表 13.3. 子字段

子字段	描述
deviceName	包含 Linux 设备名称的字符串（如 **/dev/vda** 或 **/dev/disk/by-path/** ）。建议您使用 **/dev/disk/by-path/<device_path>** 链接到存储位置。hint 必须与实际值完全匹配。
hctl	包含类似 **0:0:0:0:0 的 SCSI 总线地址的字符串**。hint 必须与实际值完全匹配。
model	包含特定厂商的设备标识符的字符串。hint 可以是实际值的子字符串。
vendor	包含该设备厂商或制造商名称的字符串。hint 可以是实际值的子字符串。
serialNumber	包含设备序列号的字符串。hint 必须与实际值完全匹配。
minSizeGigabytes	以 GB 为单位代表设备的最小大小的整数。
wwn	包含唯一存储标识符的字符串。hint 必须与实际值完全匹配。
rotational	指明该设备为旋转磁盘(true)还是非旋转磁盘(false)的布尔值。

用法示例

```
- name: master-0
  role: master
  rootDeviceHints:
    deviceName: "/dev/sda"
```

13.1.6. 关于网络

在生成代理 ISO 时，必须知道 **rendezvous IP**，以便在初始引导过程中，所有主机都可以检查辅助服务。如果使用动态主机配置协议 (DHCP) 服务器分配 IP 地址，则必须将 **rendezvousIP** 字段设置为将成为部署 control plane 一部分的主机的 IP 地址。在没有 DHCP 服务器的环境中，您可以静态定义 IP 地址。

除了静态 IP 地址外，您还可以应用任何采用 NMState 格式的网络配置。这包括 VLAN 和 NIC 绑定。

13.1.6.1. DHCP

首选方法：install-config.yaml 和 agent-config.yaml

您必须为 **rendezvousIP** 字段指定值。**networkConfig** 字段可以留空：

agent-config.yaml.file 示例

```
apiVersion: v1alpha1
kind: AgentConfig
metadata:
  name: sno-cluster
rendezvousIP: 192.168.111.80 (1)
```

(1) rendezvous 主机的 IP 地址。

13.1.6.2. 静态网络

a. 首选方法：**install-config.yaml** 和 **agent-config.yaml**

agent-config.yaml.file 示例

```
cat > agent-config.yaml << EOF
apiVersion: v1alpha1
kind: AgentConfig
metadata:
  name: sno-cluster
rendezvousIP: 192.168.111.80 (1)
hosts:
  - hostname: master-0
    interfaces:
      - name: eno1
        macAddress: 00:ef:44:21:e6:a5 (2)
    networkConfig:
      interfaces:
        - name: eno1
          type: ethernet
          state: up
          mac-address: 00:ef:44:21:e6:a5
          ipv4:
            enabled: true
            address:
              - ip: 192.168.111.80 (3)
                prefix-length: 23 (4)
            dhcp: false
      dns-resolver:
        config:
          server:
            - 192.168.111.1 (5)
      routes:
        config:
          - destination: 0.0.0.0/0
            next-hop-address: 192.168.111.1 (6)
```

```
        next-hop-interface: eno1
        table-id: 254
EOF
```

1. 如果没有为 **rendezvousIP** 字段指定值，则会从 **networkConfig** 字段中指定的静态 IP 地址选择一个地址。
2. 主机上接口的 MAC 地址，用于决定要将配置应用到的主机。
3. 目标裸机主机的静态 IP 地址。
4. 目标裸机主机的静态 IP 地址子网前缀。
5. 目标裸机主机的 DNS 服务器。
6. 节点流量的下一跳地址。这必须与为特定接口设定的 IP 地址位于同一个子网中。

b. 可选方法：GitOps ZTP 清单
GitOps ZTP 自定义资源的可选方法包含 6 个自定义资源；您可以在 **nmstateconfig.yaml** 文件中配置静态 IP。

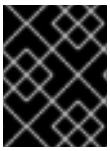
```
apiVersion: agent-install.openshift.io/v1beta1
kind: NMStateConfig
metadata:
  name: master-0
  namespace: openshift-machine-api
  labels:
    cluster0-nmstate-label-name: cluster0-nmstate-label-value
spec:
  config:
    interfaces:
      - name: eth0
        type: ethernet
        state: up
        mac-address: 52:54:01:aa:aa:a1
        ipv4:
          enabled: true
          address:
            - ip: 192.168.122.2 1
              prefix-length: 23 2
          dhcp: false
    dns-resolver:
      config:
        server:
          - 192.168.122.1 3
    routes:
      config:
        - destination: 0.0.0.0/0
          next-hop-address: 192.168.122.1 4
          next-hop-interface: eth0
          table-id: 254
  interfaces:
    - name: eth0
      macAddress: 52:54:01:aa:aa:a1 5
```

1. 目标裸机主机的静态 IP 地址。
2. 目标裸机主机的静态 IP 地址子网前缀。
3. 目标裸机主机的 DNS 服务器。
4. 节点流量的下一跳地址。这必须与为特定接口设定的 IP 地址位于同一个子网中。
5. 主机上接口的 MAC 地址，用于决定要将配置应用到的主机。

rendezvous IP 从 **config** 字段中指定的静态 IP 地址中选择。

13.1.7. 使用平台"none"选项对集群的要求

本节介绍了配置为使用平台 **none** 选项的基于 Agent 的 OpenShift Container Platform 安装的要求。



重要

在尝试在虚拟化或云环境中安装 OpenShift Container Platform 集群前，请参阅有关在未经测试的平台上部署 OpenShift Container Platform 的指南 中的信息。

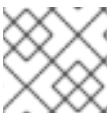
13.1.7.1. 平台"none" DNS 要求

在 OpenShift Container Platform 部署中，以下组件需要 DNS 名称解析：

- The Kubernetes API
- OpenShift Container Platform 应用程序通配符
- control plane 和计算机器

Kubernetes API、control plane 机器和计算机器还需要反向 DNS 解析。

DNS A/AAAA 或 CNAME 记录用于名称解析，PTR 记录用于反向名称解析。反向记录很重要，因为 Red Hat Enterprise Linux CoreOS(RHCOS)使用反向记录为所有节点设置主机名，除非 DHCP 提供主机名。另外，反向记录用于生成 OpenShift Container Platform 需要操作的证书签名请求(CSR)。



注意

建议使用 DHCP 服务器为每个群集节点提供主机名。

使用 platform **none** 选项的 OpenShift Container Platform 集群需要以下 DNS 记录，且必须在安装前就位它们。在每个记录中，**<cluster_name>** 是集群名称，**<base_domain>** 是您在 **install-config.yaml** 文件中指定的基域。完整的 DNS 记录采用以下形式：**<component>.<cluster_name>.<base_domain>.**。

表 13.4. 所需的 DNS 记录

组件	记录	描述
Kubernetes API	**api.<cluster_name>.<base_domain>.**	DNS A/AAAA 或 CNAME 记录，以及用于标识 API 负载均衡器的 DNS PTR 记录。这些记录必须由集群外的客户端和集群中的所有节点解析。

组件	记录	描述
	api-int.<cluster_name>.<base_domain>.	DNS A/AAAA 或 CNAME 记录，以及用于内部标识 API 负载均衡器的 DNS PTR 记录。这些记录必须可以从集群中的所有节点解析。  **重要** API 服务器必须能够根据 Kubernetes 中记录的主机名解析 worker 节点。如果 API 服务器无法解析节点名称，则代理的 API 调用会失败，且您无法从 pod 检索日志。
Routes	***.apps.<cluster_name>.<base_domain>.**	通配符 DNS A/AAAA 或 CNAME 记录，指向应用程序入口负载均衡器。应用程序入口负载均衡器以运行 Ingress Controller Pod 的机器为目标。默认情况下，Ingress Controller Pod 在计算机器上运行。这些记录必须由集群外的客户端和集群中的所有节点解析。 例如，console **-openshift-console.apps.<cluster_name>.<base_domain>** 用作到 OpenShift Container Platform 控制台的通配符路由。
control plane 机器	**<master><n>.<cluster_name>.<base_domain>.**	DNS A/AAAA 或 CNAME 记录，以识别 control plane 节点的每台机器。这些记录必须由集群中的节点解析。
计算机器	**<worker><n>.<cluster_name>.<base_domain>.**	DNS A/AAAA 或 CNAME 记录，用于识别 worker 节点的每台机器。这些记录必须由集群中的节点解析。

注意

在 OpenShift Container Platform 4.4 及更新的版本中，您不需要在 DNS 配置中指定 etcd 主机和 SRV 记录。

提示

您可以使用 **dig** 命令验证名称和反向名称解析。

13.1.7.1.1. 平台 "none" 集群的 DNS 配置示例

本节提供了 A 和 PTR 记录配置示例，它满足使用平台 **none** 选项部署 OpenShift Container Platform 的 DNS 要求。样本不是为选择一个 DNS 解决方案提供建议。

在这个示例中，集群名称为 **ocp4**，基域是 **example.com**。

平台 "none" 集群的 DNS A 记录配置示例

以下示例是 BIND 区域文件，它显示了使用 platform **none** 选项在集群中名称解析的 A 记录示例。

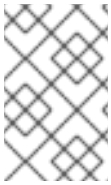
例 13.1. DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
   2019070700 ; serial
   3H  ; refresh (3 hours)
   30M  ; retry (30 minutes)
   2W  ; expiry (2 weeks)
   1W )  ; minimum (1 week)
 IN NS ns1.example.com.
 IN MX 10 smtp.example.com.
;
;
ns1.example.com.  IN A 192.168.1.5
smtp.example.com.  IN A 192.168.1.5
;
helper.example.com.  IN A 192.168.1.5
helper.ocp4.example.com. IN A 192.168.1.5
;
api.ocp4.example.com.  IN A 192.168.1.5 1
api-int.ocp4.example.com. IN A 192.168.1.5 2
;
*.apps.ocp4.example.com. IN A 192.168.1.5 3
;
master0.ocp4.example.com. IN A 192.168.1.97 4
master1.ocp4.example.com. IN A 192.168.1.98 5
master2.ocp4.example.com. IN A 192.168.1.99 6
;
worker0.ocp4.example.com. IN A 192.168.1.11 7
worker1.ocp4.example.com. IN A 192.168.1.7 8
;
;EOF
```

1 为 Kubernetes API 提供名称解析。记录引用 API 负载均衡器的 IP 地址。

2 为 Kubernetes API 提供名称解析。记录引用 API 负载均衡器的 IP 地址，用于内部集群通信。

3 为通配符路由提供名称解析。记录引用应用程序入口负载均衡器的 IP 地址。应用程序入口负载均衡器以运行 Ingress Controller Pod 的机器为目标。默认情况下，Ingress Controller Pod 在计算机器上运行。



注意

在这个示例中，将相同的负载均衡器用于 Kubernetes API 和应用入口流量。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。

4 5 6 为 control plane 机器提供名称解析。

7 8 为计算机器提供名称解析。

平台 "none" 集群的 DNS PTR 记录配置示例

下面的 BIND 区文件示例显示集群中使用 platform **none** 选项反向名称解析的 PTR 记录示例。

例 13.2. 反向记录的 DNS 区数据库示例

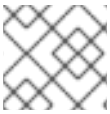
```
$TTL 1W
@ IN SOA ns1.example.com. root (
  2019070700 ; serial
  3H  ; refresh (3 hours)
  30M  ; retry (30 minutes)
  2W  ; expiry (2 weeks)
  1W )  ; minimum (1 week)
 IN NS ns1.example.com.
;
5.1.168.192.in-addr.arpa. IN PTR api.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. IN PTR api-int.ocp4.example.com. 2
;
97.1.168.192.in-addr.arpa. IN PTR master0.ocp4.example.com. 3
98.1.168.192.in-addr.arpa. IN PTR master1.ocp4.example.com. 4
99.1.168.192.in-addr.arpa. IN PTR master2.ocp4.example.com. 5
;
11.1.168.192.in-addr.arpa. IN PTR worker0.ocp4.example.com. 6
7.1.168.192.in-addr.arpa. IN PTR worker1.ocp4.example.com. 7
;
;EOF
```

1 为 Kubernetes API 提供反向 DNS 解析。PTR 记录引用 API 负载均衡器的记录名称。

2 为 Kubernetes API 提供反向 DNS 解析。PTR 记录引用 API 负载均衡器的记录名称，用于内部集群通信。

3 4 5 为 control plane 机器提供反向 DNS 解析。

6 7 为计算机器提供反向 DNS 解析。



注意

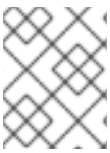
OpenShift Container Platform 应用程序通配符不需要 PTR 记录。

13.1.7.2. 平台"none"负载均衡要求

在安装 OpenShift Container Platform 前，您必须置备 API 和应用程序入口负载均衡基础架构。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。

注意

这些要求不适用于使用 platform **none** 选项的单节点 OpenShift 集群。



注意

如果要使用 Red Hat Enterprise Linux (RHEL) 实例部署 API 和应用程序入口负载均衡器，您必须单独购买 RHEL 订阅。

负载平衡基础架构必须满足以下要求：

1. **API 负载均衡器** ：提供一个通用端点，供用户（包括人工和机器）与平台交互和配置。配置以下条件：

   - 仅第 4 层负载均衡.这可称为 Raw TCP、SSL Passthrough 或 SSL 网桥模式。如果使用 SSL Bridge 模式，必须为 API 路由启用 Server Name Indication(SNI)。
   - 无状态负载平衡算法。这些选项根据负载均衡器的实施而有所不同。

   重要

   不要为 API 负载均衡器配置会话持久性。

   在负载均衡器的前端和后端配置以下端口：

   表 13.5. API 负载均衡器

port	后端机器（池成员）	internal	外部	描述
6443	控制平面。您必须为 API 服务器健康检查探测配置 **/readyz** 端点。	X	X	Kubernetes API 服务器
22623	控制平面。	X		机器配置服务器

   注意

   负载均衡器必须配置为，从 API 服务器关闭 **/readyz** 端点到从池中移除 API 服务器实例时最多需要 30 秒。在 **/readyz** 返回错误或健康后的时间范围内，端点必须被删除或添加。每 5 秒或 10 秒探测一次，有两个成功请求处于健康状态，三个成为不健康的请求是经过良好测试的值。

2. **应用程序入口负载均衡器** ：为应用程序流量从集群外部流提供入口点。OpenShift Container Platform 集群需要正确配置入口路由器。
   配置以下条件：

   - 仅第 4 层负载均衡.这可称为 Raw TCP、SSL Passthrough 或 SSL 网桥模式。如果使用 SSL Bridge 模式，您必须为入口路由启用 Server Name Indication(SNI)。
   - 建议根据可用选项以及平台上托管的应用程序类型，使用基于连接的或基于会话的持久性。

   提示

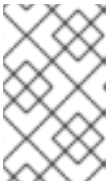
   如果应用程序入口负载均衡器可以看到客户端的真实 IP 地址，启用基于 IP 的会话持久性可以提高使用端到端 TLS 加密的应用程序的性能。

   在负载均衡器的前端和后端配置以下端口：

   表 13.6. 应用程序入口负载均衡器

port	后端机器（池成员）	internal	外部	描述

port	后端机器（池成员）	internal	外部	描述
443	默认情况下，运行 Ingress Controller Pod、计算或 worker 的机器。	X	X	HTTPS 流量
80	默认情况下，运行 Ingress Controller Pod、计算或 worker 的机器。	X	X	HTTP 流量



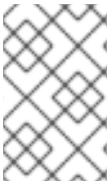
注意

如果要部署一个带有零计算节点的三节点集群，Ingress Controller Pod 在 control plane 节点上运行。在三节点集群部署中，您必须配置应用程序入口负载均衡器，将 HTTP 和 HTTPS 流量路由到 control plane 节点。

13.1.7.2.1. 平台 "none" 集群的负载均衡器配置示例

本节提供了一个满足平台 **none** 选项的集群的负载均衡要求的 API 和应用程序 Ingress 负载均衡器配置示例。示例是 HAProxy 负载均衡器的 **/etc/haproxy/haproxy.cfg** 配置。这个示例不是为选择一个负载平衡解决方案提供建议。

在这个示例中，将相同的负载均衡器用于 Kubernetes API 和应用入口流量。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。



注意

如果您使用 HAProxy 作为负载均衡器，并且 SELinux 设置为 **enforcing**，您必须通过运行 **setsebool -P haproxy_connect_any=1** 来确保 HAProxy 服务可以绑定到配置的 TCP 端口。

例 13.3. API 和应用程序入口负载均衡器配置示例

```
global
  log         127.0.0.1 local2
  pidfile     /var/run/haproxy.pid
  maxconn     4000
  daemon
defaults
  mode                    http
  log                     global
  option                  dontlognull
  option http-server-close
  option                  redispatch
  retries                 3
  timeout http-request    10s
  timeout queue           1m
  timeout connect         10s
  timeout client          1m
  timeout server          1m
  timeout http-keep-alive 10s
  timeout check           10s
  maxconn                 3000
```

```
listen api-server-6443 1
  bind *:6443
  mode tcp
  server master0 master0.ocp4.example.com:6443 check inter 1s
  server master1 master1.ocp4.example.com:6443 check inter 1s
  server master2 master2.ocp4.example.com:6443 check inter 1s
listen machine-config-server-22623 2
  bind *:22623
  mode tcp
  server master0 master0.ocp4.example.com:22623 check inter 1s
  server master1 master1.ocp4.example.com:22623 check inter 1s
  server master2 master2.ocp4.example.com:22623 check inter 1s
listen ingress-router-443 3
  bind *:443
  mode tcp
  balance source
  server worker0 worker0.ocp4.example.com:443 check inter 1s
  server worker1 worker1.ocp4.example.com:443 check inter 1s
listen ingress-router-80 4
  bind *:80
  mode tcp
  balance source
  server worker0 worker0.ocp4.example.com:80 check inter 1s
  server worker1 worker1.ocp4.example.com:80 check inter 1s
```

1 端口 **6443** 处理 Kubernetes API 流量并指向 control plane 机器。

2 端口 **22623** 处理机器配置服务器流量并指向 control plane 机器。

3 端口 **443** 处理 HTTPS 流量，并指向运行 Ingress Controller pod 的机器。默认情况下，Ingress Controller Pod 在计算机器上运行。

4 端口 **80** 处理 HTTP 流量，并指向运行 Ingress Controller pod 的机器。默认情况下，Ingress Controller Pod 在计算机器上运行。

注意

如果要部署一个带有零计算节点的三节点集群，Ingress Controller Pod 在 control plane 节点上运行。在三节点集群部署中，您必须配置应用程序入口负载均衡器，将 HTTP 和 HTTPS 流量路由到 control plane 节点。

提示

如果您使用 HAProxy 作为负载均衡器，您可以通过在 HAProxy 节点上运行 **netstat -nltupe** 来检查 **haproxy** 进程是否在侦听端口 **6443**、**22623**、**443** 和 **80**。

13.1.8. 示例：绑定和 VLAN 接口节点网络配置

以下 **agent-config.yaml** 文件是绑定和 VLAN 接口的清单示例。

```
apiVersion: v1alpha1
kind: AgentConfig
```

```
rendezvousIP: 10.10.10.14
hosts:
  - hostname: master0
    role: master
    interfaces:
     - name: enp0s4
       macAddress: 00:21:50:90:c0:10
     - name: enp0s5
       macAddress: 00:21:50:90:c0:20
    networkConfig:
      interfaces:
        - name: bond0.300 (1)
          type: vlan (2)
          state: up
          vlan:
            base-iface: bond0
            id: 300
          ipv4:
            enabled: true
            address:
              - ip: 10.10.10.14
                prefix-length: 24
            dhcp: false
        - name: bond0 (3)
          type: bond (4)
          state: up
          mac-address: 00:21:50:90:c0:10 (5)
          ipv4:
            enabled: false
          ipv6:
            enabled: false
          link-aggregation:
            mode: active-backup (6)
            options:
              miimon: "150" (7)
            port:
              - enp0s4
              - enp0s5
      dns-resolver: (8)
        config:
          server:
            - 10.10.10.11
            - 10.10.10.12
      routes:
        config:
          - destination: 0.0.0.0/0
            next-hop-address: 10.10.10.10 (9)
            next-hop-interface: bond0.300 (10)
            table-id: 254
```

(1) (3) 接口的名称。

(2) 接口的类型。这个示例创建了一个 VLAN。

(4) 接口的类型。这个示例创建了一个绑定。

5 接口的 mac 地址。

6 **mode** 属性指定绑定模式。

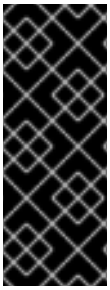
7 以毫秒为单位指定 MII 链接监控频率。这个示例每 150 毫秒检查绑定链接。

8 可选：指定 DNS 服务器的搜索和服务器设置。

9 节点流量的下一跳地址。这必须与为特定接口设定的 IP 地址位于同一个子网中。

10 节点流量的下一跳接口。

13.1.9. 示例：绑定和 SR-IOV 双节点网络配置



重要

支持与为 SR-IOV 设备启用 NIC 分区关联的第 1 天操作只是一个技术预览功能。技术预览功能不受红帽产品服务等级协议（SLA）支持，且功能可能并不完整。红帽不推荐在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。

有关红帽技术预览功能支持范围的更多信息，请参阅技术预览功能支持范围。

以下 **agent-config.yaml** 文件是带有绑定和 SR-IOV 接口的双端口 NIC 的清单示例：

```
apiVersion: v1alpha1
kind: AgentConfig
rendezvousIP: 10.10.10.14
hosts:
  - hostname: worker-1
    interfaces:
      - name: eno1
        macAddress: 0c:42:a1:55:f3:06
      - name: eno2
        macAddress: 0c:42:a1:55:f3:07
    networkConfig: 1
      interfaces: 2
        - name: eno1 3
          type: ethernet 4
          state: up
          mac-address: 0c:42:a1:55:f3:06
          ipv4:
            enabled: true
            dhcp: false 5
          ethernet:
            sr-iov:
              total-vfs: 2 6
          ipv6:
            enabled: false
        - name: sriov:eno1:0
          type: ethernet
          state: up 7
          ipv4:
```

```
    enabled: false (8)
  ipv6:
    enabled: false
    dhcp: false
- name: sriov:eno1:1
  type: ethernet
  state: down
- name: eno2
  type: ethernet
  state: up
  mac-address: 0c:42:a1:55:f3:07
  ipv4:
    enabled: true
  ethernet:
    sr-iov:
      total-vfs: 2
  ipv6:
    enabled: false
- name: sriov:eno2:0
  type: ethernet
  state: up
  ipv4:
    enabled: false
  ipv6:
    enabled: false
- name: sriov:eno2:1
  type: ethernet
  state: down
- name: bond0
  type: bond
  state: up
  min-tx-rate: 100 (9)
  max-tx-rate: 200 (10)
  link-aggregation:
    mode: active-backup (11)
    options:
      primary: sriov:eno1:0 (12)
    port:
      - sriov:eno1:0
      - sriov:eno2:0
  ipv4:
    address:
      - ip: 10.19.16.57 (13)
        prefix-length: 23
    dhcp: false
    enabled: true
  ipv6:
    enabled: false
  dns-resolver:
    config:
      server:
        - 10.11.5.160
        - 10.2.70.215
  routes:
    config:
```

```
    - destination: 0.0.0.0/0
      next-hop-address: 10.19.17.254
      next-hop-interface: bond0 14
      table-id: 254
```

1. **networkConfig** 字段包含有关主机的网络配置的信息，子字段包括 **接口**、**dns-resolver** 和 **routes**。
2. **interfaces** 字段是为主机定义的网络接口数组。
3. 接口的名称。
4. 接口的类型。这个示例创建了一个以太网接口。
5. 如果物理功能 (PF) 没有严格要求，则将其设置为 **false** 以禁用 DHCP。
6. 把它设置为要实例化的 SR-IOV 虚拟功能 (VF) 的数量。
7. 把它设置为 **up**。
8. 把它设置为 **false**，以禁用附加到绑定的 VF 的 IPv4 寻址。
9. 为 VF 设置最小传输率（以 Mbps 为单位）。这个示例值设置 100 Mbps 的速度。
    - 这个值必须小于或等于最大传输率。
    - Intel NIC 不支持 **min-tx-rate** 参数。如需更多信息，请参阅 BZ#1772847。
10. 为 VF 设置最大传输率（以 Mbps 为单位）。此示例值设置 200 Mbps 的速度。
11. 设置所需的绑定模式。
12. 设置绑定接口的首选端口。主设备是要使用的绑定接口的第一个，除非失败，否则不会被取消。当绑定接口中的一个 NIC 速度更快时，此设置特别有用，因此可以处理较大的负载。只有在绑定接口处于 **active-backup** 模式（模式 1）和 **balance-tlb** （模式 5）时，此设置才有效。
13. 为绑定接口设置静态 IP 地址。这是节点 IP 地址。
14. 将 **bond0** 设置为默认路由的网关。

其他资源

- 配置网络绑定

13.1.10. 裸机 install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多详情，或修改所需参数的值。

```
apiVersion: v1
baseDomain: example.com 1
compute: 2
- name: worker
  replicas: 0 3
controlPlane: 4
```

```
  name: master
  replicas: 1 5
metadata:
  name: sno-cluster 6
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14 7
    hostPrefix: 23 8
  networkType: OVNKubernetes 9
  serviceNetwork: 10
  - 172.30.0.0/16
platform:
  none: {} 11
fips: false 12
pullSecret: '{"auths": ...}' 13
sshKey: 'ssh-ed25519 AAAA...' 14
```

1 集群的基域。所有 DNS 记录都必须是这个基域的子域，并包含集群名称。

2 4 **controlPlane** 部分是一个单个映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，controlPlane **部分** 的第一行则不以连字符开头。仅使用一个 control plane 池。

3 此参数控制基于代理的安装在触发安装过程前等待发现的计算机器数量。它是必须使用生成的 ISO 引导的计算机器数量。

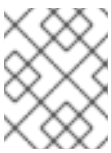
注意

如果要安装一个三节点集群，在安装 Red Hat Enterprise Linux CoreOS(RHCOS)机器时不要部署任何计算机器。

5 您添加到集群的 control plane 机器数量。由于集群使用这些值作为集群中的 etcd 端点数量，所以该值必须与您部署的 control plane 机器数量匹配。

6 您在 DNS 记录中指定的集群名称。

7 从中分配 Pod IP 地址的 IP 地址块。此块不得与现有物理网络重叠。这些 IP 地址用于 pod 网络。如果需要从外部网络访问 pod，您必须配置负载均衡器和路由器来管理流量。



注意

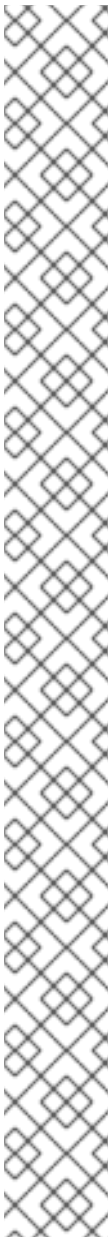
类 E CIDR 范围被保留以供以后使用。要使用 Class E CIDR 范围，您必须确保您的网络环境接受 Class E CIDR 范围内的 IP 地址。

8 分配给每个节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从 given **cidr** 中分配 a **/23** 子网，这样就能有 510（2^(32 - 23)- 2）个 pod IP 地址。如果需要从外部网络访问节点，请配置负载均衡器和路由器来管理流量。

9 要安装的集群网络插件。默认值 **OVNKubernetes** 是唯一支持的值。

10 用于服务 IP 地址的 IP 地址池。您只能输入一个 IP 地址池。此块不得与现有物理网络重叠。如果您需要从外部网络访问服务，请配置负载均衡器和路由器来管理流量。

11 对于单节点集群，您必须将平台设置为 **none**。对于多节点集群，您可以将平台设置为 **vsphere**、**baremetal** 或 **none**。



注意

如果将平台设置为 **vsphere** 或 **baremetal**，您可以以三种方式为集群节点配置 IP 地址端点：

- IPv4
- IPv6
- IPv4 和 IPv6 并行 (dual-stack)

双栈网络示例

```
networking:
  clusterNetwork:
    - cidr: 172.21.0.0/16
      hostPrefix: 23
    - cidr: fd02::/48
      hostPrefix: 64
  machineNetwork:
    - cidr: 192.168.11.0/16
    - cidr: 2001:DB8::/32
  serviceNetwork:
    - 172.22.0.0/16
    - fd03::/112
  networkType: OVNKubernetes
platform:
  baremetal:
    apiVIPs:
    - 192.168.11.3
    - 2001:DB8::4
    ingressVIPs:
    - 192.168.11.4
    - 2001:DB8::5
```

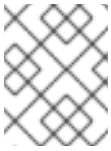
12 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS)时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。

13 此 pull secret 允许您与所含授权机构提供的服务进行身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

14 Red Hat Enterprise Linux CoreOS(RHCOS)中 **core** 用户的 SSH 公钥。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

13.1.11. 在代理 ISO 创建前验证检查

基于代理的安装程序在创建 ISO 之前对用户定义的 YAML 文件执行验证检查。验证成功后，会创建代理 ISO。

install-config.yaml

- 支持**裸机**, **vsphere** 和 **none** 平台。
- 如果平台为 **none**，则 **networkType** 参数需要为 **OVNKubernetes**。
- 如果是裸机和 vSphere 平台，则需要设置 **apiVIPs** 和 **ingressVIPs** 参数。
- 在 **agent-config.yaml** 文件中，裸机平台配置中有一些特定于主机的字段将被忽略。如果设置了这些字段，则会记录警告消息。

agent-config.yaml

- 每个接口都必须有一个定义的 MAC 地址。另外，所有接口都必须具有不同的 MAC 地址。
- 每个主机必须至少定义一个接口。
- root 设备提示不支持全局名称 (WWN) 供应商扩展。
- **host** 对象中的 **role** 参数的值必须是 **master** 或 **worker**。

13.1.11.1. ZTP 清单

agent-cluster-install.yaml

- 对于 IPv6，**networkType** 参数唯一支持的值是 **OVNKubernetes**。**OpenshiftSDN** 值只能用于 IPv4。

cluster-image-set.yaml

- **ReleaseImage** 参数必须与安装程序中定义的发行版本匹配。

13.1.12. 后续步骤

- 使用基于代理的安装程序安装集群

13.2. 了解断开连接的安装镜像

您可以使用镜像 registry 进行断开连接的安装，并确保集群只使用满足机构对外部内容控制的容器镜像。在受限网络中置备的基础架构上安装集群前，您必须将所需的容器镜像镜像(mirror)到那个环境中。要镜像容器镜像，您必须有一个 registry 才能进行镜像(mirror)。

13.2.1. 通过基于代理的安装程序为断开连接的安装镜像镜像

您可以使用以下流程之一将 OpenShift Container Platform 镜像存储库镜像到您的镜像 registry：

- 为断开连接的安装 mirror 镜像
- 使用 oc-mirror 插件为断开连接的安装镜像镜像

13.2.2. 关于为断开连接的 registry 镜像 OpenShift Container Platform 镜像存储库

要将镜像镜像用于基于代理的安装程序的断开连接的安装，您必须修改 **install-config.yaml** 文件。

您可以使用 **oc adm release mirror** 或 **oc mirror** 命令的输出来镜像发行镜像。这取决于您用来设置镜像 registry 的命令。

以下示例显示了 **oc adm release mirror** 命令的输出。

```
$ oc adm release mirror
```

输出示例

```
To use the new mirrored repository to install, add the following
section to the install-config.yaml:

imageContentSources:

mirrors:
virthost.ostest.test.metalkube.org:5000/localimages/local-release-image
source: quay.io/openshift-release-dev/ocp-v4.0-art-dev
mirrors:
virthost.ostest.test.metalkube.org:5000/localimages/local-release-image
source: registry.ci.openshift.org/ocp/release
```

以下示例显示了 oc-mirror 插件生成的 **imageContentSourcePolicy.yaml** 文件的一部分。该文件可以在结果目录中找到，如 **oc-mirror-workspace/results-1682697932/**。

imageContentSourcePolicy.yaml 文件示例

```
spec:
  repositoryDigestMirrors:
  - mirrors:
    - virthost.ostest.test.metalkube.org:5000/openshift/release
    source: quay.io/openshift-release-dev/ocp-v4.0-art-dev
  - mirrors:
    - virthost.ostest.test.metalkube.org:5000/openshift/release-images
    source: quay.io/openshift-release-dev/ocp-release
```

13.2.2.1. 配置基于代理的安装程序以使用镜像的镜像

您必须使用 **oc adm release mirror** 命令的输出或 oc-mirror 插件来配置基于 Agent 的安装程序以使用镜像镜像。

流程

1. 如果您使用 oc-mirror 插件来镜像发行镜像：
   a. 打开位于结果目录中的 **imageContentSourcePolicy.yaml**，如 **oc-mirror-workspace/results-1682697932/**。
   b. 在 yaml 文件的 **repositoryDigestMirrors** 部分中复制文本。
2. 如果使用 **oc adm release mirror** 命令镜像发行镜像：
   - 在命令输出的 **imageContentSources** 部分中复制文本。
3. 将复制的文本粘贴到 **install-config.yaml** 文件的 **imageContentSources** 字段中。
4. 将用于镜像 registry 的证书文件添加到 yaml 文件的 **additionalTrustBundle** 字段中。

重要

该值必须是您用于镜像 registry 的证书文件内容。证书文件可以是现有的可信证书颁发机构，也可以是您为镜像 registry 生成的自签名证书。

install-config.yaml 文件示例

```
additionalTrustBundle: |
  -----BEGIN CERTIFICATE-----
  ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
  -----END CERTIFICATE-----
```

5. 如果使用 GitOps ZTP 清单：将 **registry.conf** 和 **ca-bundle.crt** 文件添加到 **mirror** 路径中，以便在代理 ISO 镜像中添加镜像配置。

注意

您可以从 **oc adm release mirror** 命令的输出或 **oc mirror** 插件创建 **registries.conf** 文件。**/etc/containers/registries.conf** 文件的格式已更改。现在它是第 2 版，采用 TOML 格式。

registry.conf 文件示例

```
[[registry]]
location = "registry.ci.openshift.org/ocp/release" mirror-by-digest-only = true

[[registry.mirror]] location = "virthost.ostest.test.metalkube.org:5000/localimages/local-release-image"

[[registry]]
location = "quay.io/openshift-release-dev/ocp-v4.0-art-dev" mirror-by-digest-only = true

[[registry.mirror]] location = "virthost.ostest.test.metalkube.org:5000/localimages/local-release-image"
```

13.3. 使用基于代理的安装程序安装 OPENSHIFT CONTAINER PLATFORM 集群

使用以下步骤使用基于代理的安装程序安装 OpenShift Container Platform 集群。

13.3.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读选择集群安装方法并为用户准备它的文档。
- 如果使用防火墙或代理，将其配置为允许集群需要访问的站点。

13.3.2. 使用基于代理的安装程序安装 OpenShift Container Platform

以下流程在断开连接的环境中部署单节点 OpenShift Container Platform。您可以使用这些步骤作为基础，并根据您的要求进行修改。

13.3.2.1. 下载基于代理的安装程序

使用这个流程下载安装所需的基于代理的安装程序和 CLI。

流程

1. 使用您的登录凭证登录到 OpenShift Container Platform web 控制台。
2. 进入到 Datacenter。
3. **在本地点 Run Agent-based Installer。**
4. 为 **OpenShift Installer** 和**命令行界面**选择操作系统和架构。
5. 点 **Download Installer** 下载并提取安装程序。
6. 您可以通过点 **Download pull secret** 或 **Copy pull secret** 下载或复制 pull secret。
7. 点 **Download command-line tools**，将 **openshift-install** 二进制文件放在 **PATH** 中的目录中。

13.3.2.2. 验证安装基于代理的安装程序集群的支持的架构

在使用基于代理的安装程序安装 OpenShift Container Platform 集群前，您可以验证要在其上安装集群的支持的架构。这个过程是可选的。

先决条件

- 已安装 OpenShift CLI（**oc**）。
- 您已下载了安装程序。

流程

1. 登录 OpenShift CLI (**oc**)。
2. 运行以下命令检查您的发行镜像有效负载：

```
$ ./openshift-install version
```

输出示例

```
./openshift-install 4.16.0
built from commit abc123def456
release image quay.io/openshift-release-dev/ocp-
release@sha256:123abc456def789ghi012jkl345mno678pqr901stu234vwx567yz0
release architecture amd64
```

如果您使用带有 **multi** 有效负载的发行镜像，这个命令的输出中显示的发**行架构**是默认的架构。

3. 要检查有效负载的架构，请运行以下命令：

```
$ oc adm release info <release_image> -o jsonpath="{ .metadata.metadata}" 1
```

1 将 **<release_image>** 替换为发行镜像。例如：**quay.io/openshift-release-dev/ocp-release@sha256:123abc456def789ghi012jkl345mno678pqr901stu234vwx567yz0**。

当发行镜像使用 **multi** 有效负载时的输出示例：

```
{"release.openshift.io architecture":"multi"}
```

如果使用带有 **multi** 有效负载的发行镜像，则可以在不同的架构上安装集群，如 **arm64**, **amd64**, **s390x**, 和 **ppc64le**。否则，您只能在 **openshift-install version** 命令的输出中所显示的发**行构架**上安装集群。

13.3.2.3. 创建首选配置输入

使用这个流程创建用于创建代理镜像的首选配置输入。

流程

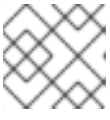
1. 运行以下命令来安装 **nmstate** 依赖项：

```
$ sudo dnf install /usr/bin/nmstatectl -y
```

2. 将 **openshift-install** 二进制文件放到 PATH 中的目录中。

3. 运行以下命令，创建一个目录来存储安装配置：

```
$ mkdir ~/<directory_name>
```



注意

这是基于代理的安装的首选方法。使用 GitOps ZTP 清单是可选的。

4. 创建 **install-config.yaml** 文件：

```
$ cat << EOF > ./my-cluster/install-config.yaml
apiVersion: v1
baseDomain: test.example.com
compute:
- architecture: amd64 1
```

```
  hyperthreading: Enabled
  name: worker
  replicas: 0
controlPlane:
  architecture: amd64
  hyperthreading: Enabled
  name: master
  replicas: 1
metadata:
  name: sno-cluster 2
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 192.168.0.0/16
  networkType: OVNKubernetes 3
  serviceNetwork:
  - 172.30.0.0/16
platform: 4
  none: {}
pullSecret: '<pull_secret>' 5
sshKey: '<ssh_pub_key>' 6
EOF
```

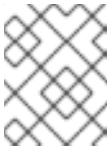
1 指定系统架构。有效值为 **amd64**,**arm64**,**ppc64le**, 和 **s390x**。

如果使用带有 **multi** 有效负载的发行镜像，则可以在不同的架构上安装集群，如 **arm64**, **amd64**, **s390x**, 和 **ppc64le**。否则，您只能在 **openshift-install version** 命令的输出中所显示的发**行构架**上安装集群。如需更多信息，请参阅"验证安装基于代理的安装程序集群的受支持架构"。

2 必需。指定集群名称。

3 要安装的集群网络插件。默认值 **OVNKubernetes** 是唯一支持的值。

4 指定您的平台。



注意

对于裸机平台，默认使用 **install-config.yaml** 文件的 platform 部分中进行的主机设置，除非它们被 **agent-config.yaml** 文件中的配置覆盖。

5 指定 pull secret。

6 指定 SSH 公钥。

注意

如果将平台设置为 **vSphere** 或 **baremetal**，您可以使用三种方式为集群节点配置 IP 地址端点：

- IPv4
- IPv6
- IPv4 和 IPv6 并行 (dual-stack)

IPv6 仅在裸机平台上被支持。

双栈网络示例

```
networking:
  clusterNetwork:
    - cidr: 172.21.0.0/16
      hostPrefix: 23
    - cidr: fd02::/48
      hostPrefix: 64
  machineNetwork:
    - cidr: 192.168.11.0/16
    - cidr: 2001:DB8::/32
  serviceNetwork:
    - 172.22.0.0/16
    - fd03::/112
  networkType: OVNKubernetes
platform:
  baremetal:
    apiVIPs:
    - 192.168.11.3
    - 2001:DB8::4
    ingressVIPs:
    - 192.168.11.4
    - 2001:DB8::5
```

注意

使用断开连接的镜像 registry 时，您必须将之前为镜像 registry 创建的证书文件添加到 **install-config.yaml** 文件的 **additionalTrustBundle** 字段中。

5. 创建 **agent-config.yaml** 文件：

```
$ cat > agent-config.yaml << EOF
apiVersion: v1beta1
kind: AgentConfig
metadata:
  name: sno-cluster
rendezvousIP: 192.168.111.80 (1)
hosts: (2)
  - hostname: master-0 (3)
    interfaces:
      - name: eno1
```

```
      macAddress: 00:ef:44:21:e6:a5
  rootDeviceHints: 4
    deviceName: /dev/sdb
  networkConfig: 5
    interfaces:
      - name: eno1
        type: ethernet
        state: up
        mac-address: 00:ef:44:21:e6:a5
        ipv4:
          enabled: true
          address:
            - ip: 192.168.111.80
              prefix-length: 23
          dhcp: false
    dns-resolver:
      config:
        server:
          - 192.168.111.1
    routes:
      config:
        - destination: 0.0.0.0/0
          next-hop-address: 192.168.111.2
          next-hop-interface: eno1
          table-id: 254
EOF
```

1. 此 IP 地址用于确定哪些节点执行 bootstrap 过程，以及运行 **assisted-service** 组件。当您没有在 **networkConfig** 参数中指定至少一个主机的 IP 地址时，您必须提供 rendezvous IP 地址。如果没有提供此地址，则会从提供的主机的 **networkConfig** 中选择一个 IP 地址。
2. 可选：主机配置。定义的主机数量不能超过 **install-config.yaml** 文件中定义的主机总数，这是 **compute.replicas** 和 **controlPlane.replicas** 参数的值的总和。
3. 可选：覆盖从动态主机配置协议(DHCP)或反向 DNS 查找中获取的主机名。每个主机必须具有由这些方法提供的唯一主机名。
4. 启用将 Red Hat Enterprise Linux CoreOS (RHCOS)镜像置备到特定设备。安装程序会按照发现设备的顺序检查设备，并将发现的值与 hint 值进行比较。它使用第一个与 hint 值匹配的发现设备。
5. 可选：以 NMState 格式配置主机的网络接口。

其他资源

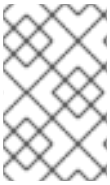
- 为 VMware vCenter 配置区域和区域
- 验证安装基于代理的安装程序集群的支持的架构
- 配置基于代理的安装程序以使用镜像的镜像

13.3.2.4. 可选：创建额外的清单文件

您可以创建额外的清单，以便在 **install-config.yaml** 和 **agent-config.yaml** 文件中提供的配置之外进一步配置集群。

13.3.2.4.1. 创建包含额外清单的目录

如果创建额外的清单，以在 **install-config.yaml** 和 **agent-config.yaml** 文件外配置基于 Agent 的安装，则必须在安装目录中创建 **openshift** 子目录。所有附加机器配置都必须位于此子目录中。



注意

您可以添加的附加清单的最常见类型是 **MachineConfig** 对象。有关您可以在基于代理的安装过程中添加的 **MachineConfig** 对象示例，请参阅"添加资源"部分中的"使用 MachineConfig 对象来配置节点"。

流程

- 在安装主机上，运行以下命令在安装目录中创建一个 **openshift** 子目录：

```
$ mkdir <installation_directory>/openshift
```

其他资源

- 使用 MachineConfig 对象配置节点

13.3.2.4.2. 磁盘分区

通常，您应该使用在 RHCOS 安装过程中创建的默认磁盘分区。然而，在有些情况下您可能需要为预期增长的目录创建独立分区。

OpenShift Container Platform 支持添加单个分区将存储附加到 **/var** 目录或 **/var** 的子目录中。例如：

- **/var/lib/containers** ：保存随着系统中添加更多镜像和容器而增长的容器相关内容。
- **/var/lib/etcd** ：保存您可能希望独立保留的数据，比如 etcd 存储的性能优化。
- **/var** ：保存您可能希望独立保留的数据，以满足审计等目的。

重要

对于大于 100GB 的磁盘大小，特别是磁盘大小大于 1TB，请创建一个独立的 **/var** 分区。

通过单独存储 **/var** 目录的内容，可以更轻松地根据需要为区域扩展存储，并在以后重新安装 OpenShift Container Platform，并保持该数据的完整性。使用这个方法，您不必再次拉取所有容器，在更新系统时也不必复制大量日志文件。

将独立分区用于 **/var** 目录或 **/var** 的子目录也会防止分区目录中的数据增加填充根文件系统。

以下流程通过添加机器配置清单来设置独立的 **/var** 分区，该清单会在安装准备阶段封装到节点类型的 Ignition 配置文件中。

先决条件

- 您已在安装目录中创建了 **openshift** 子目录。

流程

1. 创建用于配置额外分区的 Butane 配置。例如，将文件命名为 **$HOME/clusterconfig/98-var-partition.bu**，将磁盘设备名称改为 **worker** 系统上存储设备的名称，并根据情况设置存储大小。这个示例将 **/var** 目录放在一个单独的分区中：

```
variant: openshift
version: 4.16.0
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
storage:
  disks:
  - device: /dev/disk/by-id/<device_name> 1
    partitions:
    - label: var
      start_mib: <partition_start_offset> 2
      size_mib: <partition_size> 3
      number: 5
  filesystems:
    - device: /dev/disk/by-partlabel/var
      path: /var
      format: xfs
      mount_options: [defaults, prjquota] 4
      with_mount_unit: true
```

1. 要分区的磁盘的存储设备名称。
2. 当在引导磁盘中添加数据分区时，推荐最少使用偏移值 25000 兆字节。root 文件系统会自动调整大小以填充所有可用空间（最多到指定的偏移值）。如果没有指定偏移值，或者指定的值小于推荐的最小值，则生成的 root 文件系统会太小，而在以后进行的 RHCOS 重新安装可能会覆盖数据分区的开始部分。
3. 以兆字节为单位的数据分区大小。
4. 对于用于容器存储的文件系统，必须启用 **prjquota** 挂载选项。

注意

在创建单独的 **/var** 分区时，如果不同的实例类型没有相同的设备名称，则无法将不同的实例类型用于计算节点。

2. 从 Butane 配置创建一个清单，并将它保存到 **clusterconfig/openshift** 目录中。例如，运行以下命令：

```
$ butane $HOME/clusterconfig/98-var-partition.bu -o $HOME/clusterconfig/openshift/98-var-partition.yaml
```

13.3.2.5. 可选：使用 ZTP 清单

您可以使用 GitOps Zero Touch Provisioning (ZTP) 清单，在通过 **install-config.yaml** 和 **agent-config.yaml** 文件选项外配置安装。

注意

GitOps ZTP 清单可以使用或不提前配置 **install-config.yaml** 和 **agent-config.yaml** 文件生成。如果您选择配置 **install-config.yaml** 和 **agent-config.yaml** 文件，则配置会在生成时导入到 ZTP 集群清单中。

先决条件

- 您已将 **openshift-install** 二进制文件放在 **PATH** 中的目录中。
- 可选：您已创建并配置了 **install-config.yaml** 和 **agent-config.yaml** 文件。

流程

1. 使用以下命令生成 ZTP 集群清单：

```
$ openshift-install agent create cluster-manifests --dir <installation_directory>
```

重要

如果您已创建了 **install-config.yaml** 和 **agent-config.yaml** 文件，则这些文件将被删除，并替换为通过这个命令生成的集群清单。

在运行 **openshift-install agent create cluster-manifests** 命令时，对 **install-config.yaml** 和 **agent-config.yaml** 文件所做的任何配置都会导入到 ZTP 集群清单中。

2. 进入 **cluster-manifests** 目录：

```
$ cd <installation_directory>/cluster-manifests
```

3. 在 **cluster-manifests** 目录中配置清单文件。如需示例文件，请参阅 "Sample GitOps ZTP 自定义资源" 部分。
4. 断开连接的集群： 如果您在生成 ZTP 清单前没有在 **install-config.yaml** 文件中定义镜像配置，请执行以下步骤：
   a. 进入 **mirror** 目录：

```
$ cd ../mirror
```

   b. 在 **mirror** 目录中配置清单文件。

其他资源

- GitOps ZTP 自定义资源示例。
- 请参阅 网络边缘的挑战， 以了解更多有关 GitOps Zero Touch Provisioning (ZTP)的信息。

13.3.2.6. 可选：加密磁盘

在使用基于代理的安装程序安装 OpenShift Container Platform 时，使用此流程加密您的磁盘或分区。

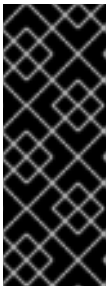
先决条件

- 您已创建并配置了 **install-config.yaml** 和 **agent-config.yaml** 文件，除非您使用 ZTP 清单。
- 您已将 **openshift-install** 二进制文件放在 **PATH** 中的目录中。

流程

1. 使用以下命令生成 ZTP 集群清单：

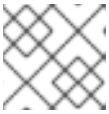
```
$ openshift-install agent create cluster-manifests --dir <installation_directory>
```



重要

如果您已创建了 **install-config.yaml** 和 **agent-config.yaml** 文件，则这些文件将被删除，并替换为通过这个命令生成的集群清单。

在运行 **openshift-install agent create cluster-manifests** 命令时，对 **install-config.yaml** 和 **agent-config.yaml** 文件所做的任何配置都会导入到 ZTP 集群清单中。



注意

如果您已经生成了 ZTP 清单，请跳过这一步。

2. 进入 **cluster-manifests** 目录：

```
$ cd <installation_directory>/cluster-manifests
```

3. 在 **agent-cluster-install.yaml** 文件中添加以下部分：

```
diskEncryption:
  enableOn: all (1)
  mode: tang (2)
  tangServers: "server1": "http://tang-server-1.example.com:7500" (3)
```

(1) 指定要启用磁盘加密的节点。有效值为 'none'、'all'、'master' 和 'worker'。

(2) 指定要使用的磁盘加密模式。有效值为 'tpmv2' 和 'tang'。

(3) 可选：如果您使用 Tang，请指定 Tang 服务器。

其他资源

- 关于磁盘加密

13.3.2.7. 创建并引导代理镜像

使用这个流程在机器上引导代理镜像。

流程

1. 运行以下命令来创建代理镜像：

```
$ openshift-install --dir <install_directory> agent create image
```

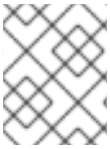
注意

Red Hat Enterprise Linux CoreOS (RHCOS) 支持主磁盘上的多路径，允许对硬件故障进行更强大的弹性，以实现更高的主机可用性。在代理 ISO 镜像中默认启用多路径，默认 **/etc/multipath.conf** 配置。

2. 在裸机机器上引导 **agent.x86_64.iso**,**agent.aarch64.iso**, 或 **agent.s390x.iso** 镜像。

13.3.2.8. 使用 RHEL KVM 添加 IBM Z (R)代理

使用以下步骤使用 RHEL KVM 手动添加 IBM Z® 代理。



注意

目前，IBM Z® (**s390x**) 上的 ISO 引导支持仅适用于 RHEL KVM，这为选择 PXE 或基于 ISO 的安装提供了灵活性。对于使用 z/VM 的安装，只支持 PXE 引导。

流程

1. 引导 RHEL KVM 机器。

2. 要部署虚拟服务器，请使用以下参数运行 **virt-install** 命令：

```
$ virt-install
    --name <vm_name> \
    --autostart \
    --memory=<memory> \
    --cpu host \
    --vcpus=<vcpus> \
    --cdrom <agent.iso_image> \ 1
    --disk pool=default,size=<disk_pool_size> \
    --network network:default,mac=<mac_address> \
    --graphics none \
    --noautoconsole \
    --os-variant rhel9.0 \
    --wait=-1
```

1 对于 **--cdrom** 参数，指定 HTTP 或 HTTPS 服务器中的 ISO 镜像的位置。

13.3.2.9. 验证当前安装主机是否可以拉取发行镜像

引导代理镜像和网络服务可用于主机后，代理控制台应用会执行拉取检查，以验证当前主机是否可以检索发行镜像。

如果主拉取检查通过，您可以退出应用程序以继续安装。如果拉取检查失败，应用程序会执行额外的检查，如 TUI 的**额外检查**部分中所示，以帮助您对问题进行故障排除。只要主拉取检查成功，则对任何其他检查失败不一定至关重要。

如果有可能导致安装失败的主机网络配置问题，您可以使用控制台应用程序调整网络配置。

重要

如果代理控制台应用程序检测到主机网络配置问题，则安装工作流将停止，直到用户手动停止控制台应用程序并信号继续操作。

流程

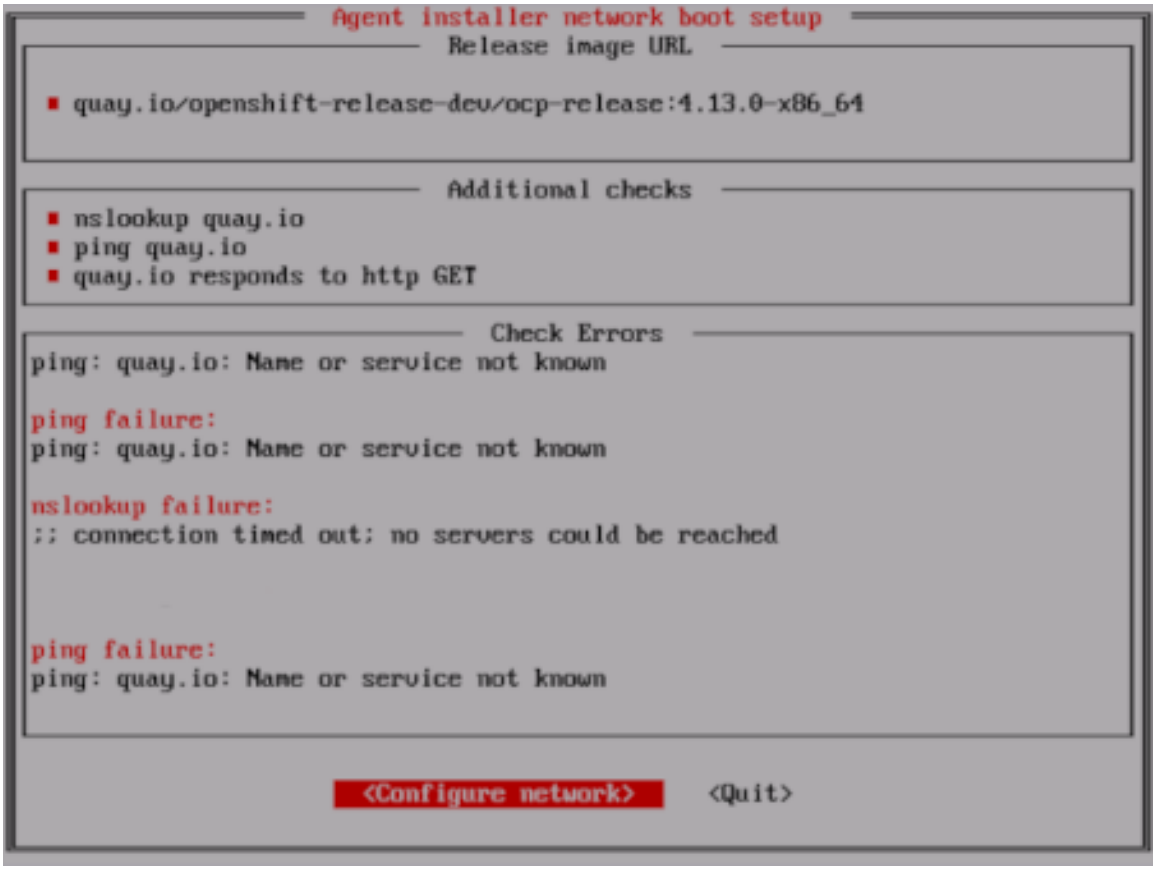
1. 等待代理控制台应用程序检查是否可以从 registry 中拉取配置的发行镜像。
2. 如果代理控制台应用程序指出安装程序连接检查已通过，请等待提示符超时。

注意

您仍然可以选择查看或更改网络配置设置，即使连接检查已通过了。

但是，如果您选择与代理控制台应用程序交互，而不是让其超时，您必须手动退出 TUI 才能继续安装。

3. 如果代理控制台应用程序检查失败（由 **发行镜像 URL** pull 检查旁的红色图标表示），请按照以下步骤重新配置主机的网络设置：
   a. 阅读 TUI 的**检查错误**部分。本节显示特定于失败检查的错误消息。



   b. 选择 **Configure network** 以启动 NetworkManager TUI。
   c. 选择 **Edit a connection** 并选择您要重新配置的连接。
   d. 编辑配置并选择 **OK** 保存您的更改。
   e. 选择 **Back** 返回到 NetworkManager TUI 的主屏幕。
   f. 选择 **Activate a Connection**。

g. 选择重新配置的网络来取消激活它。

h. 再次选择重新配置的网络来重新激活它。

i. 选择 **Back**，然后选择 **Quit** 以返回到代理控制台应用程序。

j. 至少等待五秒，以便持续网络检查使用新的网络配置重新启动。

k. 如果 **Release image URL** pull 检查成功，并显示 URL 旁边的绿色图标，请选择 **Quit** 退出代理控制台应用程序并继续安装。

13.3.2.10. 跟踪并验证安装进度

使用以下步骤跟踪安装进度并验证安装是否成功。

先决条件

- 您已为 Kubernetes API 服务器配置了 DNS 记录。

流程

1. 可选：要了解 bootstrap 主机（渲染主机）何时重启，请运行以下命令：

```
$ ./openshift-install --dir <install_directory> agent wait-for bootstrap-complete \ 1
    --log-level=info 2
```

1 对于 **<install_directory>**，请指定到生成代理 ISO 的目录的路径。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不是 **info**。

输出示例

```
.....
.....
INFO Bootstrap configMap status is complete
INFO cluster bootstrap is complete
```

当 Kubernetes API 服务器提示已在 control plane 机器上引导它时，该命令会成功。

2. 要跟踪进度并验证安装是否成功，请运行以下命令：

```
$ openshift-install --dir <install_directory> agent wait-for install-complete 1
```

1 对于 **<install_directory>** 目录，请指定到生成代理 ISO 的目录的路径。

输出示例

```
.....
.....
INFO Cluster is installed
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run
```

```
INFO     export KUBECONFIG=/home/core/installer/auth/kubeconfig
INFO Access the OpenShift web-console here: https://console-openshift-console.apps.sno-cluster.test.example.com
```

注意

如果使用可选的 GitOps ZTP 清单方法，您可以以三种方式通过 **AgentClusterInstall.yaml** 文件为集群节点配置 IP 地址端点：

- IPv4
- IPv6
- IPv4 和 IPv6 并行 (dual-stack)

IPv6 仅在裸机平台上被支持。

双栈网络示例

```
apiVIP: 192.168.11.3
ingressVIP: 192.168.11.4
clusterDeploymentRef:
  name: mycluster
imageSetRef:
  name: openshift-4.16
networking:
  clusterNetwork:
  - cidr: 172.21.0.0/16
    hostPrefix: 23
  - cidr: fd02::/48
    hostPrefix: 64
  machineNetwork:
  - cidr: 192.168.11.0/16
  - cidr: 2001:DB8::/32
  serviceNetwork:
  - 172.22.0.0/16
  - fd03::/112
  networkType: OVNKubernetes
```

其他资源

- 请参阅使用双栈网络进行部署。
- 请参阅配置 install-config yaml 文件。
- 请参阅配置三节点集群以在裸机环境中部署三节点集群。
- 请参阅关于 root 设备提示。
- 请参阅 NMState 状态示例。

13.3.3. GitOps ZTP 自定义资源示例

可选：您可以使用 GitOps Zero Touch Provisioning (ZTP) 自定义资源 (CR) 对象来使用基于代理的安装程序安装 OpenShift Container Platform 集群。

您可以自定义以下 GitOps ZTP 自定义资源，以指定有关 OpenShift Container Platform 集群的更多详情。以下 GitOps ZTP 自定义资源示例是单节点集群。

agent-cluster-install.yaml

```
apiVersion: extensions.hive.openshift.io/v1beta1
kind: AgentClusterInstall
metadata:
  name: test-agent-cluster-install
  namespace: cluster0
spec:
  clusterDeploymentRef:
    name: ostest
  imageSetRef:
    name: openshift-4.16
  networking:
    clusterNetwork:
    - cidr: 10.128.0.0/14
      hostPrefix: 23
    serviceNetwork:
    - 172.30.0.0/16
  provisionRequirements:
    controlPlaneAgents: 1
    workerAgents: 0
  sshPublicKey: <YOUR_SSH_PUBLIC_KEY>
```

cluster-deployment.yaml

```
apiVersion: hive.openshift.io/v1
kind: ClusterDeployment
metadata:
  name: ostest
  namespace: cluster0
spec:
  baseDomain: test.metalkube.org
  clusterInstallRef:
    group: extensions.hive.openshift.io
    kind: AgentClusterInstall
    name: test-agent-cluster-install
    version: v1beta1
  clusterName: ostest
  controlPlaneConfig:
    servingCertificates: {}
  platform:
    agentBareMetal:
      agentSelector:
        matchLabels:
          bla: aaa
  pullSecretRef:
    name: pull-secret
```

cluster-image-set.yaml

-

```
apiVersion: hive.openshift.io/v1
kind: ClusterImageSet
metadata:
  name: openshift-4.16
spec:
  releaseImage: registry.ci.openshift.org/ocp/release:4.16.0-0.nightly-2022-06-06-025509
```

infra-env.yaml

```
apiVersion: agent-install.openshift.io/v1beta1
kind: InfraEnv
metadata:
  name: myinfraenv
  namespace: cluster0
spec:
  clusterRef:
    name: ostest
    namespace: cluster0
  cpuArchitecture: aarch64
  pullSecretRef:
    name: pull-secret
  sshAuthorizedKey: <YOUR_SSH_PUBLIC_KEY>
  nmStateConfigLabelSelector:
    matchLabels:
      cluster0-nmstate-label-name: cluster0-nmstate-label-value
```

nmstateconfig.yaml

```
apiVersion: agent-install.openshift.io/v1beta1
kind: NMStateConfig
metadata:
  name: master-0
  namespace: openshift-machine-api
  labels:
    cluster0-nmstate-label-name: cluster0-nmstate-label-value
spec:
  config:
    interfaces:
      - name: eth0
        type: ethernet
        state: up
        mac-address: 52:54:01:aa:aa:a1
        ipv4:
          enabled: true
          address:
            - ip: 192.168.122.2
              prefix-length: 23
          dhcp: false
    dns-resolver:
      config:
        server:
          - 192.168.122.1
    routes:
      config:
        - destination: 0.0.0.0/0
```

```
        next-hop-address: 192.168.122.1
        next-hop-interface: eth0
        table-id: 254
  interfaces:
    - name: "eth0"
      macAddress: 52:54:01:aa:aa:a1
```

pull-secret.yaml

```
apiVersion: v1
kind: Secret
type: kubernetes.io/dockerconfigjson
metadata:
  name: pull-secret
  namespace: cluster0
stringData:
  .dockerconfigjson: 'YOUR_PULL_SECRET'
```

其他资源

- 请参阅 网络边缘的挑战， 以了解更多有关 GitOps Zero Touch Provisioning (ZTP)的信息。

13.3.4. 从基于代理的安装收集日志数据

使用以下步骤收集有关基于代理的安装失败的日志数据，以便为支持问题单提供。

先决条件

- 您已为 Kubernetes API 服务器配置了 DNS 记录。

流程

1. 运行以下命令并收集输出：

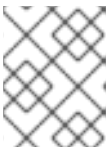
   ```
   $ ./openshift-install --dir <install_directory> agent wait-for bootstrap-complete --log-level=debug
   ```

   错误信息示例

   ```
   ...
   ERROR Bootstrap failed to complete: : bootstrap process timed out: context deadline exceeded
   ```

2. 如果上一命令的输出显示失败，或者 bootstrap 没有进展，请运行以下命令连接到 rendezvous 主机并收集输出：

   ```
   $ ssh core@<node-ip> agent-gather -O >agent-gather.tar.xz
   ```

   注意

   红帽支持可以使用 rendezvous 主机收集的数据诊断大多数问题，但如果某些主机无法注册，从每个主机收集这些数据可能会很有用。

3. 如果 bootstrap 完成且集群节点重启，请运行以下命令并收集输出：

```
$ ./openshift-install --dir <install_directory> agent wait-for install-complete --log-level=debug
```

4. 如果上一命令的输出显示失败，请执行以下步骤：

   a. 运行以下命令，将 **kubeconfig** 文件导出到您的环境：

```
$ export KUBECONFIG=<install_directory>/auth/kubeconfig
```

   b. 要收集用于调试的信息，请运行以下命令：

```
$ oc adm must-gather
```

   c. 运行以下命令，从工作目录中刚刚创建的 **must-gather** 目录创建一个压缩文件：

```
$ tar cvaf must-gather.tar.gz <must_gather_directory>
```

5. 排除 **/auth** 子目录，将部署期间使用的安装目录附加到 红帽客户门户网站上的 支持问题单中。

6. 将从此流程收集的所有其他数据添加到您的支持问题单中。

13.4. 为 OPENSHIFT CONTAINER PLATFORM 准备 PXE 资产

使用以下步骤创建基于代理的安装程序，创建 PXE 引导 OpenShift Container Platform 集群所需的资产。

您在这些步骤中创建的资产将部署单节点 OpenShift Container Platform 安装。您可以使用这些步骤作为基础并根据您的要求修改配置。

13.4.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。

13.4.2. 下载基于代理的安装程序

使用这个流程下载安装所需的基于代理的安装程序和 CLI。

流程

1. 使用您的登录凭证登录到 OpenShift Container Platform web 控制台。
2. 进入到 Datacenter。
3. **在本地点 Run Agent-based Installer。**
4. 为 **OpenShift Installer** 和**命令行界面**选择操作系统和架构。
5. 点 **Download Installer** 下载并提取安装程序。
6. 您可以通过点 **Download pull secret** 或 **Copy pull secret** 下载或复制 pull secret。
7. 点 **Download command-line tools**，将 **openshift-install** 二进制文件放在 **PATH** 中的目录中。

13.4.3. 创建首选配置输入

使用这个流程创建用于创建 PXE 文件的首选配置输入。

流程

1. 运行以下命令来安装 **nmstate** 依赖项：

```
$ sudo dnf install /usr/bin/nmstatectl -y
```

2. 将 **openshift-install** 二进制文件放到 PATH 中的目录中。

3. 运行以下命令，创建一个目录来存储安装配置：

```
$ mkdir ~/<directory_name>
```

注意

这是基于代理的安装的首选方法。使用 GitOps ZTP 清单是可选的。

4. 创建 **install-config.yaml** 文件：

```
$ cat << EOF > ./my-cluster/install-config.yaml
apiVersion: v1
baseDomain: test.example.com
compute:
- architecture: amd64 (1)
  hyperthreading: Enabled
  name: worker
  replicas: 0
controlPlane:
  architecture: amd64
  hyperthreading: Enabled
  name: master
  replicas: 1
metadata:
  name: sno-cluster (2)
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 192.168.0.0/16
  networkType: OVNKubernetes (3)
  serviceNetwork:
  - 172.30.0.0/16
platform: (4)
  none: {}
pullSecret: '<pull_secret>' (5)
sshKey: '<ssh_pub_key>' (6)
EOF
```

(1) 指定系统架构。有效值为 **amd64**,**arm64**,**ppc64le**, 和 **s390x**。

如果使用带有 **multi** 有效负载的发行镜像，则可以在不同的架构上安装集群，如 **arm64**, **amd64**, **s390x**, 和 **ppc64le**。否则，您只能在 **openshift-install version** 命令的输出中所显示的发**行构架**上安装集群。如需更多信息，请参阅"验证安装基于代理的安装程序集群的受支持架构"。

2 必需。指定集群名称。

3 要安装的集群网络插件。默认值 **OVNKubernetes** 是唯一支持的值。

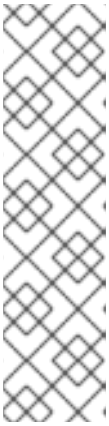
4 指定您的平台。

注意

对于裸机平台，默认使用 **install-config.yaml** 文件的 platform 部分中进行的主机设置，除非它们被 **agent-config.yaml** 文件中的配置覆盖。

5 指定 pull secret。

6 指定 SSH 公钥。



注意

如果将平台设置为 **vSphere** 或 **baremetal**，您可以使用三种方式为集群节点配置 IP 地址端点：

- IPv4
- IPv6
- IPv4 和 IPv6 并行 (dual-stack)

IPv6 仅在裸机平台上被支持。

双栈网络示例

```
networking:
  clusterNetwork:
    - cidr: 172.21.0.0/16
      hostPrefix: 23
    - cidr: fd02::/48
      hostPrefix: 64
  machineNetwork:
    - cidr: 192.168.11.0/16
    - cidr: 2001:DB8::/32
  serviceNetwork:
    - 172.22.0.0/16
    - fd03::/112
  networkType: OVNKubernetes
platform:
  baremetal:
    apiVIPs:
    - 192.168.11.3
    - 2001:DB8::4
```

```
ingressVIPs:
- 192.168.11.4
- 2001:DB8::5
```

注意

使用断开连接的镜像 registry 时，您必须将之前为镜像 registry 创建的证书文件添加到 **install-config.yaml** 文件的 **additionalTrustBundle** 字段中。

5. 创建 **agent-config.yaml** 文件：

```
$ cat > agent-config.yaml << EOF
apiVersion: v1beta1
kind: AgentConfig
metadata:
  name: sno-cluster
rendezvousIP: 192.168.111.80 (1)
hosts: (2)
  - hostname: master-0 (3)
    interfaces:
      - name: eno1
        macAddress: 00:ef:44:21:e6:a5
    rootDeviceHints: (4)
      deviceName: /dev/sdb
    networkConfig: (5)
      interfaces:
        - name: eno1
          type: ethernet
          state: up
          mac-address: 00:ef:44:21:e6:a5
          ipv4:
            enabled: true
            address:
              - ip: 192.168.111.80
                prefix-length: 23
            dhcp: false
      dns-resolver:
        config:
          server:
            - 192.168.111.1
      routes:
        config:
          - destination: 0.0.0.0/0
            next-hop-address: 192.168.111.2
            next-hop-interface: eno1
            table-id: 254
EOF
```

(1) 此 IP 地址用于确定哪些节点执行 bootstrap 过程，以及运行 **assisted-service** 组件。当您没有在 **networkConfig** 参数中指定至少一个主机的 IP 地址时，您必须提供 rendezvous IP 地址。如果没有提供此地址，则会从提供的主机的 **networkConfig** 中选择一个 IP 地址。

(2) 可选：主机配置。定义的主机数量不能超过 **install-config.yaml** 文件中定义的主机总数，这是 **compute.replicas** 和 **controlPlane.replicas** 参数的值的总和。

3 可选：覆盖从动态主机配置协议(DHCP)或反向 DNS 查找中获取的主机名。每个主机必须具有由这些方法提供的唯一主机名。

4 启用将 Red Hat Enterprise Linux CoreOS (RHCOS)镜像置备到特定设备。安装程序会按照发现设备的顺序检查设备，并将发现的值与 hint 值进行比较。它使用第一个与 hint 值匹配的发现设备。

5 可选：以 NMState 格式配置主机的网络接口。

6. 可选： 要创建 iPXE 脚本，请将 **bootArtifactsBaseURL** 添加到 **agent-config.yaml** 文件中：

```
apiVersion: v1beta1
kind: AgentConfig
metadata:
  name: sno-cluster
rendezvousIP: 192.168.111.80
bootArtifactsBaseURL: <asset_server_URL>
```

其中 **<asset_server_URL>** 是要将 PXE 资产上传到的服务器的 URL。

其他资源

- 使用双栈网络进行部署。
- 配置 install-config yaml 文件。
- 请参阅配置三节点集群以在裸机环境中部署三节点集群。
- 关于 root 设备提示.
- nmstate 状态示例.
- 可选：创建额外的清单文件

13.4.4. 创建 PXE 资产

使用以下步骤创建资产和可选脚本，以便在 PXE 基础架构中实现。

流程

1. 运行以下命令来创建 PXE 资产：

```
$ openshift-install agent create pxe-files
```

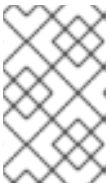
生成的 PXE 资产和可选 iPXE 脚本可在 **boot-artifacts** 目录中找到。

带有 PXE 资产和可选 iPXE 脚本的文件系统示例

```
boot-artifacts
    ├─ agent.x86_64-initrd.img
    ├─ agent.x86_64.ipxe
    ├─ agent.x86_64-rootfs.img
    └─ agent.x86_64-vmlinuz
```

重要

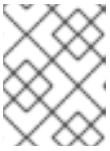
boot-artifacts 目录的内容因指定的架构而异。



注意

Red Hat Enterprise Linux CoreOS (RHCOS) 支持主磁盘上的多路径，允许对硬件故障进行更强大的弹性，以实现更高的主机可用性。在代理 ISO 镜像中默认启用多路径，默认 **/etc/multipath.conf** 配置。

2. 将 PXE 资产和可选脚本上传到您的基础架构，以便在引导过程中访问它们。



注意

如果您生成 iPXE 脚本，资产的位置必须与添加到 **agent-config.yaml** 文件中的 **bootArtifactsBaseURL** 匹配。

13.4.5. 手动添加 IBM Z 代理

创建 PXE 资产后，您可以添加 IBM Z® 代理。

根据您的 IBM Z® 环境，您可以从以下选项中选择：

- 使用 z/VM 添加 IBM Z® 代理
- 使用 RHEL KVM 添加 IBM Z® 代理

13.4.5.1. 使用 z/VM 添加 IBM Z 代理

使用以下步骤使用 z/VM 手动添加 IBM Z® 代理。

流程

1. 为 z/VM 客户机创建一个参数文件：

参数文件示例

```
rd.neednet=1 \
console=ttysclp0 \
coreos.live.rootfs_url=<rootfs_url> \ 1
ip=172.18.78.2::172.18.78.1:255.255.255.0:::none nameserver=172.18.78.1 \ 2
zfcp.allow_lun_scan=0 \ 3
rd.znet=qeth,0.0.bdd0,0.0.bdd1,0.0.bdd2,layer2=1 \
rd.dasd=0.0.4411 \ 4
rd.zfcp=0.0.8001,0x50050763040051e3,0x4000406300000000 \ 5
random.trust_cpu=on rd.luks.options=discard \
ignition.firstboot ignition.platform.id=metal \
console=tty1 console=ttyS1,115200n8 \
coreos.inst.persistent-kargs="console=tty1 console=ttyS1,115200n8"
```

1 对于 **coreos.live.rootfs_url** 工件，请为您要引导的**内核** 和 **initramfs** 指定匹配的 **rootfs** 工件。仅支持 HTTP 和 HTTPS 协议。

2 对于 **ip** 参数，使用 DHCP 自动分配 IP 地址，或者手动分配 IP 地址，如"在 IBM Z® 和 IBM® LinuxONE 上使用 z/VM 安装集群"中所述。

3 默认值为 **1**。使用 OSA 网络适配器时省略此条目。

4 对于在 DASD 类型磁盘中安装，请使用 **rd.dasd** 指定要安装 Red Hat Enterprise Linux CoreOS (RHCOS) 的 DASD。为 FCP 类型磁盘省略此条目。

5 对于在 FCP 类型磁盘中安装，请使用 **rd.zfcp=<adapter>,<wwpn>,<lun>** 指定要安装 RHCOS 的 FCP 磁盘。为 DASD 类型磁盘省略这个条目。

所有其他参数保持不变。

2. 将 **kernel.img**、**common.parm** 和 **initrd.img** 文件与 z/VM 客户机虚拟机的虚拟读取器中解放。如需更多信息，请参阅 IBM 文档中的 PUNCH。

提示

您可以使用 **CP PUNCH** 命令，或者使用 Linux (**vmur** 命令) 在两个 z/VM 虚拟机之间传输文件。

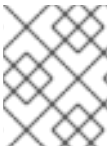
3. 登录到 bootstrap 机器上的对话监控系统 (CMS)。
4. 运行以下命令，从 reader IPL bootstrap 机器：

```
$ ipl c
```

如需更多信息，请参阅 IBM 文档中的 IPL。

13.4.5.2. 使用 RHEL KVM 添加 IBM Z (R)代理

使用以下步骤使用 RHEL KVM 手动添加 IBM Z® 代理。



注意

目前，IBM Z® (**s390x**) 上的 ISO 引导支持仅适用于 RHEL KVM，这为选择 PXE 或基于 ISO 的安装提供了灵活性。对于使用 z/VM 的安装，只支持 PXE 引导。

流程

1. 引导 RHEL KVM 机器。
2. 要部署虚拟服务器，请使用以下参数运行 **virt-install** 命令：

```
$ virt-install \
  --name <vm_name> \
  --autostart \
  --ram=16384 \
  --cpu host \
  --vcpus=8 \
  --location <path_to_kernel_initrd_image>,kernel=kernel.img,initrd=initrd.img \ 1
  --disk <qcow_image_path> \
  --network network:macvtap ,mac=<mac_address> \
  --graphics none \
  --noautoconsole \
```

```
--wait=-1 \
--extra-args "rd.neednet=1 nameserver=<nameserver>" \
--extra-args "ip=<IP>::<nameserver>::<hostname>:enc1:none" \
--extra-args "coreos.live.rootfs_url=http://<http_server>:8080/agent.s390x-rootfs.img" \
--extra-args "random.trust_cpu=on rd.luks.options=discard" \
--extra-args "ignition.firstboot ignition.platform.id=metal" \
--extra-args "console=tty1 console=ttyS1,115200n8" \
--extra-args "coreos.inst.persistent-kargs=console=tty1 console=ttyS1,115200n8" \
--osinfo detect=on,require=off
```

1 对于 **--location** 参数，指定 HTTP 或 HTTPS 服务器中的 kernel/initrd 的位置。

13.4.6. 其他资源

- 请参阅使用基于代理的安装程序安装 OpenShift Container Platform 集群，以了解更多有关基于代理的安装程序可用的配置。

13.5. 为 KUBERNETES OPERATOR 的多集群引擎准备基于 AGENT 的集群

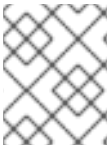
您可以安装多集群引擎 Operator，并使用基于 Agent 的 OpenShift Container Platform 安装程序部署 hub 集群。以下流程部分自动化，在部署初始集群后需要手动步骤。

13.5.1. 先决条件

- 您已阅读了以下文档：
  - 带有多集群引擎 operator 的集群生命周期概述。
  - 使用本地卷的持久性存储。
  - 使用 GitOps ZTP 在网络边缘置备集群。
  - 准备使用基于代理的安装程序进行安装。
  - 关于断开连接的安装镜像。
- 您可以访问互联网来获取所需的容器镜像。
- 已安装 OpenShift CLI(**oc**)。
- 如果要在断开连接的环境中安装，则必须为断开连接的安装镜像配置了本地镜像 registry。

13.5.2. 断开连接时为 Kubernetes Operator 准备基于代理的集群部署

您可以在断开连接的环境中将所需的 OpenShift Container Platform 容器镜像、多集群引擎 Operator 和 Local Storage Operator (LSO) 镜像 (LSO) 到断开连接的环境中的本地镜像 registry 中。确保您注意了镜像 registry 的本地 DNS 主机名和端口。



注意

要将 OpenShift Container Platform 镜像存储库镜像到您的镜像 registry，您可以使用 **oc adm release image** 或 **oc mirror** 命令。在此过程中，**oc mirror** 命令被用作示例。

流程

1. 创建一个 **<assets_directory>** 文件夹，使其包含有效的 **install-config.yaml** 和 **agent-config.yaml** 文件。此目录用于存储所有资产。

2. 要镜像 OpenShift Container Platform 镜像存储库、多集群引擎和 LSO，请使用以下设置创建一个 **ImageSetConfiguration.yaml** 文件：

   示例 ImageSetConfiguration.yaml

```
kind: ImageSetConfiguration
apiVersion: mirror.openshift.io/v1alpha2
archiveSize: 4 1
storageConfig: 2
  imageURL: <your-local-registry-dns-name>:<your-local-registry-port>/mirror/oc-mirror-
metadata 3
  skipTLS: true
mirror:
  platform:
    architectures:
      - "amd64"
    channels:
      - name: stable-4.16 4
        type: ocp
  additionalImages:
    - name: registry.redhat.io/ubi9/ubi:latest
  operators:
    - catalog: registry.redhat.io/redhat/redhat-operator-index:v4.16 5
      packages: 6
        - name: multicluster-engine 7
        - name: local-storage-operator 8
```

1. 指定镜像集合中每个文件的最大大小（以 GiB 为单位）。
2. 设置后端位置，以接收镜像设置元数据。此位置可以是 registry 或本地目录。必须指定 **storageConfig** 值。
3. 设置存储后端的 registry URL。
4. 设置包含您要安装的版本的 OpenShift Container Platform 镜像的频道。
5. 设置包含您要安装的 OpenShift Container Platform 镜像的 Operator 目录。
6. 仅指定要包含在镜像集中的特定 Operator 软件包和频道。删除此字段以检索目录中的所有软件包。
7. 多集群引擎软件包和频道。
8. LSO 软件包和频道。

注意

在镜像内容时，**oc mirror** 命令需要此文件。

3. 要镜像特定的 OpenShift Container Platform 镜像存储库、多集群引擎和 LSO，请运行以下命令：

```
$ oc mirror --dest-skip-tls --config ocp-mce-imageset.yaml docker://<your-local-registry-dns-name>:<your-local-registry-port>
```

4. 更新 **install-config.yaml** 文件中的 registry 和证书：

示例 imageContentSources.yaml

```
imageContentSources:
  - source: "quay.io/openshift-release-dev/ocp-release"
    mirrors:
      - "<your-local-registry-dns-name>:<your-local-registry-port>/openshift/release-images"
  - source: "quay.io/openshift-release-dev/ocp-v4.0-art-dev"
    mirrors:
      - "<your-local-registry-dns-name>:<your-local-registry-port>/openshift/release"
  - source: "registry.redhat.io/ubi9"
    mirrors:
      - "<your-local-registry-dns-name>:<your-local-registry-port>/ubi9"
  - source: "registry.redhat.io/multicluster-engine"
    mirrors:
      - "<your-local-registry-dns-name>:<your-local-registry-port>/multicluster-engine"
  - source: "registry.redhat.io/rhel8"
    mirrors:
      - "<your-local-registry-dns-name>:<your-local-registry-port>/rhel8"
  - source: "registry.redhat.io/redhat"
    mirrors:
      - "<your-local-registry-dns-name>:<your-local-registry-port>/redhat"
```

另外，请确保您的证书存在于 **install-config.yaml** 的 **additionalTrustBundle** 字段中。

示例 install-config.yaml

```
additionalTrustBundle: |
  -----BEGIN CERTIFICATE-----
  zzzzzzzzzzz
  -----END CERTIFICATE-------
```

重要

oc mirror 命令会创建一个名为 **oc-mirror-workspace** 的文件夹，其中包含几个输出。这包括 **imageContentSourcePolicy.yaml** 文件，用于标识 OpenShift Container Platform 和所选 Operator 所需的所有镜像。

5. 运行以下命令来生成集群清单：

```
$ openshift-install agent create cluster-manifests
```

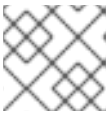
此命令更新集群 manifests 文件夹，使其包含包含您的镜像配置的 **mirror** 文件夹。

13.5.3. 连接时为 Kubernetes Operator 准备基于代理的集群部署

为多集群引擎 Operator、Local Storage Operator (LSO) 创建所需的清单，并将基于代理的 OpenShift Container Platform 集群部署为 hub 集群。

流程

1. 在 **<assets_directory>** 文件夹中创建一个名为 **openshift** 的子文件夹。此子文件夹用于存储在安装过程中应用的额外清单，以进一步自定义部署的集群。**<assets_directory>** 文件夹包含所有资产，包括 **install-config.yaml** 和 **agent-config.yaml** 文件。



注意

安装程序不会验证额外的清单。

2. 对于多集群引擎，创建以下清单并将其保存到 **<assets_directory>/openshift** 文件夹中：

示例 mce_namespace.yaml

```
apiVersion: v1
kind: Namespace
metadata:
  labels:
    openshift.io/cluster-monitoring: "true"
  name: multicluster-engine
```

示例 mce_operatorgroup.yaml

```
apiVersion: operators.coreos.com/v1
kind: OperatorGroup
metadata:
  name: multicluster-engine-operatorgroup
  namespace: multicluster-engine
spec:
  targetNamespaces:
  - multicluster-engine
```

示例 mce_subscription.yaml

```
apiVersion: operators.coreos.com/v1alpha1
kind: Subscription
metadata:
  name: multicluster-engine
  namespace: multicluster-engine
spec:
  channel: "stable-2.3"
  name: multicluster-engine
  source: redhat-operators
  sourceNamespace: openshift-marketplace
```

注意

您可以使用支持的安装程序 (AI) 使用 Red Hat Advanced Cluster Management (RHACM) 大规模安装分布式单元 (DU)。这些分布式单元必须在 hub 集群中启用。AI 服务需要手动创建的持久性卷 (PV)。

3. 对于 AI 服务，请创建以下清单并将其保存到 **<assets_directory>/openshift** 文件夹中：

示例 lso_namespace.yaml

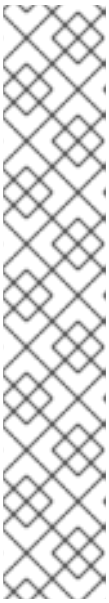
```
apiVersion: v1
kind: Namespace
metadata:
  annotations:
    openshift.io/cluster-monitoring: "true"
  name: openshift-local-storage
```

示例 lso_operatorgroup.yaml

```
apiVersion: operators.coreos.com/v1
kind: OperatorGroup
metadata:
  name: local-operator-group
  namespace: openshift-local-storage
spec:
  targetNamespaces:
    - openshift-local-storage
```

示例 lso_subscription.yaml

```
apiVersion: operators.coreos.com/v1alpha1
kind: Subscription
metadata:
  name: local-storage-operator
  namespace: openshift-local-storage
spec:
  installPlanApproval: Automatic
  name: local-storage-operator
  source: redhat-operators
  sourceNamespace: openshift-marketplace
```



注意

创建所有清单后，文件系统必须显示如下：

文件系统示例

```
<assets_directory>
    ├─ install-config.yaml
    ├─ agent-config.yaml
    └─ /openshift
       ├─ mce_namespace.yaml
       ├─ mce_operatorgroup.yaml
       ├─ mce_subscription.yaml
       ├─ lso_namespace.yaml
       ├─ lso_operatorgroup.yaml
       └─ lso_subscription.yaml
```

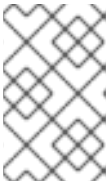
4. 运行以下命令来创建代理 ISO 镜像：

```
$ openshift-install agent create image --dir <assets_directory>
```

5. 镜像就绪后，引导目标机器并等待安装完成。

6. 要监控安装，请运行以下命令：

```
$ openshift-install agent wait-for install-complete --dir <assets_directory>
```



注意

要配置功能齐全的 hub 集群，必须创建以下清单，并通过运行 **$ oc apply -f <manifest-name>** 命令手动应用它们。清单创建的顺序非常重要，如果需要，会显示等待条件。

7. 对于 AI 服务所需的 PV，请创建以下清单：

```
apiVersion: local.storage.openshift.io/v1
kind: LocalVolume
metadata:
 name: assisted-service
 namespace: openshift-local-storage
spec:
 logLevel: Normal
 managementState: Managed
 storageClassDevices:
   - devicePaths:
       - /dev/vda
       - /dev/vdb
     storageClassName: assisted-service
     volumeMode: Filesystem
```

8. 在应用后续清单前，使用以下命令等待 PV 的可用性：

```
$ oc wait localvolume -n openshift-local-storage assisted-service --for condition=Available --
timeout 10m
```

注意

```
The `devicePath` is an example and may vary depending on the actual
hardware configuration used.
```

9. 为多集群引擎实例创建清单。

示例 MultiClusterEngine.yaml

```
apiVersion: multicluster.openshift.io/v1
kind: MultiClusterEngine
metadata:
  name: multiclusterengine
spec: {}
```

10. 创建清单以启用 AI 服务。

示例 agentserviceconfig.yaml

```
apiVersion: agent-install.openshift.io/v1beta1
kind: AgentServiceConfig
metadata:
  name: agent
  namespace: assisted-installer
spec:
 databaseStorage:
  storageClassName: assisted-service
  accessModes:
  - ReadWriteOnce
  resources:
   requests:
    storage: 10Gi
 filesystemStorage:
  storageClassName: assisted-service
  accessModes:
  - ReadWriteOnce
  resources:
   requests:
    storage: 10Gi
```

11. 创建清单来部署后续 spoke 集群。

示例 clusterimageset.yaml

```
apiVersion: hive.openshift.io/v1
kind: ClusterImageSet
metadata:
  name: "4.16"
spec:
  releaseImage: quay.io/openshift-release-dev/ocp-release:4.16.0-x86_64
```

12. 创建一个清单来导入安装代理（托管多集群引擎和 Assisted Service）作为 hub 集群。

示例 autoimport.yaml

```
apiVersion: cluster.open-cluster-management.io/v1
kind: ManagedCluster
metadata:
 labels:
   local-cluster: "true"
   cloud: auto-detect
   vendor: auto-detect
 name: local-cluster
spec:
 hubAcceptsClient: true
```

13. 等待受管集群创建。

```
$ oc wait -n multicluster-engine managedclusters local-cluster --for
condition=ManagedClusterJoined=True --timeout 10m
```

验证

- 要确认受管集群安装成功，请运行以下命令：

```
$ oc get managedcluster
NAME            HUB ACCEPTED   MANAGED CLUSTER URLS              JOINED   AVAILABLE
AGE
local-cluster   true           https://<your cluster url>:6443   True     True     77m
```

其他资源

- Local Storage Operator

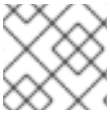
13.6. 基于代理的安装程序的安装配置参数

在使用基于代理的安装程序部署 OpenShift Container Platform 集群前，您可以提供参数来自定义集群以及托管它的平台。在创建 **install-config.yaml** 和 **agent-config.yaml** 文件时，必须为所需参数提供值，您可以使用可选参数进一步自定义集群。

13.6.1. 可用的安装配置参数

下表指定您可以在基于 Agent 的安装过程中设置的必要和可选安装配置参数。

这些值在 **install-config.yaml** 文件中指定。



注意

这些设置仅用于安装，无法在安装后修改。

13.6.1.1. 所需的配置参数

下表描述了所需的安装配置参数：

表 13.7. 所需的参数

参数	描述	值
apiVersion:	**install-config.yaml** 内容的 API 版本。当前版本为 **v1**。安装程序可能还支持旧的 API 版本。	字符串

参数	描述	值
baseDomain:	云供应商的基域。基域用于创建到 OpenShift Container Platform 集群组件的路由。集群的完整 DNS 名称是 **baseDomain** 和 **metadata.name** 参数值的组合，其格式为 **<metadata.name>.<baseDomain>**。	完全限定域名或子域名，如 **example.com**。
metadata:	Kubernetes 资源 **ObjectMeta**，其中只消耗 **name** 参数。	对象
metadata: name:	集群的名称。集群的 DNS 记录是 **{{.metadata.name}}.{{.baseDomain}}** 的子域。当您没有通过 **install-config.yaml** 或 **agent-config.yaml** 文件提供 **metadata.name** 时，例如只使用 ZTP 清单时，集群名称被设置为 **agent-cluster**。	小写字母、连字符(**-**)和句点(**.**)字符串，如 **dev**。
platform:	要执行安装的具体平台配置：**baremetal**, **external**, **none**, 或 **vsphere**。	对象

pullSecret:	从 Red Hat OpenShift Cluster Manager 获取 pull secret，验证从 Quay.io 等服务中下载 OpenShift Container Platform 组件的容器镜像。	{
		"auths":{
		"cloud.openshift.com":{
		"auth":"b3Blb=",
		"email":"you@example.com"
		},
		"quay.io":{
		"auth":"b3Blb=",
		"email":"you@example.com"
		}
		}
		}

13.6.1.2. 网络配置参数

您可以根据现有网络基础架构的要求自定义安装配置。例如，您可以扩展集群网络的 IP 地址块，或者提供不同于默认值的不同 IP 地址块。

- 如果使用 Red Hat OpenShift Networking OVN-Kubernetes 网络插件，则支持 IPv4 和 IPv6 地址系列。

如果将集群配置为使用两个 IP 地址系列，请查看以下要求：

- 两个 IP 系列都必须将相同的网络接口用于默认网关。
- 两个 IP 系列都必须具有默认网关。
- 您必须为所有网络配置参数指定 IPv4 和 IPv6 地址。例如，以下配置 IPv4 地址列在 IPv6 地址的前面。

```
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  - cidr: fd00:10:128::/56
    hostPrefix: 64
  serviceNetwork:
  - 172.30.0.0/16
  - fd00:172:16::/112
```

注意

Red Hat OpenShift Data Foundation 灾难恢复解决方案不支持 Globalnet。对于区域灾难恢复场景，请确保为每个集群中的集群和服务网络使用非重叠的专用 IP 地址。

表 13.8. 网络参数

参数	描述	值
`networking:`	集群网络的配置。	对象  **注意** 您无法在安装后修改 **网络** 对象指定的参数。
`networking:` `  networkType:`	要安装的 Red Hat OpenShift Networking 网络插件。	**OVNKubernetes**。**OVNKubernetes** 是 Linux 网络和包含 Linux 和 Windows 服务器的混合网络的 CNI 插件。默认值为 **OVNKubernetes**。

`networking:`
`  clusterNetwork:`
	pod 的 IP 地址块。
	
	默认值为 **10.128.0.0/14**，主机前缀为 **/23**。
	
	如果您指定了多个 IP 地址块，块不得重叠。
		对象数组。例如：
		
		`networking:`
		`  clusterNetwork:`
		`  - cidr: 10.128.0.0/14`
		`  hostPrefix: 23`
		`  - cidr: fd01::/48`
		`  hostPrefix: 64`

参数	描述	值
networking: clusterNetwork: cidr:	使用 **networking.clusterNetwork** 时需要此项。IP 地址块。 如果使用 OVN-Kubernetes 网络插件，您可以指定 IPv4 和 IPv6 网络。	使用 CIDR 形式的 IP 地址块。IPv4 块的前缀长度介于 **0** 到 **32** 之间。IPv6 块的前缀长度介于 **0** 到 **128** 之间。例如，**10.128.0.0/14** 或 **fd01::/48**。
networking: clusterNetwork: hostPrefix:	分配给每个节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从 given **cidr** 中分配 a/**23** 子网。**hostPrefix** 值 **23** 提供 510（2^(32 - 23)- 2）pod IP 地址。	子网前缀。 对于 IPv4 网络，默认值为 **23**。对于 IPv6 网络，默认值为 **64**。默认值也是 IPv6 的最小值。
networking: serviceNetwork:	服务的 IP 地址块。默认值为 **172.30.0.0/16**。 OVN-Kubernetes 网络插件只支持服务网络的一个 IP 地址块。 如果使用 OVN-Kubernetes 网络插件，您可以为 IPv4 和 IPv6 地址系列指定一个 IP 地址块。	CIDR 格式具有 IP 地址块的数组。例如： networking: serviceNetwork: - 172.30.0.0/16 - fd02::/112
networking: machineNetwork:	机器的 IP 地址块。 如果您指定了多个 IP 地址块，块不得重叠。	对象数组。例如： networking: machineNetwork: - cidr: 10.0.0.0/16
networking: machineNetwork: cidr:	使用 **networking.machineNetwork** 时需要此项。IP 地址块。对于 libvirt 和 IBM Power® Virtual Server 以外的所有平台，默认值为 **10.0.0.0/16**。对于 libvirt，默认值 为 **192.168.126.0/24**。对于 IBM Power® Virtual Server，默认值为 **192.168.0.0/24**。	CIDR 表示法中的 IP 网络块。 例如。**10.0.0.0/16** 或 **fd00::/48**。 **注意** 将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

13.6.1.3. 可选的配置参数

下表描述了可选的安装配置参数：

表 13.9. 可选参数

参数	描述	值
additionalTrustBundle:	添加到节点可信证书存储中的 PEM 编码 X.509 证书捆绑包。配置了代理时，也可以使用此信任捆绑包。	字符串
capabilities:	控制可选核心组件的安装。您可以通过禁用可选组件来减少 OpenShift Container Platform 集群的空间。如需更多信息，请参阅*安装*中的"集群功能"页面。	字符串数组
capabilities: baselineCapabilitySet:	选择要启用的一组初始可选功能。有效值为 **None**、**v4.11**、**v4.12** 和 **vCurrent**。默认值为 **vCurrent**。	字符串
capabilities: additionalEnabledCapabilities:	将可选功能集合扩展到您在 **baselineCapabilitySet** 中指定的范围。您可以在此参数中指定多个功能。	字符串数组
cpuPartitioningMode:	启用工作负载分区，它会隔离 OpenShift Container Platform 服务、集群管理工作负载和基础架构 pod，以便在保留的一组 CPU 上运行。工作负载分区只能在安装过程中启用，且在安装后无法禁用。虽然此字段启用工作负载分区，但它不会将工作负载配置为使用特定的 CPU。如需更多信息，请参阅 *Scalability and Performance* 部分中的 *Workload partitioning* 页面。	**None** 或 **AllNodes**.**None** 是默认值。
compute:	组成计算节点的机器的配置。	**MachinePool** 对象的数组。
compute: architecture:	决定池中机器的指令集合架构。目前，不支持具有不同架构的集群。所有池都必须指定相同的架构。有效值为 **amd64**,**arm64**,**ppc64le**, 和 **s390x**。	字符串

参数	描述	值
compute: hyperthreading:	是否在计算机器上启用或禁用并发多**线程或超线程**。默认情况下，启用并发多线程以提高机器内核的性能。 **重要** 如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。	**enabled** 或 **Disabled**
compute: name:	使用 **compute** 时需要此项。机器池的名称。	**worker**
compute: platform:	使用 **compute** 时需要此项。使用此参数指定托管 worker 机器的云供应商。此参数值必须与 **controlPlane.platform** 参数值匹配。	**baremetal**、**vsphere** 或 **{}**
compute: replicas:	要置备的计算机器数量，也称为 worker 机器。	大于或等于 **2** 的正整数。默认值为 **3**。
featureSet:	为功能集启用集群。功能集是 OpenShift Container Platform 功能的集合，默认情况下不启用。有关在安装过程中启用功能集的更多信息，请参阅"使用功能门启用功能"。	字符串.要启用的功能集的名称，如 **TechPreviewNoUpgrade**。
controlPlane:	组成 control plane 的机器的配置。	**MachinePool** 对象的数组。
controlPlane: architecture:	决定池中机器的指令集合架构。目前，不支持具有不同架构的集群。所有池都必须指定相同的架构。有效值为 **amd64**,**arm64**,**ppc64le**, 和 **s390x**。	字符串

参数	描述	值
controlPlane: hyperthreading:	是否在 control plane 机器上启用或禁用并发多**线程或超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。	**enabled** 或 **Disabled**
controlPlane: name:	使用 **controlPlane** 时需要此项。机器池的名称。	**master**
controlPlane: platform:	使用 **controlPlane** 时需要此项。使用此参数指定托管 control plane 机器的云供应商。此参数值必须与 **compute.platform** 参数值匹配。	**baremetal**、**vsphere** 或 **{}**
controlPlane: replicas:	要置备的 control plane 机器数量。	部署单节点 OpenShift 时支持的值为 **3** 或 **1**。
credentialsMode:	Cloud Credential Operator(CCO)模式。如果没有指定模式，CCO 会动态尝试决定提供的凭证的功能，在支持多个模式的平台上首选 mint 模式。	**Mint**、**Passthrough**、**Manual** 或空字符串(**""**)。[1]

参数	描述	值
fips:	启用或禁用 FIPS 模式。默认值为 **false**（禁用）。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。  **重要** 要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 Red Hat Enterprise Linux (RHEL) 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅在 FIPS 模式中安装该系统。 当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS) 时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。  **注意** 如果使用 Azure File 存储，则无法启用 FIPS 模式。	**false** 或 **true**
imageContentSources:	release-image 内容的源和存储库。	对象数组。包括一个 **source** 以及可选的 **mirrors**，如本表的以下行所述。
imageContentSources: source:	使用 **imageContentSources** 时需要此项。指定用户在镜像拉取规格中引用的存储库。	字符串

参数	描述	值
imageContentSources: mirrors:	指定可能还包含同一镜像的一个或多个仓库。	字符串数组
publish:	如何发布或公开集群的面向用户的端点，如 Kubernetes API、OpenShift 路由。	**内部或外部 .**默认值为 **External**。 在非云平台上不支持将此字段设置为 **Internal**。 **重要** 如果将字段的值设为 **Internal**，集群将无法运行。如需更多信息，请参阅 BZ#1953035。
sshKey:	用于验证对集群机器的访问的 SSH 密钥。 **注意** 对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。	例如，**sshKey: ssh-ed25519 AAAA..**。

1. 不是所有 CCO 模式都支持所有云供应商。有关 CCO 模式的更多信息，请参阅*身份验证和授权*内容中的"管理云供应商凭证"条目。

13.6.1.4. 基于代理的安装程序的其他裸机配置参数

下表描述了基于代理的安装程序的其他裸机安装配置参数：

注意

这些字段不会在集群的初始置备过程中使用，但它们可在安装集群后使用。在安装时配置这些字段，无需将它们设置为第 2 天操作。

表 13.10. 其他裸机参数

参数	描述	值
platform: baremetal: clusterProvisioningIP:	运行置备服务的集群中的 IP 地址。默认为 provisioning 子网的第三个 IP 地址。例如，**172.22.0.3** 或 **2620:52:0:1307::3**。	IPv4 或 IPv6 地址。
platform: baremetal: provisioningNetwork:	**provisioningNetwork** 配置设置决定了集群是否使用 provisioning 网络。如果存在，配置设置也会决定集群是否管理网络。 **Managed**: 默认。将此参数设置为 **Managed** 以完全管理置备网络，包括 DHCP、TFTP 等等。 **disabled** ：将此参数设置为 **Disabled**，以禁用对 provisioning 网络的要求。当设置为 **Disabled** 时，您只能根据第 2 天使用基于虚拟介质的置备。如果 **Disabled** 并使用电源管理，则必须可以从 bare-metal 网络访问 BMC。如果为 Disabled，则必须在裸机网络上提供两个用于置备服务的 IP 地址。	**Managed** 或 **Disabled**。
platform: baremetal: provisioningMACAddress:	运行置备服务的集群中的 MAC 地址。	MAC 地址。
platform: baremetal: provisioningNetworkCIDR:	用于置备的网络的 CIDR。在 provisioning 网络中不使用默认地址范围时需要这个选项。	有效的 CIDR，如 **10.0.0.0/16**。
platform: baremetal: provisioningNetworkInterface:	连接到 provisioning 网络的节点上的网络接口名称。使用 **bootMACAddress** 配置设置来启用 Ironic 标识 NIC 的 IP 地址，而不是使用 **provisioningNetworkInterface** 配置设置来标识 NIC 的名称。	字符串.

参数	描述	值
platform: baremetal: provisioningDHCPRange:	定义 provisioning 网络上节点的 IP 范围，如 **172.22.0.10,172.22.0.254**。	IP 地址范围。
platform: baremetal: hosts:	配置裸机主机。	主机配置对象的数组。
platform: baremetal: hosts: name:	主机的名称。	字符串.
platform: baremetal: hosts: bootMACAddress:	用于置备主机的 NIC 的 MAC 地址。	MAC 地址。
platform: baremetal: hosts: bmc:	主机的配置，以连接到基板管理控制器 (BMC)。	BMC 配置对象字典。
platform: baremetal: hosts: bmc: username:	BMC 的用户名。	字符串.
platform: baremetal: hosts: bmc: password:	BMC 的密码。	字符串.

参数	描述	值
platform: baremetal: hosts: bmc: address:	与主机的 BMC 控制器通信的 URL。地址配置设置指定协议。例如，**redfish+http://10.10.10.1:8000/redfish/v1/Systems/1234** 启用 Redfish。如需更多信息，请参阅裸机上的"部署安装程序置备的集群中的"BMC 寻址"。	URL。
platform: baremetal: hosts: bmc: disableCertificateVerification:	**RedFish** 和 **redfish-virtualmedia** 需要这个参数来管理 BMC 地址。当 BMC 地址使用自签名证书时，这个值应该是 **True**。	布尔值.

13.6.1.5. 其他 VMware vSphere 配置参数

下表描述了其他 VMware vSphere 配置参数：

表 13.11. 其他 VMware vSphere 集群参数

参数	描述	值
platform: vsphere:	描述托管集群的云平台中的帐户。您可以使用参数来自定义平台。如果您为机器池中的 compute 和 control plane 机器提供额外的配置设置，则不需要该参数。您只能为 OpenShift Container Platform 集群指定一个 vCenter 服务器。	vSphere 配置对象的字典
platform: vsphere: failureDomains:	建立地区和区域之间的关系。您可以使用 vCenter 对象（如 **datastore** 对象）定义故障域。故障域定义 OpenShift Container Platform 集群节点的 vCenter 位置。	故障域配置对象的数组。
platform: vsphere: failureDomains: name:	故障域的名称。	字符串

参数	描述	值
platform: vsphere: failureDomains: region:	如果为集群定义多个故障域，则必须将标签附加到每个 vCenter 数据中心。要定义区域，请使用 **openshift-region** 标签类别中的标签。对于单个 vSphere 数据中心环境，您不需要附加标签，但必须为参数输入一个字母数字值，如 **datacenter**。	字符串
platform: vsphere: failureDomains: server:	指定 VMware vCenter 服务器的完全限定主机名或 IP 地址，以便客户端可以访问故障域资源。您必须将 **server** 器角色应用到 vSphere vCenter 服务器位置。	字符串
platform: vsphere: failureDomains: zone:	如果为集群定义多个故障域，则必须为每个 vCenter 集群附加标签。要定义一个区，请使用 **openshift-zone** 标签类别中的标签。对于单个 vSphere 数据中心环境，您不需要附加标签，但必须为参数输入一个字母数字值，如 **cluster**。	字符串
platform: vsphere: failureDomains: topology: computeCluster:	vSphere 计算集群的路径。	字符串
platform: vsphere: failureDomains: topology: datacenter:	列出并定义 OpenShift Container Platform 虚拟机 (VM) 操作的数据中心。数据中心列表必须与 **vcenters** 字段中指定的数据中心列表匹配。	字符串

参数	描述	值
platform: vsphere: failureDomains: topology: datastore:	保存虚拟机文件、模板和 ISO 镜像的 vSphere 数据存储路径。  **重要** 您可以指定数据存储集群中存在的任何数据存储路径。默认情况下，Storage vMotion 会自动为数据存储集群启用。红帽不支持 Storage vMotion，因此您必须禁用 Storage vMotion 以避免 OpenShift Container Platform 集群的数据丢失问题。 如果需要在多个数据存储间指定虚拟机，请使用 **数据存储** 对象在集群 **install-config.yaml** 配置文件中指定故障域。如需更多信息，请参阅"VMware vSphere 区域和区启用"。	字符串
platform: vsphere: failureDomains: topology: folder:	可选：用户创建虚拟机的现有文件夹的绝对路径，例如 **/<datacenter_name>/vm/<folder_name>/<subfolder_name>**。	字符串
platform: vsphere: failureDomains: topology: networks:	列出 vCenter 实例中包含您配置的虚拟 IP 地址和 DNS 记录的任何网络。	字符串
platform: vsphere: failureDomains: topology: resourcePool:	可选：安装程序创建虚拟机的现有资源池的绝对路径，例如 **/<datacenter_name>/host/<cluster_name>/Resources/<resource_pool_name>/<optional_nested_resource_pool_name>**。	字符串
platform: vsphere: failureDomains: topology template:	指定预先存在的 Red Hat Enterprise Linux CoreOS (RHCOS) 镜像模板或虚拟机的绝对路径。安装程序可以使用镜像模板或虚拟机在 vSphere 主机上快速安装 RHCOS。考虑使用此参数作为在 vSphere 主机上上传 RHCOS 镜像的替代选择。此参数仅适用于安装程序置备的基础架构。	字符串

参数	描述	值
platform: vsphere: vcenters:	配置连接详情，以便服务可以与 vCenter 服务器通信。目前，只支持单个 vCenter 服务器。	vCenter 配置对象的数组。
platform: vsphere: vcenters: datacenters:	列出并定义 OpenShift Container Platform 虚拟机 (VM) 操作的数据中心。数据中心列表必须与 **failureDomains** 字段中指定的数据中心列表匹配。	字符串
platform: vsphere: vcenters: password:	与 vSphere 用户关联的密码。	字符串
platform: vsphere: vcenters: port:	用于与 vCenter 服务器通信的端口号。	整数
platform: vsphere: vcenters: server:	vCenter 服务器的完全限定主机名(FQHN)或 IP 地址。	字符串
platform: vsphere: vcenters: user:	与 vSphere 用户关联的用户名。	字符串

13.6.1.6. 弃用的 VMware vSphere 配置参数

在 OpenShift Container Platform 4.13 中，以下 vSphere 配置参数已弃用。您可以继续使用这些参数，但安装程序不会在 **install-config.yaml** 文件中自动指定这些参数。

下表列出了每个已弃用的 vSphere 配置参数：

表 13.12. 弃用的 VMware vSphere 集群参数

参数	描述	值
platform: vsphere: cluster:	安装 OpenShift Container Platform 集群的 vCenter 集群。	字符串
platform: vsphere: datacenter:	定义 OpenShift Container Platform 虚拟机 (VM) 操作的数据中心。	字符串
platform: vsphere: defaultDatastore:	用于调配卷的默认数据存储名称。	字符串
platform: vsphere: folder:	可选：安装程序创建虚拟机的现有文件夹的绝对路径。如果没有提供这个值，安装程序会创建一个文件夹，它的名称为 datacenter 虚拟机文件夹中的基础架构 ID。	字符串，如 **/<datacenter_name>/vm/<folder_name>/<subfolder_name>**。
platform: vsphere: password:	vCenter 用户名的密码。	字符串
platform: vsphere: resourcePool:	可选：安装程序创建虚拟机的现有资源池的绝对路径。如果没有指定值，安装程序会在 **/<datacenter_name>/host/<cluster_name>/Resources** 下的集群的 root 中安装资源。	字符串，例如 **/<datacenter_name>/host/<cluster_name>/Resources/<resource_pool_name>/<optional_nested_resource_pool_name>**。
platform: vsphere: username:	用于连接 vCenter 实例的用户名。此用户必须至少具有 vSphere 中 静态或动态持久性卷置备 所需的角色和权限。	字符串
platform: vsphere: vCenter:	vCenter 服务器的完全限定主机名或 IP 地址。	字符串

其他资源

- BMC 地址
- 为 VMware vCenter 配置区域和区域

13.6.2. 可用的代理配置配置参数

下表指定您可以在基于 Agent 的安装过程中设置的必要和可选代理配置参数。

这些值在 **agent-config.yaml** 文件中指定。

注意

这些设置仅用于安装，无法在安装后修改。

13.6.2.1. 所需的配置参数

下表描述了所需的代理配置参数：

表 13.13. 所需的参数

参数	描述	值
apiVersion:	**agent-config.yaml** 内容的 API 版本。当前版本是 **v1beta1**。安装程序可能还支持旧的 API 版本。	字符串
metadata:	Kubernetes 资源 **ObjectMeta**，其中只消耗 **name** 参数。	对象
metadata: name:	集群的名称。集群的 DNS 记录是 **{{.metadata.name}}.{{.baseDomain}}** 的子域。在 **agent-config.yaml** 文件中输入的值将被忽略，而是使用 **install-config.yaml** 文件中指定的值。当您没有通过 **install-config.yaml** 或 **agent-config.yaml** 文件提供 **metadata.name** 时，例如只使用 ZTP 清单时，集群名称被设置为 **agent-cluster**。	小写字母和连字符 (**-**) 的字符串，如 **dev**。

13.6.2.2. 可选的配置参数

下表描述了可选的 Agent 配置参数：

表 13.14. 可选参数

参数	描述	值
rendezvousIP:	执行 bootstrap 进程的节点的 IP 地址，以及运行 **assisted-service** 组件。当您没有在 **networkConfig** 参数中指定至少一个主机的 IP 地址时，您必须提供 rendezvous IP 地址。如果没有提供此地址，则会从提供的主机的 **networkConfig** 中选择一个 IP 地址。	IPv4 或 IPv6 地址。
bootArtifactsBaseURL:	使用基于代理的安装程序生成 iPXE 脚本时，将 Preboot Execution Environment (PXE) 资产上传到的服务器的 URL。如需更多信息，请参阅"为 OpenShift Container Platform 准备 PXE 资产"。	字符串.
additionalNTPSources:	要添加到所有集群主机的网络时间协议 (NTP) 源列表，这些源被添加到通过其他方法配置的任何 NTP 源中。	主机名或 IP 地址列表。
hosts:	主机配置。可选的主机列表。定义的主机数量不能超过 **install-config.yaml** 文件中定义的主机总数，这是 **compute.replicas** 和 **controlPlane.replicas** 参数的值的总和。	主机配置对象的数组。
hosts: hostname:	主机名.覆盖从动态主机配置协议(DHCP)或反向 DNS 查找中获取的主机名。每个主机必须具有由这些方法提供的唯一主机名，尽管通过此参数配置主机名是可选的。	字符串.
hosts: interfaces:	为主机上的接口提供名称和 MAC 地址映射表。如果在 **agent-config.yaml** 文件中提供了 **NetworkConfig** 部分，则必须包含此表，值必须与 **NetworkConfig** 部分中提供的映射匹配。	主机配置对象的数组。
hosts: interfaces: name:	主机上接口名称。	字符串.
hosts: interfaces: macAddress:	主机上接口的 MAC 地址。	一个 MAC 地址，如以下示例：**00-B0-D0-63-C2-26**。

参数	描述	值
hosts: role:	定义主机是 **master** 节点还是 **worker** 节点。如果在 **agent-config.yaml** 文件中没有定义角色，则会在集群安装过程中随机分配角色。	**master** 或 **worker**。
hosts: rootDeviceHints:	启用将 Red Hat Enterprise Linux CoreOS (RHCOS)镜像置备到特定设备。安装程序会按照发现设备的顺序检查设备，并将发现的值与 hint 值进行比较。它使用第一个与 hint 值匹配的发现设备。这是操作系统在安装过程中写入的设备。	键值对字典。如需更多信息，请参阅"为 OpenShift 安装设置环境"页面中的"Root 设备提示"。
hosts: rootDeviceHints: deviceName:	RHCOS 镜像置备为的设备的名称。	字符串.
hosts: networkConfig:	主机网络定义。配置必须与 nmstate 文档中定义的 Host Network Management API 匹配。	主机网络配置对象的字典。

其他资源

- 为 OpenShift Container Platform 准备 PXE 资产
- Root 设备提示

第 14 章 在单一节点上安装

14.1. 准备在一个节点上安装

14.1.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您已阅读了有关 选择集群安装方法并为用户准备它 的文档。

14.1.2. 关于单个节点上的 OpenShift

您可以使用标准安装方法创建单节点集群。单一节点上的 OpenShift Container Platform 是一个特殊的安装，需要创建特殊的 Ignition 配置文件。它的主要用例是用于边缘计算工作负载，包括相邻连接、便携式云和靠近基站的 5G 无线访问网络(RAN)。在单一节点上安装时的主要权衡在于缺乏高可用性。

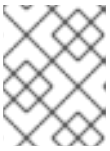
重要

不支持将 OpenShiftSDN 与单节点 OpenShift 搭配使用。OVN-Kubernetes 是单节点 OpenShift 部署的默认网络插件。

14.1.3. 在单一节点上安装 OpenShift 的要求

在单一节点上安装 OpenShift Container Platform 可降低高可用性和大型集群的一些要求。但是，您必须满足以下要求：

- **管理主机：** 您必须有一台计算机来准备 ISO、创建 USB 引导驱动器以及监控安装。

  注意

  对于 **ppc64le** 平台，主机应准备 ISO，但不需要创建 USB 引导驱动器。ISO 可以直接挂载到 PowerVM。

  注意

  IBM Z® 安装不需要 ISO。

- **CPU 架构：** 在单一节点上安装 OpenShift Container Platform 支持 **x86_64**、**arm64**、**ppc64le**、**s390x** CPU 架构。
- **支持的平台：** 在裸机、vSphere、AWS cloud、Red Hat OpenStack、OpenShift Virtualization、IBM Power® 和 IBM Z® 平台上支持在单一节点上安装 OpenShift Container Platform。
- production **-grade server：** 在单一节点上安装 OpenShift Container Platform 需要具有足够资源的服务器来运行 OpenShift Container Platform 服务和生产工作负载。

表 14.1. 最低资源要求

profile	vCPU	memory	Storage
最小值	8 个 vCPU 内核	16 GB RAM	120 GB

注意

- 当未启用并发多线程(SMT)或超线程时，一个 vCPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比率：
  （每个内核的线程数 x 内核数）x 插槽数 = vCPU
- 在安装过程中添加 Operator 可能会增加最低资源要求。

使用虚拟介质引导时，服务器必须具有基板管理控制器(BMC)。

注意

IBM Z® 和 IBM Power® 不支持 BMC。

- **网络：** 如果服务器没有连接到可路由的网络，则服务器必须可以访问互联网或访问本地注册表。服务器必须具有 Kubernetes API、Ingress 路由和集群节点域名的 DHCP 保留或静态 IP 地址。您必须配置 DNS，以将 IP 地址解析为以下完全限定域名(FQDN)：

表 14.2. 所需的 DNS 记录

用法	FQDN	描述
Kubernetes API	**api.<cluster_name>.<base_domain>**	添加 DNS A/AAAA 或 CNAME 记录。此记录必须由集群外的客户端和集群内解析。
内部 API	**api-int.<cluster_name>.<base_domain>**	在手动创建 ISO 时，添加 DNS A/AAAA 或 CNAME 记录。此记录必须由集群内的节点解析。
Ingress 路由	***.apps.<cluster_name>.<base_domain>**	添加以节点为目标的通配符 DNS A/AAAA 或 CNAME 记录。此记录必须由集群外的客户端和集群内解析。

重要

如果没有持久的 IP 地址，**apiserver** 和 **etcd** 之间的通信可能会失败。

14.2. 在单一节点上安装 OPENSHIFT

您可以使用基于 Web 的辅助安装程序和使用 Assisted Installer 生成的发现 ISO 安装单节点 OpenShift。您还可以使用 **coreos-installer** 安装单节点 OpenShift 来生成安装 ISO。

14.2.1. 使用 Assisted Installer 安装单节点 OpenShift

要在单一节点上安装 OpenShift Container Platform，请使用基于 web 的 Assisted Installer 向导来引导您完成安装过程并管理安装。

如需详细信息和配置选项，请参阅 OpenShift Container Platform 的辅助安装程序 文档。

14.2.1.1. 使用 Assisted Installer 生成发现 ISO

在单一节点上安装 OpenShift Container Platform 需要发现 ISO，辅助安装程序可生成。

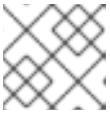
流程

1. 在管理主机上，打开浏览器并进入到 Red Hat OpenShift Cluster Manager 。
2. 单击 **Create Cluster** 以创建新集群。
3. 在 **Cluster name** 字段中输入集群名称。
4. 在 **Base domain** 字段中，输入基域。例如：

   ```
   example.com
   ```

   所有 DNS 记录都必须是这个基域的子域并包含集群名称，例如：

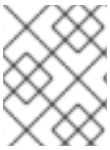
   ```
   <cluster-name>.example.com
   ```

   注意

   您不能在集群安装后更改基域或集群名称。

5. 选择 **Install single node OpenShift (SNO)** 并完成向导步骤的其余部分。下载发现 ISO。
6. 记录用于使用虚拟介质安装的发现 ISO URL。



注意

如果在此过程中启用 OpenShift Virtualization，则必须为虚拟机有第二个至少 50GiB 的本地存储设备。

其他资源

- 使用逻辑卷管理器存储的持久性存储
- OpenShift Virtualization 的作用

14.2.1.2. 使用 Assisted Installer 安装单节点 OpenShift

使用辅助安装程序安装单节点集群。

流程

1. 将 RHCOS 发现 ISO 附加到目标主机。
2. 在服务器 BIOS 设置中配置启动驱动器顺序，以便从附加的发现 ISO 启动，然后重启服务器。
3. 在管理主机上，返回到浏览器。等待主机出现在已发现的主机列表中。如有必要，重新载入 **Assisted Clusters** 页面并选择集群名称。
4. 完成安装向导步骤。添加网络详情，包括可用子网中的子网。如果需要，添加 SSH 公钥。

5. 监控安装的进度。观察集群出现的事件。在安装过程完成后，将操作系统镜像写入服务器的硬盘中，服务器会重启。

6. 删除发现 ISO，并重置服务器以从安装驱动器引导。
服务器自动重启了多次，部署 control plane。

其他资源

- 在 USB 驱动器中创建可引导 ISO 镜像
- 使用 Redfish API 从 HTTP 托管 ISO 镜像引导
- 将 worker 节点添加到单节点 OpenShift 集群

14.2.2. 手动安装单节点 OpenShift

要在单一节点上安装 OpenShift Container Platform，首先生成安装 ISO，然后从 ISO 引导服务器。您可以使用 **openshift-install** 安装程序监控安装。

其他资源

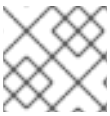
- 用户置备的基础架构对网络的要求
- 用户置备的 DNS 要求
- 配置 DHCP 或静态 IP 地址

14.2.2.1. 使用 coreos-installer 生成安装 ISO

在单一节点上安装 OpenShift Container Platform 需要安装 ISO，您可以按照以下流程生成该 ISO。

先决条件

- 安装 **podman**。



注意

如需了解网络要求，请参阅"在单一节点上安装 OpenShift 的要求，包括 DNS 记录。

流程

1. 设置 OpenShift Container Platform 版本：

```
$ OCP_VERSION=<ocp_version> 1
```

1 将 **<ocp_version>** 替换为当前版本，如 **latest-4.16**

2. 设置主机架构：

```
$ ARCH=<architecture> 1
```

1 将 **<architecture>** 替换为目标主机架构，如 **aarch64** 或 **x86_64**。

3. 输入以下命令下载 OpenShift Container Platform 客户端(**oc**)并使其可用：

```
$ curl -k https://mirror.openshift.com/pub/openshift-v4/clients/ocp/$OCP_VERSION/openshift-client-linux.tar.gz -o oc.tar.gz
```

```
$ tar zxf oc.tar.gz
```

```
$ chmod +x oc
```

4. 输入以下命令下载 OpenShift Container Platform 安装程序并使其可用：

```
$ curl -k https://mirror.openshift.com/pub/openshift-v4/clients/ocp/$OCP_VERSION/openshift-install-linux.tar.gz -o openshift-install-linux.tar.gz
```

```
$ tar zxvf openshift-install-linux.tar.gz
```

```
$ chmod +x openshift-install
```

5. 运行以下命令来检索 RHCOS ISO URL：

```
$ ISO_URL=$(./openshift-install coreos print-stream-json | grep location | grep $ARCH | grep iso | cut -d\" -f4)
```

6. 下载 RHCOS ISO：

```
$ curl -L $ISO_URL -o rhcos-live.iso
```

7. 准备 **install-config.yaml** 文件：

```
apiVersion: v1
baseDomain: <domain> (1)
compute:
- name: worker
  replicas: 0 (2)
controlPlane:
  name: master
  replicas: 1 (3)
metadata:
  name: <name> (4)
networking: (5)
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16 (6)
  networkType: OVNKubernetes
  serviceNetwork:
  - 172.30.0.0/16
platform:
  none: {}
bootstrapInPlace:
```

```
  installationDisk: /dev/disk/by-id/<disk_id> 7
pullSecret: '<pull_secret>' 8
sshKey: |
  <ssh_key> 9
```

1. 添加集群域名。
2. 将 **计算** 副本设置为 **0**。这使得 control plane 节点可以调度。
3. 将 **controlPlane** 副本设置为 **1**。与前面的 **compute** 设置结合使用，此设置可确保集群在单一节点上运行。
4. 将 **metadata** 名称设置为集群名称。
5. 设置**网络**详情。OVN-Kubernetes 是单节点集群唯一支持的网络插件类型。
6. 将 **cidr** 值设置为与单节点 OpenShift 集群的子网匹配。
7. 设置安装磁盘驱动器的路径，例如： **/dev/disk/by-id/wwn-0x64cd98f04fde100024684cf3034da5c2**。
8. 从 Red Hat OpenShift Cluster Manager 复制 pull secret ，并将内容添加到此配置设置中。
9. 从管理主机添加公共 SSH 密钥，以便您可以在安装后登录集群。

8. 运行以下命令来生成 OpenShift Container Platform 资产：

```
$ mkdir ocp
```

```
$ cp install-config.yaml ocp
```

```
$ ./openshift-install --dir=ocp create single-node-ignition-config
```

9. 运行以下命令，将 ignition 数据嵌入到 RHCOS ISO 中：

```
$ alias coreos-installer='podman run --privileged --pull always --rm \
      -v /dev:/dev -v /run/udev:/run/udev -v $PWD:/data \
      -w /data quay.io/coreos/coreos-installer:release'
```

```
$ coreos-installer iso ignition embed -fi ocp/bootstrap-in-place-for-live-iso.ign rhcos-live.iso
```

其他资源

- 如需有关在单一节点上安装 OpenShift Container Platform 的更多信息，请参阅在单一节点上安装 OpenShift 的要求。
- 有关启用 在安装前禁用的集群功能的更多信息，请参阅启用集群功能。
- 如需有关每个功能提供的功能的更多信息，请参阅 OpenShift Container Platform 4.16 中的可选集群功能。

14.2.2.2. 使用 openshift-install 监控集群安装

使用 **openshift-install** 监控单节点集群安装的进度。

流程

1. 将修改后的 RHCOS 安装 ISO 附加到目标主机。
2. 在服务器 BIOS 设置中配置启动驱动器顺序，以便从附加的发现 ISO 启动，然后重启服务器。
3. 在管理主机上，运行以下命令来监控安装：

```
$ ./openshift-install --dir=ocp wait-for install-complete
```

在部署 control plane 时服务器重启几次。

验证

- 安装完成后，运行以下命令来检查环境：

```
$ export KUBECONFIG=ocp/auth/kubeconfig
```

```
$ oc get nodes
```

输出示例

```
NAME                        STATUS   ROLES           AGE    VERSION
control-plane.example.com   Ready    master,worker   10m    v1.29.4
```

其他资源

- 在 USB 驱动器中创建可引导 ISO 镜像
- 使用 Redfish API 从 HTTP 托管 ISO 镜像引导
- 将 worker 节点添加到单节点 OpenShift 集群

14.2.3. 在云供应商上安装单节点 OpenShift

14.2.3.1. 在云供应商上安装单节点 OpenShift 的额外要求

在云供应商上安装程序置备安装的文档基于由三个 control plane 节点组成的高可用性集群。在引用文档时，请考虑单节点 OpenShift 集群要求与高可用性集群之间的区别。

- 高可用性集群需要一个临时 bootstrap 机器、三台 control plane 机器和至少两台计算机器。对于单节点 OpenShift 集群，您只需要一个临时 bootstrap 机器，另一个用于 control plane 节点，且没有 worker 节点。
- 高可用性集群安装的最低资源要求包括有 4 个 vCPU 和 100GB 存储的 control plane 节点。对于单节点 OpenShift 集群，必须至少有 8 个 vCPU 内核和 120GB 存储。
- **install-config.yaml** 文件中的 **controlPlane.replicas** 设置应设置为 **1**。
- **install-config.yaml** 文件中的 **compute.replicas** 设置应设置为 **0**。这使得 control plane 节点可以调度。

14.2.3.2. 用于单节点 OpenShift 支持的云供应商

下表包含支持的云供应商和 CPU 架构列表。

表 14.3. 支持的云供应商

云供应商	CPU 架构
Amazon Web Service (AWS)	x86_64 和 AArch64
Microsoft Azure	x86_64
Google Cloud Platform (GCP)	x86_64 和 AArch64

14.2.3.3. 在 AWS 上安装单节点 OpenShift

在 AWS 上安装单一节点集群需要使用"在带有自定义"流程的 AWS 上安装集群。

其他资源

- 使用自定义在 AWS 上安装集群

14.2.3.4. 在 Azure 上安装单节点 OpenShift

在 Azure 上安装单一节点集群需要使用"使用自定义"流程在 Azure 上安装集群。

其他资源

- 使用自定义在 Azure 上安装集群

14.2.3.5. 在 GCP 上安装单节点 OpenShift

在 GCP 上安装单一节点集群需要使用"使用自定义"流程在 GCP 上安装集群。

其他资源

- 使用自定义在 GCP 上安装集群

14.2.4. 在 USB 驱动器中创建可引导 ISO 镜像

您可以使用包含 ISO 镜像的可引导 USB 驱动器安装软件。使用 USB 驱动器引导服务器为软件安装准备服务器。

流程

1. 在管理主机上，在 USB 端口中插入 USB 驱动器。
2. 创建可引导 USB 驱动器，例如：

```
# dd if=<path_to_iso> of=<path_to_usb> status=progress
```

其中：

<path_to_iso>

是下载的 ISO 文件的相对路径，例如 **rhcos-live.iso**。

<path_to_usb>

是连接的 USB 驱动器的位置，例如 **/dev/sdb**。

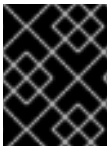
将 ISO 复制到 USB 驱动器后，您可以使用 USB 驱动器在服务器上安装软件。

14.2.5. 使用 Redfish API 从 HTTP 托管 ISO 镜像引导

您可以使用 Redfish Baseboard Management Controller (BMC) API 安装的 ISO 来置备网络中的主机。

注意

这个示例步骤演示了在 Dell 服务器上进行的步骤。



重要

确保具有与硬件兼容的 iDRAC 的最新固件版本。如果硬件或固件有问题，您必须联系相关供应商。

先决条件

- 下载安装 Red Hat Enterprise Linux CoreOS (RHCOS) ISO。
- 使用与 iDRAC9 兼容的 Dell PowerEdge 服务器。

流程

1. 将 ISO 文件复制到网络中的 HTTP 服务器。
2. 从托管 ISO 文件引导主机，例如：
   a. 运行以下命令，调用 Redfish API 将托管的 ISO 设置为 **VirtualMedia** 引导介质：

```
$ curl -k -u <bmc_username>:<bmc_password> -d '{"Image":"<hosted_iso_file>",
"Inserted": true}' -H "Content-Type: application/json" -X POST
<host_bmc_address>/redfish/v1/Managers/iDRAC.Embedded.1/VirtualMedia/CD/Actions/Vi
rtualMedia.InsertMedia
```

其中：

<bmc_username>:<bmc_password>

是目标主机 BMC 的用户名和密码。

<hosted_iso_file>

是托管安装 ISO 的 URL，例如：**http://webserver.example.com/rhcos-live-minimal.iso**。ISO 必须从目标主机机器中访问。

<host_bmc_address>

是目标主机计算机的 BMC IP 地址。

   b. 运行以下命令，将主机设置为从 **VirtualMedia** 设备引导：

```
$ curl -k -u <bmc_username>:<bmc_password> -X PATCH -H 'Content-Type:
```

```
application/json' -d '{"Boot": {"BootSourceOverrideTarget": "Cd",
"BootSourceOverrideMode": "UEFI", "BootSourceOverrideEnabled": "Once"}}'
<host_bmc_address>/redfish/v1/Systems/System.Embedded.1
```

c. 重启主机：

```
$ curl -k -u <bmc_username>:<bmc_password> -d '{"ResetType": "ForceRestart"}' -H
'Content-type: application/json' -X POST
<host_bmc_address>/redfish/v1/Systems/System.Embedded.1/Actions/ComputerSystem.R
eset
```

d. 可选：如果主机已关闭，您可以使用 **{"ResetType": "On"}** 开关引导它。运行以下命令：

```
$ curl -k -u <bmc_username>:<bmc_password> -d '{"ResetType": "On"}' -H 'Content-
type: application/json' -X POST
<host_bmc_address>/redfish/v1/Systems/System.Embedded.1/Actions/ComputerSystem.R
eset
```

14.2.6. 为远程服务器访问创建自定义 live RHCOS ISO

在某些情况下，您无法将外部磁盘驱动器附加到服务器，但您需要远程访问服务器来调配节点。建议您启用对服务器的 SSH 访问。您可以创建一个启用了 SSHd 并预定义的凭证的 live RHCOS ISO，以便您可以在服务器引导后访问服务器。

先决条件

- 已安装 **butane** 工具。

流程

1. 从 **coreos-installer** image mirror 页面下载 **coreos-installer** 二进制文件。
2. 从 mirror.openshift.com 下载最新的 live RHCOS ISO。
3. 创建 **butane** 实用程序用来创建 Ignition 文件的 **embedded.yaml** 文件：

```
variant: openshift
version: 4.16.0
metadata:
  name: sshd
  labels:
    machineconfiguration.openshift.io/role: worker
passwd:
  users:
    - name: core (1)
      ssh_authorized_keys:
        - '<ssh_key>'
```

(1) **core** 用户具有 sudo 权限。

4. 运行以下命令，运行 **butane** 工具来创建 Ignition 文件：

```
$ butane -pr embedded.yaml -o embedded.ign
```

5. 创建 Ignition 文件后，您可以使用 **coreos-installer** 工具将配置包含在新的 live RHCOS ISO 中，名为 **rhcos-sshd-4.16.0-x86_64-live.x86_64.iso** ：

```
$ coreos-installer iso ignition embed -i embedded.ign rhcos-4.16.0-x86_64-live.x86_64.iso -o
rhcos-sshd-4.16.0-x86_64-live.x86_64.iso
```

验证

- 运行以下命令，检查自定义 live ISO 是否可用于引导服务器：

```
# coreos-installer iso ignition show rhcos-sshd-4.16.0-x86_64-live.x86_64.iso
```

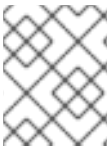
输出示例

```
{
  "ignition": {
    "version": "3.2.0"
  },
  "passwd": {
    "users": [
      {
        "name": "core",
        "sshAuthorizedKeys": [
          "ssh-rsa
AAAAB3NzaC1yc2EAAAADAQABAAABAQCZnG8AIzlDAhpyENpK2qKiTT8EbRWOrz7NXjRzo
pbPu215mocaJgjjwJjh1cYhgPhpAp6M/ttTk7I4OI7g4588Apx4bwJep6oWTU35LkY8ZxkGVPAJL
8kVITdKQviDv3XX12l4QfnDom4tm4gVbRH0gNT1wzhnLP+LKYm2Ohr9D7p9NBnAdro6k++X
WgkDeijLRUTwdEyWunldW1f8G0Mg8Y1Xzr13BUo3+8aey7HLKJMDtobkz/C8ESYA/f7HJc5Fx
F0XbapWWovSSDJrr9OmlL9f4TfE+cQk3s+eoKiz2bgNPRgEEwihVbGsCN4grA+RzLCAOpec+
2dTJrQvFqsD alosadag@sonnelicht.local"
        ]
      }
    ]
  }
}
```

14.2.7. 使用 IBM Z 和 IBM LinuxONE 安装单节点 OpenShift

在 IBM Z® 和 IBM® LinuxONE 上安装单一节点集群需要使用以下方法之一安装：

- 在 IBM Z® 和 IBM® LinuxONE 上使用 z/VM 安装集群
- 在 IBM Z® 和 IBM® LinuxONE 上使用 RHEL KVM 安装集群
- 在 IBM Z® 和 IBM® LinuxONE 上的 LPAR 上安装集群



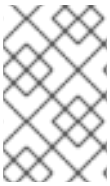
注意

在 IBM Z® 上安装单节点集群简化了开发和测试环境的安装，在条目级别需要较少的资源要求。

硬件要求

- 等同于两个用于 Linux (IFL)的集成设施，每个集群都启用了 SMT2。

- 至少一个网络连接连接到 **LoadBalancer** 服务，并为集群外的流量提供数据。



注意

您可以使用专用或共享的 IFL 来分配足够的计算资源。资源共享是 IBM Z® 的关键优势之一。但是，您必须正确调整每个虚拟机监控程序层上的容量，并确保每个 OpenShift Container Platform 集群都有足够资源。

14.2.7.1. 在 IBM Z 和 IBM LinuxONE 中使用 z/VM 安装单节点 OpenShift

先决条件

- 已安装 **podman**。

流程

1. 运行以下命令设置 OpenShift Container Platform 版本：

```
$ OCP_VERSION=<ocp_version> 1
```

1 将 **<ocp_version>** 替换为当前版本。例如：**latest-4.16**。

2. 运行以下命令来设置主机架构：

```
$ ARCH=<architecture> 1
```

1 将 **<architecture>** 替换为目标主机架构 **s390x**。

3. 输入以下命令下载 OpenShift Container Platform 客户端(**oc**)并使其可用：

```
$ curl -k https://mirror.openshift.com/pub/openshift-v4/${ARCH}/clients/ocp/${OCP_VERSION}/openshift-client-linux.tar.gz -o oc.tar.gz
```

```
$ tar zxf oc.tar.gz
```

```
$ chmod +x oc
```

4. 输入以下命令下载 OpenShift Container Platform 安装程序并使其可用：

```
$ curl -k https://mirror.openshift.com/pub/openshift-v4/${ARCH}/clients/ocp/${OCP_VERSION}/openshift-install-linux.tar.gz -o openshift-install-linux.tar.gz
```

```
$ tar zxvf openshift-install-linux.tar.gz
```

```
$ chmod +x openshift-install
```

5. 准备 **install-config.yaml** 文件：

```
apiVersion: v1
baseDomain: <domain> (1)
compute:
- name: worker
  replicas: 0 (2)
controlPlane:
  name: master
  replicas: 1 (3)
metadata:
  name: <name> (4)
networking: (5)
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16 (6)
  networkType: OVNKubernetes
  serviceNetwork:
  - 172.30.0.0/16
platform:
  none: {}
bootstrapInPlace:
  installationDisk: /dev/disk/by-id/<disk_id> (7)
pullSecret: '<pull_secret>' (8)
sshKey: |
  <ssh_key> (9)
```

1. 添加集群域名。
2. 将 **计算** 副本设置为 **0**。这使得 control plane 节点可以调度。
3. 将 **controlPlane** 副本设置为 **1**。与前面的 **compute** 设置结合使用，此设置可确保集群在单一节点上运行。
4. 将 **metadata** 名称设置为集群名称。
5. 设置**网络**详情。OVN-Kubernetes 是单节点集群唯一支持的网络插件类型。
6. 将 **cidr** 值设置为与单节点 OpenShift 集群的子网匹配。
7. 设置安装磁盘驱动器的路径，例如： **/dev/disk/by-id/wwn-0x64cd98f04fde100024684cf3034da5c2**。
8. 从 Red Hat OpenShift Cluster Manager 复制 pull secret ，并将内容添加到此配置设置中。
9. 从管理主机添加公共 SSH 密钥，以便您可以在安装后登录集群。

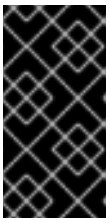
6. 运行以下命令来生成 OpenShift Container Platform 资产：

```
$ mkdir ocp
```

```
$ cp install-config.yaml ocp
```

```
$ ./openshift-install --dir=ocp create single-node-ignition-config
```

7. 从 Red Hat Customer Portal 的产品下载页，或从 RHCOS image mirror 页获取 RHEL **kernel**, **initramfs**, 和 **rootfs** 工件。



重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每个发行版本而改变。您必须下载最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。仅使用以下流程中描述的适当 **kernel**、**initramfs** 和 **rootfs** 工件。

文件名包含 OpenShift Container Platform 版本号。它们类似以下示例：

kernel

rhcos-<version>-live-kernel-<architecture>

initramfs

rhcos-<version>-live-initramfs.<architecture>.img

rootfs

rhcos-<version>-live-rootfs.<architecture>.img

注意

FCP 和 DASD 的 **rootfs** 镜像是相同的。

8. 将以下工件和文件移到 HTTP 或 HTTPS 服务器中：
   - 下载的 RHEL live **kernel**、**initramfs** 和 **rootfs** 工件
   - Ignition 文件
9. 为特定虚拟机创建参数文件：

参数文件示例

```
rd.neednet=1 \
console=ttysclp0 \
coreos.live.rootfs_url=<rhcos_liveos>:8080/rootfs.img \ 1
ignition.firstboot ignition.platform.id=metal \
ignition.config.url=<rhcos_ign>:8080/ignition/bootstrap-in-place-for-live-iso.ign \ 2
ip=encbdd0:dhcp::02:00:00:02:34:02 3
rd.znet=qeth,0.0.bdd0,0.0.bdd1,0.0.bdd2,layer2=1 \
rd.dasd=0.0.4411 \ 4
rd.zfcp=0.0.8001,0x50050763040051e3,0x4000406300000000 \ 5
zfcp.allow_lun_scan=0 \
rd.luks.options=discard
```

1 对于 **coreos.live.rootfs_url=** 工件，请您引导的 **kernel`and `initramfs** 指定匹配的 **rootfs** 工件。仅支持 HTTP 和 HTTPS 协议。

2 对于 **ignition.config.url=** 参数，请为机器角色指定 Ignition 文件。仅支持 HTTP 和 HTTPS 协议。

3 对于 **ip=** 参数，使用 DHCP 自动分配 IP 地址，或者手动分配 IP 地址，如"在 IBM Z® 和 IBM® LinuxONE 上使用 z/VM 安装集群"中所述。

4 对于在 DASD 类型磁盘中安装，请使用 **rd.dasd=** 指定要安装 RHCOS 的 DASD。为 FCP 类型磁盘省略此条目。

5 对于在 FCP 类型磁盘中安装，请使用 **rd.zfcp=<adapter>,<wwpn>,<lun>** 指定要安装 RHCOS 的 FCP 磁盘。为 DASD 类型磁盘省略这个条目。

所有其他参数保持不变。

10. 将以下工件、文件和镜像传送到 z/VM。例如，使用 FTP:
    - **kernel** 和 **initramfs** 工件
    - 参数文件
    - RHCOS 镜像
      有关如何使用 FTP 传输文件并从虚拟读取器引导的详情，请参阅 在 Z/VM 中安装。
11. 将文件 punch 到 z/VM 客户机虚拟机的虚拟读取器，即成为 bootstrap 节点。
12. 在 bootstrap 机器上登录到 CMS。
13. 运行以下命令，从 reader IPL bootstrap 机器：

```
$ cp ipl c
```

14. 首次重启虚拟机后，在另一个虚拟机后直接运行以下命令：
    a. 要在首次重启后引导 DASD 设备，请运行以下命令：

```
$ cp i <devno> clear loadparm prompt
```

其中：

<devno>
指定客户机看到的引导设备的设备号。

```
$ cp vi vmsg 0 <kernel_parameters>
```

其中：

<kernel_parameters>
指定要存储为系统控制程序数据(SCPDATA)的一组内核参数。引导 Linux 时，这些内核参数将串联到引导配置所使用的现有内核参数的末尾。combined 参数字符串不能超过 896 个字符。

    b. 要在首次重启后引导 FCP 设备，请运行以下命令：

```
$ cp set loaddev portname <wwpn> lun <lun>
```

其中：

<wwpn>

以十六进制格式指定目标端口和 **<lun>** 逻辑单元。

```
$ cp set loaddev bootprog <n>
```

其中：

<n>

指定要引导的内核。

```
$ cp set loaddev scpdata {APPEND|NEW} '<kernel_parameters>'
```

其中：

<kernel_parameters>

指定要存储为系统控制程序数据(SCPDATA)的一组内核参数。引导 Linux 时，这些内核参数将串联到引导配置所使用的现有内核参数的末尾。combined 参数字符串不能超过 896 个字符。

<APPEND|NEW>

可选：指定 **APPEND** 将内核参数附加到现有 SCPDATA 中。这是默认值。指定 **NEW** 来替换现有的 SCPDATA。

Example

```
$ cp set loaddev scpdata
'rd.zfcp=0.0.8001,0x500507630a0350a4,0x4000409D00000000
ip=encbdd0:dhcp::02:00:00:02:34:02 rd.neednet=1'
```

要启动 IPL 和引导过程，请运行以下命令：

```
$ cp i <devno>
```

其中：

<devno>

指定客户机看到的引导设备的设备号。

14.2.7.2. 在 IBM Z 和 IBM LinuxONE 上使用 RHEL KVM 安装单节点 OpenShift

先决条件

- 已安装 **podman**。

流程

1. 运行以下命令设置 OpenShift Container Platform 版本：

```
$ OCP_VERSION=<ocp_version> 1
```

1 将 **<ocp_version>** 替换为当前版本。例如：**latest-4.16**。

2. 运行以下命令来设置主机架构：

```
$ ARCH=<architecture> 1
```

1 将 **<architecture>** 替换为目标主机架构 **s390x**。

3. 输入以下命令下载 OpenShift Container Platform 客户端(**oc**)并使其可用：

```
$ curl -k https://mirror.openshift.com/pub/openshift-v4/${ARCH}/clients/ocp/${OCP_VERSION}/openshift-client-linux.tar.gz -o oc.tar.gz
```

```
$ tar zxf oc.tar.gz
```

```
$ chmod +x oc
```

4. 输入以下命令下载 OpenShift Container Platform 安装程序并使其可用：

```
$ curl -k https://mirror.openshift.com/pub/openshift-v4/${ARCH}/clients/ocp/${OCP_VERSION}/openshift-install-linux.tar.gz -o openshift-install-linux.tar.gz
```

```
$ tar zxvf openshift-install-linux.tar.gz
```

```
$ chmod +x openshift-install
```

5. 准备 **install-config.yaml** 文件：

```
apiVersion: v1
baseDomain: <domain> 1
compute:
- name: worker
  replicas: 0 2
controlPlane:
  name: master
  replicas: 1 3
metadata:
  name: <name> 4
networking: 5
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16 6
  networkType: OVNKubernetes
  serviceNetwork:
  - 172.30.0.0/16
platform:
  none: {}
```

```
bootstrapInPlace:
  installationDisk: /dev/disk/by-id/<disk_id> 7
pullSecret: '<pull_secret>' 8
sshKey: |
  <ssh_key> 9
```

1. 添加集群域名。
2. 将 **计算** 副本设置为 **0**。这使得 control plane 节点可以调度。
3. 将 **controlPlane** 副本设置为 **1**。与前面的 **compute** 设置结合使用，此设置可确保集群在单一节点上运行。
4. 将 **metadata** 名称设置为集群名称。
5. 设置**网络**详情。OVN-Kubernetes 是单节点集群唯一支持的网络插件类型。
6. 将 **cidr** 值设置为与单节点 OpenShift 集群的子网匹配。
7. 设置安装磁盘驱动器的路径，例如： **/dev/disk/by-id/wwn-0x64cd98f04fde100024684cf3034da5c2**。
8. 从 Red Hat OpenShift Cluster Manager 复制 pull secret ， 并将内容添加到此配置设置中。
9. 从管理主机添加公共 SSH 密钥，以便您可以在安装后登录集群。

6. 运行以下命令来生成 OpenShift Container Platform 资产：

```
$ mkdir ocp
```

```
$ cp install-config.yaml ocp
```

```
$ ./openshift-install --dir=ocp create single-node-ignition-config
```

7. 从 Red Hat Customer Portal 的产品下载页，或从 RHCOS image mirror 页获取 RHEL **kernel**, **initramfs**, 和 **rootfs** 工件。

重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每个发行版本而改变。您必须下载最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。仅使用以下流程中描述的适当 **kernel**、**initramfs** 和 **rootfs** 工件。

文件名包含 OpenShift Container Platform 版本号。它们类似以下示例：

kernel

rhcos-<version>-live-kernel-<architecture>

initramfs

rhcos-<version>-live-initramfs.<architecture>.img

rootfs

rhcos-<version>-live-rootfs.<architecture>.img

8. 在启动 **virt-install** 前，将以下文件和工件移到 HTTP 或 HTTPS 服务器中：
   - 下载的 RHEL live **kernel**、**initramfs** 和 **rootfs** 工件
   - Ignition 文件
9. 使用以下组件创建 KVM 客户机节点：
   - RHEL **kernel** 和 **initramfs** 工件
   - Ignition 文件
   - 新磁盘镜像
   - 调整了 parm 行参数

```
$ virt-install \
  --name <vm_name> \
  --autostart \
  --memory=<memory_mb> \
  --cpu host \
  --vcpus <vcpus> \
  --location <media_location>,kernel=<rhcos_kernel>,initrd=<rhcos_initrd> \ 1
  --disk size=100 \
  --network network=<virt_network_parm> \
  --graphics none \
  --noautoconsole \
  --extra-args "ip=<ip>::<gateway>:<mask>:<hostname>::none" \
  --extra-args "nameserver=<name_server>" \
  --extra-args "ip=dhcp rd.neednet=1 ignition.platform.id=metal ignition.firstboot" \
  --extra-args "coreos.live.rootfs_url=<rhcos_liveos>" \ 2
  --extra-args "ignition.config.url=<rhcos_ign>" \ 3
  --extra-args "random.trust_cpu=on rd.luks.options=discard" \
  --extra-args "console=ttysclp0" \
  --wait
```

1. 对于 **--location** 参数，指定 HTTP 或 HTTPS 服务器中的 kernel/initrd 的位置。
2. 对于 **coreos.live.rootfs_url=** 工件，请为您引导的 **kernel** 和 **initramfs** 指定匹配的 **rootfs** 工件。仅支持 HTTP 和 HTTPS 协议。
3. 对于 **ignition.config.url=** 参数，请为机器角色指定 Ignition 文件。仅支持 HTTP 和 HTTPS 协议。

14.2.7.3. 在 IBM Z 和 IBM LinuxONE 上的 LPAR 中安装单节点 OpenShift

先决条件

- 如果要部署单节点集群，其中没有计算节点，则 Ingress Controller pod 会在 control plane 节点上运行。在单节点集群部署中，您必须配置应用程序入口负载均衡器，将 HTTP 和 HTTPS 流量路由到 control plane 节点。如需更多信息，*请参阅用户置备的基础架构的负载平衡要求* 部分。

流程

1. 运行以下命令设置 OpenShift Container Platform 版本：

```
$ OCP_VERSION=<ocp_version> 1
```

1 将 **<ocp_version>** 替换为当前版本。例如：**latest-4.16**。

2. 运行以下命令来设置主机架构：

```
$ ARCH=<architecture> 1
```

1 将 **<architecture>** 替换为目标主机架构 **s390x**。

3. 输入以下命令下载 OpenShift Container Platform 客户端(**oc**)并使其可用：

```
$ curl -k https://mirror.openshift.com/pub/openshift-v4/${ARCH}/clients/ocp/${OCP_VERSION}/openshift-client-linux.tar.gz -o oc.tar.gz
```

```
$ tar zxvf oc.tar.gz
```

```
$ chmod +x oc
```

4. 输入以下命令下载 OpenShift Container Platform 安装程序并使其可用：

```
$ curl -k https://mirror.openshift.com/pub/openshift-v4/${ARCH}/clients/ocp/${OCP_VERSION}/openshift-install-linux.tar.gz -o openshift-install-linux.tar.gz
```

```
$ tar zxvf openshift-install-linux.tar.gz
```

```
$ chmod +x openshift-install
```

5. 准备 **install-config.yaml** 文件：

```
apiVersion: v1
baseDomain: <domain> 1
compute:
- name: worker
  replicas: 0 2
controlPlane:
  name: master
  replicas: 1 3
metadata:
  name: <name> 4
networking: 5
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16 6
  networkType: OVNKubernetes
```

```
  serviceNetwork:
  - 172.30.0.0/16
platform:
  none: {}
pullSecret: '<pull_secret>' 7
sshKey: |
  <ssh_key> 8
```

1. 添加集群域名。
2. 将 **计算** 副本设置为 **0**。这使得 control plane 节点可以调度。
3. 将 **controlPlane** 副本设置为 **1**。与前面的 **compute** 设置结合使用，此设置可确保集群在单一节点上运行。
4. 将 **metadata** 名称设置为集群名称。
5. 设置**网络**详情。OVN-Kubernetes 是单节点集群唯一支持的网络插件类型。
6. 将 **cidr** 值设置为与单节点 OpenShift 集群的子网匹配。
7. 从 Red Hat OpenShift Cluster Manager 复制 pull secret ，并将内容添加到此配置设置中。
8. 从管理主机添加公共 SSH 密钥，以便您可以在安装后登录集群。

6. 运行以下命令来生成 OpenShift Container Platform 资产：

```
$ mkdir ocp
```

```
$ cp install-config.yaml ocp
```

7. 进入包含 OpenShift Container Platform 安装程序的目录，并为集群生成 Kubernetes 清单：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1. 对于 **<installation_directory>**，请指定包含您创建的 **install-config.yaml** 文件的安装目录。

8. 检查 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** Kubernetes 清单文件中的 **mastersSchedulable** 参数是否已设置为 **true**。
   a. 打开 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** 文件。
   b. 找到 **mastersSchedulable** 参数，并确保它被设置为 **true**，如以下 **spec** 小节中所示：

```
spec:
  mastersSchedulable: true
status: {}
```

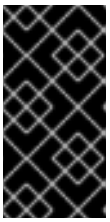
   c. 保存并退出 文件。

9. 要创建 Ignition 配置文件，请从包含安装程序的目录运行以下命令：

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定相同的安装目录。

10. 从 Red Hat Customer Portal 的产品下载页，或从 RHCOS image mirror 页获取 RHEL **kernel**, **initramfs**, 和 **rootfs** 工件。



重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每个发行版本而改变。您必须下载最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。仅使用以下流程中描述的适当 **kernel**、**initramfs** 和 **rootfs** 工件。

文件名包含 OpenShift Container Platform 版本号。它们类似以下示例：

kernel

rhcos-<version>-live-kernel-<architecture>

initramfs

rhcos-<version>-live-initramfs.<architecture>.img

rootfs

rhcos-<version>-live-rootfs.<architecture>.img

注意

FCP 和 DASD 的 **rootfs** 镜像是相同的。

11. 将以下工件和文件移到 HTTP 或 HTTPS 服务器中：

- 下载的 RHEL live **kernel**、**initramfs** 和 **rootfs** 工件
- Ignition 文件

12. 在 LPAR 中为 bootstrap 创建参数文件：

bootstrap 机器的参数文件示例

```
cio_ignore=all,!condev rd.neednet=1 \
console=ttysclp0 \
coreos.inst.install_dev=/dev/<block_device> \1
coreos.inst.ignition_url=http://<http_server>/bootstrap.ign \2
coreos.live.rootfs_url=http://<http_server>/rhcos-<version>-live-rootfs.<architecture>.img \3
ip=<ip>::<gateway>:<netmask>:<hostname>::none nameserver=<dns> \4
rd.znet=qeth,0.0.1140,0.0.1141,0.0.1142,layer2=1,portno=0 \
rd.dasd=0.0.4411 \5
rd.zfcp=0.0.8001,0x50050763040051e3,0x4000406300000000 \6
zfcp.allow_lun_scan=0
```

1. 指定要安装到的系统中的块设备。对于在 DASD 类型磁盘中的安装，使用 **dasda**；对于在 FCP 类型磁盘中的安装，使用 **sda**。
2. 指定 **bootstrap.ign** 配置文件的位置。仅支持 HTTP 和 HTTPS 协议。
3. 对于 **coreos.live.rootfs_url=** 工件，请您引导的 **kernel`and `initramfs** 指定匹配的 **rootfs** 工件。仅支持 HTTP 和 HTTPS 协议。
4. 对于 **ip=** 参数，手动分配 IP 地址，如"在 IBM Z® 和 IBM® LinuxONE 上的 LPAR 中安装集群中所述。
5. 对于在 DASD 类型磁盘中安装，请使用 **rd.dasd=** 指定要安装 RHCOS 的 DASD。为 FCP 类型磁盘省略此条目。
6. 对于在 FCP 类型磁盘中安装，请使用 **rd.zfcp=<adapter>,<wwpn>,<lun>** 指定要安装 RHCOS 的 FCP 磁盘。为 DASD 类型磁盘省略这个条目。

如果需要，您可以对参数进行进一步的调整。

13. 在 LPAR 中为 control plane 创建参数文件：

control plane 机器的参数文件示例

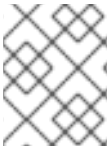
```
cio_ignore=all,!condev rd.neednet=1 \
console=ttysclp0 \
coreos.inst.install_dev=/dev/<block_device> \
coreos.inst.ignition_url=http://<http_server>/master.ign \ 1
coreos.live.rootfs_url=http://<http_server>/rhcos-<version>-live-rootfs.<architecture>.img \
ip=<ip>::<gateway>:<netmask>:<hostname>::none nameserver=<dns> \
rd.znet=qeth,0.0.1140,0.0.1141,0.0.1142,layer2=1,portno=0 \
rd.dasd=0.0.4411 \
rd.zfcp=0.0.8001,0x50050763040051e3,0x4000406300000000 \
zfcp.allow_lun_scan=0
```

1. 指定 **master.ign** 配置文件的位置。仅支持 HTTP 和 HTTPS 协议。

14. 将以下工件、文件和镜像传送到 LPAR。例如，使用 FTP:
    - **kernel** 和 **initramfs** 工件
    - 参数文件
    - RHCOS 镜像
      有关如何使用 FTP 和引导传输文件的详情，请参考在 LPAR 中安装。
15. 引导 bootstrap 机器。
16. 引导 control plane 机器。

14.2.8. 使用 IBM Power 安装单节点 OpenShift

在 IBM Power® 上安装单节点集群需要使用"使用 IBM Power® 安装集群"流程进行用户置备的安装。

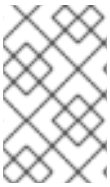


注意

在 IBM Power® 上安装单节点集群简化了开发和测试环境的安装，在条目级别需要较少的资源要求。

硬件要求

- 等同于两个用于 Linux (IFL)的集成设施，每个集群都启用了 SMT2。
- 至少一个网络连接连接到 **LoadBalancer** 服务，并为集群外的流量提供数据。



注意

您可以使用专用或共享的 IFL 来分配足够的计算资源。资源共享是 IBM Power® 的关键优势之一。但是，您必须正确调整每个虚拟机监控程序层上的容量，并确保每个 OpenShift Container Platform 集群都有足够资源。

其他资源

- 在 IBM Power® 上安装集群。

14.2.8.1. 使用 IBM Power 为单节点 OpenShift 设置 basion

在 IBM Power® 上安装单节点 OpenShift 之前，您必须设置 bastion。为 IBM Power® 上的单节点 OpenShift 设置堡垒服务器需要配置以下服务：

- PXE 用于单节点 OpenShift 集群安装。PXE 需要配置并运行以下服务：
  - DNS 定义 api、api-int 和 Ifapps
  - DHCP 服务启用 PXE，并为单节点 OpenShift 节点分配 IP 地址
  - HTTP 提供 ignition 和 RHCOS rootfs 镜像
  - TFTP 启用 PXE
- 您必须安装 **dnsmasq** 以支持 DNS、DHCP 和 PXE，httpd 用于 HTTP。

使用以下步骤配置满足这些要求的堡垒服务器。

流程

1. 使用以下命令安装 **grub2**，这是为 PowerVM 启用 PXE 所需的：

```
grub2-mknetdir --net-directory=/var/lib/tftpboot
```

/var/lib/tftpboot/boot/grub2/grub.cfg 文件示例

```
default=0
fallback=1
timeout=1
if [ ${net_default_mac} == fa:b0:45:27:43:20 ]; then
menuentry "CoreOS (BIOS)" {
  echo "Loading kernel"
  linux "/rhcos/kernel" ip=dhcp rd.neednet=1 ignition.platform.id=metal ignition.firstboot
```

```
coreos.live.rootfs_url=http://192.168.10.5:8000/install/rootfs.img
ignition.config.url=http://192.168.10.5:8000/ignition/sno.ign
  echo "Loading initrd"
  initrd  "/rhcos/initramfs.img"
}
fi
```

2. 使用以下命令，从镜像仓库下载 RHCOS 镜像文件用于 PXE。

   a. 输入以下命令为 **RHCOS_URL** 变量分配以下 4.12 URL：

   ```
   $ export RHCOS_URL=https://mirror.openshift.com/pub/openshift-v4/ppc64le/dependencies/rhcos/4.12/latest/
   ```

   b. 输入以下命令进入 **/var/lib/tftpboot/rhcos** 目录：

   ```
   $ cd /var/lib/tftpboot/rhcos
   ```

   c. 输入以下命令从 **RHCOS_URL** 变量中存储的 URL 下载指定的 RHCOS kernel 文件：

   ```
   $ wget ${RHCOS_URL}/rhcos-live-kernel-ppc64le -o kernel
   ```

   d. 输入以下命令从 **RHCOS_URL** 变量中存储的 URL 下载 RHCOS **initramfs** 文件：

   ```
   $ wget ${RHCOS_URL}/rhcos-live-initramfs.ppc64le.img -o initramfs.img
   ```

   e. 输入以下命令进入 **/var/var/www/html/install/** 目录：

   ```
   $ cd /var//var/www/html/install/
   ```

   f. 输入以下命令从 **RHCOS_URL** 变量中存储的 URL 中下载并保存 RHCOS **根文件系统**镜像文件：

   ```
   $ wget ${RHCOS_URL}/rhcos-live-rootfs.ppc64le.img -o rootfs.img
   ```

3. 要为单节点 OpenShift 集群创建 ignition 文件，您必须创建 **install-config.yaml** 文件。

   a. 输入以下命令来创建包含该文件的工作目录：

   ```
   $ mkdir -p ~/sno-work
   ```

   b. 输入以下命令进入到 **~/sno-work** 目录：

   ```
   $ cd ~/sno-work
   ```

   c. 使用以下示例文件可以在 **~/sno-work** 目录中创建所需的 **install-config.yaml** ：

   ```
   apiVersion: v1
   baseDomain: <domain> 1
   compute:
   - name: worker
     replicas: 0 2
   ```

```
controlPlane:
  name: master
  replicas: 1 3
metadata:
  name: <name> 4
networking: 5
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16 6
  networkType: OVNKubernetes
  serviceNetwork:
  - 172.30.0.0/16
platform:
  none: {}
bootstrapInPlace:
  installationDisk: /dev/disk/by-id/<disk_id> 7
pullSecret: '<pull_secret>' 8
sshKey: |
  <ssh_key> 9
```

1. 添加集群域名。
2. 将 **计算** 副本设置为 **0**。这使得 control plane 节点可以调度。
3. 将 **controlPlane** 副本设置为 **1**。与前面的 **compute** 设置结合使用，此设置可确保集群在单个节点上运行。
4. 将 **metadata** 名称设置为集群名称。
5. 设置**网络**详情。OVN-Kubernetes 是单节点集群唯一支持的网络插件类型。
6. 将 **cidr** 值设置为与单节点 OpenShift 集群的子网匹配。
7. 设置安装磁盘驱动器的路径，例如： **/dev/disk/by-id/wwn-0x64cd98f04fde100024684cf3034da5c2**。
8. 从 Red Hat OpenShift Cluster Manager 复制 pull secret ， 并将内容添加到此配置设置中。
9. 从管理主机添加公共 SSH 密钥，以便您可以在安装后登录集群。

4. 下载 **openshift-install** 镜像以创建 ignition 文件并将其复制到 **http** 目录中。

   a. 输入以下命令下载 **openshift-install-linux-4.12.0** .tar 文件：

   ```
   $ wget https://mirror.openshift.com/pub/openshift-v4/ppc64le/clients/ocp/4.12.0/openshift-install-linux-4.12.0.tar.gz
   ```

   b. 输入以下命令解包 **openshift-install-linux-4.12.0.tar.gz** 归档：

   ```
   $ tar xzvf openshift-install-linux-4.12.0.tar.gz
   ```

c. 输入以下命令

```
$ ./openshift-install --dir=~/sno-work create create single-node-ignition-config
```

d. 运行以下命令来创建 ignition 文件：

```
$ cp ~/sno-work/single-node-ignition-config.ign /var/www/html/ignition/sno.ign
```

e. 输入以下命令为 **/var/www/html** 目录恢复 SELinux 文件：

```
$ restorecon -vR /var/www/html || true
```

堡垒现在具有所有必需的文件，并正确配置以安装单节点 OpenShift。

14.2.8.2. 使用 IBM Power 安装单节点 OpenShift

先决条件

- 您已设置了 bastion。

流程

单节点 OpenShift 集群安装有两个步骤。首先，单节点 OpenShift 逻辑分区 (LPAR) 需要使用 PXE 引导，然后您需要监控安装进度。

1. 使用以下命令通过 netboot 引导 powerVM:

```
$ lpar_netboot -i -D -f -t ent -m <sno_mac> -s auto -d auto -S <server_ip> -C <sno_ip> -G
<gateway> <lpar_name> default_profile <cec_name>
```

其中：

sno_mac
指定单节点 OpenShift 集群的 MAC 地址。

sno_ip
指定单节点 OpenShift 集群的 IP 地址。

server_ip
指定 bastion (PXE 服务器的 IP 地址)。

gateway
指定网络的网关 IP。

lpar_name
在 HMC 中指定单节点 OpenShift lpar 名称。

cec_name
指定 sno_lpar 所在的系统名称

2. 在单节点 OpenShift LPAR 使用 PXE 引导后，使用 **openshift-install** 命令监控安装进度：

a. bootstrap 完成后运行以下命令：

```
./openshift-install wait-for bootstrap-complete
```

b. 成功返回后运行以下命令：

```
./openshift-install wait-for install-complete
```

第 15 章 在裸机上安装

15.1. 准备裸机集群安装

15.1.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您已阅读了有关 选择集群安装方法并为用户准备它 的文档。

15.1.2. 为 OpenShift Virtualization 规划裸机集群

如果使用 OpenShift Virtualization，必须在安装裸机集群前了解一些要求。

- 如果要使用实时迁移功能，*在集群安装时*需要有多个 worker 节点。这是因为实时迁移需要集群级别的高可用性 (HA) 标记设置为 true。当安装集群时，会设置 HA 标志，之后无法更改。如果在安装集群时定义少于两个 worker 节点，则集群生命周期中的 HA 标记被设置为 false。

注意

您可以在单节点集群中安装 OpenShift Virtualization，但单节点 OpenShift 不支持高可用性。

- 实时迁移需要共享存储。OpenShift Virtualization 的存储必须支持并使用 ReadWriteMany (RWX) 访问模式。
- 如果您计划使用单根 I/O 虚拟化 (SR-IOV)，请确保 OpenShift Container Platform 支持网络接口控制器 (NIC)。

其他资源

- OpenShift Virtualization 入门
- 为 OpenShift Virtualization 准备集群
- 关于单根 I/O 虚拟化（SR-IOV）硬件网络
- 将虚拟机连接到 SR-IOV 网络

15.1.3. SR-IOV 设备的 NIC 分区（技术预览）

OpenShift Container Platform 可以部署到具有双端口网络接口卡(NIC)的服务器上。您可以将单个、高速的双端口 NIC 分区到多个虚拟功能 (VF) 并启用 SR-IOV。

此功能支持使用绑定来实现链路聚合控制协议 (LACP) 的高可用性。

注意

物理 NIC 只能声明一个 LACP。

OpenShift Container Platform 集群可以使用以下方法之一部署到 2 个物理功能 (PF)上带有 2 个 VF 的绑定接口中：

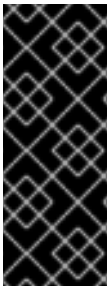
- 基于代理的安装程序

注意

nmstate 的最低所需版本是：

- **1.4.2-4** 对于 RHEL 8 版本
- **2.2.7** 对于 RHEL 9 版本

- 安装程序置备的基础架构安装
- 用户置备的基础架构安装



重要

支持与为 SR-IOV 设备启用 NIC 分区关联的第 1 天操作只是一个技术预览功能。技术预览功能不受红帽产品服务等级协议（SLA）支持，且功能可能并不完整。红帽不推荐在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。

有关红帽技术预览功能支持范围的更多信息，请参阅技术预览功能支持范围。

其他资源

- 示例：绑定和 SR-IOV 双节点网络配置
- 可选：为双端口 NIC 配置主机网络接口
- 将多个 SR-IOV 网络接口绑定到双端口 NIC 接口

15.1.4. 选择在裸机上安装 OpenShift Container Platform 的方法

OpenShift Container Platform 安装程序提供四个部署集群的方法：

- **交互式** ：您可以使用基于 Web 的 辅助安装程序（Assisted Installer）部署集群。这是使用连接到互联网的网络的集群推荐的方法。Assisted Installer 是安装 OpenShift Container Platform 的最简单方法，它提供智能默认值，并在安装集群前执行预动态验证。它还提供了一个 RESTful API 用于自动化和高级配置场景。
- **本地基于代理的** ：您可以使用基于代理的安装程序为 air-gapped 或受限网络在本地部署集群。它提供了 Assisted Installer 的许多优点，但您必须首先下载并配置基于代理的安装程序。使用命令行界面完成配置。这个方法对于 air-gapped 或受限网络是一个理想的方法。
- **自动** ：您可以在安装程序置备的基础架构中部署集群及其维护的集群。安装程序使用每个集群主机的基板管理控制器 (BMC) 进行置备。您可以使用连接的网络或 air-gapped 网络或受限网络部署集群。
- **完全控制**：您可以在自己准备和维护的基础架构上部署集群，这种方法提供了最大的定制性。您可以使用连接的网络或 air-gapped 网络或受限网络部署集群。

集群有以下特征：

- 默认提供无单点故障的高可用性基础架构。

- 管理员可以控制要应用的更新内容和时间。

如需有关安装程序置备和用户置备的安装过程的更多信息，请参阅 安装过程。

15.1.4.1. 在安装程序置备的基础架构上安装集群

您可以使用以下方法在 OpenShift Container Platform 安装程序置备的裸机基础架构上安装集群：

- **在裸机上安装安装程序置备的集群** ：您可以使用安装程序置备在裸机上安装 OpenShift Container Platform。

15.1.4.2. 在用户置备的基础架构上安装集群

您可以使用以下方法之一在您置备的裸机基础架构上安装集群：

- **在裸机上安装用户置备的集群** ：您可以在您置备的裸机基础架构上安装 OpenShift Container Platform。对于包含用户置备的基础架构的集群，您必须部署所有所需的机器。
- **使用自定义网络安装用户置备的裸机集群：您可以使用网络自定义** 在用户置备的基础架构上安装裸机集群。通过自定义网络配置，您的集群可以与环境中现有的 IP 地址分配共存，并与现有的 MTU 和 VXLAN 配置集成。大多数网络自定义必须在安装阶段应用。
- **在受限网络中安装用户置备的裸机集群：** 您可以使用镜像 registry 在受限网络或断开连接的网络中安装用户置备的裸机集群。您还可以使用此安装方法来确保集群只使用满足您组织对外部内容控制的容器镜像。

15.2. 在裸机上安装用户置备的集群

在 OpenShift Container Platform 4.16 中，您可以在您置备的裸机基础架构上安装集群。

重要

虽然您可能能够按照以下步骤在虚拟化或云环境中部署集群，但您必须了解非裸机平台的其他注意事项。在尝试在此类环境中安装 OpenShift Container Platform 集群前，请参阅有关在未经测试的平台上部署 OpenShift Container Platform 的指南 中的信息。

15.2.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读有关 选择集群安装方法的文档，并为用户准备它。
- 如果使用防火墙，则会 将其配置为允许集群需要访问的站点。

注意

如果要配置代理，请务必查看此站点列表。

15.2.2. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类型的基础架构上执行受限网络安装。在此过程中，您可以下载所需的内容，并使用它为镜像 registry 填充安装软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群前，您要更新镜像 registry 的内容。

其他资源

- 有关 在您置备的裸机基础架构上执行受限网络安装的更多信息，请参阅在受限网络上安装用户置备的裸机集群。

15.2.3. 具有用户置备基础架构的集群的要求

对于包含用户置备的基础架构的集群，您必须部署所有所需的机器。

本节论述了在用户置备的基础架构上部署 OpenShift Container Platform 的要求。

15.2.3.1. 集群安装所需的机器

最小的 OpenShift Container Platform 集群需要以下主机：

表 15.1. 最低所需的主机

主机	描述
一个临时 bootstrap 机器	集群需要 bootstrap 机器在三台 control plane 机器上部署 OpenShift Container Platform 集群。您可在安装集群后删除 bootstrap 机器。
三台 control plane 机器	control plane 机器运行组成 control plane 的 Kubernetes 和 OpenShift Container Platform 服务。
至少两台计算机器，也称为 worker 机器。	OpenShift Container Platform 用户请求的工作负载在计算机器上运行。

注意

作为例外，您可以在裸机集群中运行零台计算机器，它们仅由三台 control plane 机器组成。这为集群管理员和开发人员提供了更小、效率更高的集群，用于测试、开发和生产。不支持运行一台计算机器。

重要

要保持集群的高可用性，请将独立的物理主机用于这些集群机器。

bootstrap 和 control plane 机器必须使用 Red Hat Enterprise Linux CoreOS(RHCOS)作为操作系统。但是，计算机器可以在 Red Hat Enterprise Linux CoreOS(RHCOS)、Red Hat Enterprise Linux(RHEL) 8.6 和更高的版本。

请注意，RHCOS 基于 Red Hat Enterprise Linux(RHEL) 9.2，并继承其所有硬件认证和要求。查看 红帽企业 Linux 技术功能和限制。

15.2.3.2. 集群安装的最低资源要求

每台集群机器都必须满足以下最低要求：

表 15.2. 最低资源要求

机器	操作系统	CPU [1]	RAM	Storage	每秒输入/输出 (IOPS) [2]
bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane（控制平面）	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS、RHEL 8.6 及更新版本 [3]	2	8 GB	100 GB	300

1. 当未启用并发多线程 (SMT) 或超线程时，一个 CPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比例：（每个内核数的线程）× sockets = CPU。
2. OpenShift Container Platform 和 Kubernetes 对磁盘性能非常敏感，建议使用更快的存储速度，特别是 control plane 节点上需要 10 ms p99 fsync 持续时间的 etcd。请注意，在许多云平台上，存储大小和 IOPS 可一起扩展，因此您可能需要过度分配存储卷来获取足够的性能。
3. 与所有用户置备的安装一样，如果您选择在集群中使用 RHEL 计算机器，则负责所有操作系统生命周期管理和维护，包括执行系统更新、应用补丁和完成所有其他必要的任务。RHEL 7 计算机器的使用已弃用，并已在 OpenShift Container Platform 4.10 及更新的版本中删除。

注意

从 OpenShift Container Platform 版本 4.13 开始，RHCOS 基于 RHEL 版本 9.2，它更新了微架构要求。以下列表包含每个架构需要的最小指令集架构 (ISA)：

- x86-64 体系结构需要 x86-64-v2 ISA
- ARM64 架构需要 ARMv8.0-A ISA
- IBM Power 架构需要 Power 9 ISA
- s390x 架构需要 z14 ISA

如需更多信息，请参阅 RHEL 架构。

如果平台的实例类型满足集群机器的最低要求，则 OpenShift Container Platform 支持使用它。

其他资源

- 优化存储

15.2.3.3. 证书签名请求管理

在使用您置备的基础架构时，集群只能有限地访问自动机器管理，因此您必须提供一种在安装后批准集群证书签名请求 (CSR) 的机制。**kube-controller-manager** 只能批准 kubelet 客户端 CSR。 **machine-approver** 无法保证使用 kubelet 凭证请求的提供证书的有效性，因为它不能确认是正确的机器发出了该请求。您必须决定并实施一种方法，以验证 kubelet 提供证书请求的有效性并进行批准。

其他资源

- 有关在裸机 环境中部署三节点集群的详情，请参阅配置 三节点集群。
- 有关 在安装后批准集群证书签名请求的更多信息，请参阅批准机器 的证书签名请求。

15.2.3.4. vSphere 上裸机集群的要求

确保在集群中的所有虚拟机上启用 **disk.EnableUUID** 参数。

其他资源

- 如需了解为用户置备的基础架构在 VMware vSphere 上将 **disk.EnableUUID** 参数的值设置为 **TRUE** 的详情，请参阅安装 RHCOS 并启动 OpenShift Container Platform bootstrap 过程 。

15.2.3.5. 用户置备的基础架构对网络的要求

所有 Red Hat Enterprise Linux CoreOS(RHCOS)机器都需要在启动时在 **initramfs** 中配置联网，以获取它们的 Ignition 配置文件。

在初次启动过程中，机器需要 IP 地址配置，该配置通过 DHCP 服务器或静态设置，提供所需的引导选项。建立网络连接后，机器会从 HTTP 或 HTTPS 服务器下载 Ignition 配置文件。然后，Ignition 配置文件用于设置每台机器的确切状态。Machine Config Operator 在安装后完成对机器的更多更改，如应用新证书或密钥。

建议使用 DHCP 服务器对集群机器进行长期管理。确保 DHCP 服务器已配置为向集群机器提供持久的 IP 地址、DNS 服务器信息和主机名。

注意

如果用户置备的基础架构没有 DHCP 服务，您可以在 RHCOS 安装时向节点提供 IP 网络配置和 DNS 服务器地址。如果要从 ISO 镜像安装，这些参数可作为引导参数传递。如需有关静态 IP 置备和高级网络选项的更多信息，请参阅 *安装 RHCOS 并启动 OpenShift Container Platform bootstrap 过程* 部分。

Kubernetes API 服务器必须能够解析集群机器的节点名称。如果 API 服务器和 worker 节点位于不同的区域中，您可以配置默认 DNS 搜索区域，以允许 API 服务器解析节点名称。另一种支持的方法是始终通过节点对象和所有 DNS 请求中的完全限定域名引用主机。

15.2.3.5.1. 通过 DHCP 设置集群节点主机名

在 Red Hat Enterprise Linux CoreOS(RHCOS)机器上，主机名是通过 NetworkManager 设置的。默认情况下，机器通过 DHCP 获取其主机名。如果主机名不是由 DHCP 提供，请通过内核参数或者其它方法进

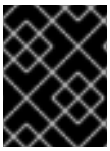
行静态设置，请通过反向 DNS 查找获取。反向 DNS 查找在网络初始化后进行，可能需要一些时间来解决。其他系统服务可以在此之前启动，并将主机名检测为 **localhost** 或类似的内容。您可以使用 DHCP 为每个集群节点提供主机名来避免这种情况。

另外，通过 DHCP 设置主机名可以绕过实施 DNS split-horizon 的环境中的手动 DNS 记录名称配置错误。

15.2.3.5.2. 网络连接要求

您必须配置机器之间的网络连接，以允许 OpenShift Container Platform 集群组件进行通信。每台机器都必须能够解析集群中所有其他机器的主机名。

本节详细介绍了所需的端口。



重要

在连接的 OpenShift Container Platform 环境中，所有节点都需要访问互联网才能为平台容器拉取镜像，并向红帽提供遥测数据。

表 15.3. 用于全机器到所有机器通信的端口

协议	port	描述
ICMP	N/A	网络可访问性测试
TCP	**1936**	指标
	9000-**9999**	主机级别的服务，包括端口 9 **100**-**9101** 上的节点导出器，以及端口 **9099** 上的 Cluster Version Operator。
	10250-**10259**	Kubernetes 保留的默认端口
UDP	**4789**	VXLAN
	6081	Geneve
	9000-**9999**	主机级别的服务，包括端口 **9100**到**9101** 上的节点导出器。
	500	IPsec IKE 数据包
	4500	IPsec NAT-T 数据包
	123	UDP 端口 **123** 上的网络时间协议 (NTP) 如果配置了外部 NTP 时间服务器，需要打开 UDP 端口 **123**。
TCP/UDP	**30000**-**32767**	Kubernetes 节点端口
ESP	N/A	IPsec Encapsulating Security Payload(ESP)

表 15.4. 用于所有机器控制平面通信的端口

协议	port	描述
TCP	**6443**	Kubernetes API

表 15.5. control plane 机器用于 control plane 机器通信的端口

协议	port	描述
TCP	**2379-2380**	etcd 服务器和对等端口

用户置备的基础架构的 NTP 配置

OpenShift Container Platform 集群被配置为默认使用公共网络时间协议(NTP)服务器。如果要使用本地企业 NTP 服务器，或者集群部署在断开连接的网络中，您可以将集群配置为使用特定的时间服务器。如需更多信息，*请参阅配置 chrony 时间服务* 的文档。

如果 DHCP 服务器提供 NTP 服务器信息，Red Hat Enterprise Linux CoreOS(RHCOS)机器上的 chrony 时间服务会读取信息，并可以把时钟与 NTP 服务器同步。

其他资源

- 配置 chrony 时间服务

15.2.3.6. 用户置备的 DNS 要求

在 OpenShift Container Platform 部署中，以下组件需要 DNS 名称解析：

- The Kubernetes API
- OpenShift Container Platform 应用程序通配符
- bootstrap、control plane 和计算机器

Kubernetes API、bootstrap 机器、control plane 机器和计算机器也需要反向 DNS 解析。

DNS A/AAAA 或 CNAME 记录用于名称解析，PTR 记录用于反向名称解析。反向记录很重要，因为 Red Hat Enterprise Linux CoreOS(RHCOS)使用反向记录为所有节点设置主机名，除非 DHCP 提供主机名。另外，反向记录用于生成 OpenShift Container Platform 需要操作的证书签名请求(CSR)。

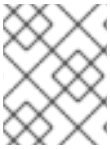
注意

建议使用 DHCP 服务器为每个群集节点提供主机名。如需更多信息，*请参阅用户置备的基础架构部分的 DHCP 建议*。

用户置备的 OpenShift Container Platform 集群需要以下 DNS 记录，这些记录必须在安装前就位。在每个记录中，**<cluster_name>** 是集群名称，**<base_domain>** 是您在 **install-config.yaml** 文件中指定的基域。完整的 DNS 记录采用以下形式：**<component>.<cluster_name>.<base_domain>.**。

表 15.6. 所需的 DNS 记录

组件	记录	描述
Kubernetes API	**api.<cluster_name>.<base_domain>.**	DNS A/AAAA 或 CNAME 记录，以及用于标识 API 负载均衡器的 DNS PTR 记录。这些记录必须由集群外的客户端和集群中的所有节点解析。
	api-int.<cluster_name>.<base_domain>.	DNS A/AAAA 或 CNAME 记录，以及用于内部标识 API 负载均衡器的 DNS PTR 记录。这些记录必须可以从集群中的所有节点解析。  **重要** API 服务器必须能够根据 Kubernetes 中记录的主机名解析 worker 节点。如果 API 服务器无法解析节点名称，则代理的 API 调用会失败，且您无法从 pod 检索日志。
Routes	***.apps.<cluster_name>.<base_domain>.**	通配符 DNS A/AAAA 或 CNAME 记录，指向应用程序入口负载均衡器。应用程序入口负载均衡器以运行 Ingress Controller Pod 的机器为目标。默认情况下，Ingress Controller Pod 在计算机器上运行。这些记录必须由集群外的客户端和集群中的所有节点解析。 例如，console **-openshift-console.apps.<cluster_name>.<base_domain>** 用作到 OpenShift Container Platform 控制台的通配符路由。
bootstrap 机器	**bootstrap.<cluster_name>.<base_domain>.**	DNS A/AAAA 或 CNAME 记录，以及用于标识 bootstrap 机器的 DNS PTR 记录。这些记录必须由集群中的节点解析。
control plane 机器	**<control_plane><n>.<cluster_name>.<base_domain>.**	DNS A/AAAA 或 CNAME 记录，以识别 control plane 节点的每台机器。这些记录必须由集群中的节点解析。
计算机器	**<compute><n>.<cluster_name>.<base_domain>.**	DNS A/AAAA 或 CNAME 记录，用于识别 worker 节点的每台机器。这些记录必须由集群中的节点解析。



注意

在 OpenShift Container Platform 4.4 及更新的版本中，您不需要在 DNS 配置中指定 etcd 主机和 SRV 记录。

提示

您可以使用 **dig** 命令验证名称和反向名称解析。如需了解详细的 *验证步骤，请参阅为用户置备的基础架构验证 DNS 解析* 一节。

15.2.3.6.1. 用户置备的集群的 DNS 配置示例

本节提供 A 和 PTR 记录配置示例，它们满足了在用户置备的基础架构上部署 OpenShift Container Platform 的 DNS 要求。样本不是为选择一个 DNS 解决方案提供建议。

在这个示例中，集群名称为 **ocp4**，基域是 **example.com**。

用户置备的集群的 DNS A 记录配置示例

以下示例是 BIND 区域文件，其中显示了用户置备的集群中名称解析的 A 记录示例。

例 15.1. DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
  2019070700 ; serial
  3H  ; refresh (3 hours)
  30M  ; retry (30 minutes)
  2W  ; expiry (2 weeks)
  1W )  ; minimum (1 week)
 IN NS ns1.example.com.
 IN MX 10 smtp.example.com.
;
;
ns1.example.com.  IN A 192.168.1.5
smtp.example.com.  IN A 192.168.1.5
;
helper.example.com.  IN A 192.168.1.5
helper.ocp4.example.com. IN A 192.168.1.5
;
api.ocp4.example.com.  IN A 192.168.1.5 1
api-int.ocp4.example.com. IN A 192.168.1.5 2
;
*.apps.ocp4.example.com. IN A 192.168.1.5 3
;
bootstrap.ocp4.example.com. IN A 192.168.1.96 4
;
control-plane0.ocp4.example.com. IN A 192.168.1.97 5
control-plane1.ocp4.example.com. IN A 192.168.1.98 6
control-plane2.ocp4.example.com. IN A 192.168.1.99 7
;
compute0.ocp4.example.com. IN A 192.168.1.11 8
compute1.ocp4.example.com. IN A 192.168.1.7 9
;
;EOF
```

1. 为 Kubernetes API 提供名称解析。记录引用 API 负载均衡器的 IP 地址。
2. 为 Kubernetes API 提供名称解析。记录引用 API 负载均衡器的 IP 地址，用于内部集群通信。
3. 为通配符路由提供名称解析。记录引用应用程序入口负载均衡器的 IP 地址。应用程序入口负载均衡器以运行 Ingress Controller Pod 的机器为目标。默认情况下，Ingress Controller Pod 在计算机器上运行。

注意

在这个示例中，将相同的负载均衡器用于 Kubernetes API 和应用入口流量。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。

4 为 bootstrap 机器提供名称解析。

5 6 7 为 control plane 机器提供名称解析。

8 9 为计算机器提供名称解析。

用户置备的集群的 DNS PTR 记录配置示例

以下示例 BIND 区域文件显示了用户置备的集群中反向名称解析的 PTR 记录示例。

例 15.2. 反向记录的 DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
  2019070700 ; serial
  3H  ; refresh (3 hours)
  30M  ; retry (30 minutes)
  2W  ; expiry (2 weeks)
  1W )  ; minimum (1 week)
 IN NS ns1.example.com.
;
5.1.168.192.in-addr.arpa. IN PTR api.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. IN PTR api-int.ocp4.example.com. 2
;
96.1.168.192.in-addr.arpa. IN PTR bootstrap.ocp4.example.com. 3
;
97.1.168.192.in-addr.arpa. IN PTR control-plane0.ocp4.example.com. 4
98.1.168.192.in-addr.arpa. IN PTR control-plane1.ocp4.example.com. 5
99.1.168.192.in-addr.arpa. IN PTR control-plane2.ocp4.example.com. 6
;
11.1.168.192.in-addr.arpa. IN PTR compute0.ocp4.example.com. 7
7.1.168.192.in-addr.arpa. IN PTR compute1.ocp4.example.com. 8
;
;EOF
```

1 为 Kubernetes API 提供反向 DNS 解析。PTR 记录引用 API 负载均衡器的记录名称。

2 为 Kubernetes API 提供反向 DNS 解析。PTR 记录引用 API 负载均衡器的记录名称，用于内部集群通信。

3 为 bootstrap 机器提供反向 DNS 解析。

4 5 6 为 control plane 机器提供反向 DNS 解析。

7 8 为计算机器提供反向 DNS 解析。

注意

OpenShift Container Platform 应用程序通配符不需要 PTR 记录。

其他资源

- 验证用户置备的基础架构的 DNS 解析

15.2.3.7. 用户置备的基础架构的负载均衡要求

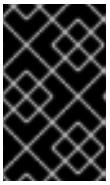
在安装 OpenShift Container Platform 前，您必须置备 API 和应用程序入口负载均衡基础架构。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。

注意

如果要使用 Red Hat Enterprise Linux (RHEL) 实例部署 API 和应用程序入口负载均衡器，您必须单独购买 RHEL 订阅。

负载平衡基础架构必须满足以下要求：

1. **API 负载均衡器** ：提供一个通用端点，供用户（包括人工和机器）与平台交互和配置。配置以下条件：
   - 仅第 4 层负载均衡.这可被称为 Raw TCP 或 SSL Passthrough 模式。
   - 无状态负载平衡算法。这些选项根据负载均衡器的实施而有所不同。



重要

不要为 API 负载均衡器配置会话持久性。为 Kubernetes API 服务器配置会话持久性可能会导致出现过量 OpenShift Container Platform 集群应用程序流量，以及过量的在集群中运行的 Kubernetes API。

在负载均衡器的前端和后端配置以下端口：

表 15.7. API 负载均衡器

port	后端机器（池成员）	internal	外部	描述
6443	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。您必须为 API 服务器健康检查探测配置 **/readyz** 端点。	X	X	Kubernetes API 服务器
22623	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。	X		机器配置服务器

注意

负载均衡器必须配置为，从 API 服务器关闭 **/readyz** 端点到从池中移除 API 服务器实例时最多需要 30 秒。在 **/readyz** 返回错误或健康后的时间范围内，端点必须被删除或添加。每 5 秒或 10 秒探测一次，有两个成功请求处于健康状态，三个成为不健康的请求是经过良好测试的值。

2. **应用程序入口负载均衡器** ：为应用程序流量从集群外部流提供入口点。OpenShift Container Platform 集群需要正确配置入口路由器。
配置以下条件：

   - 仅第 4 层负载均衡.这可被称为 Raw TCP 或 SSL Passthrough 模式。
   - 建议根据可用选项以及平台上托管的应用程序类型，使用基于连接的或基于会话的持久性。

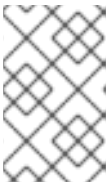
提示

如果应用程序入口负载均衡器可以看到客户端的真实 IP 地址，启用基于 IP 的会话持久性可以提高使用端到端 TLS 加密的应用程序的性能。

在负载均衡器的前端和后端配置以下端口：

表 15.8. 应用程序入口负载均衡器

port	后端机器（池成员）	internal	外部	描述
443	默认情况下，运行 Ingress Controller Pod、计算或 worker 的机器。	X	X	HTTPS 流量
80	默认情况下，运行 Ingress Controller Pod、计算或 worker 的机器。	X	X	HTTP 流量



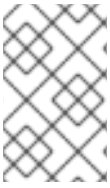
注意

如果要部署一个带有零计算节点的三节点集群，Ingress Controller Pod 在 control plane 节点上运行。在三节点集群部署中，您必须配置应用程序入口负载均衡器，将 HTTP 和 HTTPS 流量路由到 control plane 节点。

15.2.3.7.1. 用户置备的集群的负载均衡器配置示例

本节提供了一个满足用户置备集群的负载均衡要求的 API 和应用程序入口负载均衡器配置示例。示例是 HAProxy 负载均衡器的 **/etc/haproxy/haproxy.cfg** 配置。这个示例不是为选择一个负载平衡解决方案提供建议。

在这个示例中，将相同的负载均衡器用于 Kubernetes API 和应用入口流量。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。



注意

如果您使用 HAProxy 作为负载均衡器，并且 SELinux 设置为 **enforcing**，您必须通过运行 **setsebool -P haproxy_connect_any=1** 来确保 HAProxy 服务可以绑定到配置的 TCP 端口。

例 15.3. API 和应用程序入口负载均衡器配置示例

```
global
  log         127.0.0.1 local2
  pidfile     /var/run/haproxy.pid
  maxconn     4000
  daemon
defaults
  mode                    http
  log                     global
  option                  dontlognull
  option http-server-close
  option                  redispatch
  retries                 3
  timeout http-request    10s
  timeout queue           1m
  timeout connect         10s
  timeout client          1m
  timeout server          1m
  timeout http-keep-alive 10s
  timeout check           10s
  maxconn                 3000
listen api-server-6443 (1)
  bind *:6443
  mode tcp
  option  httpchk GET /readyz HTTP/1.0
  option  log-health-checks
  balance roundrobin
  server bootstrap bootstrap.ocp4.example.com:6443 verify none check check-ssl inter 10s fall 2
rise 3 backup (2)
  server master0 master0.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
fall 2 rise 3
  server master1 master1.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
fall 2 rise 3
  server master2 master2.ocp4.example.com:6443 weight 1 verify none check check-ssl inter 10s
fall 2 rise 3
listen machine-config-server-22623 (3)
  bind *:22623
  mode tcp
  server bootstrap bootstrap.ocp4.example.com:22623 check inter 1s backup (4)
  server master0 master0.ocp4.example.com:22623 check inter 1s
  server master1 master1.ocp4.example.com:22623 check inter 1s
  server master2 master2.ocp4.example.com:22623 check inter 1s
listen ingress-router-443 (5)
  bind *:443
  mode tcp
  balance source
  server compute0 compute0.ocp4.example.com:443 check inter 1s
  server compute1 compute1.ocp4.example.com:443 check inter 1s
listen ingress-router-80 (6)
  bind *:80
  mode tcp
  balance source
  server compute0 compute0.ocp4.example.com:80 check inter 1s
  server compute1 compute1.ocp4.example.com:80 check inter 1s
```

1. 端口 **6443** 处理 Kubernetes API 流量并指向 control plane 机器。
2. 4. bootstrap 条目必须在 OpenShift Container Platform 集群安装前就位，且必须在 bootstrap 过程完成后删除它们。
3. 端口 **22623** 处理机器配置服务器流量并指向 control plane 机器。
5. 端口 **443** 处理 HTTPS 流量，并指向运行 Ingress Controller pod 的机器。默认情况下，Ingress Controller Pod 在计算机器上运行。
6. 端口 **80** 处理 HTTP 流量，并指向运行 Ingress Controller pod 的机器。默认情况下，Ingress Controller Pod 在计算机器上运行。

注意

如果要部署一个带有零计算节点的三节点集群，Ingress Controller Pod 在 control plane 节点上运行。在三节点集群部署中，您必须配置应用程序入口负载均衡器，将 HTTP 和 HTTPS 流量路由到 control plane 节点。

提示

如果您使用 HAProxy 作为负载均衡器，您可以通过在 HAProxy 节点上运行 **netstat -nltupe** 来检查 **haproxy** 进程是否在侦听端口 **6443**、**22623**、**443** 和 **80**。

15.2.4. 准备用户置备的基础架构

在用户置备的基础架构上安装 OpenShift Container Platform 之前，您必须准备底层基础架构。

本节详细介绍了设置集群基础架构以准备 OpenShift Container Platform 安装所需的高级别步骤。这包括为您的集群节点配置 IP 网络和网络连接，通过防火墙启用所需的端口，以及设置所需的 DNS 和负载均衡基础架构。

准备后，集群基础架构必须满足 *带有用户置备的基础架构部分的集群要求*。

先决条件

- 您已参阅 OpenShift Container Platform 4.x Tested Integrations 页面。
- 您已查看了 *具有用户置备基础架构的集群要求部分中详述的基础架构* 要求。

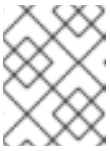
流程

1. 如果您使用 DHCP 向集群节点提供 IP 网络配置，请配置 DHCP 服务。
   a. 将节点的持久 IP 地址添加到您的 DHCP 服务器配置。在您的配置中，将相关网络接口的 MAC 地址与每个节点的预期 IP 地址匹配。
   b. 当您使用 DHCP 为集群机器配置 IP 寻址时，机器还通过 DHCP 获取 DNS 服务器信息。定义集群节点通过 DHCP 服务器配置使用的持久性 DNS 服务器地址。

注意

如果没有使用 DHCP 服务，则必须在 RHCOS 安装时为节点提供 IP 网络配置和 DNS 服务器地址。如果要从 ISO 镜像安装，这些参数可作为引导参数传递。如需有关静态 IP 置备和高级网络选项的更多信息，请参阅 *安装 RHCOS 并启动 OpenShift Container Platform bootstrap 过程* 部分。

c. 在 DHCP 服务器配置中定义集群节点的主机名。有关 *主机名注意事项的详情，请参阅通过 DHCP 设置集群节点* 主机名部分。



注意

如果没有使用 DHCP 服务，集群节点可以通过反向 DNS 查找来获取其主机名。

2. 确保您的网络基础架构提供集群组件之间所需的网络连接。有关 *要求的详情，请参阅用户置备的基础架构* 的网络要求部分。

3. 将防火墙配置为启用 OpenShift Container Platform 集群组件进行通信所需的端口。*如需有关所需端口的详细信息，请参阅用户置备的基础架构* 部分的网络要求。

重要

默认情况下，OpenShift Container Platform 集群可以访问端口 **1936**，因为每个 control plane 节点都需要访问此端口。

避免使用 Ingress 负载均衡器公开此端口，因为这样做可能会导致公开敏感信息，如统计信息和指标（与 Ingress Controller 相关的统计信息和指标）。

4. 为集群设置所需的 DNS 基础架构。

   a. 为 Kubernetes API、应用程序通配符、bootstrap 机器、control plane 机器和计算机器配置 DNS 名称解析。

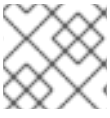
   b. 为 Kubernetes API、bootstrap 机器、control plane 机器和计算机器配置反向 DNS 解析。如需有关 *OpenShift Container Platform DNS 要求的更多信息，请参阅用户置备* DNS 要求部分。

5. 验证您的 DNS 配置。

   a. 从安装节点，针对 Kubernetes API 的记录名称、通配符路由和集群节点运行 DNS 查找。验证响应中的 IP 地址是否与正确的组件对应。

   b. 从安装节点，针对负载均衡器和集群节点的 IP 地址运行反向 DNS 查找。验证响应中的记录名称是否与正确的组件对应。
   有关详细的 *DNS 验证步骤，请参阅用户置备的基础架构* 验证 DNS 解析部分。

6. 置备所需的 API 和应用程序入口负载平衡基础架构。有关 *要求的更多信息，请参阅用户置备的基础架构的负载平衡* 要求部分。



注意

某些负载平衡解决方案要求在初始化负载平衡之前，对群集节点进行 DNS 名称解析。

其他资源

- 具有用户置备基础架构的集群的要求
- 安装 RHCOS 并启动 OpenShift Container Platform bootstrap 过程
- 通过 DHCP 设置集群节点主机名
- 高级 RHCOS 安装配置
- 用户置备的基础架构对网络的要求
- 用户置备的 DNS 要求
- 验证用户置备的基础架构的 DNS 解析
- 用户置备的基础架构的负载均衡要求

15.2.5. 验证用户置备的基础架构的 DNS 解析

您可以在在用户置备的基础架构上安装 OpenShift Container Platform 前验证 DNS 配置。

重要

本节中详述的验证步骤必须在安装集群前成功。

先决条件

- 已为您的用户置备的基础架构配置了所需的 DNS 记录。

流程

1. 从安装节点，针对 Kubernetes API 的记录名称、通配符路由和集群节点运行 DNS 查找。验证响应中包含的 IP 地址是否与正确的组件对应。

   a. 对 Kubernetes API 记录名称执行查询。检查结果是否指向 API 负载均衡器的 IP 地址：

   ```
   $ dig +noall +answer @<nameserver_ip> api.<cluster_name>.<base_domain> 1
   ```

   1 将 **<nameserver_ip>** 替换为 nameserver 的 IP 地址， **<cluster_name>** 替换为您的集群名称，**<base_domain>** 替换为您的基本域名。

   输出示例

   ```
   api.ocp4.example.com.  604800 IN A 192.168.1.5
   ```

   b. 对 Kubernetes 内部 API 记录名称执行查询。检查结果是否指向 API 负载均衡器的 IP 地址：

   ```
   $ dig +noall +answer @<nameserver_ip> api-int.<cluster_name>.<base_domain>
   ```

   输出示例

   ```
   api-int.ocp4.example.com.  604800 IN A 192.168.1.5
   ```

c. 测试 ***.apps.<cluster_name>.<base_domain>* DNS** 通配符查找示例。所有应用程序通配符查询都必须解析为应用程序入口负载均衡器的 IP 地址：

```
$ dig +noall +answer @<nameserver_ip> random.apps.<cluster_name>.<base_domain>
```

输出示例

```
random.apps.ocp4.example.com.  604800 IN A 192.168.1.5
```

注意

在示例中，将相同的负载均衡器用于 Kubernetes API 和应用程序入口流量。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。

您可以使用另一个通配符值替换 **random**。例如，您可以查询到 OpenShift Container Platform 控制台的路由：

```
$ dig +noall +answer @<nameserver_ip> console-openshift-console.apps.
<cluster_name>.<base_domain>
```

输出示例

```
console-openshift-console.apps.ocp4.example.com. 604800 IN A 192.168.1.5
```

d. 针对 bootstrap DNS 记录名称运行查询。检查结果是否指向 bootstrap 节点的 IP 地址：

```
$ dig +noall +answer @<nameserver_ip> bootstrap.<cluster_name>.<base_domain>
```

输出示例

```
bootstrap.ocp4.example.com.  604800 IN A 192.168.1.96
```

e. 使用此方法对 control plane 和计算节点的 DNS 记录名称执行查找。检查结果是否与每个节点的 IP 地址对应。

2. 从安装节点，针对负载均衡器和集群节点的 IP 地址运行反向 DNS 查找。验证响应中包含的记录名称是否与正确的组件对应。

a. 对 API 负载均衡器的 IP 地址执行反向查找。检查响应是否包含 Kubernetes API 和 Kubernetes 内部 API 的记录名称：

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.5
```

输出示例

```
5.1.168.192.in-addr.arpa. 604800 IN PTR api-int.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. 604800 IN PTR api.ocp4.example.com. 2
```

1 为 Kubernetes 内部 API 提供记录名称。

2 为 Kubernetes API 提供记录名称。

注意

OpenShift Container Platform 应用程序通配符不需要 PTR 记录。针对应用程序入口负载均衡器的 IP 地址解析反向 DNS 解析不需要验证步骤。

b. 对 bootstrap 节点的 IP 地址执行反向查找。检查结果是否指向 bootstrap 节点的 DNS 记录名称：

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.96
```

输出示例

```
96.1.168.192.in-addr.arpa. 604800 IN PTR bootstrap.ocp4.example.com.
```

c. 使用此方法对 control plane 和计算节点的 IP 地址执行反向查找。检查结果是否与每个节点的 DNS 记录名称对应。

其他资源

- 用户置备的 DNS 要求
- 用户置备的基础架构的负载均衡要求

15.2.6. 为集群节点 SSH 访问生成密钥对

在 OpenShift Container Platform 安装过程中，您可以为安装程序提供 SSH 公钥。密钥通过它们的 Ignition 配置文件传递给 Red Hat Enterprise Linux CoreOS(RHCOS)节点，用于验证对节点的 SSH 访问。密钥添加到每个节点上 **core** 用户的 **~/.ssh/authorized_keys** 列表中，这将启用免密码身份验证。

将密钥传递给节点后，您可以使用密钥对作为用户 **核心** 通过 SSH 连接到 RHCOS 节点。若要通过 SSH 访问节点，必须由 SSH 为您的本地用户管理私钥身份。

如果要通过 SSH 连接到集群节点来执行安装调试或灾难恢复，则必须在安装过程中提供 SSH 公钥。**./openshift-install gather** 命令还需要在集群节点上设置 SSH 公钥。

重要

不要在生产环境中跳过这个过程，在生产环境中需要灾难恢复和调试。

注意

您必须使用本地密钥，而不是使用特定平台方法配置 的密钥，如 AWS 密钥对。

流程

1. 如果您在本地计算机上没有可用于在集群节点上进行身份验证的现有 SSH 密钥对，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name>
```
1

1 指定新 SSH 密钥的路径和文件名，如 **~/.ssh/id_ed25519**。如果您已有密钥对，请确保您的公钥位于 **~/.ssh** 目录中。

注意

如果您计划在 **x86_64**、**ppc64le** 和 **s390x** 架构上安装使用 RHEL 加密库（这些加密库已提交给 NIST 用于 FIPS 140-2/140-3 验证）的 OpenShift Container Platform 集群，则不要创建使用 **ed25519** 算法的密钥。相反，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 查看公共 SSH 密钥：

```
$ cat <path>/<file_name>.pub
```

例如，运行以下命令来查看 **~/.ssh/id_ed25519.pub** 公钥：

```
$ cat ~/.ssh/id_ed25519.pub
```

3. 将 SSH 私钥身份添加到本地用户的 SSH 代理（如果尚未添加）。在集群节点上，或者要使用 **./openshift-install gather** 命令，需要对该密钥进行 SSH 代理管理，才能在集群节点上进行免密码 SSH 身份验证。

注意

在某些发行版中，自动管理默认 SSH 私钥身份，如 **~/.ssh/id_rsa** 和 **~/.ssh/id_dsa**。

a. 如果 **ssh-agent** 进程尚未为您的本地用户运行，请将其作为后台任务启动：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果集群处于 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

4. 将 SSH 私钥添加到 **ssh-agent** ：

```
$ ssh-add <path>/<file_name> 1
```

1 指定 SSH 私钥的路径和文件名，如 **~/.ssh/id_ed25519.pub**

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

后续步骤

- 安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。如果在您置备的基础架构上安装集群，则必须为安装程序提供密钥。

其他资源

- 验证节点健康状况

15.2.7. 获取安装程序

在安装 OpenShift Container Platform 前，将安装文件下载到您用于安装的主机上。

先决条件

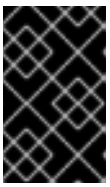
- 您有一台运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB。

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐户，请使用您的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入到安装类型的页面，下载与您的主机操作系统和架构对应的安装程序，并将该文件放在您要存储安装配置文件的目录中。

重要

安装程序会在用来安装集群的计算机上创建几个文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。



重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，请为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret 。此 pull secret 允许您与所含授权机构提供的服务进行身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

15.2.8. 安装 OpenShift CLI

您可以安装 OpenShift CLI(**oc**)来使用命令行界面与 OpenShift Container Platform 进行交互。您可以在 Linux、Windows 或 macOS 上安装 **oc**。

重要

如果安装了旧版本的 **oc**，则可能无法使用 OpenShift Container Platform 4.16 中的所有命令。下载并安装新版本的 **oc**。

在 Linux 上安装 OpenShift CLI
您可以按照以下流程在 Linux 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **产品变体** 下拉列表中选择架构。
3. 从 **版本** 下拉列表中选择适当的版本。
4. 点 **OpenShift v4.16 Linux Client** 条目旁的 **Download Now** 来保存文件。
5. 解包存档：

```
$ tar xvf <file>
```

6. 将 **oc** 二进制文件放到 **PATH 中的目录中**。
要查看您的 **PATH**，请执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

在 Windows 上安装 OpenShift CLI
您可以按照以下流程在 Windows 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 Windows Client** 条目旁的 **Download Now** 来保存文件。
4. 使用 ZIP 程序解压存档。
5. 将 **oc** 二进制文件移到 **PATH 中的目录中**。
要查看您的 **PATH**，请打开命令提示并执行以下命令：

```
C:\> path
```

验证

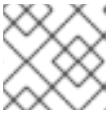
- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
C:\> oc <command>
```

在 macOS 上安装 OpenShift CLI
您可以按照以下流程在 macOS 上安装 OpenShift CLI(**oc**)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 **版本** 下拉列表中选择适当的版本。
3. 点 **OpenShift v4.16 macOS Client** 条目旁的 **Download Now** 来保存文件。



注意

对于 macOS arm64，请选择 **OpenShift v4.16 macOS arm64 Client** 条目。

4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
要查看您的 **PATH**，请打开终端并执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

15.2.9. 手动创建安装配置文件

安装集群要求您手动创建安装配置文件。

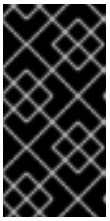
先决条件

- 您在本地机器上有一个 SSH 公钥来提供给安装程序。该密钥将用于在集群节点上进行 SSH 身份验证，以进行调试和灾难恢复。
- 已获取 OpenShift Container Platform 安装程序和集群的 pull secret。

流程

1. 创建一个安装目录来存储所需的安装资产：

```
$ mkdir <installation_directory>
```



重要

您必须创建一个目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

2. 自定义提供的 **install-config.yaml** 文件模板示例，并将其保存在 **<installation_directory>** 中。

注意

此配置文件必须命名为 **install-config.yaml**。

3. 备份 **install-config.yaml** 文件，以便您可以使用它安装多个集群。

重要

install-config.yaml 文件会在安装过程的下一步中使用。现在必须备份它。

其他资源

- 裸机的安装配置参数

15.2.9.1. 裸机 install-config.yaml 文件示例

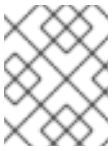
您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多详情，或修改所需参数的值。

```
apiVersion: v1
baseDomain: example.com (1)
compute: (2)
- hyperthreading: Enabled (3)
  name: worker
  replicas: 0 (4)
controlPlane: (5)
  hyperthreading: Enabled (6)
  name: master
  replicas: 3 (7)
metadata:
  name: test (8)
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14 (9)
    hostPrefix: 23 (10)
  networkType: OVNKubernetes (11)
  serviceNetwork: (12)
  - 172.30.0.0/16
platform:
  none: {} (13)
fips: false (14)
pullSecret: '{"auths": ...}' (15)
sshKey: 'ssh-ed25519 AAAA...' (16)
```

1 集群的基域。所有 DNS 记录都必须是这个基域的子域，并包含集群名称。

2 5 **controlPlane** 部分是一个单个映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，controlPlane **部分** 的第一行则不以连字符开头。仅使用一个 control plane 池。

3 6 指定要启用或禁用并发多线程(SMT)还是超线程。默认情况下，启用 SMT 可提高机器中内核的性能。您可以通过将 参数值设置为 **Disabled** 来禁用它。如果禁用 SMT，则必须在所有集群机器中禁用它；这包括 control plane 和计算机器。



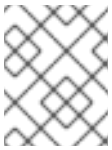
注意

默认启用并发多线程(SMT)。如果您的 BIOS 设置中没有启用 SMT，**超线程** 参数无效。

重要

如果您禁用 **超线程**，无论是在 BIOS 中，还是在 **install-config.yaml** 文件中，请确保您的容量规划考虑机器性能显著降低的情况。

4 在用户置备的基础架构上安装 OpenShift Container Platform 时，必须将这个值设置为 **0**。在安装程序置备的安装中，参数控制集群为您创建和管理的计算机器数量。在用户置备的安装中，您必须在完成集群安装前手动部署计算机器。



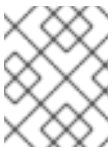
注意

如果要安装一个三节点集群，在安装 Red Hat Enterprise Linux CoreOS(RHCOS)机器时不要部署任何计算机器。

7 您添加到集群的 control plane 机器数量。由于集群使用这些值作为集群中的 etcd 端点数量，所以该值必须与您部署的 control plane 机器数量匹配。

8 您在 DNS 记录中指定的集群名称。

9 从中分配 Pod IP 地址的 IP 地址块。此块不得与现有物理网络重叠。这些 IP 地址用于 pod 网络。如果需要从外部网络访问 pod，您必须配置负载均衡器和路由器来管理流量。



注意

类 E CIDR 范围被保留以供以后使用。要使用 Class E CIDR 范围，您必须确保您的网络环境接受 Class E CIDR 范围内的 IP 地址。

10 分配给每个节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从 given **cidr** 中分配 a **/23** 子网，这样就能有 510（2^(32 - 23)- 2）个 pod IP 地址。如果需要从外部网络访问节点，请配置负载均衡器和路由器来管理流量。

11 要安装的集群网络插件。默认值 **OVNKubernetes** 是唯一支持的值。

12 用于服务 IP 地址的 IP 地址池。您只能输入一个 IP 地址池。此块不得与现有物理网络重叠。如果您需要从外部网络访问服务，请配置负载均衡器和路由器来管理流量。

13 您必须将平台设置为 **none**。您无法为您的平台提供额外的平台配置变量。

重要

使用平台类型 **none** 安装的集群无法使用一些功能，如使用 Machine API 管理计算机器。即使附加到集群的计算机器安装在通常支持该功能的平台上，也会应用这个限制。在安装后无法更改此参数。

14 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

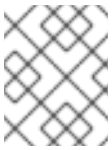
重要

要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 Red Hat Enterprise Linux (RHEL) 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅在 FIPS 模式中安装该系统。

当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS)时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。

15 Red Hat OpenShift Cluster Manager 的 pull secret 。此 pull secret 允许您与所含授权机构提供的服务进行身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

16 Red Hat Enterprise Linux CoreOS(RHCOS)中 **core** 用户的 SSH 公钥。



注意

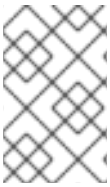
对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。

其他资源

- 如需有关 API 和应用程序入口负载均衡要求的更多信息，请参阅用户置备的基础架构 的负载平衡要求。
- 有关启用 在安装前禁用的集群功能的更多信息，请参阅启用集群功能。
- 如需有关每个功能提供的功能的更多信息，请参阅 OpenShift Container Platform 4.16 中的可选集群功能。

15.2.9.2. 在安装过程中配置集群范围的代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。



注意

对于裸机安装，如果您没有从 **install-config.yaml** 文件中的 **networking.machineNetwork[].cidr** 字段指定的范围分配节点 IP 地址，您必须将其包括在 **proxy.noProxy** 字段中。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 **Proxy** 对象的 **spec.noProxy** 字段中添加站点来绕过代理。

注意

Proxy 对象 **status.noProxy** 字段使用安装配置中的 **networking.machineNetwork[].cidr**、**networking.clusterNetwork[].cidr** 和 **networking.serviceNetwork[]** 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，**Proxy** 对象 **status.noProxy** 字段也会使用实例元数据端点填充(**169.254.169.254**)。

流程

1. 编辑 **install-config.yaml** 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 **http**。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3. 要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 **.** 以仅匹配子域。例如，**.y.com** 匹配 **x.y.com**，但不匹配 **y.com**。使用 ***** 绕过所有目的地的代理。
4. 如果提供，安装程序会在 **openshift-config** 命名空间中生成名为 **user-ca-bundle** 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 **trusted-ca-bundle** 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， **Proxy** 对象的 **trustedCA** 字段中也会引用此配置映射。**additionalTrustBundle** 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。
5. 可选：决定 **Proxy** 对象的配置以引用 **trustedCA** 字段中 **user-ca-bundle** 配置映射的策略。允许的值是 **Proxyonly** 和 **Always**。仅在配置了 **http/https** 代理时，使用 **Proxyonly** 引用 **user-ca-bundle** 配置映射。使用 **Always** 始终引用 **user-ca-bundle** 配置映射。默认值为 **Proxyonly**。

注意

安装程序不支持代理的 **readinessEndpoints** 字段。

注意

如果安装程序超时，重启并使用安装程序的 **wait-for** 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. 保存该文件并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 cluster 的集群范围代理，该代理 **使用** 提供的 **install-config.yaml** 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 **cluster Proxy** 对象，但它会有一个空 **spec**。

注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

15.2.9.3. 配置三节点集群

另外，您可以在只由三台 control plane 机器组成的裸机集群中部署零台计算机器。这为集群管理员和开发人员提供了更小、效率更高的集群，用于测试、开发和生产。

在三节点 OpenShift Container Platform 环境中，三台 control plane 机器可以调度，这意味着应用程序工作负载被调度到它们上运行。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。

流程

- 确保 **install-config.yaml** 文件中的计算副本数量设置为 **0**，如以下 **计算** 小节所示：

```
compute:
- name: worker
  platform: {}
  replicas: 0
```

注意

在用户置备的基础架构上安装 OpenShift Container Platform 时，无论您要部署的计算机器数量有多少，您必须将计算机器的 **replicas** 参数值设置为 **0**。在安装程序置备的安装中，参数控制集群为您创建和管理的计算机器数量。这不适用于手动部署计算机器的用户置备安装。

对于三节点集群安装，请按照以下步骤执行：

- 如果要部署一个带有零计算节点的三节点集群，Ingress Controller Pod 在 control plane 节点上运行。在三节点集群部署中，您必须配置应用程序入口负载均衡器，将 HTTP 和 HTTPS 流量路由到 control plane 节点。如需更多信息，*请参阅用户置备的基础架构的负载平衡要求* 部分。

- 在以下步骤中创建 Kubernetes 清单文件时，请确保 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** 文件中的 **mastersSchedulable** 参数被设置为 **true**。这可让应用程序工作负载在 control plane 节点上运行。
- 在创建 Red Hat Enterprise Linux CoreOS(RHCOS)机器时，不要部署任何计算节点。

15.2.10. 创建 Kubernetes 清单和 Ignition 配置文件

由于您必须修改一些集群定义文件并手动启动集群机器，因此您必须生成 Kubernetes 清单和 Ignition 配置文件来配置机器。

安装配置文件转换为 Kubernetes 清单。清单嵌套到 Ignition 配置文件中，稍后用于配置集群机器。

重要

- OpenShift Container Platform 安装程序生成的 Ignition 配置文件包含 24 小时后过期的证书，然后在该时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 **node-bootstrapper** 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

先决条件

- 已获得 OpenShift Container Platform 安装程序。
- 已创建 **install-config.yaml** 安装配置文件。

流程

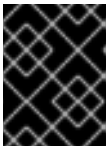
1. 进入包含 OpenShift Container Platform 安装程序的目录，并为集群生成 Kubernetes 清单：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定包含您创建的 **install-config.yaml** 文件的安装目录。

警告

如果您要安装一个三节点集群，请跳过以下步骤，以便可以调度 control plane 节点。



重要

当您将 control plane 节点从默认的不可调度配置为可以调度时，需要额外的订阅。这是因为 control plane 节点变为计算节点。

2. 检查 **<installation_directory>/manifests/cluster-scheduler-02-config.yml Kubernetes 清单文件中的 mastersSchedulable** 参数是否已设置为 **false**。此设置可防止在 control plane 机器上调度 pod：

   a. 打开 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** 文件。

   b. 找到 **mastersSchedulable** 参数，并确保它被设置为 **false**。

   c. 保存并退出 文件。

3. 要创建 Ignition 配置文件，请从包含安装程序的目录运行以下命令：

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定相同的安装目录。

为安装目录中的 bootstrap、control plane 和计算节点创建 Ignition 配置文件。**kubeadmin-password** 和 **kubeconfig** 文件在 **./<installation_directory>/auth** 目录中创建：

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

其他资源

- 如需有关 恢复 kubelet 证书的更多信息，请参阅恢复已过期的 control plane 证书。

15.2.11. 安装 RHCOS 并启动 OpenShift Container Platform bootstrap 过程

要在您置备的裸机基础架构上安装 OpenShift Container Platform，您必须在机器上安装 Red Hat Enterprise Linux CoreOS(RHCOS)。安装 RHCOS 时，您必须为您要安装的机器类型提供 OpenShift Container Platform 安装程序生成的 Ignition 配置文件。如果您配置了适当的网络、DNS 和负载均衡基础架构，OpenShift Container Platform bootstrap 过程会在 RHCOS 机器重启后自动启动。

要在机器上安装 RHCOS，请按照以下步骤使用 ISO 镜像或网络 PXE 引导。

注意

本安装文档中包括的计算节点部署步骤特定于 RHCOS。如果您选择部署基于 RHEL 的计算节点，您需要负责所有操作系统生命周期管理和维护，包括执行系统更新、应用补丁和完成所有其他必要的任务。仅支持 RHEL 8 计算机器。

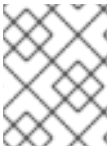
您可以使用以下方法在 ISO 和 PXE 安装过程中配置 RHCOS：

- 内核参数：您可以使用内核参数来提供特定于安装的信息。例如，您可以指定上传到 HTTP 服务器的 RHCOS 安装文件的位置以及您要安装的节点类型的 Ignition 配置文件的位置。对于 PXE 安装，您可以使用 **APPEND** 参数将参数传递给 live 安装程序的内核。对于 ISO 安装，您可以中断实时安装引导过程来添加内核参数。在这两个安装情形中，您可以使用特殊的 **coreos.inst.*** 参数来指示实时安装程序，以及标准安装引导参数来打开或关闭标准内核服务。

- Ignition 配置：OpenShift Container Platform Ignition 配置文件(***.ign**)特定于您要安装的节点类型。您可以在 RHCOS 安装过程中传递 bootstrap、control plane 或计算节点 Ignition 配置文件的位置，以便在首次启动时生效。特殊情况下，您可以创建单独的、有限的 Ignition 配置以传递给 live 系统。该 Ignition 配置可以执行特定的任务，如在安装完成后向置备系统报告成功。此特殊的 Ignition 配置由 **coreos-installer** 使用，以便在首次引导安装的系统时应用。不要直接为实时 ISO 提供标准 control plane 和计算节点 Ignition 配置。

- **coreos-installer** ：您可以将 live ISO 安装程序引导到 shell 提示符，这可让您在第一次引导前以多种方式准备持久性系统。特别是，您可以运行 **coreos-installer** 命令来识别要包含的各种工件、使用磁盘分区和设置网络。在某些情况下，您可以配置 live 系统上的功能并将其复制到安装的系统中。

使用 ISO 安装还是 PXE 安装取决于您的情况。PXE 安装需要可用的 DHCP 服务并进行更多准备，但可以使安装过程更加自动化。ISO 安装是一个更手动的过程，如果您设置的机器数超过几台，则可能不方便。



注意

从 OpenShift Container Platform 4.6 开始，RHCOS ISO 和其他安装工件支持在带有 4K 扇区的磁盘上安装。

15.2.11.1. 使用 ISO 镜像安装 RHCOS

您可以使用 ISO 镜像在机器上安装 RHCOS。

先决条件

- 已为集群创建 Ignition 配置文件。
- 您已配置了适当的网络、DNS 和负载平衡基础架构。
- 您有一个可以从计算机以及您创建的机器访问的 HTTP 服务器。
- 您已参阅 *Advanced RHCOS 安装配置* 部分来了解配置功能的不同方法，如网络和磁盘分区。

流程

1. 获取每个 Ignition 配置文件的 SHA512 摘要。例如，您可以在运行 Linux 的系统上使用以下内容来获取 **bootstrap.ign** Ignition 配置文件的 SHA512 摘要：

```
$ sha512sum <installation_directory>/bootstrap.ign
```

   后续步骤中会向 **coreos-installer** 提供摘要，以验证集群节点上 Ignition 配置文件的真实性。

2. 将安装程序创建的 bootstrap、control plane 和计算节点 Ignition 配置文件上传到 HTTP 服务器。注意这些文件的 URL。

重要

您可以在 Ignition 配置中添加或更改配置设置，然后将其保存到 HTTP 服务器。如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

3. 从安装主机上，验证 Ignition 配置文件是否在 URL 上可用。以下示例获取 bootstrap 节点的 Ignition 配置文件：

```
$ curl -k http://<HTTP_server>/bootstrap.ign 1
```

输出示例

```
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0{"ignition":
{"version":"3.2.0"},"passwd":{"users":[{"name":"core","sshAuthorizedKeys":["ssh-rsa...
```

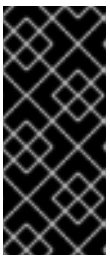
在命令中将 **bootstrap.ign** 替换为 **master.ign** 或 **worker.ign**，以验证 control plane 和计算节点的 Ignition 配置文件是否可用。

4. 虽然可以从 RHCOS 镜像镜像页面获取您选择的操作系统实例安装方法所需的 RHCOS 镜像，但推荐的方法是从 **openshift-install** 命令的输出获取 RHCOS 镜像的正确版本：

```
$ openshift-install coreos print-stream-json | grep '\.iso[^.]'
```

输出示例

```
"location": "<url>/art/storage/releases/rhcos-4.16-aarch64/<release>/aarch64/rhcos-
<release>-live.aarch64.iso",
"location": "<url>/art/storage/releases/rhcos-4.16-ppc64le/<release>/ppc64le/rhcos-
<release>-live.ppc64le.iso",
"location": "<url>/art/storage/releases/rhcos-4.16-s390x/<release>/s390x/rhcos-<release>-
live.s390x.iso",
"location": "<url>/art/storage/releases/rhcos-4.16/<release>/x86_64/rhcos-<release>-
live.x86_64.iso",
```



重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每个发行版本而改变。您必须下载最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。如果可用，请使用与 OpenShift Container Platform 版本匹配的镜像版本。这个过程只使用 ISO 镜像。此安装类型不支持 RHCOS qcow2 镜像。

ISO 文件名类似以下示例：

rhcos-<version>-live.<architecture>.iso

5. 使用 ISO 启动 RHCOS 安装。使用以下安装选项之一：

- 将 ISO 映像刻录到磁盘并直接启动。
- 使用 light-out 管理(LOM)接口使用 ISO 重定向。

6. 在不指定任何选项或中断实时引导序列的情况下引导 RHCOS ISO 镜像。等待安装程序在 RHCOS live 环境中引导进入 shell 提示符。

注意

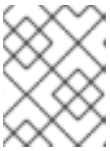
可以中断 RHCOS 安装引导过程来添加内核参数。但是，在这个 ISO 过程中，您应该使用以下步骤中所述的 **coreos-installer** 命令，而不是添加内核参数。

7. 运行 **coreos-installer** 命令并指定满足您的安装要求的选项。您至少必须指定指向节点类型的 Ignition 配置文件的 URL，以及您要安装到的设备：

```
$ sudo coreos-installer install --ignition-url=http://<HTTP_server>/<node_type>.ign <device>
--ignition-hash=sha512-<digest> 1 2
```

1 您必须使用 **sudo** 运行 **coreos-installer** 命令，因为 **core** 用户没有执行安装所需的 root 权限。

2 当 Ignition 配置文件通过 HTTP URL 获取时，需要 **--ignition-hash** 选项来验证集群节点上 Ignition 配置文件的真实性。**<digest>** 是上一步中获取的 Ignition 配置文件 SHA512 摘要。



注意

如果要通过使用 TLS 的 HTTPS 服务器提供 Ignition 配置文件，您可以在运行 **coreos-installer** 前将内部证书颁发机构(CA)添加到系统信任存储中。

以下示例将引导节点安装初始化到 **/dev/sda** 设备。bootstrap 节点的 Ignition 配置文件从 IP 地址 192.168.1.2 的 HTTP Web 服务器获取：

```
$ sudo coreos-installer install --ignition-
url=http://192.168.1.2:80/installation_directory/bootstrap.ign /dev/sda --ignition-hash=sha512-
a5a2d43879223273c9b60af66b44202a1d1248fc01cf156c46d4a79f552b6bad47bc8cc78ddf011
6e80c59d2ea9e32ba53bc807afbca581aa059311def2c3e3b
```

8. 在机器的控制台上监控 RHCOS 安装的进度。

重要

在开始安装 OpenShift Container Platform 之前，确保每个节点中安装成功。观察安装过程可以帮助确定可能会出现 RHCOS 安装问题的原因。

9. 安装 RHCOS 后，您必须重启系统。系统重启过程中，它会应用您指定的 Ignition 配置文件。

10. 检查控制台输出，以验证 Ignition 是否运行。

示例命令

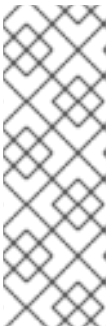
```
Ignition: ran on 2022/03/14 14:48:33 UTC (this boot)
Ignition: user-provided config was applied
```

11. 继续为集群创建其他机器。

重要

此时您必须创建 bootstrap 和 control plane 机器。如果 control plane 机器不可调度，请在安装 OpenShift Container Platform 前至少创建两台计算机器。

如果存在所需的网络、DNS 和负载均衡器基础架构，OpenShift Container Platform bootstrap 过程会在 RHCOS 节点重启后自动启动。



注意

RHCOS 节点不包含 **core** 用户的默认密码。您可以使用可访问 SSH 私钥的用户的身份运行 **ssh core@<node>.<cluster_name>.<base_domain>** 来访问节点，该私钥与您在 **install_config.yaml** 文件中指定的公钥配对。运行 RHCOS 的 OpenShift Container Platform 4 集群节点不可变，它依赖于 Operator 来应用集群更改。不建议使用 SSH 访问集群节点。但是，当调查安装问题时，如果 OpenShift Container Platform API 不可用，或者 kubelet 在目标节点上无法正常工作，则调试或灾难恢复可能需要 SSH 访问。

15.2.11.2. 使用 PXE 或 iPXE 启动安装 RHCOS

您可以使用 PXE 或 iPXE 启动在机器上安装 RHCOS。

先决条件

- 已为集群创建 Ignition 配置文件。
- 您已配置了适当的网络、DNS 和负载平衡基础架构。
- 您已配置了合适的 PXE 或 iPXE 基础架构。
- 您有一个可以从计算机以及您创建的机器访问的 HTTP 服务器。
- 您已参阅 *Advanced RHCOS 安装配置* 部分来了解配置功能的不同方法，如网络和磁盘分区。

流程

1. 将安装程序创建的 bootstrap、control plane 和计算节点 Ignition 配置文件上传到 HTTP 服务器。注意这些文件的 URL。

重要

您可以在 Ignition 配置中添加或更改配置设置，然后将其保存到 HTTP 服务器。如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

2. 从安装主机上，验证 Ignition 配置文件是否在 URL 上可用。以下示例获取 bootstrap 节点的 Ignition 配置文件：

```
$ curl -k http://<HTTP_server>/bootstrap.ign 1
```

输出示例

```
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
```

```
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0{"ignition":
{"version":"3.2.0"},"passwd":{"users":[{"name":"core","sshAuthorizedKeys":["ssh-rsa...
```

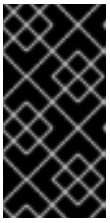
在命令中将 **bootstrap.ign** 替换为 **master.ign** 或 **worker.ign**，以验证 control plane 和计算节点的 Ignition 配置文件是否可用。

3. 虽然可以从 RHCOS image mirror 页面获取您选择的操作系统实例所需的 RHCOS **kernel**、**initramfs** 和 **rootfs** 文件，但推荐的方法是从 **openshift-install** 命令的输出中获取 RHCOS 文件的正确版本：

```
$ openshift-install coreos print-stream-json | grep -Eo '"https.*(kernel-|initramfs.|rootfs.)\w+
(\.img)?"'
```

输出示例

```
"<url>/art/storage/releases/rhcos-4.16-aarch64/<release>/aarch64/rhcos-<release>-live-
kernel-aarch64"
"<url>/art/storage/releases/rhcos-4.16-aarch64/<release>/aarch64/rhcos-<release>-live-
initramfs.aarch64.img"
"<url>/art/storage/releases/rhcos-4.16-aarch64/<release>/aarch64/rhcos-<release>-live-
rootfs.aarch64.img"
"<url>/art/storage/releases/rhcos-4.16-ppc64le/49.84.202110081256-0/ppc64le/rhcos-
<release>-live-kernel-ppc64le"
"<url>/art/storage/releases/rhcos-4.16-ppc64le/<release>/ppc64le/rhcos-<release>-live-
initramfs.ppc64le.img"
"<url>/art/storage/releases/rhcos-4.16-ppc64le/<release>/ppc64le/rhcos-<release>-live-
rootfs.ppc64le.img"
"<url>/art/storage/releases/rhcos-4.16-s390x/<release>/s390x/rhcos-<release>-live-kernel-
s390x"
"<url>/art/storage/releases/rhcos-4.16-s390x/<release>/s390x/rhcos-<release>-live-
initramfs.s390x.img"
"<url>/art/storage/releases/rhcos-4.16-s390x/<release>/s390x/rhcos-<release>-live-
rootfs.s390x.img"
"<url>/art/storage/releases/rhcos-4.16/<release>/x86_64/rhcos-<release>-live-kernel-
x86_64"
"<url>/art/storage/releases/rhcos-4.16/<release>/x86_64/rhcos-<release>-live-
initramfs.x86_64.img"
"<url>/art/storage/releases/rhcos-4.16/<release>/x86_64/rhcos-<release>-live-
rootfs.x86_64.img"
```



重要

RHCOS 工件可能不会随着 OpenShift Container Platform 的每个发行版本而改变。您必须下载最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。这个过程只使用下面描述的适当 **kernel**、**initram fs** 和 **rootfs** 工件。此安装类型不支持 RHCOS QCOW2 镜像。

文件名包含 OpenShift Container Platform 版本号。它们类似以下示例：

- **kernel**:**rhcos-<version>-live-kernel-<architecture>**
- **initramfs**: **rhcos-<version>-live-initramfs.<architecture>.img**
- **rootfs**: **rhcos-<version>-live-rootfs.<architecture>.img**

4. 将 **rootfs**、**kernel** 和 **initramfs** 文件上传到 HTTP 服务器。

重要

如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

5. 配置网络引导基础架构，以便在安装 RHCOS 后机器从本地磁盘启动。

6. 为 RHCOS 镜像配置 PXE 或 iPXE 安装并开始安装。
为环境修改以下示例菜单条目之一，并验证能否正确访问镜像和 Ignition 文件：

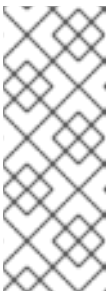
- 对于 PXE(**x86_64**)：

```
DEFAULT pxeboot
TIMEOUT 20
PROMPT 0
LABEL pxeboot
    KERNEL http://<HTTP_server>/rhcos-<version>-live-kernel-<architecture> (1)
    APPEND initrd=http://<HTTP_server>/rhcos-<version>-live-initramfs.
<architecture>.img coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-
rootfs.<architecture>.img coreos.inst.install_dev=/dev/sda
coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign (2) (3)
```

(1) (1) 指定上传到 HTTP 服务器的 live **kernel** 文件位置。URL 必须是 HTTP、TFTP 或 FTP；不支持 HTTPS 和 NFS。

(2) 如果您使用多个 NIC，请在 **ip** 选项中指定一个接口。例如，要在名为 **eno1** 的 NIC 上使用 DHCP，请设置 **ip=eno1:dhcp**。

(3) 指定上传到 HTTP 服务器的 RHCOS 文件的位置。**initrd** 参数值是 **initramfs** 文件的位置，**coreos.live.rootfs_url** 参数值是 **rootfs** 文件的位置，**coreos.inst.ignition_url** 参数值则是 bootstrap Ignition 配置文件的位置。您还可以在 **APPEND** 行中添加更多内核参数来配置联网或其他引导选项。



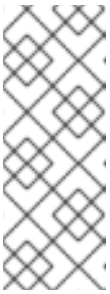
注意

此配置不会在图形控制台的机器上启用串行控制台访问。要配置不同的控制台，请在 **APPEND** 行中添加一个或多个 **console=** 参数。例如，添加 **console=tty0 console=ttyS0** 以将第一个 PC 串口设置为主控制台，并将图形控制台设置为二级控制台。如需更多信息，请参阅如何在 Red Hat Enterprise Linux 中设置串行终端和/或控制台？ 和"启用 PXE 和 ISO 安装的串行控制台"部分。

- 对于 iPXE (**x86_64** + **aarch64**)：

```
kernel http://<HTTP_server>/rhcos-<version>-live-kernel-<architecture> initrd=main
coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-rootfs.
<architecture>.img coreos.inst.install_dev=/dev/sda
coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign (1) (2)
initrd --name main http://<HTTP_server>/rhcos-<version>-live-initramfs.
<architecture>.img (3)
boot
```

1. 指定上传到 HTTP 服务器的 RHCOS 文件的位置。**kernel** 参数值是 **kernel** 文件的位置，init **rd=main** 参数用于在 UEFI 系统中引导， **coreos.live.rootfs_url** 参数值是
2. 如果您使用多个 NIC，请在 **ip** 选项中指定一个接口。例如，要在名为 **eno1** 的 NIC 上使用 DHCP，请设置 **ip=eno1:dhcp**。
3. 指定上传到 HTTP 服务器的 **initramfs** 文件的位置。



注意

此配置不会在图形控制台的机器上启用串行控制台访问。要配置不同的控制台，请在 **内核参数** 中添加一个或多个 **console=** 参数。例如，添加 **console=tty0 console=ttyS0** 以将第一个 PC 串口设置为主控制台，并将图形控制台设置为二级控制台。如需更多信息，请参阅如何在 Red Hat Enterprise Linux 中设置串行终端和/或控制台？ 和"启用 PXE 和 ISO 安装的串行控制台"部分。

注意

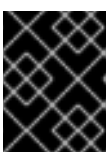
要在 **aarch64** 架构中网络引导 CoreOS **内核**，您需要使用启用了 **IMAGE_GZIP** 选项的 iPXE 构建版本。请参阅 iPXE 中的**IMAGE_GZIP** 选项。

- 对于 **aarch64** 中的 PXE（使用 UEFI 和 Grub 作为第二阶段）：

```
menuentry 'Install CoreOS' {
    linux rhcos-<version>-live-kernel-<architecture>
coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-rootfs.
<architecture>.img coreos.inst.install_dev=/dev/sda
coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign 1 2
    initrd rhcos-<version>-live-initramfs.<architecture>.img 3
}
```

1. 指定上传到 HTTP/TFTP 服务器的 RHCOS 文件的位置。**kernel** 参数值是 TFTP 服务器中的 **kernel** 文件的位置。**coreos.live.rootfs_url** 参数值是 **rootfs** 文件的位置，**coreos.inst.ignition_url** 参数值是 HTTP 服务器上的 bootstrap Ignition 配置文件的位置。
2. 如果您使用多个 NIC，请在 **ip** 选项中指定一个接口。例如，要在名为 **eno1** 的 NIC 上使用 DHCP，请设置 **ip=eno1:dhcp**。
3. 指定上传到 TFTP 服务器的 **initramfs** 文件的位置。

7. 在机器的控制台上监控 RHCOS 安装的进度。



重要

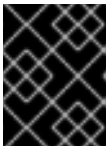
在开始安装 OpenShift Container Platform 之前，确保每个节点中安装成功。观察安装过程可以帮助确定可能会出现 RHCOS 安装问题的原因。

8. 安装 RHCOS 后，系统会重启。在重启过程中，系统会应用您指定的 Ignition 配置文件。
9. 检查控制台输出，以验证 Ignition 是否运行。

示例命令

```
Ignition: ran on 2022/03/14 14:48:33 UTC (this boot)
Ignition: user-provided config was applied
```

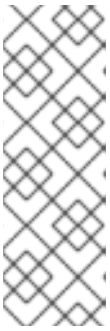
10. 继续为集群创建机器。



重要

此时您必须创建 bootstrap 和 control plane 机器。如果 control plane 机器不可调度，请在安装集群前至少创建两台计算机器。

如果存在所需的网络、DNS 和负载均衡器基础架构，OpenShift Container Platform bootstrap 过程会在 RHCOS 节点重启后自动启动。



注意

RHCOS 节点不包含 **core** 用户的默认密码。您可以使用可访问 SSH 私钥的用户的身份运行 **ssh core@<node>.<cluster_name>.<base_domain>** 来访问节点，该私钥与您在 **install_config.yaml** 文件中指定的公钥配对。运行 RHCOS 的 OpenShift Container Platform 4 集群节点不可变，它依赖于 Operator 来应用集群更改。不建议使用 SSH 访问集群节点。但是，当调查安装问题时，如果 OpenShift Container Platform API 不可用，或者 kubelet 在目标节点上无法正常工作，则调试或灾难恢复可能需要 SSH 访问。

15.2.11.3. 高级 RHCOS 安装配置

为 OpenShift Container Platform 手动置备 Red Hat Enterprise Linux CoreOS(RHCOS)节点的一个关键优点是能够进行通过默认的 OpenShift Container Platform 安装方法无法进行的配置。本节介绍了您可以使用的一些技术进行配置，其中包括：

- 将内核参数传递给实时安装程序
- 从 live 系统手动运行 **coreos-installer**
- 自定义实时 ISO 或 PXE 引导镜像

本节详述了与 Red Hat Enterprise Linux CoreOS(RHCOS)手动安装的高级配置相关的内容，如磁盘分区、网络以及使用 Ignition 配置的不同方式相关。

15.2.11.3.1. 使用高级网络选项进行 PXE 和 ISO 安装

OpenShift Container Platform 节点的网络默认使用 DHCP 来收集所有必要的配置设置。要设置静态 IP 地址或配置特殊设置，如绑定，您可以执行以下操作之一：

- 引导 live 安装程序时传递特殊内核参数。
- 使用机器配置将网络文件复制到安装的系统中。
- 从 live 安装程序 shell 提示符配置网络，然后将这些设置复制到安装的系统上，以便在安装的系统第一次引导时生效。

要配置 PXE 或 iPXE 安装，请使用以下选项之一：

- 请参阅"高级 RHCOS 安装参考"表。
- 使用机器配置将网络文件复制到安装的系统中。

要配置 ISO 安装，请使用以下步骤：

流程

1. 引导 ISO 安装程序.
2. 在 live 系统 shell 提示符下，使用可用的 RHEL 工具（如 **nmcli** 或 **nmtui** ）为 live 系统配置网络。
3. 运行 **coreos-installer** 命令以安装系统，添加 **--copy-network** 选项来复制网络配置。例如：

```
$ sudo coreos-installer install --copy-network \
    --ignition-url=http://host/worker.ign /dev/disk/by-id/scsi-<serial_number>
```

重要

copy-network 选项仅复制在 **/etc/NetworkManager/system-connections** 下找到的网络配置。特别是，它不会复制系统主机名。

4. 重启安装的系统。

其他资源

- 有关 **nmcli** 和 **nmtui** 工具的更多信息，请参阅 RHEL 8 文档中的 Getting started with nmcli and Getting started with nmtui。

15.2.11.3.2. 磁盘分区

磁盘分区是在 Red Hat Enterprise Linux CoreOS(RHCOS)安装过程中在 OpenShift Container Platform 集群节点上创建的。特定架构的每个 RHCOS 节点使用相同的分区布局，除非覆盖默认的分区配置。在 RHCOS 安装过程中，根文件系统的大小会增加，以使用目标设备中剩余的可用空间。

重要

在节点上使用自定义分区方案可能会导致 OpenShift Container Platform 在某些节点分区上监控或警报。如果要覆盖默认分区，请参阅 了解 OpenShift 文件系统监控（驱除条件）以了解有关 OpenShift Container Platform 如何监控主机文件系统的更多信息。

OpenShift Container Platform 监控以下两个文件系统标识符：

- **nodefs**，这是包含 **/var/lib/kubelet** 的文件系统
- **imagefs**，这是包含 **/var/lib/containers** 的文件系统

对于默认分区方案，**nodefs** 和 **imagefs** 监控相同的根文件系统 /。

要在 OpenShift Container Platform 集群节点上安装 RHCOS 时覆盖默认分区，您必须创建单独的分区。您可能想要为容器和容器镜像添加单独的存储分区。例如，通过在独立分区中挂载 **/var/lib/containers**，kubelet 会单独监控 **/var/lib/containers** 作为 **imagefs** 目录，以及 root 文件系统作为 **nodefs** 目录。

重要

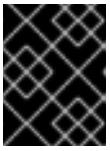
如果您已将磁盘大小调整为托管更大的文件系统，请考虑创建单独的 **/var/lib/containers** 分区。考虑重新定义具有 **xfs** 格式的磁盘大小，以减少大量分配组导致的 CPU 时间问题。

15.2.11.3.2.1. 创建独立 /var 分区

通常，您应该使用在 RHCOS 安装过程中创建的默认磁盘分区。然而，在有些情况下您可能需要为预期增长的目录创建独立分区。

OpenShift Container Platform 支持添加单个分区将存储附加到 **/var** 目录或 **/var** 的子目录中。例如：

- **/var/lib/containers** ：保存随着系统中添加更多镜像和容器而增长的容器相关内容。
- **/var/lib/etcd** ：保存您可能希望独立保留的数据，比如 etcd 存储的性能优化。
- **/var** ：保存您可能希望独立保留的数据，以满足审计等目的。



重要

对于大于 100GB 的磁盘大小，特别是磁盘大小大于 1TB，请创建一个独立的 **/var** 分区。

通过单独存储 **/var** 目录的内容，可以更轻松地根据需要为区域扩展存储，并在以后重新安装 OpenShift Container Platform，并保持该数据的完整性。使用这个方法，您不必再次拉取所有容器，在更新系统时也不必复制大量日志文件。

将独立分区用于 **/var** 目录或 **/var** 的子目录也会防止分区目录中的数据增加填充根文件系统。

以下流程通过添加机器配置清单来设置独立的 **/var** 分区，该清单会在安装准备阶段封装到节点类型的 Ignition 配置文件中。

流程

1. 在安装主机上，切换到包含 OpenShift Container Platform 安装程序的目录，并为集群生成 Kubernetes 清单：

```
$ openshift-install create manifests --dir <installation_directory>
```

2. 创建用于配置额外分区的 Butane 配置。例如，将文件命名为 **$HOME/clusterconfig/98-var-partition.bu**，将磁盘设备名称改为 **worker** 系统上存储设备的名称，并根据情况设置存储大小。这个示例将 **/var** 目录放在一个单独的分区中：

```
variant: openshift
version: 4.16.0
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
storage:
  disks:
  - device: /dev/disk/by-id/<device_name> 1
    partitions:
    - label: var
      start_mib: <partition_start_offset> 2
```

```
    size_mib: <partition_size> 3
    number: 5
filesystems:
  - device: /dev/disk/by-partlabel/var
    path: /var
    format: xfs
    mount_options: [defaults, prjquota] 4
    with_mount_unit: true
```

1 要分区的磁盘的存储设备名称。

2 当在引导磁盘中添加数据分区时，推荐最少使用偏移值 25000 兆字节。root 文件系统会自动调整大小以填充所有可用空间（最多到指定的偏移值）。如果没有指定偏移值，或者指定的值小于推荐的最小值，则生成的 root 文件系统会太小，而在以后进行的 RHCOS 重新安装可能会覆盖数据分区的开始部分。

3 以兆字节为单位的数据分区大小。

4 对于用于容器存储的文件系统，必须启用 **prjquota** 挂载选项。

注意

在创建单独的 **/var** 分区时，如果不同的实例类型没有相同的设备名称，则无法将不同的实例类型用于计算节点。

3. 从 Butane 配置创建一个清单，并将它保存到 **clusterconfig/openshift** 目录中。例如，运行以下命令：

```
$ butane $HOME/clusterconfig/98-var-partition.bu -o $HOME/clusterconfig/openshift/98-var-partition.yaml
```

4. 创建 Ignition 配置文件：

```
$ openshift-install create ignition-configs --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定相同的安装目录。

为安装目录中的 bootstrap、control plane 和计算节点创建 Ignition 配置文件：

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

<installation_directory>/manifest 和 **<installation_directory>/openshift** 目录中的文件被嵌套到 Ignition 配置文件中，包括包含 **98-var-partition** 自定义 **MachineConfig** 对象的文件。

后续步骤

- 您可以通过在 RHCOS 安装过程中引用 Ignition 配置文件来应用自定义磁盘分区。

15.2.11.3.2.2. 保留现有分区

对于 ISO 安装，您可以在 **coreos-installer** 命令中添加可让安装程序维护一个或多个现有分区的选项。对于 PXE 安装，您可以在 **APPEND** 参数中添加 **coreos.inst.*** 选项来保留分区。

保存的分区可能是来自现有 OpenShift Container Platform 系统的数据分区。您可以通过分区标签或编号识别您要保留的磁盘分区。

注意

如果您保存了现有分区，且这些分区没有为 RHCOS 留下足够空间，则安装将失败，而不影响保存的分区。

在 ISO 安装过程中保留现有分区

这个示例保留分区标签以 **数据开头的任何分区(data *)**：

```
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign \
      --save-partlabel 'data*' /dev/disk/by-id/scsi-<serial_number>
```

以下示例演示了在运行 **coreos-installer** 时要保留磁盘上的第 6 个分区：

```
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign \
      --save-partindex 6 /dev/disk/by-id/scsi-<serial_number>
```

这个示例保留分区 5 及更高分区：

```
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign
      --save-partindex 5- /dev/disk/by-id/scsi-<serial_number>
```

在前面已保存分区的示例中，**coreos-installer** 会立即重新创建分区。

在 PXE 安装过程中保留现有分区

这个 **APPEND** 选项保留分区标签以 'data'('data*')开头的任何分区：

```
coreos.inst.save_partlabel=data*
```

这个 **APPEND** 选项保留分区 5 及更高分区：

```
coreos.inst.save_partindex=5-
```

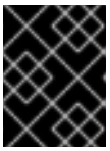
这个 **APPEND** 选项保留分区 6:

```
coreos.inst.save_partindex=6
```

15.2.11.3.3. 识别 Ignition 配置

在进行 RHCOS 手动安装时，您可以提供两种 Ignition 配置类型，它们有不同的原因：

- **永久安装 Ignition 配置** ：每个手动 RHCOS 安装都需要传递 **openshift-installer** 生成的 Ignition 配置文件之一，如 **bootstrap.ign**、**master.ign** 和 **worker.ign**，才能进行安装。



重要

不建议直接修改这些 Ignition 配置文件。您可以更新嵌套到 Ignition 配置文件中的清单文件，如上一节示例中所述。

对于 PXE 安装，您可以使用 **coreos.inst.ignition_url=** 选项在 **APPEND** 行上传递 Ignition 配置。对于 ISO 安装，在 ISO 引导至 shell 提示符后，您可以使用带有 **--ignition-url=** 选项的 **coreos-installer** 命令行。在这两种情况下，只支持 HTTP 和 HTTPS 协议。

- 实时**安装 Ignition 配置** ：可使用 **coreos-installer customize** 子命令及其各种选项来创建此类型。使用此方法，Ignition 配置会传递到 live 安装介质，在引导时立即运行，并在 RHCOS 系统安装到磁盘之前或之后执行设置任务。这个方法只用于必须执行一次且之后不能再次应用的任务，比如不能使用机器配置进行的高级分区。
  对于 PXE 或 ISO 引导，您可以创建 Ignition 配置，**APP** END **ignition.config.url=** 选项来标识 Ignition 配置的位置。您还需要附加 **ignition.firstboot ignition.platform.id=metal** 或 **ignition.config.url** 选项。

15.2.11.3.4. 默认控制台配置

从 OpenShift Container Platform 4.16 引导镜像安装的 Red Hat Enterprise Linux CoreOS (RHCOS) 节点使用默认控制台，旨在识别大多数虚拟化和裸机设置。不同的云和虚拟化平台可能会根据所选的架构使用不同的默认设置。裸机安装使用内核默认设置，这通常意味着图形控制台是主控制台，并且禁用串行控制台。

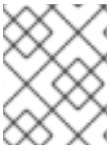
默认控制台可能与特定的硬件配置不匹配，或者您可能具有需要调整默认控制台的特定需求。例如：

- 您希望访问控制台上的紧急 shell 进行调试。
- 您的云平台没有提供到图形控制台的互动访问，但提供了一个串行控制台。
- 您需要启用多个控制台。

控制台配置继承自引导镜像。这意味着现有集群中的新节点不受默认控制台的影响。

您可以使用以下方法为裸机安装配置控制台：

- 在命令行中手动使用 **coreos-installer**。
- 使用带有 **--dest-console** 选项的 **coreos-installer iso customize** 或 **coreos-installer pxe customize** 子命令，以创建可自动执行进程的自定义镜像。



注意

对于高级自定义，请使用 **coreos-installer iso** 或 **coreos-installer pxe** 子命令而不是内核参数来执行控制台配置。

15.2.11.3.5. 为 PXE 和 ISO 安装启用串行控制台

默认情况下，Red Hat Enterprise Linux CoreOS (RHCOS) 串行控制台被禁用，所有输出都会写入图形控制台。您可以为 ISO 安装启用串行控制台并重新配置引导装载程序，以便输出同时发送到串行控制台和图形控制台。

流程

1. 引导 ISO 安装程序.
2. 运行 **coreos-installer** 命令来安装系统，添加 **--console** 选项一次来指定图形控制台，然后第二次指定串行控制台：

```
$ coreos-installer install \
  --console=tty0 \ 1
  --console=ttyS0,<options> \ 2
  --ignition-url=http://host/worker.ign /dev/disk/by-id/scsi-<serial_number>
```

1. 所需的二级控制台。在这种情况下，是图形控制台。省略这个选项将禁用图形控制台。
2. 所需的主控制台。在这种情况下，是串行控制台。**options** 字段定义 baud 速率和其他设置。此字段的一个常见值为 **11520n8**。如果没有提供选项，则使用默认内核值 **9600n8**。有关这个选项格式的更多信息，请参阅 Linux 内核串口控制台 文档。

3. 重启安装的系统。

注意

可以使用 **coreos-installer install --append-karg** 选项来获取类似的结果，并使用 **console=** 指定控制台。但是，这只会为内核设置控制台，而不为引导装载程序设置控制台。

要配置 PXE 安装，请确保省略 **coreos.inst.install_dev** 内核命令行选项，并使用 shell 提示符使用上述 ISO 安装过程手动运行 **coreos-installer**。

15.2.11.3.6. 自定义 live RHCOS ISO 或 PXE 安装

您可以通过将 Ignition 配置文件直接注入镜像中来使用 live ISO 镜像或 PXE 环境来安装 RHCOS。这会创建一个自定义镜像，供您用来置备系统。

对于 ISO 镜像，此操作的机制是 **coreos-installer iso custom** 子命令，它使用您的配置修改 **.iso** 文件。同样，PXE 环境的机制是 **coreos-installer pxe customize** 子命令，它会创建一个包含自定义的新 **initramfs** 文件。

custom 子命令是一个通用工具，也可以嵌入其他类型的自定义。以下任务是一些更常见的自定义示例：

- 当公司安全策略需要使用时，注入自定义的 CA 证书。
- 在不需要内核参数的情况下配置网络设置。
- 嵌入任意预安装和安装后脚本或二进制文件。

15.2.11.3.7. 自定义 live RHCOS ISO 镜像

您可以使用 **coreos-installer iso custom** 子命令直接自定义 live RHCOS ISO 镜像。当您引导 ISO 镜像时，会自动应用自定义。

您可以使用此功能配置 ISO 镜像来自动安装 RHCOS。

流程

1. 从 **coreos-installer** 镜像镜像页面下载 **coreos-installer** 二进制文件。

2. 从 RHCOS 镜像镜像页面和 Ignition 配置文件检索 RHCOS ISO 镜像，然后运行以下命令来直接将 Ignition 配置注入 ISO 镜像：

```
$ coreos-installer iso customize rhcos-<version>-live.x86_64.iso \
  --dest-ignition bootstrap.ign \ 1
  --dest-device /dev/disk/by-id/scsi-<serial_number> 2
```

1. 从 **openshift-installer** 安装程序生成的 Ignition 配置文件。

2. 当您指定这个选项时，ISO 镜像会自动运行安装。否则，镜像为安装配置，但不会自动安装，除非您指定了 **coreos.inst.install_dev** 内核参数。

3. 可选：要删除 ISO 镜像自定义并将镜像返回到其 pristine 状态，请运行：

```
$ coreos-installer iso reset rhcos-<version>-live.x86_64.iso
```

现在，您可以重新自定义 live ISO 镜像，或者在其 pristine 状态中使用它。

应用您的自定义会影响每个后续 RHCOS 引导。

15.2.11.3.7.1. 修改实时安装 ISO 镜像以启用串行控制台

在使用 OpenShift Container Platform 4.12 及更高版本安装的集群中，串行控制台默认被禁用，所有输出都会写入图形控制台。您可以按照以下流程启用串行控制台。

流程

1. 从 **coreos-installer** 镜像镜像页面下载 **coreos-installer** 二进制文件。

2. 从 RHCOS image mirror 页中获取 RHCOS ISO 镜像，并运行以下命令来自定义 ISO 镜像，使串行控制台能够接收输出：

```
$ coreos-installer iso customize rhcos-<version>-live.x86_64.iso \
  --dest-ignition <path> \ 1
  --dest-console tty0 \ 2
  --dest-console ttyS0,<options> \ 3
  --dest-device /dev/disk/by-id/scsi-<serial_number> 4
```

1. 要安装 Ignition 配置的位置。

2. 所需的二级控制台。在这种情况下，是图形控制台。省略这个选项将禁用图形控制台。

3. 所需的主控制台。在这种情况下，是串行控制台。**options** 字段定义 baud 速率和其他设置。此字段的一个常见值为 **115200n8**。如果没有提供选项，则使用默认内核值 **9600n8**。有关这个选项格式的更多信息，请参阅 Linux 内核串口控制台 文档。

4. 要安装到的指定磁盘。如果省略这个选项，ISO 镜像会自动运行安装程序，除非还指定了 **coreos.inst.install_dev** 内核参数。

注意

--dest-console 选项会影响安装的系统，而不是实时 ISO 系统。要修改 live ISO 系统的控制台，请使用 **--live-karg-append** 选项并使用 **console=** 指定控制台。

自定义会被应用，并影响每个后续 ISO 镜像引导。

3. 可选：要删除 ISO 镜像自定义并将镜像返回到其原始状态，请运行以下命令：

```
$ coreos-installer iso reset rhcos-<version>-live.x86_64.iso
```

现在，您可以重新自定义 live ISO 镜像，或者在其原始状态中使用它。

15.2.11.3.7.2. 修改实时安装 ISO 镜像以使用自定义证书颁发机构

您可以使用 **custom** 子命令的 **--ignition-ca** 标志向 Ignition 提供证书颁发机构(CA)证书。您可以在安装过程中使用 CA 证书，并在置备安装的系统时使用 CA 证书。

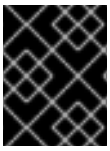
注意

自定义 CA 证书会影响 Ignition 获取远程资源的方式，但它们不会影响安装到系统中的证书。

流程

1. 从 **coreos-installer** 镜像镜像页面下载 **coreos-installer** 二进制文件。
2. 从 RHCOS 镜像镜像页面检索 RHCOS ISO 镜像，并运行以下命令来自定义 ISO 镜像以用于自定义 CA：

```
$ coreos-installer iso customize rhcos-<version>-live.x86_64.iso --ignition-ca cert.pem
```



重要

coreos.inst.ignition_url 内核参数无法使用 **--ignition-ca** 标志。您必须使用 **--dest-ignition** 标志为每个集群创建自定义镜像。

应用自定义 CA 证书会影响每个后续 RHCOS 引导。

15.2.11.3.7.3. 使用自定义网络设置修改实时安装 ISO 镜像

您可以将 NetworkManager 密钥文件嵌入到 live ISO 镜像中，并使用 **customize** 子命令的 **--network-keyfile** 标志将其传递给安装的系统。

警告

在创建连接配置文件时，您必须在连接配置文件的文件名中使用 **.nmconnection** 文件名扩展名。如果不使用 **.nmconnection** 文件名扩展，集群会将连接配置集应用到 live 环境，但它不会在集群首次启动节点时应用配置，从而导致无法正常工作的设置。

流程

1. 从 **coreos-installer** 镜像镜像页面下载 **coreos-installer** 二进制文件。
2. 为绑定接口创建连接配置集。例如，在本地目录中创建 **bond0.nmconnection** 文件，其内容如下：

```
[connection]
id=bond0
type=bond
interface-name=bond0
multi-connect=1
permissions=

[ethernet]
mac-address-blacklist=

[bond]
miimon=100
mode=active-backup

[ipv4]
method=auto

[ipv6]
method=auto

[proxy]
```

3. 为二级接口创建连接配置集，以添加到绑定中。例如，在本地目录中创建 **bond0-proxy-em1.nmconnection** 文件，其内容如下：

```
[connection]
id=em1
type=ethernet
interface-name=em1
master=bond0
multi-connect=1
permissions=
slave-type=bond

[ethernet]
mac-address-blacklist=
```

4. 为二级接口创建连接配置集，以添加到绑定中。例如，在本地目录中创建 **bond0-proxy-em2.nmconnection** 文件，其内容如下：

```
[connection]
id=em2
type=ethernet
interface-name=em2
master=bond0
multi-connect=1
permissions=
slave-type=bond

[ethernet]
mac-address-blacklist=
```

5. 从 RHCOS 镜像镜像页面检索 RHCOS ISO 镜像，并运行以下命令来使用您配置网络自定义 ISO 镜像：

```
$ coreos-installer iso customize rhcos-<version>-live.x86_64.iso \
    --network-keyfile bond0.nmconnection \
    --network-keyfile bond0-proxy-em1.nmconnection \
    --network-keyfile bond0-proxy-em2.nmconnection
```

网络设置应用于实时系统，并传输到目标系统。

15.2.11.3.7.4. 为 iSCSI 引导设备自定义实时安装 ISO 镜像

您可以设置 iSCSI 目标和 initiator 值，以便使用自定义的 live RHCOS 镜像版本自动挂载、引导和配置。

先决条件

1. 您有一个 iSCSI 目标，您要在其中安装 RHCOS。

流程

1. 从 **coreos-installer** 镜像镜像页面下载 **coreos-installer** 二进制文件。
2. 从 RHCOS 镜像镜像页面检索 RHCOS ISO 镜像，并运行以下命令使用以下信息自定义 ISO 镜像：

```
$ coreos-installer iso customize \
    --pre-install mount-iscsi.sh \ 1
    --post-install unmount-iscsi.sh \ 2
    --dest-device /dev/disk/by-path/<IP_address>:<port>-iscsi-<target_iqn>-lun-<lun> \ 3
    --dest-ignition config.ign \ 4
    --dest-karg-append rd.iscsi.initiator=<initiator_iqn> \ 5
    --dest-karg-append netroot=<target_iqn> \ 6
    -o custom.iso rhcos-<version>-live.x86_64.iso
```

1. 安装之前运行的脚本。它应包含用于挂载 iSCSI 目标以及启用多路径的任何命令的 **iscsiadm** 命令。
2. 安装后运行的脚本。它应包含命令 **iscsiadm --mode node --logout=all**。

3. 目标系统的位置。您必须提供目标门户的 IP 地址、关联的端口号、目标 iSCSI 节点采用 IQN 格式，以及 iSCSI 逻辑单元号(LUN)。
4. 目标系统的 Ignition 配置。
5. iSCSI 启动器或客户端，以 IQN 格式的名称。启动器构成连接到 iSCSI 目标的一个会话。
6. iSCSI 目标或服务器的名称（IQN 格式）。

有关 **dracut** 支持的 iSCSI 选项的更多信息，请参阅 **dracut.cmdline** 手册页。

15.2.11.3.7.5. 使用 iBFT 为 iSCSI 引导设备自定义实时安装 ISO 镜像

您可以设置 iSCSI 目标和 initiator 值，以便使用自定义的 live RHCOS 镜像版本自动挂载、引导和配置。

先决条件

1. 您有一个 iSCSI 目标，您要在其中安装 RHCOS。
2. 可选：有多路径 iSCSI 目标。

流程

1. 从 **coreos-installer** 镜像镜像页面下载 **coreos-installer** 二进制文件。
2. 从 RHCOS 镜像镜像页面检索 RHCOS ISO 镜像，并运行以下命令使用以下信息自定义 ISO 镜像：

```
$ coreos-installer iso customize \
    --pre-install mount-iscsi.sh \ 1
    --post-install unmount-iscsi.sh \ 2
    --dest-device /dev/mapper/mpatha \ 3
    --dest-ignition config.ign \ 4
    --dest-karg-append rd.iscsi.firmware=1 \ 5
    --dest-karg-append rd.multipath=default \ 6
    -o custom.iso rhcos-<version>-live.x86_64.iso
```

1. 安装之前运行的脚本。它应包含用于挂载 iSCSI 目标以及启用多路径的任何命令的 **iscsiadm** 命令。
2. 安装后运行的脚本。它应包含命令 **iscsiadm --mode node --logout=all**。
3. 设备的路径。如果连接了多个多路径设备，或者需要明确指定，您使用多路径（**/dev/mapper/mpatha**），您可以使用 **/dev/disk/by-path** 中可用的 World Wide Name (WWN) 符号链接。
4. 目标系统的 Ignition 配置。
5. iSCSI 参数从 BIOS 固件读取。
6. 可选：如果您要启用多路径，请包含此参数。

有关 **dracut** 支持的 iSCSI 选项的更多信息，请参阅 **dracut.cmdline** 手册页。

15.2.11.3.8. 自定义 live RHCOS PXE 环境

您可以使用 **coreos-installer pxe customize** 子命令直接自定义 live RHCOS PXE 环境。当您引导 PXE 环境时，会自动应用自定义。

您可以使用此功能配置 PXE 环境来自动安装 RHCOS。

流程

1. 从 **coreos-installer** 镜像镜像页面下载 **coreos-installer** 二进制文件。
2. 从 RHCOS 镜像镜像页面和 Ignition 配置文件获取 RHCOS **kernel**, **initramfs** 和 **rootfs** 文件，然后运行以下命令创建一个包含 Ignition 配置中的自定义的新 **initramfs** 文件：

```
$ coreos-installer pxe customize rhcos-<version>-live-initramfs.x86_64.img \
  --dest-ignition bootstrap.ign \ 1
  --dest-device /dev/disk/by-id/scsi-<serial_number> \ 2
  -o rhcos-<version>-custom-initramfs.x86_64.img 3
```

1. 从 **openshift-installer** 生成的 Ignition 配置文件。
2. 当您指定这个选项时，PXE 环境会自动运行安装。否则，为安装配置了镜像，除非指定了 **coreos.inst.install_dev** 内核参数，否则不会自动这样做。
3. 在 PXE 配置中使用自定义 **initramfs** 文件。添加 **ignition.firstboot** 和 **ignition.platform.id=metal** 内核参数（如果它们尚不存在）。

应用您的自定义会影响每个后续 RHCOS 引导。

15.2.11.3.8.1. 修改实时安装 PXE 环境以启用串行控制台。

在使用 OpenShift Container Platform 4.12 及更高版本安装的集群中，串行控制台默认被禁用，所有输出都会写入图形控制台。您可以按照以下流程启用串行控制台。

流程

1. 从 **coreos-installer** 镜像镜像页面下载 **coreos-installer** 二进制文件。
2. 从 RHCOS image mirror 页面获取 **kernel**, **initramfs** 和 **rootfs** 文件，然后运行以下命令来创建新的自定义 **initramfs** 文件，以便串行控制台接收输出：

```
$ coreos-installer pxe customize rhcos-<version>-live-initramfs.x86_64.img \
  --dest-ignition <path> \ 1
  --dest-console tty0 \ 2
  --dest-console ttyS0,<options> \ 3
  --dest-device /dev/disk/by-id/scsi-<serial_number> \ 4
  -o rhcos-<version>-custom-initramfs.x86_64.img 5
```

1. 要安装 Ignition 配置的位置。
2. 所需的二级控制台。在这种情况下，是图形控制台。省略这个选项将禁用图形控制台。
3.

所需的主控制台。在这种情况下，是串行控制台。**options** 字段定义 baud 速率和其他设置。此字段的一个常见值为 **115200n8**。如果没有提供选项，则使用默认内核值 **9600n8**。有

4 要安装到的指定磁盘。如果省略这个选项，PXE 环境会自动运行安装程序，除非还指定了 **coreos.inst.install_dev** 内核参数。

5 在 PXE 配置中使用自定义 **initramfs** 文件。添加 **ignition.firstboot** 和 **ignition.platform.id=metal** 内核参数（如果它们尚不存在）。

自定义会被应用，并影响 PXE 环境的每个后续引导。

15.2.11.3.8.2. 修改实时安装 PXE 环境以使用自定义证书颁发机构

您可以使用 **custom** 子命令的 **--ignition-ca** 标志向 Ignition 提供证书颁发机构(CA)证书。您可以在安装过程中使用 CA 证书，并在置备安装的系统时使用 CA 证书。

注意

自定义 CA 证书会影响 Ignition 获取远程资源的方式，但它们不会影响安装到系统中的证书。

流程

1. 从 **coreos-installer** 镜像镜像页面下载 **coreos-installer** 二进制文件。
2. 从 RHCOS 镜像镜像页面获取 RHCOS **kernel**、**initramfs** 和 **rootfs** 文件，并运行以下命令来创建一个新的自定义 **initramfs** 文件以用于自定义 CA：

```
$ coreos-installer pxe customize rhcos-<version>-live-initramfs.x86_64.img \
    --ignition-ca cert.pem \
    -o rhcos-<version>-custom-initramfs.x86_64.img
```

3. 在 PXE 配置中使用自定义 **initramfs** 文件。添加 **ignition.firstboot** 和 **ignition.platform.id=metal** 内核参数（如果它们尚不存在）。

重要

coreos.inst.ignition_url 内核参数无法使用 **--ignition-ca** 标志。您必须使用 **--dest-ignition** 标志为每个集群创建自定义镜像。

应用自定义 CA 证书会影响每个后续 RHCOS 引导。

15.2.11.3.8.3. 使用自定义网络设置修改实时安装 PXE 环境

您可以将 NetworkManager 密钥文件嵌入到 live PXE 环境中，并使用 **customize** 子命令的 **--network-keyfile** 标志将其传递给安装的系统。

警告

在创建连接配置文件时，您必须在连接配置文件的文件名中使用 **.nmconnection** 文件名扩展名。如果不使用 **.nmconnection** 文件名扩展，集群会将连接配置集应用到 live 环境，但它不会在集群首次启动节点时应用配置，从而导致无法正常工作的设置。

流程

1. 从 **coreos-installer** 镜像镜像页面下载 **coreos-installer** 二进制文件。
2. 为绑定接口创建连接配置集。例如，在本地目录中创建 **bond0.nmconnection** 文件，其内容如下：

```
[connection]
id=bond0
type=bond
interface-name=bond0
multi-connect=1
permissions=

[ethernet]
mac-address-blacklist=

[bond]
miimon=100
mode=active-backup

[ipv4]
method=auto

[ipv6]
method=auto

[proxy]
```

3. 为二级接口创建连接配置集，以添加到绑定中。例如，在本地目录中创建 **bond0-proxy-em1.nmconnection** 文件，其内容如下：

```
[connection]
id=em1
type=ethernet
interface-name=em1
master=bond0
multi-connect=1
permissions=
slave-type=bond

[ethernet]
mac-address-blacklist=
```

4. 为二级接口创建连接配置集，以添加到绑定中。例如，在本地目录中创建 **bond0-proxy-em2.nmconnection** 文件，其内容如下：

```
[connection]
id=em2
type=ethernet
interface-name=em2
master=bond0
multi-connect=1
permissions=
slave-type=bond

[ethernet]
mac-address-blacklist=
```

5. 从 RHCOS 镜像镜像页面获取 RHCOS **kernel**、**initramfs** 和 **rootfs** 文件，并运行以下命令来创建一个新的自定义 **initramfs** 文件，它包括您的配置网络：

```
$ coreos-installer pxe customize rhcos-<version>-live-initramfs.x86_64.img \
    --network-keyfile bond0.nmconnection \
    --network-keyfile bond0-proxy-em1.nmconnection \
    --network-keyfile bond0-proxy-em2.nmconnection \
    -o rhcos-<version>-custom-initramfs.x86_64.img
```

6. 在 PXE 配置中使用自定义 **initramfs** 文件。添加 **ignition.firstboot** 和 **ignition.platform.id=metal** 内核参数（如果它们尚不存在）。
网络设置应用于实时系统，并传输到目标系统。

15.2.11.3.8.4. 为 iSCSI 引导设备自定义实时安装 PXE 环境

您可以设置 iSCSI 目标和 initiator 值，以便使用自定义的 live RHCOS 镜像版本自动挂载、引导和配置。

先决条件

1. 您有一个 iSCSI 目标，您要在其中安装 RHCOS。

流程

1. 从 **coreos-installer** 镜像镜像页面下载 **coreos-installer** 二进制文件。

2. 从 RHCOS image mirror 页获取 RHCOS **kernel**、**initramfs** 和 **rootfs** 文件，并运行以下命令来使用以下信息创建新的自定义 **initramfs** 文件：

```
$ coreos-installer pxe customize \
    --pre-install mount-iscsi.sh \ 1
    --post-install unmount-iscsi.sh \ 2
    --dest-device /dev/disk/by-path/<IP_address>:<port>-iscsi-<target_iqn>-lun-<lun> \ 3
    --dest-ignition config.ign \ 4
    --dest-karg-append rd.iscsi.initiator=<initiator_iqn> \ 5
    --dest-karg-append netroot=<target_iqn> \ 6
    -o custom.img rhcos-<version>-live-initramfs.x86_64.img
```

1. 安装之前运行的脚本。它应包含用于挂载 iSCSI 目标以及启用多路径的任何命令的 **iscsiadm** 命令。
2. 安装后运行的脚本。它应包含命令 **iscsiadm --mode node --logout=all**。
3. 目标系统的位置。您必须提供目标门户的 IP 地址、关联的端口号、目标 iSCSI 节点采用 IQN 格式，以及 iSCSI 逻辑单元号(LUN)。
4. 目标系统的 Ignition 配置。
5. iSCSI 启动器或客户端，以 IQN 格式的名称。启动器构成连接到 iSCSI 目标的一个会话。
6. iSCSI 目标或服务器的名称（IQN 格式）。

有关 **dracut** 支持的 iSCSI 选项的更多信息，请参阅 **dracut.cmdline** 手册页。

15.2.11.3.8.5. 使用 iBFT 为 iSCSI 引导设备自定义实时安装 PXE 环境

您可以设置 iSCSI 目标和 initiator 值，以便使用自定义的 live RHCOS 镜像版本自动挂载、引导和配置。

先决条件

1. 您有一个 iSCSI 目标，您要在其中安装 RHCOS。
2. 可选：有多路径 iSCSI 目标。

流程

1. 从 **coreos-installer** 镜像镜像页面下载 **coreos-installer** 二进制文件。
2. 从 RHCOS image mirror 页获取 RHCOS **kernel**、**initramfs** 和 **rootfs** 文件，并运行以下命令来使用以下信息创建新的自定义 **initramfs** 文件：

```
$ coreos-installer pxe customize \
    --pre-install mount-iscsi.sh \ 1
    --post-install unmount-iscsi.sh \ 2
    --dest-device /dev/mapper/mpatha \ 3
    --dest-ignition config.ign \ 4
    --dest-karg-append rd.iscsi.firmware=1 \ 5
    --dest-karg-append rd.multipath=default \ 6
    -o custom.img rhcos-<version>-live-initramfs.x86_64.img
```

1. 安装之前运行的脚本。它应包含用于挂载 iSCSI 目标的 **iscsiadm** 命令。
2. 安装后运行的脚本。它应包含命令 **iscsiadm --mode node --logout=all**。
3. 设备的路径。如果连接了多个多路径设备，或者需要明确指定，您使用多路径（**/dev/mapper/mpatha**），您可以使用 **/dev/disk/by-path** 中可用的 World Wide Name (WWN) 符号链接。
4. 目标系统的 Ignition 配置。
5. iSCSI 参数从 BIOS 固件读取。

6 可选：如果您要启用多路径，请包含此参数。

有关 **dracut** 支持的 iSCSI 选项的更多信息，请参阅 **dracut.cmdline** 手册页。

15.2.11.3.9. 高级 RHCOS 安装参考

本节演示了网络配置和其他高级选项，允许您修改 Red Hat Enterprise Linux CoreOS(RHCOS)手动安装过程。下表描述了您可以用于 RHCOS live 安装程序和 **coreos-installer** 命令的内核参数和命令行选项。

15.2.11.3.9.1. ISO 安装的网络和绑定选项

如果从 ISO 镜像安装 RHCOS，您可以在引导镜像时手动添加内核参数，以便为节点配置网络。如果没有指定网络参数，当 RHCOS 检测到需要网络来获取 Ignition 配置文件时，在 initramfs 中激活 DHCP。

重要

在手动添加网络参数时，还必须添加 **rd.neednet=1** 内核参数，以便在 initramfs 中启动网络。

以下信息提供了在 RHCOS 节点上为 ISO 安装配置网络和绑定的示例。示例描述了如何使用 **ip=**、**name server =** 和 **bond=** 内核参数。

注意

添加内核参数时顺序非常重要： **ip=**、**name server=**，然后 **bond=**。

网络选项在系统引导过程中传递给 **dracut** 工具。有关 **dracut** 支持的网络选项的更多信息，请参阅 **dracut.cmdline** 手册页。

以下示例是 ISO 安装的网络选项。

配置 DHCP 或静态 IP 地址
要配置 IP 地址，可使用 DHCP(**ip=dhcp**)或设置单独的静态 IP 地址(**ip=<host_ip>**)。如果设置静态 IP，则必须在每个节点上识别 DNS 服务器 IP 地址（**名称服务器=<dns_ip>**）。以下示例集：

- 节点的 IP 地址为 **10.10.10.2**
- 网关地址为 **10.10.10.254**
- 子网掩码为 **255.255.255.0**
- 到 **core0.example.com** 的主机名
- DNS 服务器地址为 **4.4.4.41**
- 自动配置值为 **none**。当以静态方式配置 IP 网络时，不需要自动配置。

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
nameserver=4.4.4.41
```

注意

当您使用 DHCP 为 RHCOS 机器配置 IP 寻址时，机器还通过 DHCP 获取 DNS 服务器信息。对于基于 DHCP 的部署，您可以通过 DHCP 服务器配置定义 RHCOS 节点使用的 DNS 服务器地址。

配置没有静态主机名的 IP 地址
您可以在不分配静态主机名的情况下配置 IP 地址。如果用户没有设置静态主机名，则会提取并通过反向 DNS 查找自动设置。要在没有静态主机名的情况下配置 IP 地址，请参考以下示例：

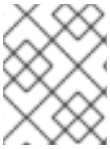
- 节点的 IP 地址为 **10.10.10.2**
- 网关地址为 **10.10.10.254**
- 子网掩码为 **255.255.255.0**
- DNS 服务器地址为 **4.4.4.41**
- 自动配置值为 **none**。当以静态方式配置 IP 网络时，不需要自动配置。

```
ip=10.10.10.2::10.10.10.254:255.255.255.0::enp1s0:none
nameserver=4.4.4.41
```

指定多个网络接口
您可以通过设置多个 **ip=** 条目来指定多个网络接口。

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
ip=10.10.10.3::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none
```

配置默认网关和路由
可选：您可以通过设置 a **rd.route=** 值来配置到额外网络的路由。



注意

当您配置一个或多个网络时，需要一个默认网关。如果额外网络网关与主要网络网关不同，则默认网关必须是主要网络网关。

- 运行以下命令来配置默认网关：

  ```
  ip=::10.10.10.254::::
  ```

- 输入以下命令为额外网络配置路由：

  ```
  rd.route=20.20.20.0/24:20.20.20.254:enp2s0
  ```

在单个接口中禁用 DHCP
您可以在单一接口中禁用 DHCP，例如当有两个或者多个网络接口时，且只有一个接口被使用。在示例中，**enp1s0** 接口具有一个静态网络配置，而 **enp2s0** 禁用了 DHCP，不使用它：

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
ip=::::core0.example.com:enp2s0:none
```

合并 DHCP 和静态 IP 配置

您可以将系统上的 DHCP 和静态 IP 配置与多个网络接口合并，例如：

```
ip=enp1s0:dhcp
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none
```

在独立接口上配置 VLAN
可选： 您可以使用 **vlan=** 参数在单个接口上配置 VLAN。

- 要在网络接口中配置 VLAN 并使用静态 IP 地址，请运行以下命令：

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0.100:none
vlan=enp2s0.100:enp2s0
```

- 要在网络接口中配置 VLAN 并使用 DHCP，请运行以下命令：

```
ip=enp2s0.100:dhcp
vlan=enp2s0.100:enp2s0
```

提供多个 DNS 服务器
您可以通过为每个服务器添加一个 **nameserver=** 条目来提供多个 DNS 服务器，例如

```
nameserver=1.1.1.1
nameserver=8.8.8.8
```

将多个网络接口绑定到一个接口
可选： 您可以使用 **bond=** 选项将多个网络接口绑定到一个接口。请参见以下示例：

- 配置绑定接口的语法为： **bond=<name>[:<network_interfaces>][:options]**
  <name> 是绑定设备名称 (**bond0**)、**<network_interfaces>** 代表以逗号分隔的物理（以太网）接口列表(**em1,em2**)，*options* 是用逗号分开的绑定选项列表。输入 **modinfo bonding** 查看可用选项。

- 当使用 **bond=** 创建绑定接口时，您必须指定如何分配 IP 地址以及绑定接口的其他信息。
  - 要将绑定接口配置为使用 DHCP，请将绑定的 IP 地址设置为 **dhcp**。例如：

```
bond=bond0:em1,em2:mode=active-backup
ip=bond0:dhcp
```

  - 要将绑定接口配置为使用静态 IP 地址，请输入您需要的特定 IP 地址和相关信息。例如：

```
bond=bond0:em1,em2:mode=active-backup
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0:none
```

将多个 SR-IOV 网络接口绑定到双端口 NIC 接口

重要

支持与为 SR-IOV 设备启用 NIC 分区关联的第 1 天操作只是一个技术预览功能。技术预览功能不受红帽产品服务等级协议（SLA）支持，且功能可能并不完整。红帽不推荐在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。

有关红帽技术预览功能支持范围的更多信息，请参阅技术预览功能支持范围。

可选： 您可以使用 **bond=** 选项将多个 SR-IOV 网络接口绑定到双端口 NIC 接口。

在每个节点上，您必须执行以下任务：

1. 按照管理 SR-IOV 设备中的指导创建 SR-IOV 虚拟功能(VF)。按照"将 SR-IOV 网络设备附加到虚拟机"部分中的步骤操作。
2. 创建绑定，将所需的 VF 附加到绑定，并根据配置网络绑定的指导设置绑定链接状态。按照任何描述的步骤创建绑定。

以下示例演示了您必须使用的语法：

- 配置绑定接口的语法为：**bond=<name>[:<network_interfaces>][:options]**
  <name> 是绑定设备名称 (**bond0**)、**<network_interfaces>** 由内核中已知的名称来代表虚拟功能 (VF)，并显示在 **ip link** 命令的输出中 (**eno1f0,eno2f0**)，*options* 是以逗号分隔的绑定选项列表。输入 **modinfo bonding** 查看可用选项。
- 当使用 **bond=** 创建绑定接口时，您必须指定如何分配 IP 地址以及绑定接口的其他信息。
  - 要将绑定接口配置为使用 DHCP，请将绑定的 IP 地址设置为 **dhcp**。例如：

    ```
    bond=bond0:eno1f0,eno2f0:mode=active-backup
    ip=bond0:dhcp
    ```

  - 要将绑定接口配置为使用静态 IP 地址，请输入您需要的特定 IP 地址和相关信息。例如：

    ```
    bond=bond0:eno1f0,eno2f0:mode=active-backup
    ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0:none
    ```

使用网络团队
可选： 您可以使用 **team=** 参数来将网络团队用作绑定的替代选择：

- 配置组接口的语法为： **team=name[:network_interfaces]**
  name 是组设备名称(**team0**)，*network_interfaces* 代表以逗号分隔的物理（以太网）接口（**em1、em2**）列表。

注意

当 RHCOS 切换到即将推出的 RHEL 版本时，团队(team)功能被计划弃用。如需更多信息，请参阅红帽知识库文章。

使用以下示例配置网络团队：

```
team=team0:em1,em2
ip=team0:dhcp
```

15.2.11.3.9.2. ISO 和 PXE 安装的coreos-installer 选项

从 ISO 镜像引导 RHCOS live 环境后，您可以通过 **在命令提示符下运行 coreos-installer install <options> <device** > 来安装 RHCOS。

下表显示了您可以传递给 **coreos-installer** 命令的子命令、选项和参数。

表 15.9. coreos-installer 子命令、命令行选项和参数

coreos-installer install 子命令	
子命令	***描述***
$ coreos-installer install <options> <device>	在 ISO 镜像中嵌入 Ignition 配置。
coreos-installer install 子命令选项	
选项	***描述***
-u, --image-url <url>	手动指定镜像 URL。
-f,--image-file <path>	手动指定本地镜像文件。用于调试。
-i, --ignition-file <path>	从文件中嵌入 Ignition 配置。
-i,--ignition-url <URL>	从 URL 嵌入 Ignition 配置。
--ignition-hash <digest>	Ignition 配置 **的 type-value** 的摘要值。
-p, --platform <name>	覆盖已安装系统的 Ignition 平台 ID。
--console <spec>	为安装的系统设置内核和引导装载程序控制台。有关 **<spec>** 格式的更多信息，请参阅 Linux 内核串口控制台文档。
--append-karg <arg>...	将默认内核参数附加到安装的系统中。
--delete-karg <arg>...	从安装的系统中删除默认内核参数。
-n, --copy-network	从安装环境中复制网络配置。  **重要** **copy-network** 选项仅复制在 **/etc/NetworkManager/system-connections** 下找到的网络配置。特别是，它不会复制系统主机名。
--network-dir <path>	用于 **-n**。默认为 **/etc/NetworkManager/system-connections/**。
--save-partlabel <lx>..	使用这个标签 glob 保存分区。
--save-partindex <id>...	使用这个数值或范围保存分区。
--insecure	跳过 RHCOS 镜像签名验证。

--insecure-ignition	允许没有 HTTPS 或 hash 的 Ignition URL。
--architecture <name>	目标 CPU 架构.有效值为 **x86_64** 和 **aarch64**。
--preserve-on-error	出错时不要清除分区表。
-h,--help	打印帮助信息.
coreos-installer install 子命令参数	
参数	***描述***
<device>	目标设备.
coreos-installer ISO 子命令	
子命令	***描述***
$ coreos-installer iso customize <options> <ISO_image>	自定义 RHCOS live ISO 镜像。
coreos-installer iso reset <options> <ISO_image>	将 RHCOS live ISO 镜像恢复到默认设置。
coreos-installer iso ignition remove <options> <ISO_image>	从 ISO 镜像中删除嵌入的 Ignition 配置。
coreos-installer ISO customize 子命令选项	
选项	***描述***
--dest-ignition <path>	将指定的 Ignition 配置文件合并到目标系统的新配置片段中。
--dest-console <spec>	为目标系统指定内核和引导装载程序控制台。
--dest-device <path>	安装并覆盖指定的目标设备。
--dest-karg-append <arg>	为每个目标系统引导添加一个内核参数。
--dest-karg-delete <arg>	从目标系统的每个引导中删除内核参数。
--network-keyfile <path>	使用指定的 NetworkManager 密钥文件进行实时和目标系统配置网络。
--ignition-ca <path>	指定要被 Ignition 信任的额外 TLS 证书颁发机构。

--pre-install <path>	在安装之前运行指定的脚本。
--post-install <path>	安装后运行指定的脚本。
--installer-config <path>	应用指定的安装程序配置文件。
--live-ignition <path>	将指定的 Ignition 配置文件合并到实时环境的新配置片段中。
--live-karg-append <arg>	为每个实时环境引导添加一个内核参数。
--live-karg-delete <arg>	从实时环境每次引导时删除内核参数。
--live-karg-replace <k=o=n>	在每次启动 live 环境时替换内核参数，格式为 **key=old=new**。
-f,--force	覆盖现有的 Ignition 配置。
-o,--output <path>	将 ISO 写入到新的输出文件。
-h,--help	打印帮助信息.
coreos-installer PXE 子命令	
子命令	***描述***
请注意，并非所有子命令都接受所有这些选项。	
coreos-installer pxe customize <options> <path>	自定义 RHCOS live PXE 引导配置。
coreos-installer pxe ignition wrap <options>	在镜像中嵌套 Ignition 配置。
coreos-installer pxe ignition unwrap <options> <image_name>	在镜像中显示嵌套的 Ignition 配置。
coreos-installer PXE customize 子命令选项	
选项	***描述***
请注意，并非所有子命令都接受所有这些选项。	
--dest-ignition <path>	将指定的 Ignition 配置文件合并到目标系统的新配置片段中。
--dest-console <spec>	为目标系统指定内核和引导装载程序控制台。

--dest-device <path>	安装并覆盖指定的目标设备。
--network-keyfile <path>	使用指定的 NetworkManager 密钥文件进行实时和目标系统配置网络。
--ignition-ca <path>	指定要被 Ignition 信任的额外 TLS 证书颁发机构。
--pre-install <path>	在安装之前运行指定的脚本。
post-install <path>	安装后运行指定的脚本。
--installer-config <path>	应用指定的安装程序配置文件。
--live-ignition <path>	将指定的 Ignition 配置文件合并到实时环境的新配置片段中。
-o, --output <path>	将 initramfs 写入一个新输出文件。  **注意** PXE 环境需要这个选项。
-h,--help	打印帮助信息.

15.2.11.3.9.3. coreos.inst 引导选项用于 ISO 或 PXE 安装

您可以通过将 **coreos.inst** boot 参数传递给 RHCOS live 安装程序，在引导时自动调用 **coreos-installer** 选项。这些是在标准引导参数之外提供的。

- 对于 ISO 安装，可以通过在启动加载器菜单中中断自动引导来添加 **coreos.inst** 选项。您可以在突出显示 **RHEL CoreOS(Live)** 菜单选项时按 **TAB** 来中断自动引导。
- 对于 PXE 或 iPXE 安装，在引导 RHCOS live 安装程序前，**coreos.inst** 选项必须添加到 **APPEND** 行。

下表显示了用于 ISO 和 PXE 安装的 RHCOS live 安装程序 **coreos.inst** 引导选项。

表 15.10. coreos.inst 引导选项

参数	描述
coreos.inst.install_dev	必需。要安装到的系统中的块设备。建议您使用完整路径，如 **/dev/sda**，但 **允许使用**。
coreos.inst.ignition_url	可选：嵌入到安装的系统中的 Ignition 配置的 URL。如果没有指定 URL，则不会嵌入 Ignition 配置。仅支持 HTTP 和 HTTPS 协议。

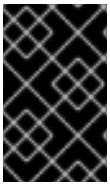
参数	描述
coreos.inst.save_partlabel	可选：在安装过程中要保留的分区分离标签。允许使用 glob 风格的通配符。指定分区不需要存在。
coreos.inst.save_partindex	可选：在安装过程中压缩要保留的分区索引。允许 **m-n** 范围，**m** 或 **n** 可以被省略。指定分区不需要存在。
coreos.inst.insecure	可选：将 **coreos.inst.image_url** 指定的 OS 镜像提交取消签名。
coreos.inst.image_url	可选：下载并安装指定的 RHCOS 镜像。 ● 这个参数不应该在生产环境中使用，而是只用于调试目的。 ● 虽然此参数可用于安装与 live 介质不匹配的 RHCOS 版本，但建议您使用与您要安装版本匹配的介质。 ● 如果您使用 **coreos.inst.image_url**，还必须使用 **coreos.inst.insecure**。这是因为，裸机介质没有为 OpenShift Container Platform 进行 GPG 签名。 ● 仅支持 HTTP 和 HTTPS 协议。
coreos.inst.skip_reboot	可选：安装后系统不会重启。安装完成后，您将收到提示，提示您检查在安装过程中发生的情况。这个参数不应该在生产环境中使用，而是只用于调试目的。
coreos.inst.platform_id	可选：安装 RHCOS 镜像的平台的 Ignition 平台 ID。默认为 **metal**。这个选项决定是否从云供应商（如 VMware）请求 Ignition 配置。例如：**coreos.inst.platform_id=vmware**。
ignition.config.url	可选：用于实时引导的 Ignition 配置的 URL。例如，这可用于自定义调用 **coreos-installer** 的方式，或者用于在安装前或安装后运行代码。这与 **coreos.inst.ignition_url**（这是已安装系统的 Ignition 配置）不同。

15.2.11.4. 在 RHCOS 上启用带有内核参数的多路径

RHCOS 支持主磁盘上的多路径，支持更强大的硬件故障弹性，以获得更高的主机可用性。

您可以在安装时为 OpenShift Container Platform 4.8 或更高版本置备的节点启用多路径。虽然安装后支持可以通过机器配置激活多路径来实现，但建议在安装过程中启用多路径。

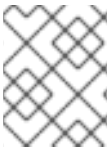
在任何 I/O 到未优化路径会导致 I/O 系统错误的设置中，您必须在安装时启用多路径。



重要

在 IBM Z® 和 IBM® LinuxONE 中，您只能在在安装过程中为它配置集群时启用多路径。如需更多信息，请参阅在 *IBM Z® 和 IBM® LinuxONE 上安装使用 z/VM 的集群"安装* RHCOS 并启动 OpenShift Container Platform bootstrap 过程"。

以下流程在安装时启用多路径，并在 **coreos-installer install** 命令中附加内核参数，以便安装的系统本身将使用从第一次引导开始的多路径。



注意

OpenShift Container Platform 不支持在从 4.6 或更早版本升级的节点上启用多路径作为 2 天的活动。

流程

1. 要启用多路径并启动 **multipathd** 守护进程，请在安装主机上运行以下命令：

```
$ mpathconf --enable && systemctl start multipathd.service
```

   - 可选：如果引导 PXE 或 ISO，则可以通过从内核命令行添加 **rd.multipath=default** 来启用多路径。

2. 通过调用 **coreos-installer** 程序附加内核参数：

   - 如果只有一个多路径设备连接到计算机，则应在路径 **/dev/mapper/mpatha** 上可用。例如：

```
$ coreos-installer install /dev/mapper/mpatha \ 1
--append-karg rd.multipath=default \
--append-karg root=/dev/disk/by-label/dm-mpath-root \
--append-karg rw
```

     1 表示单一多路径设备的路径。

   - 如果有多个多路径设备连接到计算机，或者更为明确，而不是使用 **/dev/mapper/mpatha**，则建议使用 **/dev/disk/by-id** 中可用的 World Wide Name(WWN)符号链接。例如：

```
$ coreos-installer install /dev/disk/by-id/wwn-<wwn_ID> \ 1
--append-karg rd.multipath=default \
--append-karg root=/dev/disk/by-label/dm-mpath-root \
--append-karg rw
```

     1 表示目标多路径设备的 WWN ID。例如： **0xx194e957fcedb4841**。

     当使用特殊 **coreos.inst.*** 参数指示 live 安装程序时，这个符号链接也可以用作 **coreos.inst.install_dev** 内核参数。如需更多信息，请参阅"安装 RHCOS 和启动 OpenShift Container Platform bootstrap 过程"。

3. 前往其中一个 worker 节点并列出内核命令行参数（主机上的 **/proc/cmdline** 中），以检查内核参数是否正常工作：

```
$ oc debug node/ip-10-0-141-105.ec2.internal
```

输出示例

```
Starting pod/ip-10-0-141-105ec2internal-debug ...
To use host binaries, run `chroot /host`

sh-4.2# cat /host/proc/cmdline
...
rd.multipath=default root=/dev/disk/by-label/dm-mpath-root
...

sh-4.2# exit
```

您应看到添加的内核参数。

15.2.11.5. 在 iSCSI 引导设备中手动安装 RHCOS

您可以在 iSCSI 目标上手动安装 RHCOS。

先决条件

1. 您在 RHCOS live 环境中。
2. 您有一个要在其上安装 RHCOS 的 iSCSI 目标。

流程

1. 运行以下命令，从 live 环境中挂载 iSCSI 目标：

```
$ iscsiadm \
    --mode discovery \
    --type sendtargets
    --portal <IP_address> \ 1
    --login
```

1 目标门户的 IP 地址。

2. 运行以下命令并使用必要的内核参数将 RHCOS 安装到 iSCSI 目标上，例如：

```
$ coreos-installer install \
    /dev/disk/by-path/ip-<IP_address>:<port>-iscsi-<target_iqn>-lun-<lun> \ 1
    --append-karg rd.iscsi.initiator=<initiator_iqn> \ 2
    --append.karg netroot=<target_iqn> \ 3
    --console ttyS0,115200n8
    --ignition-file <path_to_file>
```

1 您要安装到的位置。您必须提供目标门户的 IP 地址、关联的端口号、目标 iSCSI 节点采用 IQN 格式，以及 iSCSI 逻辑单元号(LUN)。

2 iSCSI 启动器或客户端，以 IQN 格式的名称。启动器构成连接到 iSCSI 目标的一个会话。

3 iSCSI 目标或服务器的名称（IQN 格式）。

有关 **dracut** 支持的 iSCSI 选项的更多信息，请参阅 **dracut.cmdline** 手册页。

3. 使用以下命令卸载 iSCSI 磁盘：

```
$ iscsiadm --mode node --logoutall=all
```

此过程也可以使用 **coreos-installer iso customize** 或 **coreos-installer pxe customize** 子命令来执行。

15.2.11.6. 使用 iBFT 在 iSCSI 引导设备中安装 RHCOS

在一个完全无盘机器上，也可以通过 iBFT. iSCSI 多路径传递 iSCSI 目标和启动器值。

先决条件

1. 您在 RHCOS live 环境中。
2. 您有一个 iSCSI 目标，您要在其中安装 RHCOS。
3. 可选：有多路径 iSCSI 目标。

流程

1. 运行以下命令，从 live 环境中挂载 iSCSI 目标：

```
$ iscsiadm \
    --mode discovery \
    --type sendtargets
    --portal <IP_address> \ 1
    --login
```

1 目标门户的 IP 地址。

2. 可选：使用以下命令启用多路径并启动守护进程：

```
$ mpathconf --enable && systemctl start multipathd.service
```

3. 运行以下命令并使用必要的内核参数将 RHCOS 安装到 iSCSI 目标上，例如：

```
$ coreos-installer install \
    /dev/mapper/mpatha \ 1
    --append-karg rd.iscsi.firmware=1 \ 2
    --append-karg rd.multipath=default \ 3
    --console ttyS0 \
    --ignition-file <path_to_file>
```

1 单一多路径设备的路径。如果连接了多个多路径设备，或者需要明确指定，您可以使用 **/dev/disk/by-path** 中可用的 World Wide Name (WWN) 符号链接。

2 iSCSI 参数从 BIOS 固件读取。

3 可选：如果您要启用多路径，请包含此参数。

有关 **dracut** 支持的 iSCSI 选项的更多信息，请参阅 **dracut.cmdline** 手册页。

4. 卸载 iSCSI 磁盘：

```
$ iscsiadm --mode node --logout=all
```

此过程也可以使用 **coreos-installer iso customize** 或 **coreos-installer pxe customize** 子命令来执行。

其他资源

- 如需有关使用特殊 **coreos.inst.*** 参数指定 live 安装程序的更多信息，请参阅 安装 RHCOS 并启动 OpenShift Container Platform bootstrap 过程。

15.2.12. 等待 bootstrap 过程完成

OpenShift Container Platform bootstrap 过程在集群节点首次引导到安装到磁盘的持久 RHCOS 环境后开始。通过 Ignition 配置文件提供的配置信息用于初始化 bootstrap 过程并在机器上安装 OpenShift Container Platform。您必须等待 bootstrap 过程完成。

先决条件

- 已为集群创建 Ignition 配置文件。
- 您已配置了适当的网络、DNS 和负载平衡基础架构。
- 已获得安装程序，并为集群生成 Ignition 配置文件。
- 已在集群机器上安装 RHCOS，并提供 OpenShift Container Platform 安装程序生成的 Ignition 配置文件。
- 您的机器可以直接访问互联网，或者有 HTTP 或 HTTPS 代理可用。

流程

1. 监控 bootstrap 过程：

```
$ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \ 1
    --log-level=info 2
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不是 **info**。

输出示例

```
INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...
INFO API v1.29.4 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
```

当 Kubernetes API 服务器提示已在 control plane 机器上引导它时，该命令会成功。

2. bootstrap 过程完成后，从负载均衡器中删除 bootstrap 机器。

重要

此时您必须从负载均衡器中删除 bootstrap 机器。您还可以删除或重新格式化 bootstrap 机器本身。

其他资源

- 如需有关 监控安装日志的更多信息，请参阅监控安装进度，并在出现安装问题时检索诊断数据。

15.2.13. 使用 CLI 登录集群

您可以通过导出集群 **kubeconfig** 文件，以默认系统用户身份登录集群。**kubeconfig** 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。
- 已安装 **oc** CLI。

流程

1. 导出 **kubeadmin** 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 **<installation_directory>**，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 **oc** 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

15.2.14. 批准机器的证书签名请求

当您将机器添加到集群时，会为您添加的每台机器生成两个待处理证书签名请求(CSR)。您必须确认这些 CSR 已获得批准，或根据需要自行批准。必须首先批准客户端请求，然后批准服务器请求。

先决条件

- 您已将机器添加到集群中。

流程

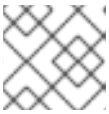
1. 确认集群可以识别这些机器：

```
$ oc get nodes
```

输出示例

```
NAME      STATUS    ROLES   AGE  VERSION
master-0  Ready     master  63m  v1.29.4
master-1  Ready     master  63m  v1.29.4
master-2  Ready     master  64m  v1.29.4
```

输出中列出了您创建的所有机器。



注意

在有些 CSR 被批准前，前面的输出可能不包括计算节点（也称为 worker 节点）。

2. 检查待处理的 CSR，并确保添加到集群中的每台机器都有 **Pending** 或 **Approved** 状态的客户端请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE    REQUESTOR                                                    CONDITION
csr-8b2br   15m    system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper   Pending
csr-8vnps   15m    system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper   Pending
...
```

在本例中，两台机器加入集群。您可能会在列表中看到更多已批准的 CSR。

3. 如果 CSR 没有获得批准，在您添加的机器的所有待处理 CSR 都处于 **Pending 状态** 后，请批准集群机器的 CSR：

注意

由于 CSR 会自动轮转，因此请在将机器添加到集群后一小时内批准您的 CSR。如果没有在一小时内批准它们，证书将会轮转，每个节点会存在多个证书。您必须批准所有这些证书。批准客户端 CSR 后，Kubelet 为服务证书创建一个二级 CSR，这需要手动批准。然后，如果 Kubelet 请求具有相同参数的新证书，则后续提供证书续订请求由 **machine-approver** 自动批准。

注意

对于在未启用机器 API 的平台上运行的集群，如裸机和其他用户置备的基础架构，您必须实施一种方法来自动批准 kubelet 提供证书请求(CSR)。如果没有批准请求，则 **oc exec**、oc**rsh** 和 **oc logs** 命令将无法成功，因为 API 服务器连接到 kubelet 时需要服务证书。与 Kubelet 端点联系的任何操作都需要此证书批准。该方法必须监视新的 CSR，确认 CSR 由 system: **node** 或 **system:admin** 组中的 **node-bootstrapper** 服务帐户提交，并确认节点的身份。

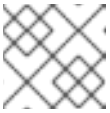
- 要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```



注意

在有些 CSR 被批准前，一些 Operator 可能无法使用。

4. 现在，您的客户端请求已被批准，您必须查看添加到集群中的每台机器的服务器请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE     REQUESTOR                                                   CONDITION
csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

5. 如果剩余的 CSR 没有被批准，且处于 **Pending** 状态，请批准集群机器的 CSR：

- 要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs oc adm certificate approve
```

6. 批准所有客户端和服务器 CSR 后，机器将 处于 **Ready 状态**。运行以下命令验证：

```
$ oc get nodes
```

输出示例

```
NAME      STATUS    ROLES   AGE  VERSION
master-0  Ready     master  73m  v1.29.4
master-1  Ready     master  73m  v1.29.4
master-2  Ready     master  74m  v1.29.4
worker-0  Ready     worker  11m  v1.29.4
worker-1  Ready     worker  11m  v1.29.4
```

注意

批准服务器 CSR 后可能需要几分钟时间让机器过渡到 **Ready** 状态。

其他信息

- 如需有关 CSR 的更多信息，请参阅 证书签名请求。

15.2.15. 初始 Operator 配置

在 control plane 初始化后，您必须立即配置一些 Operator，以便它们都可用。

先决条件

- 您的 control plane 已初始化。

流程

1. 观察集群组件上线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION   AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.16.0    True        False         False      19m
baremetal                                  4.16.0    True        False         False      37m
cloud-credential                           4.16.0    True        False         False      40m
cluster-autoscaler                         4.16.0    True        False         False      37m
config-operator                            4.16.0    True        False         False      38m
console                                    4.16.0    True        False         False      26m
csi-snapshot-controller                    4.16.0    True        False         False      37m
dns                                        4.16.0    True        False         False      37m
etcd                                       4.16.0    True        False         False      36m
image-registry                             4.16.0    True        False         False      31m
ingress                                    4.16.0    True        False         False      30m
insights                                   4.16.0    True        False         False      31m
kube-apiserver                             4.16.0    True        False         False      26m
kube-controller-manager                    4.16.0    True        False         False      36m
kube-scheduler                             4.16.0    True        False         False      36m
kube-storage-version-migrator              4.16.0    True        False         False      37m
machine-api                                4.16.0    True        False         False      29m
machine-approver                           4.16.0    True        False         False      37m
machine-config                             4.16.0    True        False         False      36m
marketplace                                4.16.0    True        False         False      37m
monitoring                                 4.16.0    True        False         False      29m
network                                    4.16.0    True        False         False      38m
node-tuning                                4.16.0    True        False         False      37m
openshift-apiserver                        4.16.0    True        False         False      32m
openshift-controller-manager               4.16.0    True        False         False      30m
openshift-samples                          4.16.0    True        False         False      32m
operator-lifecycle-manager                 4.16.0    True        False         False      37m
operator-lifecycle-manager-catalog         4.16.0    True        False         False      37m
```

```
operator-lifecycle-manager-packageserver   4.16.0   True        False         False      32m
service-ca                                 4.16.0   True        False         False      38m
storage                                    4.16.0   True        False         False      37m
```

2. 配置不可用的 Operator。

其他资源

- 如需了解 在 OpenShift Container Platform 安装失败时收集数据的详细信息，请参阅从失败安装收集日志。
- 如需了解在集群中检查 Operator pod 健康状况的步骤，并收集 Operator 日志以进行诊断，请参阅故障排除 Operator 问题。

15.2.15.1. 安装过程中删除的镜像 registry

在不提供可共享对象存储的平台上，OpenShift Image Registry Operator bootstraps 本身为 **Removed**。这允许 **openshift-installer** 在这些平台类型上完成安装。

安装后，您必须编辑 Image Registry Operator 配置，将 **managementState** 从 **Removed** 切换到 **Managed**。完成此操作后，您必须配置存储。

15.2.15.2. 镜像 registry 存储配置

对于不提供默认存储的平台，Image Registry Operator 最初不可用。安装后，您必须将 registry 配置为使用存储，以便 Registry Operator 可用。

显示配置生产集群所需的持久性卷的说明。如果适用，显示有关将空目录配置为存储位置的说明，这仅适用于非生产集群。

提供了在升级过程中使用 **Recreate** rollout 策略来允许镜像 registry 使用块存储类型的说明。

15.2.15.2.1. 为裸机和其他手动安装配置 registry 存储

作为集群管理员，在安装后需要配置 registry 来使用存储。

先决条件

- 您可以使用具有 **cluster-admin** 角色的用户访问集群。
- 您有一个使用手动置备的 Red Hat Enterprise Linux CoreOS(RHCOS)节点（如裸机）的集群。
- 您已为集群置备持久性存储，如 Red Hat OpenShift Data Foundation。

重要

当您只有一个副本时，OpenShift Container Platform 支持对镜像 registry 存储的 **ReadWriteOnce** 访问。**ReadWriteOnce** 访问还要求 registry 使用 **Recreate** rollout 策略。要部署支持高可用性的镜像 registry，需要两个或多个副本，**ReadWriteMany** 访问。

- 必须具有 100Gi 容量。

流程

1. 要将 registry 配置为使用存储，修改 **configs.imageregistry/cluster** 资源中的 **spec.storage.pvc**。

注意

使用共享存储时，请查看您的安全设置以防止外部访问。

2. 验证您没有 registry pod:

```
$ oc get pod -n openshift-image-registry -l docker-registry=default
```

输出示例

```
No resources found in openshift-image-registry namespace
```

注意

如果您的输出中有一个 registry pod，则不需要继续这个过程。

3. 检查 registry 配置：

```
$ oc edit configs.imageregistry.operator.openshift.io
```

输出示例

```
storage:
  pvc:
    claim:
```

将 **claim** 字段留空以允许自动创建 **image-registry-storage** PVC。

4. 检查 **clusteroperator** 状态：

```
$ oc get clusteroperator image-registry
```

输出示例

```
NAME           VERSION      AVAILABLE   PROGRESSING   DEGRADED   SINCE
MESSAGE
image-registry   4.16           True        False         False      6h50m
```

5. 确保 registry 设置为 managed，以启用镜像的构建和推送。

- 运行：

```
$ oc edit configs.imageregistry/cluster
```

然后，更改行

```
managementState: Removed
```

至

```
managementState: Managed
```

15.2.15.2.2. 在非生产集群中为镜像 registry 配置存储

您必须为 Image Registry Operator 配置存储。对于非生产集群，您可以将镜像 registry 设置为空目录。如果您这样做，重启 registry 时会丢失所有镜像。

流程

- 将镜像 registry 存储设置为空目录：

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec":{"storage":{"emptyDir":{}}}}'
```

> **警告**
>
> 仅为非生产集群配置这个选项。

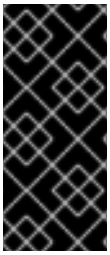
如果在 Image Registry Operator 初始化其组件前运行这个命令，**oc patch** 命令会失败并显示以下错误：

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster" not found
```

等待几分钟，然后再次运行 命令。

15.2.15.2.3. 为裸机配置块 registry 存储

要允许镜像 registry 在作为集群管理员升级过程中使用块存储类型，您可以使用 **Recreate rollout 策略**。



> **重要**
>
> 支持块存储卷或块持久性卷，但不建议在生产环境中使用镜像 registry。在块存储上配置 registry 的安装不具有高可用性，因为 registry 无法具有多个副本。
>
> 如果您选择将块存储卷与镜像 registry 搭配使用，则必须使用文件系统持久性卷声明 (PVC)。

流程

1. 输入以下命令将镜像 registry 存储设置为块存储类型，对 registry 进行补丁，使其使用 **Recreate** rollout 策略，并只使用一个副本运行：

```
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec":{"rolloutStrategy":"Recreate","replicas":1}}'
```

2. 为块存储设备置备 PV，并为该卷创建 PVC。请求的块卷使用 ReadWriteOnce(RWO)访问模式。

   a. 创建包含以下内容的 pvc.yaml 文件以定义 VMware vSphere PersistentVolumeClaim 对象：

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: image-registry-storage 1
  namespace: openshift-image-registry 2
spec:
  accessModes:
  - ReadWriteOnce 3
  resources:
    requests:
      storage: 100Gi 4
```

   1 代表 PersistentVolumeClaim 对象的唯一名称。

   2 PersistentVolumeClaim 对象的命名空间，即 openshift-image-registry。

   3 持久性卷声明的访问模式。使用 ReadWriteOnce 时，单个节点可以通过读写权限挂载该卷。

   4 持久性卷声明的大小。

   b. 输入以下命令从文件创建 PersistentVolumeClaim 对象：

```
$ oc create -f pvc.yaml -n openshift-image-registry
```

3. 输入以下命令编辑 registry 配置，使其引用正确的 PVC：

```
$ oc edit config.imageregistry.operator.openshift.io -o yaml
```

输出示例

```
storage:
  pvc:
    claim: 1
```

1

通过创建自定义 PVC，您可以将 claim 字段留空，以便默认自动创建 image-registry-storage PVC。

15.2.16. 在用户置备的基础架构上完成安装

完成 Operator 配置后，可以在您提供的基础架构上完成集群安装。

先决条件

- 您的 control plane 已初始化。
- 已完成初始 Operator 配置。

流程

1. 使用以下命令确认所有集群组件都在线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION   AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.16.0    True        False         False      19m
baremetal                                  4.16.0    True        False         False      37m
cloud-credential                           4.16.0    True        False         False      40m
cluster-autoscaler                         4.16.0    True        False         False      37m
```

```
config-operator                          4.16.0    True        False         False      38m
console                                  4.16.0    True        False         False      26m
csi-snapshot-controller                  4.16.0    True        False         False      37m
dns                                      4.16.0    True        False         False      37m
etcd                                     4.16.0    True        False         False      36m
image-registry                           4.16.0    True        False         False      31m
ingress                                  4.16.0    True        False         False      30m
insights                                 4.16.0    True        False         False      31m
kube-apiserver                           4.16.0    True        False         False      26m
kube-controller-manager                  4.16.0    True        False         False      36m
kube-scheduler                           4.16.0    True        False         False      36m
kube-storage-version-migrator            4.16.0    True        False         False      37m
machine-api                              4.16.0    True        False         False      29m
machine-approver                         4.16.0    True        False         False      37m
machine-config                           4.16.0    True        False         False      36m
marketplace                              4.16.0    True        False         False      37m
monitoring                               4.16.0    True        False         False      29m
network                                  4.16.0    True        False         False      38m
node-tuning                              4.16.0    True        False         False      37m
openshift-apiserver                      4.16.0    True        False         False      32m
openshift-controller-manager             4.16.0    True        False         False      30m
openshift-samples                        4.16.0    True        False         False      32m
operator-lifecycle-manager               4.16.0    True        False         False      37m
operator-lifecycle-manager-catalog       4.16.0    True        False         False      37m
operator-lifecycle-manager-packageserver 4.16.0    True        False         False      32m
service-ca                               4.16.0    True        False         False      38m
storage                                  4.16.0    True        False         False      37m
```

另外，当所有集群都可用时，以下命令会通知您。它还检索并显示凭证：

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

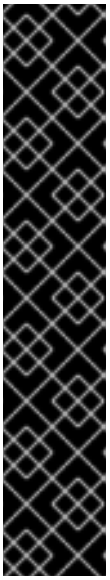
1

对于 <installation_directory>，请指定安装文件保存到的目录的路径。

输出示例

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

Cluster Version Operator 完成从 Kubernetes API 服务器部署 OpenShift Container Platform 集群时，该命令会成功。



重要

- 安装程序生成的 Ignition 配置文件包含 24 小时后过期的证书，然后在该时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 node-bootstrapper 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。

- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

2. 确认 Kubernetes API 服务器正在与 pod 通信。

   a. 要查看所有 pod 的列表，请使用以下命令：

```
$ oc get pods --all-namespaces
```

输出示例

```
NAMESPACE                  NAME                                        READY   STATUS
RESTARTS   AGE
openshift-apiserver-operator     openshift-apiserver-operator-85cb746d55-zqhs8
1/1     Running     1          9m
openshift-apiserver              apiserver-67b9g                          1/1     Running     0
3m
openshift-apiserver              apiserver-ljcmx                          1/1     Running     0
1m
openshift-apiserver              apiserver-z25h4                          1/1     Running     0
2m
openshift-authentication-operator authentication-operator-69d5d8bf84-vh2n8
1/1     Running     0          5m
...
```

.

b. 使用以下命令，查看上一命令的输出中所列 pod 的日志：

```
$ oc logs <pod_name> -n <namespace> 1
```

1 指定 pod 名称和命名空间，如上一命令的输出中所示。

如果 pod 日志显示，Kubernetes API 服务器可以与集群机器通信。

3. 对于使用光纤通道协议(FCP)的安装，还需要额外的步骤才能启用多路径。不要在安装过程中启用多路径。

如需更多信息，请参阅 *安装后机器配置任务* 文档中的"使用 RHCOS 上使用内核参数启用多路径"。

15.2.17. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于 远程健康监控

15.2.18. 后续步骤

- 验证安装.
- 自定义集群。

- 如果需要，您可以选择 不使用远程健康报告。
- 设置 registry 并配置 registry 存储。

15.3. 使用自定义网络安装用户置备的裸机集群

在 OpenShift Container Platform 4.16 中，您可以使用自定义的网络配置选项在裸机环境中安装集群。通过自定义网络配置，您的集群可以与环境中现有的 IP 地址分配共存，并与现有的 MTU 和 VXLAN 配置集成。

自定义 OpenShift Container Platform 网络时，必须在安装过程中设置大多数网络配置参数。您只能在正在运行的集群中修改 kubeProxy 网络配置参数。

15.3.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读有关 选择集群安装方法的文档，并为用户准备它。
- 如果您使用防火墙并计划使用 Telemetry 服务，则将防火墙配置为允许集群需要访问的站点。

15.3.2. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。

- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类型的基础架构上执行受限网络安装。在此过程中，您可以下载所需的内容，并使用它为镜像 registry 填充安装软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群前，您要更新镜像 registry 的内容。

其他资源

- 有关 在您置备的裸机基础架构上执行受限网络安装的更多信息，请参阅在受限网络上安装用户置备的裸机集群。

15.3.3. 具有用户置备基础架构的集群的要求

对于包含用户置备的基础架构的集群，您必须部署所有所需的机器。

本节论述了在用户置备的基础架构上部署 OpenShift Container Platform 的要求。

15.3.3.1. 集群安装所需的机器

最小的 OpenShift Container Platform 集群需要以下主机：

表 15.11. 最低所需的主机

主机	描述
一个临时 bootstrap 机器	集群需要 bootstrap 机器在三台 control plane 机器上部署 OpenShift Container Platform 集群。您可在安装集群后删除 bootstrap 机器。
三台 control plane 机器	control plane 机器运行组成 control plane 的 Kubernetes 和 OpenShift Container Platform 服务。
至少两台计算机器，也称为 worker 机器。	OpenShift Container Platform 用户请求的工作负载在计算机器上运行。

注意

作为例外，您可以在裸机集群中运行零台计算机器，它们仅由三台 control plane 机器组成。这为集群管理员和开发人员提供了更小、效率更高的集群，用于测试、开发和生产。不支持运行一台计算机器。

重要

要保持集群的高可用性，请将独立的物理主机用于这些集群机器。

bootstrap 和 control plane 机器必须使用 Red Hat Enterprise Linux CoreOS(RHCOS)作为操作系统。但是，计算机器可以在 Red Hat Enterprise Linux CoreOS(RHCOS)、Red Hat Enterprise Linux(RHEL) 8.6 和更高的版本。

请注意，RHCOS 基于 Red Hat Enterprise Linux(RHEL) 9.2，并继承其所有硬件认证和要求。查看红帽企业 Linux 技术功能和限制。

15.3.3.2. 集群安装的最低资源要求

每台集群机器都必须满足以下最低要求：

表 15.12. 最低资源要求

机器	操作系统	CPU [1]	RAM	Storage	每秒输入/输出 (IOPS) [2]
bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane（控制平面）	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS、RHEL 8.6 及更新版本 [3]	2	8 GB	100 GB	300

1. 当未启用并发多线程 (SMT) 或超线程时，一个 CPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比例：（每个内核数的线程）× sockets = CPU。

2. OpenShift Container Platform 和 Kubernetes 对磁盘性能非常敏感，建议使用更快的存储

速度，特别是 control plane 节点上需要 10 ms p99 fsync 持续时间的 etcd。请注意，在许多云平台上，存储大小和 IOPS 可一起扩展，因此您可能需要过度分配存储卷来获取足够的性能。

3. 与所有用户置备的安装一样，如果您选择在集群中使用 RHEL 计算机器，则负责所有操作系统生命周期管理和维护，包括执行系统更新、应用补丁和完成所有其他必要的任务。RHEL 7 计算机器的使用已弃用，并已在 OpenShift Container Platform 4.10 及更新的版本中删除。

注意

从 OpenShift Container Platform 版本 4.13 开始，RHCOS 基于 RHEL 版本 9.2，它更新了微架构要求。以下列表包含每个架构需要的最小指令集架构 (ISA)：

- x86-64 体系结构需要 x86-64-v2 ISA
- ARM64 架构需要 ARMv8.0-A ISA
- IBM Power 架构需要 Power 9 ISA
- s390x 架构需要 z14 ISA

如需更多信息，请参阅 RHEL 架构。

如果平台的实例类型满足集群机器的最低要求，则 OpenShift Container Platform 支持使用它。

其他资源

- 优化存储

15.3.3.3. 证书签名请求管理

在使用您置备的基础架构时，集群只能有限地访问自动机器管理，因此您必须提供一种在安装后批准集群证书签名请求 (CSR) 的机制。kube-controller-manager 只能批准 kubelet 客户端 CSR。machine-approver 无法保证使用 kubelet 凭证请求的提供证书的有效性，因为它不能确认是正确的机器发出了该请求。您必须决定并实施一种方法，以验证 kubelet 提供证书请求的有效性并进行批准。

其他资源

- 有关在裸机 环境中部署三节点集群的详情，请参阅配置 三节点集群。
- 有关 在安装后批准集群证书签名请求的更多信息，请参阅批准机器 的证书签名请求。

15.3.3.4. 用户置备的基础架构对网络的要求

所有 Red Hat Enterprise Linux CoreOS(RHCOS)机器都需要在启动时在 initramfs 中配置联网，以获取它们的 Ignition 配置文件。

在初次启动过程中，机器需要 IP 地址配置，该配置通过 DHCP 服务器或静态设置，提供所需的引导选项。建立网络连接后，机器会从 HTTP 或 HTTPS 服务器下载 Ignition 配置文件。然后，Ignition 配置文件用于设置每台机器的确切状态。Machine Config Operator 在安装后完成对机器的更多更改，如应用新证书或密钥。

建议使用 DHCP 服务器对集群机器进行长期管理。确保 DHCP 服务器已配置为向集群机器提供持久的 IP 地址、DNS 服务器信息和主机名。

注意

如果用户置备的基础架构没有 DHCP 服务，您可以在 RHCOS 安装时向节点提供 IP 网络配置和 DNS 服务器地址。如果要从 ISO 镜像安装，这些参数可作为引导参数传递。如需有关静态 IP 置备和高级网络选项的更多信息，请参阅 *安装 RHCOS 并启动 OpenShift Container Platform bootstrap 过程* 部分。

Kubernetes API 服务器必须能够解析集群机器的节点名称。如果 API 服务器和 worker 节点位于不同的区域中，您可以配置默认 DNS 搜索区域，以允许 API 服务器解析节点名称。另一种支持的方法是始终通过节点对象和所有 DNS 请求中的完全限定域名引用主机。

15.3.3.4.1. 通过 DHCP 设置集群节点主机名

在 Red Hat Enterprise Linux CoreOS(RHCOS)机器上，主机名是通过 NetworkManager 设置的。默认情况下，机器通过 DHCP 获取其主机名。如果主机名不是由 DHCP 提供，请通过内核参数或者其它方法进行静态设置，请通过反向 DNS 查找获取。反向 DNS 查找在网络初始化后进行，可能需要一些时间来解决。其他系统服务可以在此之前启动，并将主机名检测为 localhost 或类似的内容。您可以使用 DHCP 为每个集群节点提供主机名来避免这种情况。

另外，通过 DHCP 设置主机名可以绕过实施 DNS split-horizon 的环境中的手动 DNS 记录名称配置错误。

15.3.3.4.2. 网络连接要求

您必须配置机器之间的网络连接，以允许 OpenShift Container Platform 集群组件进行通信。每台机器都必须能够解析集群中所有其他机器的主机名。

本节详细介绍了所需的端口。

重要

在连接的 OpenShift Container Platform 环境中，所有节点都需要访问互联网才能为平台容器拉取镜像，并向红帽提供遥测数据。

表 15.13. 用于全机器到所有机器通信的端口

协议	port	描述
ICMP	N/A	网络可访问性测试
TCP	**1936**	指标
	9000-**9999**	主机级别的服务，包括端口 9**100**-**9101** 上的节点导出器，以及端口 **9099** 上的 Cluster Version Operator。
	10250-**10259**	Kubernetes 保留的默认端口
UDP	**4789**	VXLAN
	6081	Geneve
	9000-**9999**	主机级别的服务，包括端口 **9100**到**9101** 上的节点导出器。
	500	IPsec IKE 数据包
	4500	IPsec NAT-T 数据包
	123	UDP 端口 **123** 上的网络时间协议 (NTP) 如果配置了外部 NTP 时间服务器，需要打开 UDP 端口 **123**。
TCP/UDP	**30000**-**32767**	Kubernetes 节点端口

协议	port	描述
ESP	N/A	IPsec Encapsulating Security Payload(ESP)

表 15.14. 用于所有机器控制平面通信的端口

协议	port	描述
TCP	**6443**	Kubernetes API

表 15.15. control plane 机器用于 control plane 机器通信的端口

协议	port	描述
TCP	**2379-2380**	etcd 服务器和对等端口

用户置备的基础架构的 NTP 配置

OpenShift Container Platform 集群被配置为默认使用公共网络时间协议(NTP)服务器。如果要使用本地企业 NTP 服务器，或者集群部署在断开连接的网络中，您可以将集群配置为使用特定的时间服务器。如需更多信息，*请参阅配置 chrony 时间服务* 的文档。

如果 DHCP 服务器提供 NTP 服务器信息，Red Hat Enterprise Linux CoreOS(RHCOS)机器上的 chrony 时间服务会读取信息，并可以把时钟与 NTP 服务器同步。

其他资源

- **配置 chrony 时间服务**

15.3.3.5. 用户置备的 DNS 要求

在 OpenShift Container Platform 部署中，以下组件需要 DNS 名称解析：

- **The Kubernetes API**
- **OpenShift Container Platform 应用程序通配符**

- bootstrap、control plane 和计算机器

Kubernetes API、bootstrap 机器、control plane 机器和计算机器也需要反向 DNS 解析。

DNS A/AAAA 或 CNAME 记录用于名称解析，PTR 记录用于反向名称解析。反向记录很重要，因为 Red Hat Enterprise Linux CoreOS(RHCOS)使用反向记录为所有节点设置主机名，除非 DHCP 提供主机名。另外，反向记录用于生成 OpenShift Container Platform 需要操作的证书签名请求(CSR)。

注意

建议使用 DHCP 服务器为每个群集节点提供主机名。如需更多信息，*请参阅用户置备的基础架构部分的 DHCP 建议*。

用户置备的 OpenShift Container Platform 集群需要以下 DNS 记录，这些记录必须在安装前就位。在每个记录中，<cluster_name> 是集群名称，<base_domain> 是您在 install-config.yaml 文件中指定的基域。完整的 DNS 记录采用以下形式：<component>.<cluster_name>.<base_domain>.。

表 15.16. 所需的 DNS 记录

组件	记录	描述
Kubernetes API	**api.<cluster_name>.<base_domain>.**	DNS A/AAAA 或 CNAME 记录，以及用于标识 API 负载均衡器的 DNS PTR 记录。这些记录必须由集群外的客户端和集群中的所有节点解析。
	api-int.<cluster_name>.<base_domain>.	DNS A/AAAA 或 CNAME 记录，以及用于内部标识 API 负载均衡器的 DNS PTR 记录。这些记录必须可以从集群中的所有节点解析。  **重要** API 服务器必须能够根据 Kubernetes 中记录的主机名解析 worker 节点。如果 API 服务器无法解析节点名称，则代理的 API 调用会失败，且您无法从 pod 检索日志。

组件	记录	描述
Routes	***.apps.<cluster_name>. <base_domain>.**	通配符 DNS A/AAAA 或 CNAME 记录，指向应用程序入口负载均衡器。应用程序入口负载均衡器以运行 Ingress Controller Pod 的机器为目标。默认情况下，Ingress Controller Pod 在计算机器上运行。这些记录必须由集群外的客户端和集群中的所有节点解析。 例如，console **-openshift-console.apps. <cluster_name>.<base_domain>** 用作到 OpenShift Container Platform 控制台的通配符路由。
bootstrap 机器	**bootstrap.<cluster_name>. <base_domain>.**	DNS A/AAAA 或 CNAME 记录，以及用于标识 bootstrap 机器的 DNS PTR 记录。这些记录必须由集群中的节点解析。
control plane 机器	**<control_plane><n>. <cluster_name>. <base_domain>.**	DNS A/AAAA 或 CNAME 记录，以识别 control plane 节点的每台机器。这些记录必须由集群中的节点解析。
计算机器	**<compute><n>. <cluster_name>. <base_domain>.**	DNS A/AAAA 或 CNAME 记录，用于识别 worker 节点的每台机器。这些记录必须由集群中的节点解析。

注意

在 OpenShift Container Platform 4.4 及更新的版本中，您不需要在 DNS 配置中指定 etcd 主机和 SRV 记录。

提示

您可以使用 dig 命令验证名称和反向名称解析。如需了解详细的 *验证步骤，请参阅为用户置备的基础架构验证 DNS 解析* 一节。

15.3.3.5.1. 用户置备的集群的 DNS 配置示例

本节提供 A 和 PTR 记录配置示例，它们满足了在用户置备的基础架构上部署 OpenShift Container Platform 的 DNS 要求。样本不是为选择一个 DNS 解决方案提供建议。

在这个示例中，集群名称为 ocp4，基域是 example.com。

用户置备的集群的 DNS A 记录配置示例

以下示例是 BIND 区域文件，其中显示了用户置备的集群中名称解析的 A 记录示例。

例 15.4. DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
  2019070700 ; serial
  3H  ; refresh (3 hours)
  30M  ; retry (30 minutes)
  2W  ; expiry (2 weeks)
  1W )  ; minimum (1 week)
 IN NS ns1.example.com.
 IN MX 10 smtp.example.com.
;
;
ns1.example.com.  IN A 192.168.1.5
smtp.example.com.  IN A 192.168.1.5
;
helper.example.com.  IN A 192.168.1.5
helper.ocp4.example.com. IN A 192.168.1.5
;
api.ocp4.example.com.  IN A 192.168.1.5 1
api-int.ocp4.example.com. IN A 192.168.1.5 2
;
*.apps.ocp4.example.com. IN A 192.168.1.5 3
;
bootstrap.ocp4.example.com. IN A 192.168.1.96 4
;
control-plane0.ocp4.example.com. IN A 192.168.1.97 5
control-plane1.ocp4.example.com. IN A 192.168.1.98 6
control-plane2.ocp4.example.com. IN A 192.168.1.99 7
;
compute0.ocp4.example.com. IN A 192.168.1.11 8
compute1.ocp4.example.com. IN A 192.168.1.7 9
;
;EOF
```

1 为 Kubernetes API 提供名称解析。记录引用 API 负载均衡器的 IP 地址。

2 为 Kubernetes API 提供名称解析。记录引用 API 负载均衡器的 IP 地址，用于内部集群通信。

3 为通配符路由提供名称解析。记录引用应用程序入口负载均衡器的 IP 地址。应用程序入口负载均衡器以运行 Ingress Controller Pod 的机器为目标。默认情况下，Ingress Controller Pod 在计算机器上运行。

注意

在这个示例中，将相同的负载均衡器用于 Kubernetes API 和应用入口流量。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。

4 为 bootstrap 机器提供名称解析。

5 6 7 为 control plane 机器提供名称解析。

8 9 为计算机器提供名称解析。

用户置备的集群的 DNS PTR 记录配置示例

以下示例 BIND 区域文件显示了用户置备的集群中反向名称解析的 PTR 记录示例。

例 15.5. 反向记录的 DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
  2019070700 ; serial
  3H  ; refresh (3 hours)
  30M  ; retry (30 minutes)
  2W  ; expiry (2 weeks)
  1W )  ; minimum (1 week)
 IN NS ns1.example.com.
;
5.1.168.192.in-addr.arpa. IN PTR api.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. IN PTR api-int.ocp4.example.com. 2
;
96.1.168.192.in-addr.arpa. IN PTR bootstrap.ocp4.example.com. 3
;
97.1.168.192.in-addr.arpa. IN PTR control-plane0.ocp4.example.com. 4
98.1.168.192.in-addr.arpa. IN PTR control-plane1.ocp4.example.com. 5
99.1.168.192.in-addr.arpa. IN PTR control-plane2.ocp4.example.com. 6
;
11.1.168.192.in-addr.arpa. IN PTR compute0.ocp4.example.com. 7
```

```
7.1.168.192.in-addr.arpa. IN PTR compute1.ocp4.example.com. 8
;
;EOF
```

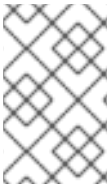
1 为 Kubernetes API 提供反向 DNS 解析。PTR 记录引用 API 负载均衡器的记录名称。

2 为 Kubernetes API 提供反向 DNS 解析。PTR 记录引用 API 负载均衡器的记录名称，用于内部集群通信。

3 为 bootstrap 机器提供反向 DNS 解析。

4 5 6 为 control plane 机器提供反向 DNS 解析。

7 8 为计算机器提供反向 DNS 解析。



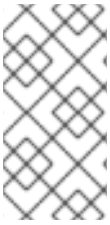
注意

OpenShift Container Platform 应用程序通配符不需要 PTR 记录。

- 验证用户置备的基础架构的 DNS 解析

15.3.3.6. 用户置备的基础架构的负载均衡要求

在安装 OpenShift Container Platform 前，您必须置备 API 和应用程序入口负载均衡基础架构。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。



注意

如果要使用 Red Hat Enterprise Linux (RHEL) 实例部署 API 和应用程序入口负载均衡器，您必须单独购买 RHEL 订阅。

负载平衡基础架构必须满足以下要求：

1. **API 负载均衡器 ：提供一个通用端点，供用户（包括人工和机器）与平台交互和配置。配置以下条件：**

   - **仅第 4 层负载均衡.这可被称为 Raw TCP 或 SSL Passthrough 模式。**
   - **无状态负载平衡算法。这些选项根据负载均衡器的实施而有所不同。**

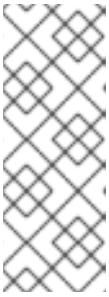
重要

不要为 API 负载均衡器配置会话持久性。为 Kubernetes API 服务器配置会话持久性可能会导致出现过量 OpenShift Container Platform 集群应用程序流量，以及过量的在集群中运行的 Kubernetes API。

在负载均衡器的前端和后端配置以下端口：

表 15.17. API 负载均衡器

port	后端机器（池成员）	internal	外部	描述
6443	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。您必须为 API 服务器健康检查探测配置 **/readyz** 端点。	X	X	Kubernetes API 服务器
22623	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。	X		机器配置服务器



注意

负载均衡器必须配置为，从 API 服务器关闭 /readyz 端点到从池中移除 API 服务器实例时最多需要 30 秒。在 /readyz 返回错误或健康后的时间范围内，端点必须被删除或添加。每 5 秒或 10 秒探测一次，有两个成功请求处于健康状态，三个成为不健康的请求是经过良好测试的值。

2.

应用程序入口负载均衡器 ：为应用程序流量从集群外部流提供入口点。OpenShift Container Platform 集群需要正确配置入口路由器。

配置以下条件：

- 仅第 4 层负载均衡.这可被称为 Raw TCP 或 SSL Passthrough 模式。
- 建议根据可用选项以及平台上托管的应用程序类型，使用基于连接的或基于会话的持久性。

提示

如果应用程序入口负载均衡器可以看到客户端的真实 IP 地址，启用基于 IP 的会话持久性可以提高使用端到端 TLS 加密的应用程序的性能。

在负载均衡器的前端和后端配置以下端口：

表 15.18. 应用程序入口负载均衡器

port	后端机器（池成员）	internal	外部	描述
443	默认情况下，运行 Ingress Controller Pod、计算或 worker 的机器。	X	X	HTTPS 流量
80	默认情况下，运行 Ingress Controller Pod、计算或 worker 的机器。	X	X	HTTP 流量

注意

如果要部署一个带有零计算节点的三节点集群，Ingress Controller Pod 在 control plane 节点上运行。在三节点集群部署中，您必须配置应用程序入口负载均衡器，将 HTTP 和 HTTPS 流量路由到 control plane 节点。

15.3.3.6.1. 用户置备的集群的负载均衡器配置示例

本节提供了一个满足用户置备集群的负载均衡要求的 API 和应用程序入口负载均衡器配置示例。示例是 HAProxy 负载均衡器的 /etc/haproxy/haproxy.cfg 配置。这个示例不是为选择一个负载平衡解决方案提供建议。

在这个示例中，将相同的负载均衡器用于 Kubernetes API 和应用入口流量。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。

注意

如果您使用 HAProxy 作为负载均衡器，并且 SELinux 设置为 enforcing，您必须通过运行 setsebool -P haproxy_connect_any=1 来确保 HAProxy 服务可以绑定到配置的 TCP 端口。

例 15.6. API 和应用程序入口负载均衡器配置示例

```
global
  log         127.0.0.1 local2
  pidfile     /var/run/haproxy.pid
  maxconn     4000
  daemon
defaults
  mode                    http
  log                     global
  option                  dontlognull
  option http-server-close
  option                  redispatch
  retries                 3
  timeout http-request    10s
  timeout queue           1m
  timeout connect         10s
  timeout client          1m
  timeout server          1m
  timeout http-keep-alive 10s
  timeout check           10s
  maxconn                 3000
listen api-server-6443 1
  bind *:6443
  mode tcp
  option  httpchk GET /readyz HTTP/1.0
```

```
  option  log-health-checks
  balance roundrobin
  server bootstrap bootstrap.ocp4.example.com:6443 verify none check check-ssl inter 10s
fall 2 rise 3 backup 2
  server master0 master0.ocp4.example.com:6443 weight 1 verify none check check-ssl
inter 10s fall 2 rise 3
  server master1 master1.ocp4.example.com:6443 weight 1 verify none check check-ssl
inter 10s fall 2 rise 3
  server master2 master2.ocp4.example.com:6443 weight 1 verify none check check-ssl
inter 10s fall 2 rise 3
listen machine-config-server-22623 3
  bind *:22623
  mode tcp
  server bootstrap bootstrap.ocp4.example.com:22623 check inter 1s backup 4
  server master0 master0.ocp4.example.com:22623 check inter 1s
  server master1 master1.ocp4.example.com:22623 check inter 1s
  server master2 master2.ocp4.example.com:22623 check inter 1s
listen ingress-router-443 5
  bind *:443
  mode tcp
  balance source
  server compute0 compute0.ocp4.example.com:443 check inter 1s
  server compute1 compute1.ocp4.example.com:443 check inter 1s
listen ingress-router-80 6
  bind *:80
  mode tcp
  balance source
  server compute0 compute0.ocp4.example.com:80 check inter 1s
  server compute1 compute1.ocp4.example.com:80 check inter 1s
```

1

端口 6443 处理 Kubernetes API 流量并指向 control plane 机器。

2 4

bootstrap 条目必须在 OpenShift Container Platform 集群安装前就位，且必须在 bootstrap 过程完成后删除它们。

3

端口 22623 处理机器配置服务器流量并指向 control plane 机器。

5

端口 443 处理 HTTPS 流量，并指向运行 Ingress Controller pod 的机器。默认情况下，Ingress Controller Pod 在计算机器上运行。

6

端口 80 处理 HTTP 流量，并指向运行 Ingress Controller pod 的机器。默认情况下，Ingress Controller Pod 在计算机器上运行。

注意

如果要部署一个带有零计算节点的三节点集群，Ingress Controller Pod 在 control plane 节点上运行。在三节点集群部署中，您必须配置应用程序入口负载均衡器，将 HTTP 和 HTTPS 流量路由到 control plane 节点。

提示

如果您使用 HAProxy 作为负载均衡器，您可以通过在 HAProxy 节点上运行 netstat -nltupe 来检查 haproxy 进程是否在侦听端口 6443、22623、443 和 80。

15.3.4. 准备用户置备的基础架构

在用户置备的基础架构上安装 OpenShift Container Platform 之前，您必须准备底层基础架构。

本节详细介绍了设置集群基础架构以准备 OpenShift Container Platform 安装所需的高级别步骤。这包括为您的集群节点配置 IP 网络和网络连接，通过防火墙启用所需的端口，以及设置所需的 DNS 和负载均衡基础架构。

准备后，集群基础架构必须满足 *带有用户置备的基础架构部分的集群要求*。

先决条件

- 您已参阅 OpenShift Container Platform 4.x Tested Integrations 页面。
- 您已查看了 *具有用户置备基础架构的集群要求部分中详述的基础架构* 要求。

流程

1. 如果您使用 DHCP 向集群节点提供 IP 网络配置，请配置 DHCP 服务。
   a. 将节点的持久 IP 地址添加到您的 DHCP 服务器配置。在您的配置中，将相关网络接口的 MAC 地址与每个节点的预期 IP 地址匹配。

b.

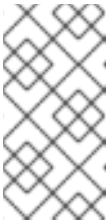
当您使用 DHCP 为集群机器配置 IP 寻址时，机器还通过 DHCP 获取 DNS 服务器信息。定义集群节点通过 DHCP 服务器配置使用的持久性 DNS 服务器地址。

注意

如果没有使用 DHCP 服务，则必须在 RHCOS 安装时为节点提供 IP 网络配置和 DNS 服务器地址。如果要从 ISO 镜像安装，这些参数可作为引导参数传递。如需有关静态 IP 置备和高级网络选项的更多信息，请参阅 *安装 RHCOS 并启动 OpenShift Container Platform bootstrap 过程* 部分。

c.

在 DHCP 服务器配置中定义集群节点的主机名。有关 *主机名注意事项的详情，请参阅通过 DHCP 设置集群节点* 主机名部分。



注意

如果没有使用 DHCP 服务，集群节点可以通过反向 DNS 查找来获取其主机名。

2.

确保您的网络基础架构提供集群组件之间所需的网络连接。有关 *要求的详情，请参阅用户置备的基础架构* 的网络要求部分。

3.

将防火墙配置为启用 OpenShift Container Platform 集群组件进行通信所需的端口。*如需有关所需端口的详细信息，请参阅用户置备的基础架构* 部分的网络要求。

重要

默认情况下，OpenShift Container Platform 集群可以访问端口 1936，因为每个 control plane 节点都需要访问此端口。

避免使用 Ingress 负载均衡器公开此端口，因为这样做可能会导致公开敏感信息，如统计信息和指标（与 Ingress Controller 相关的统计信息和指标）。

4.

为集群设置所需的 DNS 基础架构。

a. 为 Kubernetes API、应用程序通配符、bootstrap 机器、control plane 机器和计算机器配置 DNS 名称解析。

b. 为 Kubernetes API、bootstrap 机器、control plane 机器和计算机器配置反向 DNS 解析。

如需有关 *OpenShift Container Platform DNS 要求的更多信息，请参阅用户置备* DNS 要求部分。

5. 验证您的 DNS 配置。

a. 从安装节点，针对 Kubernetes API 的记录名称、通配符路由和集群节点运行 DNS 查找。验证响应中的 IP 地址是否与正确的组件对应。

b. 从安装节点，针对负载均衡器和集群节点的 IP 地址运行反向 DNS 查找。验证响应中的记录名称是否与正确的组件对应。

有关详细的 *DNS 验证步骤，请参阅用户置备的基础架构* 验证 DNS 解析部分。

6. 置备所需的 API 和应用程序入口负载平衡基础架构。有关 *要求的更多信息，请参阅用户置备的基础架构的负载平衡* 要求部分。

注意

某些负载平衡解决方案要求在初始化负载平衡之前，对群集节点进行 DNS 名称解析。

其他资源

- 具有用户置备基础架构的集群的要求
- 安装 RHCOS 并启动 OpenShift Container Platform bootstrap 过程
- 通过 DHCP 设置集群节点主机名

- 高级 RHCOS 安装配置
- 用户置备的基础架构对网络的要求
- 用户置备的 DNS 要求
- 验证用户置备的基础架构的 DNS 解析
- 用户置备的基础架构的负载均衡要求

15.3.5. 验证用户置备的基础架构的 DNS 解析

您可以在在用户置备的基础架构上安装 OpenShift Container Platform 前验证 DNS 配置。

重要

本节中详述的验证步骤必须在安装集群前成功。

先决条件

- 已为您的用户置备的基础架构配置了所需的 DNS 记录。

流程

1. 从安装节点，针对 Kubernetes API 的记录名称、通配符路由和集群节点运行 DNS 查找。验证响应中包含的 IP 地址是否与正确的组件对应。

   a. 对 Kubernetes API 记录名称执行查询。检查结果是否指向 API 负载均衡器的 IP 地址：

   ```
   $ dig +noall +answer @<nameserver_ip> api.<cluster_name>.<base_domain> 1
   ```

   1

输出示例

```
api.ocp4.example.com.  604800 IN A 192.168.1.5
```

b. 对 Kubernetes 内部 API 记录名称执行查询。检查结果是否指向 API 负载均衡器的 IP 地址：

```
$ dig +noall +answer @<nameserver_ip> api-int.<cluster_name>.<base_domain>
```

输出示例

```
api-int.ocp4.example.com.  604800 IN A 192.168.1.5
```

c. 测试 *.apps.<cluster_name>.<base_domain> DNS 通配符查找示例。所有应用程序通配符查询都必须解析为应用程序入口负载均衡器的 IP 地址：

```
$ dig +noall +answer @<nameserver_ip> random.apps.<cluster_name>.
<base_domain>
```

输出示例

```
random.apps.ocp4.example.com.  604800 IN A 192.168.1.5
```

注意

在示例中，将相同的负载均衡器用于 Kubernetes API 和应用程序入口流量。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。

您可以使用另一个通配符值替换 random。例如，您可以查询到 OpenShift Container Platform 控制台的路由：

```
$ dig +noall +answer @<nameserver_ip> console-openshift-console.apps.
<cluster_name>.<base_domain>
```

输出示例

```
console-openshift-console.apps.ocp4.example.com. 604800 IN A 192.168.1.5
```

d. 针对 bootstrap DNS 记录名称运行查询。检查结果是否指向 bootstrap 节点的 IP 地址：

```
$ dig +noall +answer @<nameserver_ip> bootstrap.<cluster_name>.
<base_domain>
```

输出示例

```
bootstrap.ocp4.example.com.  604800 IN A 192.168.1.96
```

e. 使用此方法对 control plane 和计算节点的 DNS 记录名称执行查找。检查结果是否与每个节点的 IP 地址对应。

2. 从安装节点，针对负载均衡器和集群节点的 IP 地址运行反向 DNS 查找。验证响应中包含的记录名称是否与正确的组件对应。

a. 对 API 负载均衡器的 IP 地址执行反向查找。检查响应是否包含 Kubernetes API 和 Kubernetes 内部 API 的记录名称：

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.5
```

输出示例

```
5.1.168.192.in-addr.arpa. 604800 IN PTR api-int.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. 604800 IN PTR api.ocp4.example.com. 2
```

1 为 Kubernetes 内部 API 提供记录名称。

2 为 Kubernetes API 提供记录名称。

注意

OpenShift Container Platform 应用程序通配符不需要 PTR 记录。针对应用程序入口负载均衡器的 IP 地址解析反向 DNS 解析不需要验证步骤。

b. 对 bootstrap 节点的 IP 地址执行反向查找。检查结果是否指向 bootstrap 节点的 DNS 记录名称：

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.96
```

输出示例

```
96.1.168.192.in-addr.arpa. 604800 IN PTR bootstrap.ocp4.example.com.
```

c.

使用此方法对 control plane 和计算节点的 IP 地址执行反向查找。检查结果是否与每个节点的 DNS 记录名称对应。

其他资源

- 用户置备的 DNS 要求
- 用户置备的基础架构的负载均衡要求

15.3.6. 为集群节点 SSH 访问生成密钥对

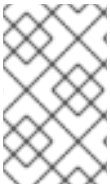
在 OpenShift Container Platform 安装过程中，您可以为安装程序提供 SSH 公钥。密钥通过它们的 Ignition 配置文件传递给 Red Hat Enterprise Linux CoreOS(RHCOS)节点，用于验证对节点的 SSH 访问。密钥添加到每个节点上 core 用户的 ~/.ssh/authorized_keys 列表中，这将启用免密码身份验证。

将密钥传递给节点后，您可以使用密钥对作为用户 核心 通过 SSH 连接到 RHCOS 节点。若要通过 SSH 访问节点，必须由 SSH 为您的本地用户管理私钥身份。

如果要通过 SSH 连接到集群节点来执行安装调试或灾难恢复，则必须在安装过程中提供 SSH 公钥。./openshift-install gather 命令还需要在集群节点上设置 SSH 公钥。

重要

不要在生产环境中跳过这个过程，在生产环境中需要灾难恢复和调试。



注意

您必须使用本地密钥，而不是使用特定平台方法配置 的密钥，如 AWS 密钥对。

流程

1. 如果您在本地计算机上没有可用于在集群节点上进行身份验证的现有 SSH 密钥对，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 ~/.ssh/id_ed25519。如果您已有密钥对，请确保您的公钥位于 ~/.ssh 目录中。

注意

如果您计划在 x86_64、ppc64le 和 s390x 架构上安装使用 RHEL 加密库（这些加密库已提交给 NIST 用于 FIPS 140-2/140-3 验证）的 OpenShift Container Platform 集群，则不要创建使用 ed25519 算法的密钥。相反，创建一个使用 rsa 或 ecdsa 算法的密钥。

2. 查看公共 SSH 密钥：

```
$ cat <path>/<file_name>.pub
```

例如，运行以下命令来查看 ~/.ssh/id_ed25519.pub 公钥：

```
$ cat ~/.ssh/id_ed25519.pub
```

3. 将 SSH 私钥身份添加到本地用户的 SSH 代理（如果尚未添加）。在集群节点上，或者要使用 ./openshift-install gather 命令，需要对该密钥进行 SSH 代理管理，才能在集群节点上进行免密码 SSH 身份验证。

注意

在某些发行版中，自动管理默认 SSH 私钥身份，如 ~/.ssh/id_rsa 和 ~/.ssh/id_dsa。

a. 如果 ssh-agent 进程尚未为您的本地用户运行，请将其作为后台任务启动：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果集群处于 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

4. 将 SSH 私钥添加到 ssh-agent ：

```
$ ssh-add <path>/<file_name> 1
```

1 指定 SSH 私钥的路径和文件名，如 ~/.ssh/id_ed25519.pub

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

后续步骤

- 安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

其他资源

- 验证节点健康状况

15.3.7. 获取安装程序

在安装 OpenShift Container Platform 前，将安装文件下载到您用于安装的主机上。

先决条件

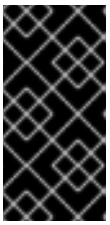
- 您有一台运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB。

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐户，请使用您的凭证登录。如果没有，请创建一个帐户。

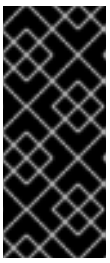
2. 选择您的基础架构供应商。

3. 进入到安装类型的页面，下载与您的主机操作系统和架构对应的安装程序，并将该文件放在您要存储安装配置文件的目录中。



重要

安装程序会在用来安装集群的计算机上创建几个文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。



重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，请为特定云供应商完成 OpenShift Container Platform 卸载流程。

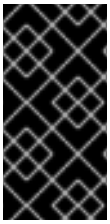
4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret。此 pull secret 允许您与所含授权机构提供的服务进行身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

15.3.8. 安装 OpenShift CLI

您可以安装 OpenShift CLI(oc)来使用命令行界面与 OpenShift Container Platform 进行交互。您可以在 Linux、Windows 或 macOS 上安装 oc。



重要

如果安装了旧版本的 oc，则可能无法使用 OpenShift Container Platform 4.16 中的所有命令。下载并安装新版本的 oc。

在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI(oc)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 产品变体 下拉列表中选择架构。
3. 从 版本 下拉列表中选择适当的版本。
4. 点 OpenShift v4.16 Linux Client 条目旁的 Download Now 来保存文件。
5. 解包存档：

```
$ tar xvf <file>
```

6. 将 oc 二进制文件放到 PATH 中的目录中。

要查看您的 PATH，请执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 oc 命令：

```
$ oc <command>
```

在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI(oc)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 版本 下拉列表中选择适当的版本。
3. 点 OpenShift v4.16 Windows Client 条目旁的 Download Now 来保存文件。
4. 使用 ZIP 程序解压存档。
5. 将 oc 二进制文件移到 PATH 中的目录中。

   要查看您的 PATH，请打开命令提示并执行以下命令：

   ```
   C:\> path
   ```

验证

- 安装 OpenShift CLI 后，可以使用 oc 命令：

  ```
  C:\> oc <command>
  ```

在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI(oc)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 版本 下拉列表中选择适当的版本。

3. 点 OpenShift v4.16 macOS Client 条目旁的 Download Now 来保存文件。

注意

对于 macOS arm64，请选择 OpenShift v4.16 macOS arm64 Client 条目。

4. 解包和解压存档。

5. 将 oc 二进制文件移到 PATH 的目录中。

要查看您的 PATH，请打开终端并执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 oc 命令：

```
$ oc <command>
```

15.3.9. 手动创建安装配置文件

安装集群要求您手动创建安装配置文件。

先决条件

- 您在本地机器上有一个 SSH 公钥来提供给安装程序。该密钥将用于在集群节点上进行 SSH 身份验证，以进行调试和灾难恢复。

- 已获取 OpenShift Container Platform 安装程序和集群的 pull secret。

流程

1. 创建一个安装目录来存储所需的安装资产：

```
$ mkdir <installation_directory>
```

重要

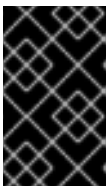
您必须创建一个目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

2. 自定义提供的 install-config.yaml 文件模板示例，并将其保存在 <installation_directory> 中。

注意

此配置文件必须命名为 install-config.yaml。

3. 备份 install-config.yaml 文件，以便您可以使用它安装多个集群。



重要

install-config.yaml 文件会在安装过程的下一步中使用。现在必须备份它。

其他资源

- 裸机的安装配置参数

15.3.9.1. 裸机 install-config.yaml 文件示例

您可以自定义 install-config.yaml 文件，以指定有关 OpenShift Container Platform 集群平台的更多详情，或修改所需参数的值。

```
apiVersion: v1
baseDomain: example.com (1)
compute: (2)
```

```
- hyperthreading: Enabled 3
  name: worker
  replicas: 0 4
controlPlane: 5
  hyperthreading: Enabled 6
  name: master
  replicas: 3 7
metadata:
  name: test 8
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14 9
    hostPrefix: 23 10
  networkType: OVNKubernetes 11
  serviceNetwork: 12
  - 172.30.0.0/16
platform:
  none: {} 13
fips: false 14
pullSecret: '{"auths": ...}' 15
sshKey: 'ssh-ed25519 AAAA...' 16
```

1

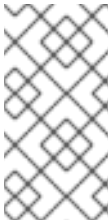
集群的基域。所有 DNS 记录都必须是这个基域的子域，并包含集群名称。

2 5

controlPlane 部分是一个单个映射，但 compute 部分是一系列映射。为满足不同数据结构的要求，compute 部分的第一行必须以连字符 - 开头，controlPlane 部分 的第一行则不以连字符开头。仅使用一个 control plane 池。

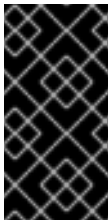
3 6

指定要启用或禁用并发多线程(SMT)还是超线程。默认情况下，启用 SMT 可提高机器中内核的性能。您可以通过将 参数值设置为 Disabled 来禁用它。如果禁用 SMT，则必须在所有集群机器中禁用它；这包括 control plane 和计算机器。



注意

默认启用并发多线程(SMT)。如果您的 BIOS 设置中没有启用 SMT，超线程 参数无效。

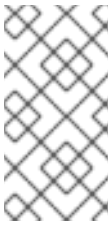


重要

如果您禁用 超线程，无论是在 BIOS 中，还是在 install-config.yaml 文件中，请确保您的容量规划考虑机器性能显著降低的情况。

4

在用户置备的基础架构上安装 OpenShift Container Platform 时，必须将这个值设置为 0。在安装程序置备的安装中，参数控制集群为您创建和管理的计算机器数量。在用户置备的安装中，您必须在完成集群安装前手动部署计算机器。



注意

如果要安装一个三节点集群，在安装 Red Hat Enterprise Linux CoreOS(RHCOS)机器时不要部署任何计算机器。

7

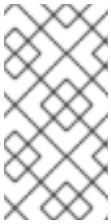
您添加到集群的 control plane 机器数量。由于集群使用这些值作为集群中的 etcd 端点数量，所以该值必须与您部署的 control plane 机器数量匹配。

8

您在 DNS 记录中指定的集群名称。

9

从中分配 Pod IP 地址的 IP 地址块。此块不得与现有物理网络重叠。这些 IP 地址用于 pod 网络。如果需要从外部网络访问 pod，您必须配置负载均衡器和路由器来管理流量。



注意

类 E CIDR 范围被保留以供以后使用。要使用 Class E CIDR 范围，您必须确保您的网络环境接受 Class E CIDR 范围内的 IP 地址。

10

分配给每个节点的子网前缀长度。例如，如果 hostPrefix 设为 23，则每个节点从 given cidr 中分配 a /23 子网，这样就能有 510（2^(32 - 23)- 2）个 pod IP 地址。如果需要从外部网络访问节点，请配置负载均衡器和路由器来管理流量。

11

要安装的集群网络插件。默认值 OVNKubernetes 是唯一支持的值。

12

用于服务 IP 地址的 IP 地址池。您只能输入一个 IP 地址池。此块不得与现有物理网络重叠。如果您需要从外部网络访问服务，请配置负载均衡器和路由器来管理流量。

13

您必须将平台设置为 none。您无法为您的平台提供额外的平台配置变量。

重要

使用平台类型 none 安装的集群无法使用一些功能，如使用 Machine API 管理计算机器。即使附加到集群的计算机器安装在通常支持该功能的平台上，也会应用这个限制。在安装后无法更改此参数。

14

是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

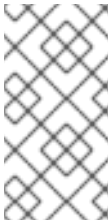
要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 Red Hat Enterprise Linux (RHEL) 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅在 FIPS 模式中安装该系统。

当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS)时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。

15

Red Hat OpenShift Cluster Manager 的 pull secret。此 pull secret 允许您与所含授权机构提供的服务进行身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

16



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 ssh-agent 进程使用的 SSH 密钥。

其他资源

- 如需有关 API 和应用程序入口负载均衡要求的更多信息，请参阅用户置备的基础架构 的负载平衡要求。

15.3.10. 网络配置阶段

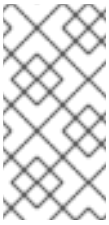
OpenShift Container Platform 安装前有两个阶段，您可以在其中自定义网络配置。

第 1 阶段

在创建清单文件前，您可以自定义 install-config.yaml 文件中的以下与网络相关的字段：

- networking.networkType
- networking.clusterNetwork
- networking.serviceNetwork
- networking.machineNetwork

有关这些字段的更多信息，请参阅 *安装配置参数*。



注意

将 networking.machineNetwork 设置为与首选 NIC 所在的 CIDR 匹配。

重要

CIDR 范围 172.17.0.0/16 由 libVirt 保留。对于集群中的任何网络，您无法使用此范围或与这个范围重叠的范围。

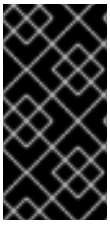
第 2 阶段

运行 openshift-install create 清单创建 清单文件后，您可以只使用您要修改的字段定义自定义 Cluster Network Operator 清单。您可以使用 清单指定高级网络配置。

您不能覆盖在 stage 2 阶段 1 中在 install-config.yaml 文件中指定的值。但是，您可以在第 2 阶段进一步自定义网络插件。

15.3.11. 指定高级网络配置

您可以使用网络插件的高级网络配置将集群集成到现有网络环境中。您只能在安装集群前指定高级网络配置。



重要

不支持通过修改安装程序创建的 OpenShift Container Platform 清单文件来自定义网络配置。支持应用您创建的清单文件，如以下流程中所示。

先决条件

- 您已创建 install-config.yaml 文件并完成对其所做的任何修改。

流程

1. 进入包含安装程序的目录并创建清单：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 <installation_directory> 指定包含集群的 install-config.yaml 文件的目录名称。

2.

在 <installation_directory>/manifests/ 目录中为高级网络配置创建一个名为 cluster-network-03-config.yml 的 stub 清单文件：

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
```

3. 在 cluster-network-03-config.yml 文件中指定集群的高级网络配置，如下例所示：

为 OVN-Kubernetes 网络供应商启用 IPsec

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  defaultNetwork:
    ovnKubernetesConfig:
      ipsecConfig:
        mode: Full
```

4. 可选：备份 manifests/cluster-network-03-config.yml 文件。创建 Ignition 配置文件时，安装程序会使用 manifests/ 目录。

15.3.12. Cluster Network Operator 配置

集群网络的配置作为 Cluster Network Operator(CNO)配置的一部分指定，并存储在名为 cluster 的自定义资源(CR)对象中。CR 指定 operator.openshift.io API 组中的 Network API 的字段。

CNO 配置在集群安装过程中从 Network.config.openshift.io API 组中的 Network API 继承以下字段：

clusterNetwork

从中分配 Pod IP 地址的 IP 地址池。

serviceNetwork

服务的 IP 地址池.

defaultNetwork.type

集群网络插件。OVNKubernetes 是安装期间唯一支持的插件。

您可以通过在名为 cluster 的 CNO 对象中设置 defaultNetwork 对象的字段来为集群指定集群网络插件配置。

15.3.12.1. Cluster Network Operator 配置对象

下表中描述了 Cluster Network Operator(CNO)的字段：

表 15.19. Cluster Network Operator 配置对象

字段	类型	描述
metadata.name	**字符串**	CNO 对象的名称。这个名称始终是 **集群**。

spec.clusterNetwork	**array**	用于指定从哪些 IP 地址块分配 Pod IP 地址以及集群中每个节点的子网前缀长度的列表。例如：
		`spec:`
		`  clusterNetwork:`
		`  - cidr: 10.128.0.0/19`
		`  hostPrefix: 23`
		`  - cidr: 10.128.32.0/19`
		`  hostPrefix: 23`

spec.serviceNetwork	**array**	服务的 IP 地址块。OpenShift SDN 和 OVN-Kubernetes 网络插件只支持服务网络的一个 IP 地址块。例如：
		`spec:`
		`  serviceNetwork:`
		`  - 172.30.0.0/14`
		您只能在创建清单前在 **install-config.yaml** 文件中自定义此字段。该值在清单文件中是只读的。

spec.defaultNetwork	**object**	为集群网络配置网络插件。
spec.kubeProxyConfig	**object**	此对象的字段指定 kube-proxy 配置。如果使用 OVN-Kubernetes 集群网络供应商，则 kube-proxy 配置不会起作用。

defaultNetwork 对象配置

下表列出了 defaultNetwork 对象的值：

表 15.20. defaultNetwork 对象

字段	类型	描述
type	**字符串**	**OVNKubernetes**。Red Hat OpenShift Networking 网络插件在安装过程中被选择。此值在集群安装后无法更改。 **注意** OpenShift Container Platform 默认使用 OVN-Kubernetes 网络插件。OpenShift SDN 不再作为新集群的安装选择提供。
ovnKubernetesConfig	**object**	此对象仅对 OVN-Kubernetes 网络插件有效。

配置 OVN-Kubernetes 网络插件

下表描述了 OVN-Kubernetes 网络插件的配置字段：

表 15.21. ovnKubernetesConfig object

字段	类型	描述
mtu	**integer**	Geneve（通用网络虚拟化封装）覆盖网络的最大传输单元 (MTU)。这根据主网络接口的 MTU 自动探测。您通常不需要覆盖检测到的 MTU。 如果自动探测的值不是您期望的值，请确认节点上主网络接口上的 MTU 是否正确。您不能使用这个选项更改节点上主网络接口的 MTU 值。 如果集群中不同节点需要不同的 MTU 值，则必须将此值设置为**比** 集群中的最低 MTU 值小 100。例如，如果集群中的某些节点的 MTU 为 **9001**，而某些节点的 MTU 为 **1500**，则必须将此值设置为 **1400**。
genevePort	**integer**	用于所有 Geneve 数据包的端口。默认值为 **6081**。此值在集群安装后无法更改。
ipsecConfig	**object**	指定用于自定义 IPsec 配置的配置对象。
ipv4	**object**	为 IPv4 设置指定配置对象。

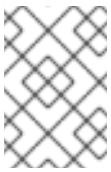
字段	类型	描述
ipv6	**object**	为 IPv6 设置指定配置对象。
policyAuditConfig	**object**	指定用于自定义网络策略审计日志的配置对象。如果未设置，则使用默认的审计日志设置。
gatewayConfig	**object**	可选：指定一个配置对象来自定义如何将出口流量发送到节点网关。  **注意** 在迁移出口流量时，工作负载和服务流量会受到一定影响，直到 Cluster Network Operator (CNO) 成功推出更改。

表 15.22. ovnKubernetesConfig.ipv4 object

字段	类型	描述
internalTransitSwitchSubnet	字符串	如果您的现有网络基础架构与 **100.88.0.0/16** IPv4 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。启用东西流量的分布式传输交换机的子网。此子网不能与 OVN-Kubernetes 或主机本身使用的任何其他子网重叠。必须足够大，以适应集群中的每个节点一个 IP 地址。 默认值为 **100.88.0.0/16**。
internalJoinSubnet	字符串	如果您的现有网络基础架构与 **100.64.0.0/16** IPv4 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。您必须确保 IP 地址范围没有与 OpenShift Container Platform 安装使用的任何其他子网重叠。IP 地址范围必须大于可添加到集群的最大节点数。例如，如果 **clusterNetwork.cidr** 值为 **10.128.0.0/14**，并且 **clusterNetwork.hostPrefix** 值为 **/23**，则最大节点数量为 **2^(23-14)=512**。 默认值为 **100.64.0.0/16**。

表 15.23. ovnKubernetesConfig.ipv6 object

字段	类型	描述
internalTransitSwitchSubnet	字符串	如果您的现有网络基础架构与 **fd97::/64** IPv6 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。启用东西流量的分布式传输交换机的子网。此子网不能与 OVN-Kubernetes 或主机本身使用的任何其他子网重叠。必须足够大，以适应集群中的每个节点一个 IP 地址。 默认值为 **fd97::/64**。

字段	类型	描述
internalJoinSubnet	字符串	如果您的现有网络基础架构与 **fd98::/64** IPv6 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。您必须确保 IP 地址范围没有与 OpenShift Container Platform 安装使用的任何其他子网重叠。IP 地址范围必须大于可添加到集群的最大节点数。 默认值为 **fd98::/64**。

表 15.24. policyAuditConfig object

字段	类型	描述
rateLimit	整数	每个节点每秒生成一次的消息数量上限。默认值为每秒 **20** 条消息。
maxFileSize	整数	审计日志的最大大小，以字节为单位。默认值为 **50000000** 或 50 MB。
maxLogFiles	整数	保留的日志文件的最大数量。
目的地	字符串	以下附加审计日志目标之一： **libc** 主机上的 journald 进程的 libc **syslog（）** 函数。 **UDP:<host>:<port>** 一个 syslog 服务器。将 **<host>:<port>** 替换为 **syslog 服务器的主机** 和端口。 **Unix:<file>** 由 **<file>** 指定的 Unix 域套接字文件。 **null** 不要将审计日志发送到任何其他目标。
syslogFacility	字符串	syslog 工具，如 as **kern**，如 RFC5424 定义。默认值为 **local0。**

表 15.25. gatewayConfig object

字段	类型	描述

字段	类型	描述
routingViaHost	**布尔值**	将此字段设置为 **true**，将来自 pod 的出口流量发送到主机网络堆栈。对于依赖于在内核路由表中手动配置路由的高级别安装和应用程序，您可能需要将出口流量路由到主机网络堆栈。默认情况下，出口流量在 OVN 中进行处理以退出集群，不受内核路由表中的特殊路由的影响。默认值为 **false**。 此字段与 Open vSwitch 硬件卸载功能有交互。如果将此字段设置为 **true**，则不会获得卸载的性能优势，因为主机网络堆栈会处理出口流量。
ipForwarding	**object**	您可以使用 **Network** 资源中的 **ipForwarding** 规格来控制 OVN-Kubernetes 管理接口上所有流量的 IP 转发。指定 **Restricted** 只允许 Kubernetes 相关流量的 IP 转发。指定 **Global** 以允许转发所有 IP 流量。对于新安装，默认值为 **Restricted**。对于到 OpenShift Container Platform 4.14 或更高版本的更新，默认值为 **Global**。
ipv4	**object**	可选：指定一个对象来为主机配置内部 OVN-Kubernetes 伪装地址，以服务 IPv4 地址的流量。
ipv6	**object**	可选：指定一个对象来为主机配置内部 OVN-Kubernetes 伪装地址，以服务 IPv6 地址的流量。

表 15.26. gatewayConfig.ipv4 对象

字段	类型	描述
internalMasqueradeSubnet	**字符串**	内部使用的伪装 IPv4 地址，以启用主机服务流量。主机配置了这些 IP 地址和共享网关网桥接口。默认值为 **169.254.169.0/29**。

表 15.27. gatewayConfig.ipv6 对象

字段	类型	描述
internalMasqueradeSubnet	**字符串**	内部使用的伪装 IPv6 地址，以启用主机服务流量。主机配置了这些 IP 地址和共享网关网桥接口。默认值为 **fd69::/125**。

表 15.28. ipsecConfig 对象

字段	类型	描述

字段	类型	描述
模式	**字符串**	指定 IPsec 实现的行为。必须是以下值之一： • **Disabled**: 在集群节点上不启用 IPsec。 • **External** ：对于带有外部主机的网络流量，启用 IPsec。 • **Full**: IPsec 为带有外部主机的 pod 流量和网络流量启用 IPsec。

启用 IPSec 的 OVN-Kubernetes 配置示例

```
defaultNetwork:
  type: OVNKubernetes
  ovnKubernetesConfig:
    mtu: 1400
    genevePort: 6081
      ipsecConfig:
        mode: Full
```

重要

使用 OVNKubernetes 可能会导致 IBM Power® 上的堆栈耗尽问题。

kubeProxyConfig 对象配置（仅限 OpenShiftSDN 容器网络接口）

kubeProxyConfig 对象的值在下表中定义：

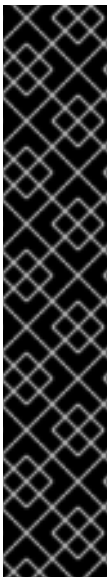
表 15.29. kubeProxyConfig object

字段	类型	描述

字段	类型	描述
iptablesSyncPeriod	**字符串**	**iptables** 规则的刷新周期。默认值为 **30s**。有效的后缀包括 **s**、**m** 和 **h**，具体参见 Go 时间 包 文档。  **注意** 由于 OpenShift Container Platform 4.3 及更高版本中引进了性能改进，不再需要调整 **iptablesSyncPeriod** 参数。
proxyArguments.iptables-min-sync-period	**array**	刷新 **iptables** 规则前的最短持续时间。此字段确保刷新的频率不会过于频繁。有效的后缀包括 **s**、**m** 和 **h**，具体参见 Go **time** 软件包。默认值为： kubeProxyConfig: proxyArguments: iptables-min-sync-period: - 0s

15.3.13. 创建 Ignition 配置文件

由于您必须手动启动集群机器，因此您必须生成 Ignition 配置文件，集群需要这些配置文件来创建其机器。



重要

- **安装程序生成的 Ignition 配置文件包含 24 小时后过期的证书，然后在该时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 node-bootstrapper 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。**

- **建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。**

先决条件

- 获取 OpenShift Container Platform 安装程序和集群的 pull secret。

流程

- 获取 Ignition 配置文件：

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

1 对于 <installation_directory>，请指定要存储安装程序创建的文件的目录名称。

重要

如果创建了 install-config.yaml 文件，请指定包含该文件的目录。否则，指定一个空目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

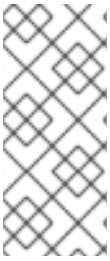
该目录中会生成以下文件：

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

15.3.14. 安装 RHCOS 并启动 OpenShift Container Platform bootstrap 过程

要在您置备的裸机基础架构上安装 OpenShift Container Platform，您必须在机器上安装 Red Hat Enterprise Linux CoreOS(RHCOS)。安装 RHCOS 时，您必须为您要安装的机器类型提供 OpenShift Container Platform 安装程序生成的 Ignition 配置文件。如果您配置了适当的网络、DNS 和负载均衡基础架构，OpenShift Container Platform bootstrap 过程会在 RHCOS 机器重启后自动启动。

要在机器上安装 RHCOS，请按照以下步骤使用 ISO 镜像或网络 PXE 引导。



注意

本安装文档中包括的计算节点部署步骤特定于 RHCOS。如果您选择部署基于 RHEL 的计算节点，您需要负责所有操作系统生命周期管理和维护，包括执行系统更新、应用补丁和完成所有其他必要的任务。仅支持 RHEL 8 计算机器。

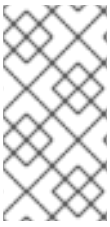
您可以使用以下方法在 ISO 和 PXE 安装过程中配置 RHCOS：

- 内核参数： 您可以使用内核参数来提供特定于安装的信息。例如，您可以指定上传到 HTTP 服务器的 RHCOS 安装文件的位置以及您要安装的节点类型的 Ignition 配置文件的位置。对于 PXE 安装，您可以使用 APPEND 参数将参数传递给 live 安装程序的内核。对于 ISO 安装，您可以中断实时安装引导过程来添加内核参数。在这两个安装情形中，您可以使用特殊的 coreos.inst.* 参数来指示实时安装程序，以及标准安装引导参数来打开或关闭标准内核服务。

- Ignition 配置：OpenShift Container Platform Ignition 配置文件(*.ign)特定于您要安装的节点类型。您可以在 RHCOS 安装过程中传递 bootstrap、control plane 或计算节点 Ignition 配置文件的位置，以便在首次启动时生效。特殊情况下，您可以创建单独的、有限的 Ignition 配置以传递给 live 系统。该 Ignition 配置可以执行特定的任务，如在安装完成后向置备系统报告成功。此特殊的 Ignition 配置由 coreos-installer 使用，以便在首次引导安装的系统时应用。不要直接为实时 ISO 提供标准 control plane 和计算节点 Ignition 配置。

- coreos-installer ：您可以将 live ISO 安装程序引导到 shell 提示符，这可让您在第一次引导前以多种方式准备持久性系统。特别是，您可以运行 coreos-installer 命令来识别要包含的各种工件、使用磁盘分区和设置网络。在某些情况下，您可以配置 live 系统上的功能并将其复制到安装的系统中。

使用 ISO 安装还是 PXE 安装取决于您的情况。PXE 安装需要可用的 DHCP 服务并进行更多准备，但可以使安装过程更加自动化。ISO 安装是一个更手动的过程，如果您设置的机器数超过几台，则可能不方便。



注意

从 OpenShift Container Platform 4.6 开始，RHCOS ISO 和其他安装工件支持在带有 4K 扇区的磁盘上安装。

15.3.14.1. 使用 ISO 镜像安装 RHCOS

您可以使用 ISO 镜像在机器上安装 RHCOS。

先决条件

- 已为集群创建 Ignition 配置文件。
- 您已配置了适当的网络、DNS 和负载平衡基础架构。
- 您有一个可以从计算机以及您创建的机器访问的 HTTP 服务器。
- 您已参阅 *Advanced RHCOS 安装配置* 部分来了解配置功能的不同方法，如网络和磁盘分区。

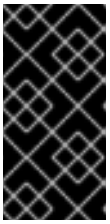
流程

1. 获取每个 Ignition 配置文件的 SHA512 摘要。例如，您可以在运行 Linux 的系统上使用以下内容来获取 bootstrap.ign Ignition 配置文件的 SHA512 摘要：

```
$ sha512sum <installation_directory>/bootstrap.ign
```

后续步骤中会向 coreos-installer 提供摘要，以验证集群节点上 Ignition 配置文件的真实性。

2. 将安装程序创建的 bootstrap、control plane 和计算节点 Ignition 配置文件上传到 HTTP 服务器。注意这些文件的 URL。



重要

您可以在 Ignition 配置中添加或更改配置设置，然后将其保存到 HTTP 服务器。如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

3. 从安装主机上，验证 Ignition 配置文件是否在 URL 上可用。以下示例获取 bootstrap 节点的 Ignition 配置文件：

```
$ curl -k http://<HTTP_server>/bootstrap.ign 1
```

输出示例

```
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0{"ignition":
{"version":"3.2.0"},"passwd":{"users":[{"name":"core","sshAuthorizedKeys":["ssh-
rsa...
```

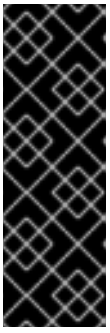
在命令中将 bootstrap.ign 替换为 master.ign 或 worker.ign，以验证 control plane 和计算节点的 Ignition 配置文件是否可用。

4. 虽然可以从 RHCOS 镜像镜像页面获取您选择的操作系统实例安装方法所需的 RHCOS 镜像，但推荐的方法是从 openshift-install 命令的输出获取 RHCOS 镜像的正确版本：

```
$ openshift-install coreos print-stream-json | grep '\.iso[^.]'
```

输出示例

```
"location": "<url>/art/storage/releases/rhcos-4.16-aarch64/<release>/aarch64/rhcos-
<release>-live.aarch64.iso",
"location": "<url>/art/storage/releases/rhcos-4.16-ppc64le/<release>/ppc64le/rhcos-
<release>-live.ppc64le.iso",
"location": "<url>/art/storage/releases/rhcos-4.16-s390x/<release>/s390x/rhcos-
<release>-live.s390x.iso",
"location": "<url>/art/storage/releases/rhcos-4.16/<release>/x86_64/rhcos-<release>-
live.x86_64.iso",
```



重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每个发行版本而改变。您必须下载最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。如果可用，请使用与 OpenShift Container Platform 版本匹配的镜像版本。这个过程只使用 ISO 镜像。此安装类型不支持 RHCOS qcow2 镜像。

ISO 文件名类似以下示例：

```
rhcos-<version>-live.<architecture>.iso
```

5. 使用 ISO 启动 RHCOS 安装。使用以下安装选项之一：

- 将 ISO 映像刻录到磁盘并直接启动。
- 使用 light-out 管理(LOM)接口使用 ISO 重定向。

6. 在不指定任何选项或中断实时引导序列的情况下引导 RHCOS ISO 镜像。等待安装程序在 RHCOS live 环境中引导进入 shell 提示符。

注意

可以中断 RHCOS 安装引导过程来添加内核参数。但是，在这个 ISO 过程中，您应该使用以下步骤中所述的 coreos-installer 命令，而不是添加内核参数。

7. 运行 coreos-installer 命令并指定满足您的安装要求的选项。您至少必须指定指向节点类型的 Ignition 配置文件的 URL，以及您要安装到的设备：

```
$ sudo coreos-installer install --ignition-url=http://<HTTP_server>/<node_type>.ign
<device> --ignition-hash=sha512-<digest> 1 2
```

1 1 您必须使用 sudo 运行 coreos-installer 命令，因为 core 用户没有执行安装所需的 root 权限。

2 当 Ignition 配置文件通过 HTTP URL 获取时，需要 --ignition-hash 选项来验证集群节点上 Ignition 配置文件的真实性。<digest> 是上一步中获取的 Ignition 配置文件 SHA512 摘要。

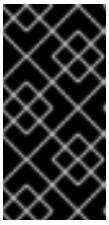
注意

如果要通过使用 TLS 的 HTTPS 服务器提供 Ignition 配置文件，您可以在运行 coreos-installer 前将内部证书颁发机构(CA)添加到系统信任存储中。

以下示例将引导节点安装初始化到 /dev/sda 设备。bootstrap 节点的 Ignition 配置文件从 IP 地址 192.168.1.2 的 HTTP Web 服务器获取：

```
$ sudo coreos-installer install --ignition-
url=http://192.168.1.2:80/installation_directory/bootstrap.ign /dev/sda --ignition-
hash=sha512-
a5a2d43879223273c9b60af66b44202a1d1248fc01cf156c46d4a79f552b6bad47bc8cc78dd
f0116e80c59d2ea9e32ba53bc807afbca581aa059311def2c3e3b
```

8. 在机器的控制台上监控 RHCOS 安装的进度。



重要

在开始安装 OpenShift Container Platform 之前，确保每个节点中安装成功。观察安装过程可以帮助确定可能会出现 RHCOS 安装问题的原因。

9. 安装 RHCOS 后，您必须重启系统。系统重启过程中，它会应用您指定的 Ignition 配置文件。

10. 检查控制台输出，以验证 Ignition 是否运行。

示例命令

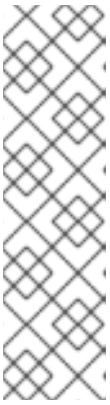
```
Ignition: ran on 2022/03/14 14:48:33 UTC (this boot)
Ignition: user-provided config was applied
```

11. 继续为集群创建其他机器。

重要

此时您必须创建 bootstrap 和 control plane 机器。如果 control plane 机器不可调度，请在安装 OpenShift Container Platform 前至少创建两台计算机器。

如果存在所需的网络、DNS 和负载均衡器基础架构，OpenShift Container Platform bootstrap 过程会在 RHCOS 节点重启后自动启动。



注意

RHCOS 节点不包含 core 用户的默认密码。您可以使用可访问 SSH 私钥的用户的身份运行 ssh core@<node>.<cluster_name>.<base_domain > 来访问节点，该私钥与您在 install_config.yaml 文件中指定的公钥配对。运行 RHCOS 的 OpenShift Container Platform 4 集群节点不可变，它依赖于 Operator 来应用集群更改。不建议使用 SSH 访问集群节点。但是，当调查安装问题时，如果 OpenShift Container Platform API 不可用，或者 kubelet 在目标节点上无法正常工作，则调试或灾难恢复可能需要 SSH 访问。

15.3.14.2. 使用 PXE 或 iPXE 启动安装 RHCOS

您可以使用 PXE 或 iPXE 启动在机器上安装 RHCOS。

先决条件

- 已为集群创建 Ignition 配置文件。
- 您已配置了适当的网络、DNS 和负载平衡基础架构。
- 您已配置了合适的 PXE 或 iPXE 基础架构。
- 您有一个可以从计算机以及您创建的机器访问的 HTTP 服务器。
- 您已参阅 *Advanced RHCOS 安装配置* 部分来了解配置功能的不同方法，如网络和磁盘分区。

流程

1. 将安装程序创建的 bootstrap、control plane 和计算节点 Ignition 配置文件上传到 HTTP 服务器。注意这些文件的 URL。

重要

您可以在 Ignition 配置中添加或更改配置设置，然后将其保存到 HTTP 服务器。如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

2. 从安装主机上，验证 Ignition 配置文件是否在 URL 上可用。以下示例获取 bootstrap 节点的 Ignition 配置文件：

```
$ curl -k http://<HTTP_server>/bootstrap.ign 1
```

输出示例

```
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0{"ignition":
{"version":"3.2.0"},"passwd":{"users":[{"name":"core","sshAuthorizedKeys":["ssh-
rsa...
```

在命令中将 bootstrap.ign 替换为 master.ign 或 worker.ign，以验证 control plane 和计算节点的 Ignition 配置文件是否可用。

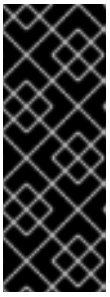
3. 虽然可以从 RHCOS image mirror 页面获取您选择的操作系统实例所需的 RHCOS kernel、initramfs 和 rootfs 文件，但推荐的方法是从 openshift-install 命令的输出中获取 RHCOS 文件的正确版本：

```
$ openshift-install coreos print-stream-json | grep -Eo '"https.*(kernel-
|initramfs.|rootfs.)\w+(\.img)?"'
```

输出示例

```
"<url>/art/storage/releases/rhcos-4.16-aarch64/<release>/aarch64/rhcos-<release>-live-
```

```
kernel-aarch64"
"<url>/art/storage/releases/rhcos-4.16-aarch64/<release>/aarch64/rhcos-<release>-live-initramfs.aarch64.img"
"<url>/art/storage/releases/rhcos-4.16-aarch64/<release>/aarch64/rhcos-<release>-live-rootfs.aarch64.img"
"<url>/art/storage/releases/rhcos-4.16-ppc64le/49.84.202110081256-0/ppc64le/rhcos-<release>-live-kernel-ppc64le"
"<url>/art/storage/releases/rhcos-4.16-ppc64le/<release>/ppc64le/rhcos-<release>-live-initramfs.ppc64le.img"
"<url>/art/storage/releases/rhcos-4.16-ppc64le/<release>/ppc64le/rhcos-<release>-live-rootfs.ppc64le.img"
"<url>/art/storage/releases/rhcos-4.16-s390x/<release>/s390x/rhcos-<release>-live-kernel-s390x"
"<url>/art/storage/releases/rhcos-4.16-s390x/<release>/s390x/rhcos-<release>-live-initramfs.s390x.img"
"<url>/art/storage/releases/rhcos-4.16-s390x/<release>/s390x/rhcos-<release>-live-rootfs.s390x.img"
"<url>/art/storage/releases/rhcos-4.16/<release>/x86_64/rhcos-<release>-live-kernel-x86_64"
"<url>/art/storage/releases/rhcos-4.16/<release>/x86_64/rhcos-<release>-live-initramfs.x86_64.img"
"<url>/art/storage/releases/rhcos-4.16/<release>/x86_64/rhcos-<release>-live-rootfs.x86_64.img"
```



重要

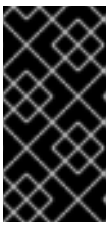
RHCOS 工件可能不会随着 OpenShift Container Platform 的每个发行版本而改变。您必须下载最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。这个过程只使用下面描述的适当 kernel、initram fs 和 rootfs 工件。此安装类型不支持 RHCOS QCOW2 镜像。

文件名包含 OpenShift Container Platform 版本号。它们类似以下示例：

- kernel:rhcos-<version>-live-kernel-<architecture>
- initramfs: rhcos-<version>-live-initramfs.<architecture>.img
- rootfs: rhcos-<version>-live-rootfs.<architecture>.img

4

4. 将 rootfs、kernel 和 initramfs 文件上传到 HTTP 服务器。



重要

如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

5. 配置网络引导基础架构，以便在安装 RHCOS 后机器从本地磁盘启动。

6. 为 RHCOS 镜像配置 PXE 或 iPXE 安装并开始安装。

为环境修改以下示例菜单条目之一，并验证能否正确访问镜像和 Ignition 文件：

- 对于 PXE(x86_64)：

```
DEFAULT pxeboot
TIMEOUT 20
PROMPT 0
LABEL pxeboot
    KERNEL http://<HTTP_server>/rhcos-<version>-live-kernel-<architecture> (1)
    APPEND initrd=http://<HTTP_server>/rhcos-<version>-live-initramfs.
<architecture>.img coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-
rootfs.<architecture>.img coreos.inst.install_dev=/dev/sda
coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign (2) (3)
```

(1) (1)

指定上传到 HTTP 服务器的 live kernel 文件位置。URL 必须是 HTTP、TFTP 或 FTP；不支持 HTTPS 和 NFS。

(2)

如果您使用多个 NIC，请在 ip 选项中指定一个接口。例如，要在名为 eno1 的 NIC 上使用 DHCP，请设置 ip=eno1:dhcp。

(3)

指定上传到 HTTP 服务器的 RHCOS 文件的位置。initrd 参数值是 initramfs 文件的位置，coreos.live.rootfs_url 参数值是 rootfs 文件的位置，coreos.inst.ignition_url 参数值则是 bootstrap Ignition 配置文件的位置。您还可以在 APPEND 行中添加更多内核参数来配置联网或其他引导选项。

注意

此配置不会在图形控制台的机器上启用串行控制台访问。要配置不同的控制台，请在 APPEND 行中添加一个或多个 console= 参数。例如，添加 console=tty0 console=ttyS0 以将第一个 PC 串口设置为主控制台，并将图形控制台设置为二级控制台。如需更多信息，请参阅如何在 Red Hat Enterprise Linux 中设置串行终端和/或控制台？ 和"启用 PXE 和 ISO 安装的串行控制台"部分。

- **对于 iPXE (x86_64 + aarch64) :**

```
kernel http://<HTTP_server>/rhcos-<version>-live-kernel-<architecture> initrd=main
coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-rootfs.
<architecture>.img coreos.inst.install_dev=/dev/sda
coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign 1 2
initrd --name main http://<HTTP_server>/rhcos-<version>-live-initramfs.
<architecture>.img 3
boot
```

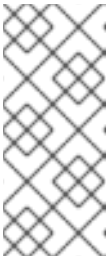
1. **指定上传到 HTTP 服务器的 RHCOS 文件的位置。kernel 参数值是 kernel 文件的位置，init rd=main 参数用于在 UEFI 系统中引导，coreos.live.rootfs_url 参数值是 rootfs 文件的位置，coreos.inst.ignition_url 参数值则是 bootstrap Ignition 配置文件的位置。**

2. **如果您使用多个 NIC，请在 ip 选项中指定一个接口。例如，要在名为 eno1 的 NIC 上使用 DHCP，请设置 ip=eno1:dhcp。**

3. **指定上传到 HTTP 服务器的 initramfs 文件的位置。**

注意

此配置不会在图形控制台的机器上启用串行控制台访问。要配置不同的控制台，请在 内核参数 中添加一个或多个 console= 参数。例如，添加 console=tty0 console=ttyS0 以将第一个 PC 串口设置为主控制台，并将图形控制台设置为二级控制台。如需更多信息，请参阅如何在 Red Hat Enterprise Linux 中设置串行终端和/或控制台？ 和"启用 PXE 和 ISO 安装的串行控制台"部分。



注意

要在 aarch64 架构中网络引导 CoreOS 内核，您需要使用启用了 IMAGE_GZIP 选项的 iPXE 构建版本。请参阅 iPXE 中的IMAGE_GZIP 选项。

- 对于 aarch64 中的 PXE（使用 UEFI 和 Grub 作为第二阶段）：

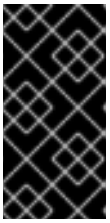
```
menuentry 'Install CoreOS' {
    linux rhcos-<version>-live-kernel-<architecture>
coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-rootfs.
<architecture>.img coreos.inst.install_dev=/dev/sda
coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign 1 2
    initrd rhcos-<version>-live-initramfs.<architecture>.img 3
}
```

1. 指定上传到 HTTP/TFTP 服务器的 RHCOS 文件的位置。kernel 参数值是 TFTP 服务器中的 kernel 文件的位置。coreos.live.rootfs_url 参数值是 rootfs 文件的位置，coreos.inst.ignition_url 参数值是 HTTP 服务器上的 bootstrap Ignition 配置文件的位置。

2. 如果您使用多个 NIC，请在 ip 选项中指定一个接口。例如，要在名为 eno1 的 NIC 上使用 DHCP，请设置 ip=eno1:dhcp。

3. 指定上传到 TFTP 服务器的 initramfs 文件的位置。

7. 在机器的控制台上监控 RHCOS 安装的进度。



重要

在开始安装 OpenShift Container Platform 之前，确保每个节点中安装成功。观察安装过程可以帮助确定可能会出现 RHCOS 安装问题的原因。

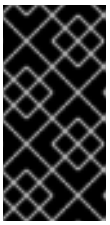
8. 安装 RHCOS 后，系统会重启。在重启过程中，系统会应用您指定的 Ignition 配置文件。

9. 检查控制台输出，以验证 Ignition 是否运行。

示例命令

```
Ignition: ran on 2022/03/14 14:48:33 UTC (this boot)
Ignition: user-provided config was applied
```

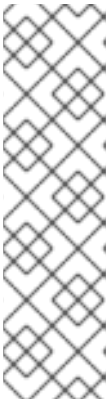
10. 继续为集群创建机器。



重要

此时您必须创建 bootstrap 和 control plane 机器。如果 control plane 机器不可调度，请在安装集群前至少创建两台计算机器。

如果存在所需的网络、DNS 和负载均衡器基础架构，OpenShift Container Platform bootstrap 过程会在 RHCOS 节点重启后自动启动。



注意

RHCOS 节点不包含 core 用户的默认密码。您可以使用可访问 SSH 私钥的用户的身份运行 ssh core@<node>.<cluster_name>.<base_domain > 来访问节点，该私钥与您在 install_config.yaml 文件中指定的公钥配对。运行 RHCOS 的 OpenShift Container Platform 4 集群节点不可变，它依赖于 Operator 来应用集群更改。不建议使用 SSH 访问集群节点。但是，当调查安装问题时，如果 OpenShift Container Platform API 不可用，或者 kubelet 在目标节点上无法正常工作，则调试或灾难恢复可能需要 SSH 访问。

15.3.14.3. 高级 RHCOS 安装配置

为 OpenShift Container Platform 手动置备 Red Hat Enterprise Linux CoreOS(RHCOS)节点的一个关键优点是能够进行通过默认的 OpenShift Container Platform 安装方法无法进行的配置。本节介绍了您可以使用的一些技术进行配置，其中包括：

- 将内核参数传递给实时安装程序

- 从 live 系统手动运行 coreos-installer
- 自定义实时 ISO 或 PXE 引导镜像

本节详述了与 Red Hat Enterprise Linux CoreOS(RHCOS)手动安装的高级配置相关的内容，如磁盘分区、网络以及使用 Ignition 配置的不同方式相关。

15.3.14.3.1. 使用高级网络选项进行 PXE 和 ISO 安装

OpenShift Container Platform 节点的网络默认使用 DHCP 来收集所有必要的配置设置。要设置静态 IP 地址或配置特殊设置，如绑定，您可以执行以下操作之一：

- 引导 live 安装程序时传递特殊内核参数。
- 使用机器配置将网络文件复制到安装的系统中。
- 从 live 安装程序 shell 提示符配置网络，然后将这些设置复制到安装的系统上，以便在安装的系统第一次引导时生效。

要配置 PXE 或 iPXE 安装，请使用以下选项之一：

- 请参阅"高级 RHCOS 安装参考"表。
- 使用机器配置将网络文件复制到安装的系统中。

要配置 ISO 安装，请使用以下步骤：

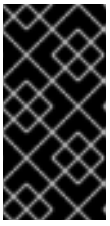
流程

1. 引导 ISO 安装程序.
2.

在 live 系统 shell 提示符下，使用可用的 RHEL 工具（如 nmcli 或 nmtui ）为 live 系统配置网络。

3. 运行 coreos-installer 命令以安装系统，添加 --copy-network 选项来复制网络配置。例如：

```
$ sudo coreos-installer install --copy-network \
    --ignition-url=http://host/worker.ign /dev/disk/by-id/scsi-<serial_number>
```



重要

copy-network 选项仅复制在 /etc/NetworkManager/system-connections 下找到的网络配置。特别是，它不会复制系统主机名。

4. 重启安装的系统。

其他资源

- 有关 nmcli 和 nmtui 工具的更多信息，请参阅 RHEL 8 文档中的 Getting started with nmcli and Getting started with nmtui。

15.3.14.3.2. 磁盘分区

磁盘分区是在 Red Hat Enterprise Linux CoreOS(RHCOS)安装过程中在 OpenShift Container Platform 集群节点上创建的。特定架构的每个 RHCOS 节点使用相同的分区布局，除非覆盖默认的分区配置。在 RHCOS 安装过程中，根文件系统的大小会增加，以使用目标设备中剩余的可用空间。

重要

在节点上使用自定义分区方案可能会导致 OpenShift Container Platform 在某些节点分区上监控或警报。如果要覆盖默认分区，请参阅 了解 OpenShift 文件系统监控（驱除条件） 以了解有关 OpenShift Container Platform 如何监控主机文件系统的更多信息。

OpenShift Container Platform 监控以下两个文件系统标识符：

- nodefs，这是包含 /var/lib/kubelet 的文件系统

- imagefs，这是包含 /var/lib/containers 的文件系统

对于默认分区方案，nodefs 和 imagefs 监控相同的根文件系统 /。

要在 OpenShift Container Platform 集群节点上安装 RHCOS 时覆盖默认分区，您必须创建单独的分区。您可能想要为容器和容器镜像添加单独的存储分区。例如，通过在独立分区中挂载 /var/lib/containers，kubelet 会单独监控 /var/lib/containers 作为 imagefs 目录，以及 root 文件系统作为 nodefs 目录。

重要

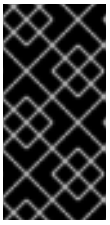
如果您已将磁盘大小调整为托管更大的文件系统，请考虑创建单独的 /var/lib/containers 分区。考虑重新定义具有 xfs 格式的磁盘大小，以减少大量分配组导致的 CPU 时间问题。

15.3.14.3.2.1. 创建独立 /var 分区

通常，您应该使用在 RHCOS 安装过程中创建的默认磁盘分区。然而，在有些情况下您可能需要为预期增长的目录创建独立分区。

OpenShift Container Platform 支持添加单个分区将存储附加到 /var 目录或 /var 的子目录中。例如：

- /var/lib/containers ：保存随着系统中添加更多镜像和容器而增长的容器相关内容。
- /var/lib/etcd ：保存您可能希望独立保留的数据，比如 etcd 存储的性能优化。
- /var ：保存您可能希望独立保留的数据，以满足审计等目的。



重要

对于大于 100GB 的磁盘大小，特别是磁盘大小大于 1TB，请创建一个独立的 /var 分区。

通过单独存储 /var 目录的内容，可以更轻松地根据需要为区域扩展存储，并在以后重新安装 OpenShift Container Platform，并保持该数据的完整性。使用这个方法，您不必再次拉取所有容器，在更新系统时也不必复制大量日志文件。

将独立分区用于 /var 目录或 /var 的子目录也会防止分区目录中的数据增加填充根文件系统。

以下流程通过添加机器配置清单来设置独立的 /var 分区，该清单会在安装准备阶段封装到节点类型的 Ignition 配置文件中。

流程

1. 在安装主机上，切换到包含 OpenShift Container Platform 安装程序的目录，并为集群生成 Kubernetes 清单：

```
$ openshift-install create manifests --dir <installation_directory>
```

2. 创建用于配置额外分区的 Butane 配置。例如，将文件命名为 $HOME/clusterconfig/98-var-partition.bu，将磁盘设备名称改为 worker 系统上存储设备的名称，并根据情况设置存储大小。这个示例将 /var 目录放在一个单独的分区中：

```
variant: openshift
version: 4.16.0
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
storage:
  disks:
  - device: /dev/disk/by-id/<device_name> 1
    partitions:
    - label: var
      start_mib: <partition_start_offset> 2
      size_mib: <partition_size> 3
      number: 5
  filesystems:
    - device: /dev/disk/by-partlabel/var
      path: /var
      format: xfs
      mount_options: [defaults, prjquota] 4
      with_mount_unit: true
```

1. 要分区的磁盘的存储设备名称。

2

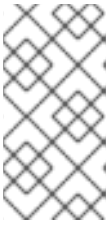
当在引导磁盘中添加数据分区时，推荐最少使用偏移值 25000 兆字节。root 文件系统会自动调整大小以填充所有可用空间（最多到指定的偏移值）。如果没有指定偏移值，或者指定的值小于推荐的最小值，则生成的 root 文件系统会太小，而在以后进行的 RHCOS 重新安装可能会覆盖数据分区的开始部分。

3

以兆字节为单位的数据分区大小。

4

对于用于容器存储的文件系统，必须启用 prjquota 挂载选项。



注意

在创建单独的 /var 分区时，如果不同的实例类型没有相同的设备名称，则无法将不同的实例类型用于计算节点。

3. 从 Butane 配置创建一个清单，并将它保存到 clusterconfig/openshift 目录中。例如，运行以下命令：

```
$ butane $HOME/clusterconfig/98-var-partition.bu -o
$HOME/clusterconfig/openshift/98-var-partition.yaml
```

4. 创建 Ignition 配置文件：

```
$ openshift-install create ignition-configs --dir <installation_directory> 1
```

1

对于 <installation_directory>，请指定相同的安装目录。

为安装目录中的 bootstrap、control plane 和计算节点创建 Ignition 配置文件：

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
```

```
├── master.ign
├── metadata.json
└── worker.ign
```

<installation_directory>/manifest 和 <installation_directory>/openshift 目录中的文件被嵌套到 Ignition 配置文件中，包括包含 98-var-partition 自定义 MachineConfig 对象的文件。

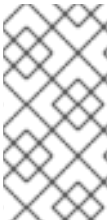
后续步骤

- 您可以通过在 RHCOS 安装过程中引用 Ignition 配置文件来应用自定义磁盘分区。

15.3.14.3.2.2. 保留现有分区

对于 ISO 安装，您可以在 coreos-installer 命令中添加可让安装程序维护一个或多个现有分区的选项。对于 PXE 安装，您可以在 APPEND 参数中添加 coreos.inst.* 选项来保留分区。

保存的分区可能是来自现有 OpenShift Container Platform 系统的数据分区。您可以通过分区标签或编号识别您要保留的磁盘分区。



注意

如果您保存了现有分区，且这些分区没有为 RHCOS 留下足够空间，则安装将失败，而不影响保存的分区。

在 ISO 安装过程中保留现有分区

这个示例保留分区标签以 数据开头的任何分区(data *)：

```
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign \
      --save-partlabel 'data*' /dev/disk/by-id/scsi-<serial_number>
```

以下示例演示了在运行 coreos-installer 时要保留磁盘上的第 6 个分区：

```
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign \
      --save-partindex 6 /dev/disk/by-id/scsi-<serial_number>
```

这个示例保留分区 5 及更高分区：

```
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign
      --save-partindex 5- /dev/disk/by-id/scsi-<serial_number>
```

在前面已保存分区的示例中，coreos-installer 会立即重新创建分区。

在 PXE 安装过程中保留现有分区

这个 APPEND 选项保留分区标签以 'data'('data*')开头的任何分区：

```
coreos.inst.save_partlabel=data*
```

这个 APPEND 选项保留分区 5 及更高分区：

```
coreos.inst.save_partindex=5-
```

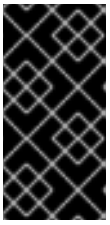
这个 APPEND 选项保留分区 6:

```
coreos.inst.save_partindex=6
```

15.3.14.3.3. 识别 Ignition 配置

在进行 RHCOS 手动安装时，您可以提供两种 Ignition 配置类型，它们有不同的原因：

- 永久安装 Ignition 配置 ：每个手动 RHCOS 安装都需要传递 openshift-installer 生成的 Ignition 配置文件之一，如 bootstrap.ign、master.ign 和 worker.ign，才能进行安装。



重要

不建议直接修改这些 Ignition 配置文件。您可以更新嵌套到 Ignition 配置文件中的清单文件，如上一节示例中所述。

对于 PXE 安装，您可以使用 coreos.inst.ignition_url= 选项在 APPEND 行上传递 Ignition 配置。对于 ISO 安装，在 ISO 引导至 shell 提示符后，您可以使用带有 --ignition-url= 选项的 coreos-installer 命令行。在这两种情况下，只支持 HTTP 和 HTTPS 协议。

- 实时安装 Ignition 配置 ：可使用 coreos-installer customize 子命令及其各种选项来创建此类型。使用此方法，Ignition 配置会传递到 live 安装介质，在引导时立即运行，并在 RHCOS

系统安装到磁盘之前或之后执行设置任务。这个方法只用于必须执行一次且之后不能再次应用的任务，比如不能使用机器配置进行的高级分区。

对于 PXE 或 ISO 引导，您可以创建 Ignition 配置，APP END ignition.config.url= 选项来标识 Ignition 配置的位置。您还需要附加 ignition.firstboot ignition.platform.id=metal 或 ignition.config.url 选项。

15.3.14.3.4. 默认控制台配置

从 OpenShift Container Platform 4.16 引导镜像安装的 Red Hat Enterprise Linux CoreOS (RHCOS) 节点使用默认控制台，旨在识别大多数虚拟化和裸机设置。不同的云和虚拟化平台可能会根据所选的架构使用不同的默认设置。裸机安装使用内核默认设置，这通常意味着图形控制台是主控制台，并且禁用串行控制台。

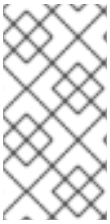
默认控制台可能与特定的硬件配置不匹配，或者您可能具有需要调整默认控制台的特定需求。例如：

- 您希望访问控制台上的紧急 shell 进行调试。
- 您的云平台没有提供到图形控制台的互动访问，但提供了一个串行控制台。
- 您需要启用多个控制台。

控制台配置继承自引导镜像。这意味着现有集群中的新节点不受默认控制台的影响。

您可以使用以下方法为裸机安装配置控制台：

- 在命令行中手动使用 coreos-installer。
- 使用带有 --dest-console 选项的 coreos-installer iso customize 或 coreos-installer pxe customize 子命令，以创建可自动执行进程的自定义镜像。



注意

对于高级自定义，请使用 coreos-installer iso 或 coreos-installer pxe 子命令而不是内核参数来执行控制台配置。

15.3.14.3.5. 为 PXE 和 ISO 安装启用串行控制台

默认情况下，Red Hat Enterprise Linux CoreOS (RHCOS) 串行控制台被禁用，所有输出都会写入图形控制台。您可以为 ISO 安装启用串行控制台并重新配置引导装载程序，以便输出同时发送到串行控制台和图形控制台。

流程

1. 引导 ISO 安装程序.

2. 运行 coreos-installer 命令来安装系统，添加 --console 选项一次来指定图形控制台，然后第二次指定串行控制台：

```
$ coreos-installer install \
  --console=tty0 \ 1
  --console=ttyS0,<options> \ 2
  --ignition-url=http://host/worker.ign /dev/disk/by-id/scsi-<serial_number>
```

1. 所需的二级控制台。在这种情况下，是图形控制台。省略这个选项将禁用图形控制台。

2. 所需的主控制台。在这种情况下，是串行控制台。options 字段定义 baud 速率和其他设置。此字段的一个常见值为 11520n8。如果没有提供选项，则使用默认内核值 9600n8。有关这个选项格式的更多信息，请参阅 Linux 内核串口控制台 文档。

3. 重启安装的系统。

注意

可以使用 coreos-installer install --append-karg 选项来获取类似的结果，并使用 console= 指定控制台。但是，这只会为内核设置控制台，而不为引导装载程序设置控制台。

要配置 PXE 安装，请确保省略 coreos.inst.install_dev 内核命令行选项，并使用 shell 提示符使用上述 ISO 安装过程手动运行 coreos-installer。

15.3.14.3.6. 自定义 live RHCOS ISO 或 PXE 安装

您可以通过将 Ignition 配置文件直接注入镜像中来使用 live ISO 镜像或 PXE 环境来安装 RHCOS。这会创建一个自定义镜像，供您用来置备系统。

对于 ISO 镜像，此操作的机制是 coreos-installer iso custom 子命令，它使用您的配置修改 .iso 文件。同样，PXE 环境的机制是 coreos-installer pxe customize 子命令，它会创建一个包含自定义的新 initramfs 文件。

custom 子命令是一个通用工具，也可以嵌入其他类型的自定义。以下任务是一些更常见的自定义示例：

- 当公司安全策略需要使用时，注入自定义的 CA 证书。
- 在不需要内核参数的情况下配置网络设置。
- 嵌入任意预安装和安装后脚本或二进制文件。

15.3.14.3.7. 自定义 live RHCOS ISO 镜像

您可以使用 coreos-installer iso custom 子命令直接自定义 live RHCOS ISO 镜像。当您引导 ISO 镜像时，会自动应用自定义。

您可以使用此功能配置 ISO 镜像来自动安装 RHCOS。

流程

1. 从 coreos-installer 镜像镜像页面下载 coreos-installer 二进制文件。

2. 从 RHCOS 镜像镜像页面和 Ignition 配置文件检索 RHCOS ISO 镜像，然后运行以下命令来直接将 Ignition 配置注入 ISO 镜像：

```
$ coreos-installer iso customize rhcos-<version>-live.x86_64.iso \
  --dest-ignition bootstrap.ign \ 1
  --dest-device /dev/disk/by-id/scsi-<serial_number> 2
```

1. 从 openshift-installer 安装程序生成的 Ignition 配置文件。

2. 当您指定这个选项时，ISO 镜像会自动运行安装。否则，镜像为安装配置，但不会自动安装，除非您指定了 coreos.inst.install_dev 内核参数。

3. 可选：要删除 ISO 镜像自定义并将镜像返回到其 pristine 状态，请运行：

```
$ coreos-installer iso reset rhcos-<version>-live.x86_64.iso
```

现在，您可以重新自定义 live ISO 镜像，或者在其 pristine 状态中使用它。

应用您的自定义会影响每个后续 RHCOS 引导。

15.3.14.3.7.1. 修改实时安装 ISO 镜像以启用串行控制台

在使用 OpenShift Container Platform 4.12 及更高版本安装的集群中，串行控制台默认被禁用，所有输出都会写入图形控制台。您可以按照以下流程启用串行控制台。

流程

1. 从 coreos-installer 镜像镜像页面下载 coreos-installer 二进制文件。

2. 从 RHCOS image mirror 页中获取 RHCOS ISO 镜像，并运行以下命令来自定义 ISO 镜像，使串行控制台能够接收输出：

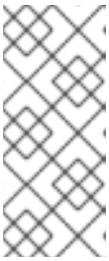
```
$ coreos-installer iso customize rhcos-<version>-live.x86_64.iso \
  --dest-ignition <path> \ 1
  --dest-console tty0 \ 2
  --dest-console ttyS0,<options> \ 3
  --dest-device /dev/disk/by-id/scsi-<serial_number> 4
```

1 要安装 Ignition 配置的位置。

2 所需的二级控制台。在这种情况下，是图形控制台。省略这个选项将禁用图形控制台。

3 所需的主控制台。在这种情况下，是串行控制台。options 字段定义 baud 速率和其他设置。此字段的一个常见值为 115200n8。如果没有提供选项，则使用默认内核值 9600n8。有关这个选项格式的更多信息，请参阅 Linux 内核串口控制台文档。

4 要安装到的指定磁盘。如果省略这个选项，ISO 镜像会自动运行安装程序，除非还指定了 coreos.inst.install_dev 内核参数。



注意

--dest-console 选项会影响安装的系统，而不是实时 ISO 系统。要修改 live ISO 系统的控制台，请使用 --live-karg-append 选项并使用 console= 指定控制台。

自定义会被应用，并影响每个后续 ISO 镜像引导。

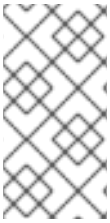
3. 可选：要删除 ISO 镜像自定义并将镜像返回到其原始状态，请运行以下命令：

```
$ coreos-installer iso reset rhcos-<version>-live.x86_64.iso
```

现在，您可以重新自定义 live ISO 镜像，或者在其原始状态中使用它。

15.3.14.3.7.2. 修改实时安装 ISO 镜像以使用自定义证书颁发机构

您可以使用 custom 子命令的 --ignition-ca 标志向 Ignition 提供证书颁发机构(CA)证书。您可以在安装过程中使用 CA 证书，并在置备安装的系统时使用 CA 证书。



注意

自定义 CA 证书会影响 Ignition 获取远程资源的方式，但它们不会影响安装到系统中的证书。

流程

1. 从 coreos-installer 镜像镜像页面下载 coreos-installer 二进制文件。

2. 从 RHCOS 镜像镜像页面检索 RHCOS ISO 镜像，并运行以下命令来自定义 ISO 镜像以用于自定义 CA：

```
$ coreos-installer iso customize rhcos-<version>-live.x86_64.iso --ignition-ca cert.pem
```

重要

coreos.inst.ignition_url 内核参数无法使用 --ignition-ca 标志。您必须使用 --dest-ignition 标志为每个集群创建自定义镜像。

应用自定义 CA 证书会影响每个后续 RHCOS 引导。

15.3.14.3.7.3. 使用自定义网络设置修改实时安装 ISO 镜像

您可以将 NetworkManager 密钥文件嵌入到 live ISO 镜像中，并使用 customize 子命令的 --network-keyfile 标志将其传递给安装的系统。

警告

在创建连接配置文件时，您必须在连接配置文件的文件名中使用 .nmconnection 文件名扩展名。如果不使用 .nmconnection 文件名扩展，集群会将连接配置集应用到 live 环境，但它不会在集群首次启动节点时应用配置，从而导致无法正常工作的设置。

流程

1. 从 coreos-installer 镜像镜像页面下载 coreos-installer 二进制文件。

2. 为绑定接口创建连接配置集。例如，在本地目录中创建 bond0.nmconnection 文件，其内容如下：

```
[connection]
id=bond0
type=bond
interface-name=bond0
multi-connect=1
permissions=

[ethernet]
mac-address-blacklist=

[bond]
miimon=100
mode=active-backup

[ipv4]
method=auto

[ipv6]
method=auto

[proxy]
```

3. 为二级接口创建连接配置集，以添加到绑定中。例如，在本地目录中创建 bond0-proxy-em1.nmconnection 文件，其内容如下：

```
[connection]
id=em1
type=ethernet
interface-name=em1
```

```
master=bond0
multi-connect=1
permissions=
slave-type=bond

[ethernet]
mac-address-blacklist=
```

4. 为二级接口创建连接配置集，以添加到绑定中。例如，在本地目录中创建 bond0-proxy-em2.nmconnection 文件，其内容如下：

```
[connection]
id=em2
type=ethernet
interface-name=em2
master=bond0
multi-connect=1
permissions=
slave-type=bond

[ethernet]
mac-address-blacklist=
```

5. 从 RHCOS 镜像镜像页面检索 RHCOS ISO 镜像，并运行以下命令来使用您配置网络自定义 ISO 镜像：

```
$ coreos-installer iso customize rhcos-<version>-live.x86_64.iso \
  --network-keyfile bond0.nmconnection \
  --network-keyfile bond0-proxy-em1.nmconnection \
  --network-keyfile bond0-proxy-em2.nmconnection
```

网络设置应用于实时系统，并传输到目标系统。

15.3.14.3.7.4. 为 iSCSI 引导设备自定义实时安装 ISO 镜像

您可以设置 iSCSI 目标和 initiator 值，以便使用自定义的 live RHCOS 镜像版本自动挂载、引导和配置。

先决条件

1. 您有一个 iSCSI 目标，您要在其中安装 RHCOS。

流程

1. 从 coreos-installer 镜像镜像页面下载 coreos-installer 二进制文件。

2. 从 RHCOS 镜像镜像页面检索 RHCOS ISO 镜像，并运行以下命令使用以下信息自定义 ISO 镜像：

```
$ coreos-installer iso customize \
  --pre-install mount-iscsi.sh \ 1
  --post-install unmount-iscsi.sh \ 2
  --dest-device /dev/disk/by-path/<IP_address>:<port>-iscsi-<target_iqn>-lun-<lun> \
3
  --dest-ignition config.ign \ 4
  --dest-karg-append rd.iscsi.initiator=<initiator_iqn> \ 5
  --dest-karg-append netroot=<target_iqn> \ 6
  -o custom.iso rhcos-<version>-live.x86_64.iso
```

1 安装之前运行的脚本。它应包含用于挂载 iSCSI 目标以及启用多路径的任何命令的 iscsiadm 命令。

2 安装后运行的脚本。它应包含命令 iscsiadm --mode node --logout=all。

3 目标系统的位置。您必须提供目标门户的 IP 地址、关联的端口号、目标 iSCSI 节点采用 IQN 格式，以及 iSCSI 逻辑单元号(LUN)。

4 目标系统的 Ignition 配置。

5 iSCSI 启动器或客户端，以 IQN 格式的名称。启动器构成连接到 iSCSI 目标的一个会话。

6 iSCSI 目标或服务器的名称（IQN 格式）。

有关 dracut 支持的 iSCSI 选项的更多信息，请参阅 dracut.cmdline 手册页。

15.3.14.3.7.5. 使用 iBFT 为 iSCSI 引导设备自定义实时安装 ISO 镜像

您可以设置 iSCSI 目标和 initiator 值，以便使用自定义的 live RHCOS 镜像版本自动挂载、引导和配置。

先决条件

1. 您有一个 iSCSI 目标，您要在其中安装 RHCOS。
2. 可选：有多路径 iSCSI 目标。

流程

1. 从 coreos-installer 镜像镜像页面下载 coreos-installer 二进制文件。
2. 从 RHCOS 镜像镜像页面检索 RHCOS ISO 镜像，并运行以下命令使用以下信息自定义 ISO 镜像：

```
$ coreos-installer iso customize \
  --pre-install mount-iscsi.sh \ 1
  --post-install unmount-iscsi.sh \ 2
  --dest-device /dev/mapper/mpatha \ 3
  --dest-ignition config.ign \ 4
  --dest-karg-append rd.iscsi.firmware=1 \ 5
  --dest-karg-append rd.multipath=default \ 6
  -o custom.iso rhcos-<version>-live.x86_64.iso
```

1. 安装之前运行的脚本。它应包含用于挂载 iSCSI 目标以及启用多路径的任何命令的 iscsiadm 命令。
2. 安装后运行的脚本。它应包含命令 iscsiadm --mode node --logout=all。
3. 设备的路径。如果连接了多个多路径设备，或者需要明确指定，您使用多路径（/dev/mapper/mpatha），您可以使用 /dev/disk/by-path 中可用的 World Wide Name (WWN) 符号链接。

4

目标系统的 Ignition 配置。

5

iSCSI 参数从 BIOS 固件读取。

6

可选：如果您要启用多路径，请包含此参数。

有关 dracut 支持的 iSCSI 选项的更多信息，请参阅 dracut.cmdline 手册页。

15.3.14.3.8. 自定义 live RHCOS PXE 环境

您可以使用 coreos-installer pxe customize 子命令直接自定义 live RHCOS PXE 环境。当您引导 PXE 环境时，会自动应用自定义。

您可以使用此功能配置 PXE 环境来自动安装 RHCOS。

流程

1. 从 coreos-installer 镜像镜像页面下载 coreos-installer 二进制文件。

2. 从 RHCOS 镜像镜像页面和 Ignition 配置文件获取 RHCOS kernel, initramfs 和 rootfs 文件，然后运行以下命令创建一个包含 Ignition 配置中的自定义的新 initramfs 文件：

```
$ coreos-installer pxe customize rhcos-<version>-live-initramfs.x86_64.img \
  --dest-ignition bootstrap.ign \ 1
  --dest-device /dev/disk/by-id/scsi-<serial_number> \ 2
  -o rhcos-<version>-custom-initramfs.x86_64.img 3
```

1

从 openshift-installer 生成的 Ignition 配置文件。

2

当您指定这个选项时，PXE 环境会自动运行安装。否则，为安装配置了镜像，除非指定了 coreos.inst.install_dev 内核参数，否则不会自动这样做。

3

在 PXE 配置中使用自定义 initramfs 文件。添加 ignition.firstboot 和 ignition.platform.id=metal 内核参数（如果它们尚不存在）。

应用您的自定义会影响每个后续 RHCOS 引导。

15.3.14.3.8.1. 修改实时安装 PXE 环境以启用串行控制台。

在使用 OpenShift Container Platform 4.12 及更高版本安装的集群中，串行控制台默认被禁用，所有输出都会写入图形控制台。您可以按照以下流程启用串行控制台。

流程

1. 从 coreos-installer 镜像镜像页面下载 coreos-installer 二进制文件。

2. 从 RHCOS image mirror 页面获取 kernel, initramfs 和 rootfs 文件，然后运行以下命令来创建新的自定义 initramfs 文件，以便串行控制台接收输出：

```
$ coreos-installer pxe customize rhcos-<version>-live-initramfs.x86_64.img \
  --dest-ignition <path> \ 1
  --dest-console tty0 \ 2
  --dest-console ttyS0,<options> \ 3
  --dest-device /dev/disk/by-id/scsi-<serial_number> \ 4
  -o rhcos-<version>-custom-initramfs.x86_64.img 5
```

1

要安装 Ignition 配置的位置。

2

所需的二级控制台。在这种情况下，是图形控制台。省略这个选项将禁用图形控制台。

3

所需的主控制台。在这种情况下，是串行控制台。options 字段定义 baud 速率和其他设置。此字段的一个常见值为 115200n8。如果没有提供选项，则使用默认内核值 9600n8。有关这个选项格式的更多信息，请参阅 Linux 内核串口控制台文档。

4

要安装到的指定磁盘。如果省略这个选项，PXE 环境会自动运行安装程序，除非还指定了 coreos.inst.install_dev 内核参数。

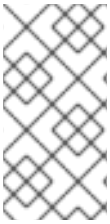
5

在 PXE 配置中使用自定义 initramfs 文件。添加 ignition.firstboot 和 ignition.platform.id=metal 内核参数（如果它们尚不存在）。

自定义会被应用，并影响 PXE 环境的每个后续引导。

15.3.14.3.8.2. 修改实时安装 PXE 环境以使用自定义证书颁发机构

您可以使用 custom 子命令的 --ignition-ca 标志向 Ignition 提供证书颁发机构(CA)证书。您可以在安装过程中使用 CA 证书，并在置备安装的系统时使用 CA 证书。



注意

自定义 CA 证书会影响 Ignition 获取远程资源的方式，但它们不会影响安装到系统中的证书。

流程

1. 从 coreos-installer 镜像镜像页面下载 coreos-installer 二进制文件。

2. 从 RHCOS 镜像镜像页面获取 RHCOS kernel、initramfs 和 rootfs 文件，并运行以下命令来创建一个新的自定义 initramfs 文件以用于自定义 CA：

```
$ coreos-installer pxe customize rhcos-<version>-live-initramfs.x86_64.img \
  --ignition-ca cert.pem \
  -o rhcos-<version>-custom-initramfs.x86_64.img
```

3. 在 PXE 配置中使用自定义 initramfs 文件。添加 ignition.firstboot 和 ignition.platform.id=metal 内核参数（如果它们尚不存在）。

重要

coreos.inst.ignition_url 内核参数无法使用 --ignition-ca 标志。您必须使用 --dest-ignition 标志为每个集群创建自定义镜像。

应用自定义 CA 证书会影响每个后续 RHCOS 引导。

15.3.14.3.8.3. 使用自定义网络设置修改实时安装 PXE 环境

您可以将 NetworkManager 密钥文件嵌入到 live PXE 环境中，并使用 customize 子命令的 --network-keyfile 标志将其传递给安装的系统。

警告

在创建连接配置文件时，您必须在连接配置文件的文件名中使用 .nmconnection 文件名扩展名。如果不使用 .nmconnection 文件名扩展，集群会将连接配置集应用到 live 环境，但它不会在集群首次启动节点时应用配置，从而导致无法正常工作的设置。

流程

1. 从 coreos-installer 镜像镜像页面下载 coreos-installer 二进制文件。

2. 为绑定接口创建连接配置集。例如，在本地目录中创建 bond0.nmconnection 文件，其内容如下：

```
[connection]
id=bond0
type=bond
interface-name=bond0
multi-connect=1
permissions=

[ethernet]
mac-address-blacklist=

[bond]
miimon=100
```

```
mode=active-backup

[ipv4]
method=auto

[ipv6]
method=auto

[proxy]
```

3. 为二级接口创建连接配置集，以添加到绑定中。例如，在本地目录中创建 bond0-proxy-em1.nmconnection 文件，其内容如下：

```
[connection]
id=em1
type=ethernet
interface-name=em1
master=bond0
multi-connect=1
permissions=
slave-type=bond

[ethernet]
mac-address-blacklist=
```

4. 为二级接口创建连接配置集，以添加到绑定中。例如，在本地目录中创建 bond0-proxy-em2.nmconnection 文件，其内容如下：

```
[connection]
id=em2
type=ethernet
interface-name=em2
master=bond0
multi-connect=1
permissions=
slave-type=bond

[ethernet]
mac-address-blacklist=
```

5. 从 RHCOS 镜像镜像页面获取 RHCOS kernel、initramfs 和 rootfs 文件，并运行以下命令来创建一个新的自定义 initramfs 文件，它包括您的配置网络：

```
$ coreos-installer pxe customize rhcos-<version>-live-initramfs.x86_64.img \
    --network-keyfile bond0.nmconnection \
    --network-keyfile bond0-proxy-em1.nmconnection \
    --network-keyfile bond0-proxy-em2.nmconnection \
    -o rhcos-<version>-custom-initramfs.x86_64.img
```

6. 在 PXE 配置中使用自定义 initramfs 文件。添加 ignition.firstboot 和 ignition.platform.id=metal 内核参数（如果它们尚不存在）。

网络设置应用于实时系统，并传输到目标系统。

15.3.14.3.8.4. 为 iSCSI 引导设备自定义实时安装 PXE 环境

您可以设置 iSCSI 目标和 initiator 值，以便使用自定义的 live RHCOS 镜像版本自动挂载、引导和配置。

先决条件

1. 您有一个 iSCSI 目标，您要在其中安装 RHCOS。

流程

1. 从 coreos-installer 镜像镜像页面下载 coreos-installer 二进制文件。

2. 从 RHCOS image mirror 页获取 RHCOS kernel、initramfs 和 rootfs 文件，并运行以下命令来使用以下信息创建新的自定义 initramfs 文件：

```
$ coreos-installer pxe customize \
    --pre-install mount-iscsi.sh \ 1
    --post-install unmount-iscsi.sh \ 2
    --dest-device /dev/disk/by-path/<IP_address>:<port>-iscsi-<target_iqn>-lun-<lun> \
3
    --dest-ignition config.ign \ 4
    --dest-karg-append rd.iscsi.initiator=<initiator_iqn> \ 5
    --dest-karg-append netroot=<target_iqn> \ 6
    -o custom.img rhcos-<version>-live-initramfs.x86_64.img
```

1. 安装之前运行的脚本。它应包含用于挂载 iSCSI 目标以及启用多路径的任何命令的 iscsiadm 命令。

2. 安装后运行的脚本。它应包含命令 iscsiadm --mode node --logout=all。

3.

目标系统的位置。您必须提供目标门户的 IP 地址、关联的端口号、目标 iSCSI 节点采用 IQN 格式，以及 iSCSI 逻辑单元号(LUN)。

4 目标系统的 Ignition 配置。

5 iSCSI 启动器或客户端，以 IQN 格式的名称。启动器构成连接到 iSCSI 目标的一个会话。

6 iSCSI 目标或服务器的名称（IQN 格式）。

有关 dracut 支持的 iSCSI 选项的更多信息，请参阅 dracut.cmdline 手册页。

15.3.14.3.8.5. 使用 iBFT 为 iSCSI 引导设备自定义实时安装 PXE 环境

您可以设置 iSCSI 目标和 initiator 值，以便使用自定义的 live RHCOS 镜像版本自动挂载、引导和配置。

先决条件

1. 您有一个 iSCSI 目标，您要在其中安装 RHCOS。
2. 可选：有多路径 iSCSI 目标。

流程

1. 从 coreos-installer 镜像镜像页面下载 coreos-installer 二进制文件。
2. 从 RHCOS image mirror 页获取 RHCOS kernel、initramfs 和 rootfs 文件，并运行以下命令来使用以下信息创建新的自定义 initramfs 文件：

```
$ coreos-installer pxe customize \
  --pre-install mount-iscsi.sh \ 1
```

```
--post-install unmount-iscsi.sh \ 2
--dest-device /dev/mapper/mpatha \ 3
--dest-ignition config.ign \ 4
--dest-karg-append rd.iscsi.firmware=1 \ 5
--dest-karg-append rd.multipath=default \ 6
-o custom.img rhcos-<version>-live-initramfs.x86_64.img
```

1

安装之前运行的脚本。它应包含用于挂载 iSCSI 目标的 iscsiadm 命令。

2

安装后运行的脚本。它应包含命令 iscsiadm --mode node --logout=all。

3

设备的路径。如果连接了多个多路径设备，或者需要明确指定，您使用多路径（/dev/mapper/mpatha），您可以使用 /dev/disk/by-path 中可用的 World Wide Name (WWN) 符号链接。

4

目标系统的 Ignition 配置。

5

iSCSI 参数从 BIOS 固件读取。

6

可选：如果您要启用多路径，请包含此参数。

有关 dracut 支持的 iSCSI 选项的更多信息，请参阅 dracut.cmdline 手册页。

15.3.14.3.9. 高级 RHCOS 安装参考

本节演示了网络配置和其他高级选项，允许您修改 Red Hat Enterprise Linux CoreOS(RHCOS)手动安装过程。下表描述了您可以用于 RHCOS live 安装程序和 coreos-installer 命令的内核参数和命令行选项。

15.3.14.3.9.1. ISO 安装的网络和绑定选项

如果从 ISO 镜像安装 RHCOS，您可以在引导镜像时手动添加内核参数，以便为节点配置网络。如

果没有指定网络参数，当 RHCOS 检测到需要网络来获取 Ignition 配置文件时，在 initramfs 中激活 DHCP。

重要

在手动添加网络参数时，还必须添加 rd.neednet=1 内核参数，以便在 initramfs 中启动网络。

以下信息提供了在 RHCOS 节点上为 ISO 安装配置网络和绑定的示例。示例描述了如何使用 ip=、name server = 和 bond= 内核参数。

注意

添加内核参数时顺序非常重要： ip=、 name server=，然后 bond=。

网络选项在系统引导过程中传递给 dracut 工具。有关 dracut 支持的网络选项的更多信息，请参阅 dracut.cmdline 手册页。

以下示例是 ISO 安装的网络选项。

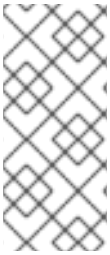
配置 DHCP 或静态 IP 地址

要配置 IP 地址，可使用 DHCP(ip=dhcp)或设置单独的静态 IP 地址(ip=<host_ip>)。如果设置静态 IP，则必须在每个节点上识别 DNS 服务器 IP 地址（名称服务器=<dns_ip>）。以下示例集：

- 节点的 IP 地址为 10.10.10.2
- 网关地址为 10.10.10.254
- 子网掩码为 255.255.255.0
- 到 core0.example.com 的主机名

- DNS 服务器地址为 4.4.4.41
- 自动配置值为 none。当以静态方式配置 IP 网络时，不需要自动配置。

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
nameserver=4.4.4.41
```



注意

当您使用 DHCP 为 RHCOS 机器配置 IP 寻址时，机器还通过 DHCP 获取 DNS 服务器信息。对于基于 DHCP 的部署，您可以通过 DHCP 服务器配置定义 RHCOS 节点使用的 DNS 服务器地址。

配置没有静态主机名的 IP 地址

您可以在不分配静态主机名的情况下配置 IP 地址。如果用户没有设置静态主机名，则会提取并通过反向 DNS 查找自动设置。要在没有静态主机名的情况下配置 IP 地址，请参考以下示例：

- 节点的 IP 地址为 10.10.10.2
- 网关地址为 10.10.10.254
- 子网掩码为 255.255.255.0
- DNS 服务器地址为 4.4.4.41
- 自动配置值为 none。当以静态方式配置 IP 网络时，不需要自动配置。

```
ip=10.10.10.2::10.10.10.254:255.255.255.0::enp1s0:none
nameserver=4.4.4.41
```

指定多个网络接口

您可以通过设置多个 ip= 条目来指定多个网络接口。

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
ip=10.10.10.3::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none
```

-

配置默认网关和路由

可选：您可以通过设置 a rd.route= 值来配置到额外网络的路由。

注意

当您配置一个或多个网络时，需要一个默认网关。如果额外网络网关与主要网络网关不同，则默认网关必须是主要网络网关。

- 运行以下命令来配置默认网关：

```
ip=::10.10.10.254::::
```

- 输入以下命令为额外网络配置路由：

```
rd.route=20.20.20.0/24:20.20.20.254:enp2s0
```

在单个接口中禁用 DHCP

您可以在单一接口中禁用 DHCP，例如当有两个或者多个网络接口时，且只有一个接口被使用。在示例中，enp1s0 接口具有一个静态网络配置，而 enp2s0 禁用了 DHCP，不使用它：

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
ip=::::core0.example.com:enp2s0:none
```

合并 DHCP 和静态 IP 配置

您可以将系统上的 DHCP 和静态 IP 配置与多个网络接口合并，例如：

```
ip=enp1s0:dhcp
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none
```

在独立接口上配置 VLAN

可选： 您可以使用 vlan= 参数在单个接口上配置 VLAN。

- 要在网络接口中配置 VLAN 并使用静态 IP 地址，请运行以下命令：

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0.100:none
vlan=enp2s0.100:enp2s0
```

- 要在网络接口中配置 VLAN 并使用 DHCP，请运行以下命令：

```
ip=enp2s0.100:dhcp
vlan=enp2s0.100:enp2s0
```

提供多个 DNS 服务器

您可以通过为每个服务器添加一个 nameserver= 条目来提供多个 DNS 服务器，例如

```
nameserver=1.1.1.1
nameserver=8.8.8.8
```

将多个网络接口绑定到一个接口

可选：您可以使用 bond= 选项将多个网络接口绑定到一个接口。请参见以下示例：

- 配置绑定接口的语法为：bond=<name>[:<network_interfaces>][:options]

  <name> 是绑定设备名称 (bond0)、<network_interfaces> 代表以逗号分隔的物理（以太网）接口列表(em1,em2)，*options* 是用逗号分开的绑定选项列表。输入 modinfo bonding 查看可用选项。

- 当使用 bond= 创建绑定接口时，您必须指定如何分配 IP 地址以及绑定接口的其他信息。

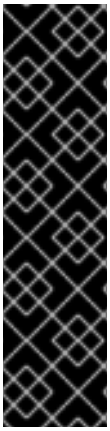
  - 要将绑定接口配置为使用 DHCP，请将绑定的 IP 地址设置为 dhcp。例如：

    ```
    bond=bond0:em1,em2:mode=active-backup
    ip=bond0:dhcp
    ```

  - 要将绑定接口配置为使用静态 IP 地址，请输入您需要的特定 IP 地址和相关信息。例如：

    ```
    bond=bond0:em1,em2:mode=active-backup
    ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0:none
    ```

将多个 SR-IOV 网络接口绑定到双端口 NIC 接口



重要

支持与为 SR-IOV 设备启用 NIC 分区关联的第 1 天操作只是一个技术预览功能。技术预览功能不受红帽产品服务等级协议（SLA）支持，且功能可能并不完整。红帽不推荐在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。

有关红帽技术预览功能支持范围的更多信息，请参阅技术预览功能支持范围。

可选： 您可以使用 bond= 选项将多个 SR-IOV 网络接口绑定到双端口 NIC 接口。

在每个节点上，您必须执行以下任务：

1. 按照管理 SR-IOV 设备中的指导创建 SR-IOV 虚拟功能(VF)。按照"将 SR-IOV 网络设备附加到虚拟机"部分中的步骤操作。

2. 创建绑定，将所需的 VF 附加到绑定，并根据配置网络绑定的指导设置绑定链接状态。按照任何描述的步骤创建绑定。

以下示例演示了您必须使用的语法：

- 配置绑定接口的语法为：bond=<name>[:<network_interfaces>][:options]

  <name> 是绑定设备名称 (bond0)、<network_interfaces> 由内核中已知的名称来代表虚拟功能(VF)，并显示在 ip link 命令的输出中 (eno1f0,eno2f0)， *options* 是以逗号分隔的绑定选项列表。输入 modinfo bonding 查看可用选项。

- 当使用 bond= 创建绑定接口时，您必须指定如何分配 IP 地址以及绑定接口的其他信息。

  - 要将绑定接口配置为使用 DHCP，请将绑定的 IP 地址设置为 dhcp。例如：

    ```
    bond=bond0:eno1f0,eno2f0:mode=active-backup
    ip=bond0:dhcp
    ```

  -

要将绑定接口配置为使用静态 IP 地址，请输入您需要的特定 IP 地址和相关信息。例如：

```
bond=bond0:eno1f0,eno2f0:mode=active-backup
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0:none
```

使用网络团队

可选：您可以使用 team= 参数来将网络团队用作绑定的替代选择：

- 配置组接口的语法为：team=name[:network_interfaces]

name 是组设备名称(team0)，*network_interfaces* 代表以逗号分隔的物理（以太网）接口（em1、em2）列表。

注意

当 RHCOS 切换到即将推出的 RHEL 版本时，团队(team)功能被计划弃用。如需更多信息，请参阅红帽知识库文章。

使用以下示例配置网络团队：

```
team=team0:em1,em2
ip=team0:dhcp
```

15.3.14.3.9.2. ISO 和 PXE 安装的 coreos-installer 选项

从 ISO 镜像引导 RHCOS live 环境后，您可以通过 在命令提示符下运行 coreos-installer install <options> <device > 来安装 RHCOS。

下表显示了您可以传递给 coreos-installer 命令的子命令、选项和参数。

表 15.30. coreos-installer 子命令、命令行选项和参数

coreos-installer install 子命令	
子命令	*描述*
$ coreos-installer install <options> <device>	在 ISO 镜像中嵌入 Ignition 配置。

coreos-installer install 子命令选项	
选项	***描述***
-u, --image-url <url>	手动指定镜像 URL。
-f,--image-file <path>	手动指定本地镜像文件。用于调试。
-i, --ignition-file <path>	从文件中嵌入 Ignition 配置。
-i,--ignition-url <URL>	从 URL 嵌入 Ignition 配置。
--ignition-hash <digest>	Ignition 配置 **的 type-value** 的摘要值。
-p, --platform <name>	覆盖已安装系统的 Ignition 平台 ID。
--console <spec>	为安装的系统设置内核和引导装载程序控制台。有关 **<spec>** 格式的更多信息，请参阅 Linux 内核串口控制台文档。
--append-karg <arg>...	将默认内核参数附加到安装的系统中。
--delete-karg <arg>...	从安装的系统中删除默认内核参数。
-n, --copy-network	从安装环境中复制网络配置。  **重要** **copy-network** 选项仅复制在 **/etc/NetworkManager/system-connections** 下找到的网络配置。特别是，它不会复制系统主机名。
--network-dir <path>	用于 **-n**。默认为 **/etc/NetworkManager/system-connections/**。
--save-partlabel <lx>..	使用这个标签 glob 保存分区。
--save-partindex <id>...	使用这个数值或范围保存分区。
--insecure	跳过 RHCOS 镜像签名验证。
--insecure-ignition	允许没有 HTTPS 或 hash 的 Ignition URL。
--architecture <name>	目标 CPU 架构.有效值为 **x86_64** 和 **aarch64**。
--preserve-on-error	出错时不要清除分区表。

-h,--help	打印帮助信息.
coreos-installer install 子命令参数	
参数	***描述***
<device>	目标设备.
coreos-installer ISO 子命令	
子命令	***描述***
$ coreos-installer iso customize <options> <ISO_image>	自定义 RHCOS live ISO 镜像。
coreos-installer iso reset <options> <ISO_image>	将 RHCOS live ISO 镜像恢复到默认设置。
coreos-installer iso ignition remove <options> <ISO_image>	从 ISO 镜像中删除嵌入的 Ignition 配置。
coreos-installer ISO customize 子命令选项	
选项	***描述***
--dest-ignition <path>	将指定的 Ignition 配置文件合并到目标系统的新配置片段中。
--dest-console <spec>	为目标系统指定内核和引导装载程序控制台。
--dest-device <path>	安装并覆盖指定的目标设备。
--dest-karg-append <arg>	为每个目标系统引导添加一个内核参数。
--dest-karg-delete <arg>	从目标系统的每个引导中删除内核参数。
--network-keyfile <path>	使用指定的 NetworkManager 密钥文件进行实时和目标系统配置网络。
--ignition-ca <path>	指定要被 Ignition 信任的额外 TLS 证书颁发机构。
--pre-install <path>	在安装之前运行指定的脚本。
--post-install <path>	安装后运行指定的脚本。
--installer-config <path>	应用指定的安装程序配置文件。

--live-ignition <path>	将指定的 Ignition 配置文件合并到实时环境的新配置片段中。
--live-karg-append <arg>	为每个实时环境引导添加一个内核参数。
--live-karg-delete <arg>	从实时环境每次引导时删除内核参数。
--live-karg-replace <k=o=n>	在每次启动 live 环境时替换内核参数，格式为 **key=old=new**。
-f,--force	覆盖现有的 Ignition 配置。
-o,--output <path>	将 ISO 写入到新的输出文件。
-h,--help	打印帮助信息.
coreos-installer PXE 子命令	
子命令	***描述***
请注意，并非所有子命令都接受所有这些选项。	
coreos-installer pxe customize <options> <path>	自定义 RHCOS live PXE 引导配置。
coreos-installer pxe ignition wrap <options>	在镜像中嵌套 Ignition 配置。
coreos-installer pxe ignition unwrap <options> <image_name>	在镜像中显示嵌套的 Ignition 配置。
coreos-installer PXE customize 子命令选项	
选项	***描述***
请注意，并非所有子命令都接受所有这些选项。	
--dest-ignition <path>	将指定的 Ignition 配置文件合并到目标系统的新配置片段中。
--dest-console <spec>	为目标系统指定内核和引导装载程序控制台。
--dest-device <path>	安装并覆盖指定的目标设备。
--network-keyfile <path>	使用指定的 NetworkManager 密钥文件进行实时和目标系统配置网络。
--ignition-ca <path>	指定要被 Ignition 信任的额外 TLS 证书颁发机构。

--pre-install <path>	在安装之前运行指定的脚本。
post-install <path>	安装后运行指定的脚本。
--installer-config <path>	应用指定的安装程序配置文件。
--live-ignition <path>	将指定的 Ignition 配置文件合并到实时环境的新配置片段中。
-o, --output <path>	将 initramfs 写入一个新输出文件。  **注意** PXE 环境需要这个选项。
-h,--help	打印帮助信息.

15.3.14.3.9.3. coreos.inst 引导选项用于 ISO 或 PXE 安装

您可以通过将 coreos.inst boot 参数传递给 RHCOS live 安装程序，在引导时自动调用 coreos-installer 选项。这些是在标准引导参数之外提供的。

- 对于 ISO 安装，可以通过在启动加载器菜单中中断自动引导来添加 coreos.inst 选项。您可以在突出显示 RHEL CoreOS(Live) 菜单选项时按 TAB 来中断自动引导。
- 对于 PXE 或 iPXE 安装，在引导 RHCOS live 安装程序前，coreos.inst 选项必须添加到 APPEND 行。

下表显示了用于 ISO 和 PXE 安装的 RHCOS live 安装程序 coreos.inst 引导选项。

表 15.31. coreos.inst 引导选项

参数	描述
coreos.inst.install_dev	必需。要安装到的系统中的块设备。建议您使用完整路径，如 **/dev/sda**，但 **允许使用**。
coreos.inst.ignition_url	可选：嵌入到安装的系统中的 Ignition 配置的 URL。如果没有指定 URL，则不会嵌入 Ignition 配置。仅支持 HTTP 和 HTTPS 协议。

参数	描述
coreos.inst.save_partlabel	可选：在安装过程中要保留的分区分离标签。允许使用 glob 风格的通配符。指定分区不需要存在。
coreos.inst.save_partindex	可选：在安装过程中压缩要保留的分区索引。允许 **m-n** 范围，**m** 或 **n** 可以被省略。指定分区不需要存在。
coreos.inst.insecure	可选：将 **coreos.inst.image_url** 指定的 OS 镜像提交取消签名。
coreos.inst.image_url	可选：下载并安装指定的 RHCOS 镜像。 ● 这个参数不应该在生产环境中使用，而是只用于调试目的。 ● 虽然此参数可用于安装与 live 介质不匹配的 RHCOS 版本，但建议您使用与您要安装版本匹配的介质。 ● 如果您使用 **coreos.inst.image_url**，还必须使用 **coreos.inst.insecure**。这是因为，裸机介质没有为 OpenShift Container Platform 进行 GPG 签名。 ● 仅支持 HTTP 和 HTTPS 协议。
coreos.inst.skip_reboot	可选：安装后系统不会重启。安装完成后，您将收到提示，提示您检查在安装过程中发生的情况。这个参数不应该在生产环境中使用，而是只用于调试目的。
coreos.inst.platform_id	可选：安装 RHCOS 镜像的平台的 Ignition 平台 ID。默认为 **metal**。这个选项决定是否从云供应商（如 VMware）请求 Ignition 配置。例如：**coreos.inst.platform_id=vmware**。
ignition.config.url	可选：用于实时引导的 Ignition 配置的 URL。例如，这可用于自定义调用 **coreos-installer** 的方式，或者用于在安装前或安装后运行代码。这与 **coreos.inst.ignition_url**（这是已安装系统的 Ignition 配置）不同。

15.3.14.4. 在 RHCOS 上启用带有内核参数的多路径

RHCOS 支持主磁盘上的多路径，支持更强大的硬件故障弹性，以获得更高的主机可用性。

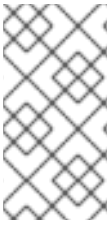
您可以在安装时为 OpenShift Container Platform 4.8 或更高版本置备的节点启用多路径。虽然安装后支持可以通过机器配置激活多路径来实现，但建议在安装过程中启用多路径。

在任何 I/O 到未优化路径会导致 I/O 系统错误的设置中，您必须在安装时启用多路径。

重要

在 IBM Z® 和 IBM® LinuxONE 中，您只能在在安装过程中为它配置集群时启用多路径。如需更多信息，请参阅在 *IBM Z® 和 IBM® LinuxONE 上安装使用 z/VM 的集群"安装 RHCOS 并启动 OpenShift Container Platform bootstrap 过程"*。

以下流程在安装时启用多路径，并在 coreos-installer install 命令中附加内核参数，以便安装的系统本身将使用从第一次引导开始的多路径。



注意

OpenShift Container Platform 不支持在从 4.6 或更早版本升级的节点上启用多路径作为 2 天的活动。

流程

1. 要启用多路径并启动 multipathd 守护进程，请在安装主机上运行以下命令：

```
$ mpathconf --enable && systemctl start multipathd.service
```

- 可选：如果引导 PXE 或 ISO，则可以通过从内核命令行添加 rd.multipath=default 来启用多路径。

2. 通过调用 coreos-installer 程序附加内核参数：

- 如果只有一个多路径设备连接到计算机，则应在路径 /dev/mapper/mpatha 上可用。例如：

```
$ coreos-installer install /dev/mapper/mpatha \ 1
--append-karg rd.multipath=default \
--append-karg root=/dev/disk/by-label/dm-mpath-root \
--append-karg rw
```

1

表示单一多路径设备的路径。

- 如果有多个多路径设备连接到计算机，或者更为明确，而不是使用 /dev/mapper/mpatha，则建议使用 /dev/disk/by-id 中可用的 World Wide Name(WWN)符号链接。例如：

```
$ coreos-installer install /dev/disk/by-id/wwn-<wwn_ID> \ 1
--append-karg rd.multipath=default \
--append-karg root=/dev/disk/by-label/dm-mpath-root \
--append-karg rw
```

1 表示目标多路径设备的 WWN ID。例如： 0xx194e957fcedb4841。

当使用特殊 coreos.inst.* 参数指示 live 安装程序时，这个符号链接也可以用作 coreos.inst.install_dev 内核参数。如需更多信息，请参阅"安装 RHCOS 和启动 OpenShift Container Platform bootstrap 过程"。

3. 前往其中一个 worker 节点并列出内核命令行参数（主机上的 /proc/cmdline 中），以检查内核参数是否正常工作：

```
$ oc debug node/ip-10-0-141-105.ec2.internal
```

输出示例

```
Starting pod/ip-10-0-141-105ec2internal-debug ...
To use host binaries, run `chroot /host`

sh-4.2# cat /host/proc/cmdline
...
rd.multipath=default root=/dev/disk/by-label/dm-mpath-root
...

sh-4.2# exit
```

您应看到添加的内核参数。

15.3.14.5. 在 iSCSI 引导设备中手动安装 RHCOS

您可以在 iSCSI 目标上手动安装 RHCOS。

先决条件

1. 您在 RHCOS live 环境中。
2. 您有一个要在其上安装 RHCOS 的 iSCSI 目标。

流程

1. 运行以下命令，从 live 环境中挂载 iSCSI 目标：

```
$ iscsiadm \
    --mode discovery \
    --type sendtargets
    --portal <IP_address> \ 1
    --login
```

1. 目标门户的 IP 地址。

2. 运行以下命令并使用必要的内核参数将 RHCOS 安装到 iSCSI 目标上，例如：

```
$ coreos-installer install \
    /dev/disk/by-path/ip-<IP_address>:<port>-iscsi-<target_iqn>-lun-<lun> \ 1
    --append-karg rd.iscsi.initiator=<initiator_iqn> \ 2
    --append.karg netroot=<target_iqn> \ 3
    --console ttyS0,115200n8
    --ignition-file <path_to_file>
```

1. 您要安装到的位置。您必须提供目标门户的 IP 地址、关联的端口号、目标 iSCSI 节点采用 IQN 格式，以及 iSCSI 逻辑单元号(LUN)。

2 iSCSI 启动器或客户端，以 IQN 格式的名称。启动器构成连接到 iSCSI 目标的一个会话。

3 iSCSI 目标或服务器的名称（IQN 格式）。

有关 dracut 支持的 iSCSI 选项的更多信息，请参阅 dracut.cmdline 手册页。

3. 使用以下命令卸载 iSCSI 磁盘：

```
$ iscsiadm --mode node --logoutall=all
```

此过程也可以使用 coreos-installer iso customize 或 coreos-installer pxe customize 子命令来执行。

15.3.14.6. 使用 iBFT 在 iSCSI 引导设备中安装 RHCOS

在一个完全无盘机器上，也可以通过 iBFT. iSCSI 多路径传递 iSCSI 目标和启动器值。

先决条件

1. 您在 RHCOS live 环境中。
2. 您有一个 iSCSI 目标，您要在其中安装 RHCOS。
3. 可选：有多路径 iSCSI 目标。

流程

1. 运行以下命令，从 live 环境中挂载 iSCSI 目标：

```
$ iscsiadm \
    --mode discovery \
```

```
--type sendtargets
--portal <IP_address> \ 1
--login
```

1 目标门户的 IP 地址。

2. 可选：使用以下命令启用多路径并启动守护进程：

```
$ mpathconf --enable && systemctl start multipathd.service
```

3. 运行以下命令并使用必要的内核参数将 RHCOS 安装到 iSCSI 目标上，例如：

```
$ coreos-installer install \
 /dev/mapper/mpatha \ 1
 --append-karg rd.iscsi.firmware=1 \ 2
 --append-karg rd.multipath=default \ 3
 --console ttyS0 \
 --ignition-file <path_to_file>
```

1 单一多路径设备的路径。如果连接了多个多路径设备，或者需要明确指定，您可以使用 /dev/disk/by-path 中可用的 World Wide Name (WWN) 符号链接。

2 iSCSI 参数从 BIOS 固件读取。

3 可选：如果您要启用多路径，请包含此参数。

有关 dracut 支持的 iSCSI 选项的更多信息，请参阅 dracut.cmdline 手册页。

4. 卸载 iSCSI 磁盘：

```
$ iscsiadm --mode node --logout=all
```

此过程也可以使用 coreos-installer iso customize 或 coreos-installer pxe customize 子命令来执行。

15.3.15. 等待 bootstrap 过程完成

OpenShift Container Platform bootstrap 过程在集群节点首次引导到安装到磁盘的持久 RHCOS 环境后开始。通过 Ignition 配置文件提供的配置信息用于初始化 bootstrap 过程并在机器上安装 OpenShift Container Platform。您必须等待 bootstrap 过程完成。

先决条件

- 已为集群创建 Ignition 配置文件。
- 您已配置了适当的网络、DNS 和负载平衡基础架构。
- 已获得安装程序，并为集群生成 Ignition 配置文件。
- 已在集群机器上安装 RHCOS，并提供 OpenShift Container Platform 安装程序生成的 Ignition 配置文件。
- 您的机器可以直接访问互联网，或者有 HTTP 或 HTTPS 代理可用。

流程

1. 监控 bootstrap 过程：

```
$ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \ 1
    --log-level=info 2
```

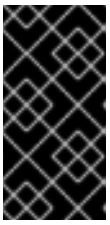
1. 对于 <installation_directory>，请指定安装文件保存到的目录的路径。
2. 要查看不同的安装详情，请指定 warn、debug 或 error，而不是 info。

输出示例

```
INFO Waiting up to 30m0s for the Kubernetes API at
https://api.test.example.com:6443...
INFO API v1.29.4 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
```

当 Kubernetes API 服务器提示已在 control plane 机器上引导它时，该命令会成功。

2. bootstrap 过程完成后，从负载均衡器中删除 bootstrap 机器。



重要

此时您必须从负载均衡器中删除 bootstrap 机器。您还可以删除或重新格式化 bootstrap 机器本身。

其他资源

- 如需有关 监控安装日志的更多信息，请参阅监控安装进度，并在出现安装问题时检索诊断数据。

15.3.16. 使用 CLI 登录集群

您可以通过导出集群 kubeconfig 文件，以默认系统用户身份登录集群。kubeconfig 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。
- 已安装 oc CLI。

流程

1. 导出 kubeadmin 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 <installation_directory>，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 oc 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

15.3.17. 批准机器的证书签名请求

当您将机器添加到集群时，会为您添加的每台机器生成两个待处理证书签名请求(CSR)。您必须确认这些 CSR 已获得批准，或根据需要自行批准。必须首先批准客户端请求，然后批准服务器请求。

先决条件

- 您已将机器添加到集群中。

流程

1. 确认集群可以识别这些机器：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0  Ready    master  63m  v1.29.4
master-1  Ready    master  63m  v1.29.4
master-2  Ready    master  64m  v1.29.4
```

输出中列出了您创建的所有机器。

注意

在有些 CSR 被批准前，前面的输出可能不包括计算节点（也称为 worker 节点）。

2. 检查待处理的 CSR，并确保添加到集群中的每台机器都有 Pending 或 Approved 状态的客户端请求：

```
$ oc get csr
```

输出示例

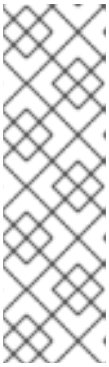
```
NAME        AGE    REQUESTOR                                                   CONDITION
csr-8b2br   15m    system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
csr-8vnps   15m    system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
...
```

在本例中，两台机器加入集群。您可能会在列表中看到更多已批准的 CSR。

3. 如果 CSR 没有获得批准，在您添加的机器的所有待处理 CSR 都处于 Pending 状态后，请批准集群机器的 CSR：

注意

由于 CSR 会自动轮转，因此请在将机器添加到集群后一小时内批准您的 CSR。如果没有在一小时内批准它们，证书将会轮转，每个节点会存在多个证书。您必须批准所有这些证书。批准客户端 CSR 后，Kubelet 为服务证书创建一个二级 CSR，这需要手动批准。然后，如果 Kubelet 请求具有相同参数的新证书，则后续提供证书续订请求由 machine-approver 自动批准。



注意

对于在未启用机器 API 的平台上运行的集群，如裸机和其他用户置备的基础架构，您必须实施一种方法来自动批准 kubelet 提供证书请求(CSR)。如果没有批准请求，则 oc exec、ocrsh 和 oc logs 命令将无法成功，因为 API 服务器连接到 kubelet 时需要服务证书。与 Kubelet 端点联系的任何操作都需要此证书批准。该方法必须监视新的 CSR，确认 CSR 由 system: node 或 system:admin 组中的 node-bootstrapper 服务帐户提交，并确认节点的身份。

- 要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 <csr_name> 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}
{{"\n"}}{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```

注意

在有些 CSR 被批准前，一些 Operator 可能无法使用。

4. 现在，您的客户端请求已被批准，您必须查看添加到集群中的每台机器的服务器请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE     REQUESTOR                                                   CONDITION
csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal        Pending
csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal        Pending
...
```

5. 如果剩余的 CSR 没有被批准，且处于 Pending 状态，请批准集群机器的 CSR：

- 要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 <csr_name> 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

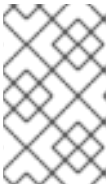
```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}
{{"\n"}}{{end}}{{end}}' | xargs oc adm certificate approve
```

6. 批准所有客户端和服务器 CSR 后，机器将 处于 Ready 状态。运行以下命令验证：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0  Ready    master  73m  v1.29.4
master-1  Ready    master  73m  v1.29.4
master-2  Ready    master  74m  v1.29.4
worker-0  Ready    worker  11m  v1.29.4
worker-1  Ready    worker  11m  v1.29.4
```



注意

批准服务器 CSR 后可能需要几分钟时间让机器过渡到 Ready 状态。

其他信息

- 如需有关 CSR 的更多信息，请参阅 证书签名请求。

15.3.18. 初始 Operator 配置

在 control plane 初始化后，您必须立即配置一些 Operator，以便它们都可用。

先决条件

- 您的 control plane 已初始化。

流程

1. 观察集群组件上线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION   AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.16.0    True        False         False      19m
baremetal                                  4.16.0    True        False         False      37m
cloud-credential                           4.16.0    True        False         False      40m
cluster-autoscaler                         4.16.0    True        False         False      37m
config-operator                            4.16.0    True        False         False      38m
console                                    4.16.0    True        False         False      26m
csi-snapshot-controller                    4.16.0    True        False         False      37m
dns                                        4.16.0    True        False         False      37m
etcd                                       4.16.0    True        False         False      36m
image-registry                             4.16.0    True        False         False      31m
ingress                                    4.16.0    True        False         False      30m
```

```
insights                                   4.16.0    True        False         False      31m
kube-apiserver                             4.16.0    True        False         False      26m
kube-controller-manager                    4.16.0    True        False         False      36m
kube-scheduler                             4.16.0    True        False         False      36m
kube-storage-version-migrator              4.16.0    True        False         False      37m
machine-api                                4.16.0    True        False         False      29m
machine-approver                           4.16.0    True        False         False      37m
machine-config                             4.16.0    True        False         False      36m
marketplace                                4.16.0    True        False         False      37m
monitoring                                 4.16.0    True        False         False      29m
network                                    4.16.0    True        False         False      38m
node-tuning                                4.16.0    True        False         False      37m
openshift-apiserver                        4.16.0    True        False         False      32m
openshift-controller-manager               4.16.0    True        False         False      30m
openshift-samples                          4.16.0    True        False         False      32m
operator-lifecycle-manager                 4.16.0    True        False         False      37m
operator-lifecycle-manager-catalog         4.16.0    True        False         False      37m
operator-lifecycle-manager-packageserver   4.16.0    True        False         False      32m
service-ca                                 4.16.0    True        False         False      38m
storage                                    4.16.0    True        False         False      37m
```

2. 配置不可用的 Operator。

其他资源

- 如需了解 在 OpenShift Container Platform 安装失败时收集数据的详细信息，请参阅从失败安装收集日志。
- 如需了解在集群中检查 Operator pod 健康状况的步骤，并收集 Operator 日志以进行诊断，请参阅故障排除 Operator 问题。

15.3.18.1. 安装过程中删除的镜像 registry

在不提供可共享对象存储的平台上，OpenShift Image Registry Operator bootstraps 本身为 Removed。这允许 openshift-installer 在这些平台类型上完成安装。

安装后，您必须编辑 Image Registry Operator 配置，将 managementState 从 Removed 切换到 Managed。完成此操作后，您必须配置存储。

15.3.18.2. 镜像 registry 存储配置

对于不提供默认存储的平台，Image Registry Operator 最初不可用。安装后，您必须将 registry 配置为使用存储，以便 Registry Operator 可用。

显示配置生产集群所需的持久性卷的说明。如果适用，显示有关将空目录配置为存储位置的说明，这仅适用于非生产集群。

提供了在升级过程中使用 Recreate rollout 策略来允许镜像 registry 使用块存储类型的说明。

15.3.18.3. 为裸机配置块 registry 存储

要允许镜像 registry 在作为集群管理员升级过程中使用块存储类型，您可以使用 Recreate rollout 策略。

重要

支持块存储卷或块持久性卷，但不建议在生产环境中使用镜像 registry。在块存储上配置 registry 的安装不具有高可用性，因为 registry 无法具有多个副本。

如果您选择将块存储卷与镜像 registry 搭配使用，则必须使用文件系统持久性卷声明(PVC)。

流程

1. 输入以下命令将镜像 registry 存储设置为块存储类型，对 registry 进行补丁，使其使用 Recreate rollout 策略，并只使用一个副本运行：

```
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec":
{"rolloutStrategy":"Recreate","replicas":1}}'
```

2. 为块存储设备置备 PV，并为该卷创建 PVC。请求的块卷使用 ReadWriteOnce(RWO)访问模式。

   a. 创建包含以下内容的 pvc.yaml 文件以定义 VMware vSphere PersistentVolumeClaim 对象：

```
kind: PersistentVolumeClaim
apiVersion: v1
```

```
metadata:
  name: image-registry-storage 1
  namespace: openshift-image-registry 2
spec:
  accessModes:
  - ReadWriteOnce 3
  resources:
    requests:
      storage: 100Gi 4
```

1 代表 PersistentVolumeClaim 对象的唯一名称。

2 PersistentVolumeClaim 对象的命名空间，即 openshift-image-registry。

3 持久性卷声明的访问模式。使用 ReadWriteOnce 时，单个节点可以通过读写权限挂载该卷。

4 持久性卷声明的大小。

b. 输入以下命令从文件创建 PersistentVolumeClaim 对象：

```
$ oc create -f pvc.yaml -n openshift-image-registry
```

3. 输入以下命令编辑 registry 配置，使其引用正确的 PVC：

```
$ oc edit config.imageregistry.operator.openshift.io -o yaml
```

输出示例

```
storage:
  pvc:
    claim: 1
```

1 通过创建自定义 PVC，您可以将 claim 字段留空，以便默认自动创建 image-registry-storage PVC。

15.3.19. 在用户置备的基础架构上完成安装

完成 Operator 配置后，可以在您提供的基础架构上完成集群安装。

先决条件

- 您的 control plane 已初始化。
- 已完成初始 Operator 配置。

流程

1. 使用以下命令确认所有集群组件都在线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION   AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.16.0    True        False         False      19m
baremetal                                  4.16.0    True        False         False      37m
cloud-credential                           4.16.0    True        False         False      40m
cluster-autoscaler                         4.16.0    True        False         False      37m
config-operator                            4.16.0    True        False         False      38m
console                                    4.16.0    True        False         False      26m
csi-snapshot-controller                    4.16.0    True        False         False      37m
dns                                        4.16.0    True        False         False      37m
etcd                                       4.16.0    True        False         False      36m
image-registry                             4.16.0    True        False         False      31m
ingress                                    4.16.0    True        False         False      30m
insights                                   4.16.0    True        False         False      31m
kube-apiserver                             4.16.0    True        False         False      26m
```

```
kube-controller-manager                    4.16.0    True        False         False      36m
kube-scheduler                             4.16.0    True        False         False      36m
kube-storage-version-migrator              4.16.0    True        False         False      37m
machine-api                                4.16.0    True        False         False      29m
machine-approver                           4.16.0    True        False         False      37m
machine-config                             4.16.0    True        False         False      36m
marketplace                                4.16.0    True        False         False      37m
monitoring                                 4.16.0    True        False         False      29m
network                                    4.16.0    True        False         False      38m
node-tuning                                4.16.0    True        False         False      37m
openshift-apiserver                        4.16.0    True        False         False      32m
openshift-controller-manager               4.16.0    True        False         False      30m
openshift-samples                          4.16.0    True        False         False      32m
operator-lifecycle-manager                 4.16.0    True        False         False      37m
operator-lifecycle-manager-catalog         4.16.0    True        False         False      37m
operator-lifecycle-manager-packageserver   4.16.0    True        False         False      32m
service-ca                                 4.16.0    True        False         False      38m
storage                                    4.16.0    True        False         False      37m
```

另外，当所有集群都可用时，以下命令会通知您。它还检索并显示凭证：

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

1

对于 <installation_directory>，请指定安装文件保存到的目录的路径。

输出示例

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

Cluster Version Operator 完成从 Kubernetes API 服务器部署 OpenShift Container Platform 集群时，该命令会成功。

重要

- 安装程序生成的 Ignition 配置文件包含 24 小时后过期的证书，然后在该时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 node-bootstrapper 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。

- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

2. 确认 Kubernetes API 服务器正在与 pod 通信。

   a. 要查看所有 pod 的列表，请使用以下命令：

   ```
   $ oc get pods --all-namespaces
   ```

   输出示例

   ```
   NAMESPACE                       NAME                                        READY   STATUS
   RESTARTS   AGE
   openshift-apiserver-operator    openshift-apiserver-operator-85cb746d55-zqhs8
   1/1     Running   1          9m
   openshift-apiserver             apiserver-67b9g                             1/1     Running   0
   3m
   openshift-apiserver             apiserver-ljcmx                             1/1     Running   0
   1m
   openshift-apiserver             apiserver-z25h4                             1/1     Running   0
   2m
   openshift-authentication-operator authentication-operator-69d5d8bf84-vh2n8
   1/1     Running   0          5m
   ...
   ```

   b. 使用以下命令，查看上一命令的输出中所列 pod 的日志：

   ```
   $ oc logs <pod_name> -n <namespace> 1
   ```

1

指定 pod 名称和命名空间，如上一命令的输出中所示。

如果 pod 日志显示，Kubernetes API 服务器可以与集群机器通信。

3. 对于使用光纤通道协议(FCP)的安装，还需要额外的步骤才能启用多路径。不要在安装过程中启用多路径。

如需更多信息，请参阅 *安装后机器配置任务* 文档中的"使用 RHCOS 上使用内核参数启用多路径"。

15.3.20. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

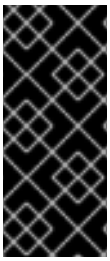
- 有关 Telemetry 服务的更多信息，请参阅关于 远程健康监控

15.3.21. 后续步骤

- 验证安装.
- 自定义集群。
- 如果需要，您可以选择 不使用远程健康报告。
- 设置 registry 并配置 registry 存储。

15.4. 在受限网络中安装用户置备的裸机集群

在 OpenShift Container Platform 4.16 中，您可以在受限网络中置备的裸机基础架构上安装集群。

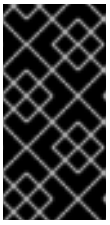


重要

虽然您可能能够按照以下步骤在虚拟化或云环境中部署集群，但您必须了解非裸机平台的其他注意事项。在尝试在此类环境中安装 OpenShift Container Platform 集群前，请参阅有关在未经测试的平台上部署 OpenShift Container Platform 的指南 中的信息。

15.4.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读有关 选择集群安装方法的文档，并为用户准备它。
- 您在镜像主机上创建 registry，并获取您的 OpenShift Container Platform 版本的 imageContentSources 数据。



重要

由于安装介质位于镜像主机上，因此您可以使用该计算机完成所有安装步骤。

- 已为集群置备了 持久性存储。要部署私有镜像 registry，您的存储必须提供 ReadWriteMany 访问模式。
- 如果您使用防火墙并计划使用 Telemetry 服务，则将防火墙配置为允许集群需要访问的站点。

注意

如果要配置代理，请务必查看此站点列表。

15.4.2. 关于在受限网络中安装

在 OpenShift Container Platform 4.16 中，可以执行不需要有效的互联网连接来获取软件组件的安装。受限网络安装可以使用安装程序置备的基础架构或用户置备的基础架构完成，具体取决于您要安装集群的云平台。

如果您选择在云平台中执行受限网络安装，您仍需要访问其云 API。有些云功能，比如 Amazon Web Service 的 Route 53 DNS 和 IAM 服务，需要访问互联网。根据您的网络，在裸机硬件、Nutanix 或 VMware vSphere 上安装可能需要较少的互联网访问。

要完成受限网络安装，您必须创建一个 registry，以镜像 OpenShift 镜像 registry 的内容并包含安装介质。您可以在镜像主机上创建此 registry，该主机可同时访问互联网和您的封闭网络，也可以使用满足您的限制条件的其他方法。

重要

由于用户置备安装配置的复杂性，在尝试使用用户置备的基础架构受限网络安装前，请考虑完成标准用户置备的基础架构安装。完成此测试安装后，您可以更轻松地隔离和排除在受限网络中安装过程中可能出现的任何问题。

15.4.2.1. 其他限制

受限网络中的集群有以下额外限制和限制：

- ClusterVersion 状态包含一个 Unable to retrieve available updates 错误。
- 默认情况下，您无法使用 Developer Catalog 的内容，因为您无法访问所需的镜像流标签。

15.4.3. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来获得用来安装集群的镜像。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。

- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

15.4.4. 具有用户置备基础架构的集群的要求

对于包含用户置备的基础架构的集群，您必须部署所有所需的机器。

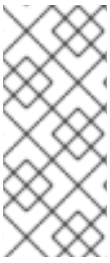
本节论述了在用户置备的基础架构上部署 OpenShift Container Platform 的要求。

15.4.4.1. 集群安装所需的机器

最小的 OpenShift Container Platform 集群需要以下主机：

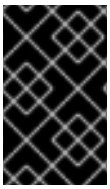
表 15.32. 最低所需的主机

主机	描述
一个临时 bootstrap 机器	集群需要 bootstrap 机器在三台 control plane 机器上部署 OpenShift Container Platform 集群。您可在安装集群后删除 bootstrap 机器。
三台 control plane 机器	control plane 机器运行组成 control plane 的 Kubernetes 和 OpenShift Container Platform 服务。
至少两台计算机器，也称为 worker 机器。	OpenShift Container Platform 用户请求的工作负载在计算机器上运行。



注意

作为例外，您可以在裸机集群中运行零台计算机器，它们仅由三台 control plane 机器组成。这为集群管理员和开发人员提供了更小、效率更高的集群，用于测试、开发和生产。不支持运行一台计算机器。



重要

要保持集群的高可用性，请将独立的物理主机用于这些集群机器。

bootstrap 和 control plane 机器必须使用 Red Hat Enterprise Linux CoreOS(RHCOS)作为操作系统。但是，计算机器可以在 Red Hat Enterprise Linux CoreOS(RHCOS)、Red Hat Enterprise Linux(RHEL) 8.6 和更高的版本。

请注意，RHCOS 基于 Red Hat Enterprise Linux(RHEL) 9.2，并继承其所有硬件认证和要求。查看红帽企业 Linux 技术功能和限制。

15.4.4.2. 集群安装的最低资源要求

每台集群机器都必须满足以下最低要求：

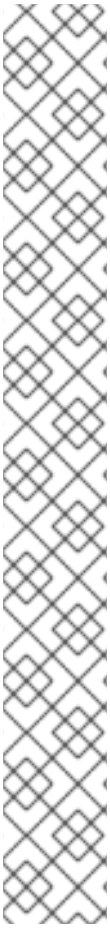
表 15.33. 最低资源要求

机器	操作系统	CPU [1]	RAM	Storage	每秒输入/输出 (IOPS) [2]
bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane（控制平面）	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS、RHEL 8.6 及更新版本 [3]	2	8 GB	100 GB	300

1. 当未启用并发多线程 (SMT) 或超线程时，一个 CPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比例：（每个内核数的线程）× sockets = CPU。

2. OpenShift Container Platform 和 Kubernetes 对磁盘性能非常敏感，建议使用更快的存储速度，特别是 control plane 节点上需要 10 ms p99 fsync 持续时间的 etcd。请注意，在许多云平台上，存储大小和 IOPS 可一起扩展，因此您可能需要过度分配存储卷来获取足够的性能。

3. 与所有用户置备的安装一样，如果您选择在集群中使用 RHEL 计算机器，则负责所有操作系统生命周期管理和维护，包括执行系统更新、应用补丁和完成所有其他必要的任务。RHEL 7 计算机器的使用已弃用，并已在 OpenShift Container Platform 4.10 及更新的版本中删除。



注意

从 OpenShift Container Platform 版本 4.13 开始，RHCOS 基于 RHEL 版本 9.2，它更新了微架构要求。以下列表包含每个架构需要的最小指令集架构 (ISA)：

- x86-64 体系结构需要 x86-64-v2 ISA
- ARM64 架构需要 ARMv8.0-A ISA
- IBM Power 架构需要 Power 9 ISA
- s390x 架构需要 z14 ISA

如需更多信息，请参阅 RHEL 架构。

如果平台的实例类型满足集群机器的最低要求，则 OpenShift Container Platform 支持使用它。

其他资源

- 优化存储

15.4.4.3. 证书签名请求管理

在使用您置备的基础架构时，集群只能有限地访问自动机器管理，因此您必须提供一种在安装后批准集群证书签名请求 (CSR) 的机制。kube-controller-manager 只能批准 kubelet 客户端 CSR。machine-approver 无法保证使用 kubelet 凭证请求的提供证书的有效性，因为它不能确认是正确的机器发出了该请求。您必须决定并实施一种方法，以验证 kubelet 提供证书请求的有效性并进行批准。

其他资源

- 有关在裸机环境中部署三节点集群的详情，请参阅配置三节点集群。
- 有关在安装后批准集群证书签名请求的更多信息，请参阅批准机器的证书签名请求。

15.4.4.4. 用户置备的基础架构对网络的要求

所有 Red Hat Enterprise Linux CoreOS(RHCOS)机器都需要在启动时在 initramfs 中配置联网，以获取它们的 Ignition 配置文件。

在初次启动过程中，机器需要 IP 地址配置，该配置通过 DHCP 服务器或静态设置，提供所需的引导选项。建立网络连接后，机器会从 HTTP 或 HTTPS 服务器下载 Ignition 配置文件。然后，Ignition 配置文件用于设置每台机器的确切状态。Machine Config Operator 在安装后完成对机器的更多更改，如应用新证书或密钥。

建议使用 DHCP 服务器对集群机器进行长期管理。确保 DHCP 服务器已配置为向集群机器提供持久的 IP 地址、DNS 服务器信息和主机名。

注意

如果用户置备的基础架构没有 DHCP 服务，您可以在 RHCOS 安装时向节点提供 IP 网络配置和 DNS 服务器地址。如果要从 ISO 镜像安装，这些参数可作为引导参数传递。如需有关静态 IP 置备和高级网络选项的更多信息，请参阅 *安装 RHCOS 并启动 OpenShift Container Platform bootstrap 过程* 部分。

Kubernetes API 服务器必须能够解析集群机器的节点名称。如果 API 服务器和 worker 节点位于不同的区域中，您可以配置默认 DNS 搜索区域，以允许 API 服务器解析节点名称。另一种支持的方法是始终通过节点对象和所有 DNS 请求中的完全限定域名引用主机。

15.4.4.4.1. 通过 DHCP 设置集群节点主机名

在 Red Hat Enterprise Linux CoreOS(RHCOS)机器上，主机名是通过 NetworkManager 设置的。默认情况下，机器通过 DHCP 获取其主机名。如果主机名不是由 DHCP 提供，请通过内核参数或者其它方法进行静态设置，请通过反向 DNS 查找获取。反向 DNS 查找在网络初始化后进行，可能需要一些时间来解决。其他系统服务可以在此之前启动，并将主机名检测为 localhost 或类似的内容。您可以使用 DHCP 为每个集群节点提供主机名来避免这种情况。

另外，通过 DHCP 设置主机名可以绕过实施 DNS split-horizon 的环境中的手动 DNS 记录名称配置错误。

15.4.4.4.2. 网络连接要求

您必须配置机器之间的网络连接，以允许 OpenShift Container Platform 集群组件进行通信。每台机器都必须能够解析集群中所有其他机器的主机名。

本节详细介绍了所需的端口。

表 15.34. 用于全机器到所有机器通信的端口

协议	port	描述
ICMP	N/A	网络可访问性测试
TCP	**1936**	指标
	9000-9999	主机级别的服务，包括端口 9**100-9101** 上的节点导出器，以及端口 **9099** 上的 Cluster Version Operator。
	10250-10259	Kubernetes 保留的默认端口
UDP	**4789**	VXLAN
	6081	Geneve
	9000-9999	主机级别的服务，包括端口 **9100**到**9101** 上的节点导出器。
	500	IPsec IKE 数据包
	4500	IPsec NAT-T 数据包
	123	UDP 端口 **123** 上的网络时间协议 (NTP) 如果配置了外部 NTP 时间服务器，需要打开 UDP 端口 **123**。
TCP/UDP	**30000-32767**	Kubernetes 节点端口
ESP	N/A	IPsec Encapsulating Security Payload(ESP)

表 15.35. 用于所有机器控制平面通信的端口

协议	port	描述
TCP	**6443**	Kubernetes API

表 15.36. control plane 机器用于 control plane 机器通信的端口

协议	port	描述
TCP	**2379-2380**	etcd 服务器和对等端口

用户置备的基础架构的 NTP 配置

OpenShift Container Platform 集群被配置为默认使用公共网络时间协议(NTP)服务器。如果要使用本地企业 NTP 服务器，或者集群部署在断开连接的网络中，您可以将集群配置为使用特定的时间服务器。如需更多信息，*请参阅配置 chrony 时间服务* 的文档。

如果 DHCP 服务器提供 NTP 服务器信息，Red Hat Enterprise Linux CoreOS(RHCOS)机器上的 chrony 时间服务会读取信息，并可以把时钟与 NTP 服务器同步。

其他资源

- 配置 chrony 时间服务

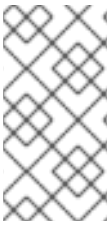
15.4.4.5. 用户置备的 DNS 要求

在 OpenShift Container Platform 部署中，以下组件需要 DNS 名称解析：

- The Kubernetes API
- OpenShift Container Platform 应用程序通配符
- bootstrap、control plane 和计算机器

Kubernetes API、bootstrap 机器、control plane 机器和计算机器也需要反向 DNS 解析。

DNS A/AAAA 或 CNAME 记录用于名称解析，PTR 记录用于反向名称解析。反向记录很重要，因为 Red Hat Enterprise Linux CoreOS(RHCOS)使用反向记录为所有节点设置主机名，除非 DHCP 提供主机名。另外，反向记录用于生成 OpenShift Container Platform 需要操作的证书签名请求(CSR)。



注意

建议使用 DHCP 服务器为每个群集节点提供主机名。如需更多信息，*请参阅用户置备的基础架构部分的 DHCP 建议*。

用户置备的 OpenShift Container Platform 集群需要以下 DNS 记录，这些记录必须在安装前就位。

在每个记录中，<cluster_name> 是集群名称，<base_domain> 是您在 install-config.yaml 文件中指定的基域。完整的 DNS 记录采用以下形式： <component>.<cluster_name>.<base_domain>.。

表 15.37. 所需的 DNS 记录

组件	记录	描述
Kubernetes API	**api.<cluster_name>.<base_domain>.**	DNS A/AAAA 或 CNAME 记录，以及用于标识 API 负载均衡器的 DNS PTR 记录。这些记录必须由集群外的客户端和集群中的所有节点解析。
	api-int.<cluster_name>.<base_domain>.	DNS A/AAAA 或 CNAME 记录，以及用于内部标识 API 负载均衡器的 DNS PTR 记录。这些记录必须可以从集群中的所有节点解析。 **重要** API 服务器必须能够根据 Kubernetes 中记录的主机名解析 worker 节点。如果 API 服务器无法解析节点名称，则代理的 API 调用会失败，且您无法从 pod 检索日志。
Routes	***.apps.<cluster_name>.<base_domain>.**	通配符 DNS A/AAAA 或 CNAME 记录，指向应用程序入口负载均衡器。应用程序入口负载均衡器以运行 Ingress Controller Pod 的机器为目标。默认情况下，Ingress Controller Pod 在计算机器上运行。这些记录必须由集群外的客户端和集群中的所有节点解析。 例如，console **-openshift-console.apps.<cluster_name>.<base_domain>** 用作到 OpenShift Container Platform 控制台的通配符路由。
bootstrap 机器	**bootstrap.<cluster_name>.<base_domain>.**	DNS A/AAAA 或 CNAME 记录，以及用于标识 bootstrap 机器的 DNS PTR 记录。这些记录必须由集群中的节点解析。
control plane 机器	**<control_plane><n>.<cluster_name>.<base_domain>.**	DNS A/AAAA 或 CNAME 记录，以识别 control plane 节点的每台机器。这些记录必须由集群中的节点解析。
计算机器	**<compute><n>.<cluster_name>.<base_domain>.**	DNS A/AAAA 或 CNAME 记录，用于识别 worker 节点的每台机器。这些记录必须由集群中的节点解析。

注意

在 OpenShift Container Platform 4.4 及更新的版本中，您不需要在 DNS 配置中指定 etcd 主机和 SRV 记录。

提示

您可以使用 dig 命令验证名称和反向名称解析。如需了解详细的 *验证步骤，请参阅为用户置备的基础架构验证 DNS 解析* 一节。

15.4.4.5.1. 用户置备的集群的 DNS 配置示例

本节提供 A 和 PTR 记录配置示例，它们满足了在用户置备的基础架构上部署 OpenShift Container Platform 的 DNS 要求。样本不是为选择一个 DNS 解决方案提供建议。

在这个示例中，集群名称为 ocp4，基域是 example.com。

用户置备的集群的 DNS A 记录配置示例

以下示例是 BIND 区域文件，其中显示了用户置备的集群中名称解析的 A 记录示例。

例 15.7. DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
  2019070700 ; serial
  3H  ; refresh (3 hours)
  30M  ; retry (30 minutes)
  2W  ; expiry (2 weeks)
  1W )  ; minimum (1 week)
 IN NS ns1.example.com.
 IN MX 10 smtp.example.com.
;
;
ns1.example.com.  IN A 192.168.1.5
smtp.example.com.  IN A 192.168.1.5
;
helper.example.com.  IN A 192.168.1.5
helper.ocp4.example.com. IN A 192.168.1.5
;
api.ocp4.example.com.  IN A 192.168.1.5 1
api-int.ocp4.example.com. IN A 192.168.1.5 2
;
*.apps.ocp4.example.com. IN A 192.168.1.5 3
;
bootstrap.ocp4.example.com. IN A 192.168.1.96 4
;
control-plane0.ocp4.example.com. IN A 192.168.1.97 5
control-plane1.ocp4.example.com. IN A 192.168.1.98 6
control-plane2.ocp4.example.com. IN A 192.168.1.99 7
;
compute0.ocp4.example.com. IN A 192.168.1.11 8
```

```
compute1.ocp4.example.com. IN A 192.168.1.7 9
;
;EOF
```

1

为 Kubernetes API 提供名称解析。记录引用 API 负载均衡器的 IP 地址。

2

为 Kubernetes API 提供名称解析。记录引用 API 负载均衡器的 IP 地址，用于内部集群通信。

3

为通配符路由提供名称解析。记录引用应用程序入口负载均衡器的 IP 地址。应用程序入口负载均衡器以运行 Ingress Controller Pod 的机器为目标。默认情况下，Ingress Controller Pod 在计算机器上运行。

注意

在这个示例中，将相同的负载均衡器用于 Kubernetes API 和应用入口流量。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。

4

为 bootstrap 机器提供名称解析。

5 6 7

为 control plane 机器提供名称解析。

8 9

为计算机器提供名称解析。

用户置备的集群的 DNS PTR 记录配置示例

以下示例 BIND 区域文件显示了用户置备的集群中反向名称解析的 PTR 记录示例。

例 15.8. 反向记录的 DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
  2019070700 ; serial
  3H  ; refresh (3 hours)
  30M  ; retry (30 minutes)
  2W  ; expiry (2 weeks)
  1W )  ; minimum (1 week)
 IN NS ns1.example.com.
;
5.1.168.192.in-addr.arpa. IN PTR api.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. IN PTR api-int.ocp4.example.com. 2
;
96.1.168.192.in-addr.arpa. IN PTR bootstrap.ocp4.example.com. 3
;
97.1.168.192.in-addr.arpa. IN PTR control-plane0.ocp4.example.com. 4
98.1.168.192.in-addr.arpa. IN PTR control-plane1.ocp4.example.com. 5
99.1.168.192.in-addr.arpa. IN PTR control-plane2.ocp4.example.com. 6
;
11.1.168.192.in-addr.arpa. IN PTR compute0.ocp4.example.com. 7
7.1.168.192.in-addr.arpa. IN PTR compute1.ocp4.example.com. 8
;
;EOF
```

1

为 Kubernetes API 提供反向 DNS 解析。PTR 记录引用 API 负载均衡器的记录名称。

2

为 Kubernetes API 提供反向 DNS 解析。PTR 记录引用 API 负载均衡器的记录名称，用于内部集群通信。

3

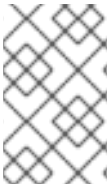
为 bootstrap 机器提供反向 DNS 解析。

4 5 6

为 control plane 机器提供反向 DNS 解析。

7 8

为计算机器提供反向 DNS 解析。



注意

OpenShift Container Platform 应用程序通配符不需要 PTR 记录。

其他资源

- 验证用户置备的基础架构的 DNS 解析

15.4.4.6. 用户置备的基础架构的负载均衡要求

在安装 OpenShift Container Platform 前，您必须置备 API 和应用程序入口负载均衡基础架构。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。

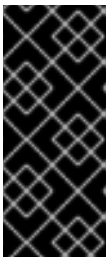
注意

如果要使用 Red Hat Enterprise Linux (RHEL) 实例部署 API 和应用程序入口负载均衡器，您必须单独购买 RHEL 订阅。

负载平衡基础架构必须满足以下要求：

1. **API 负载均衡器 ：提供一个通用端点，供用户（包括人工和机器）与平台交互和配置。配置以下条件：**

   - **仅第 4 层负载均衡.这可被称为 Raw TCP 或 SSL Passthrough 模式。**
   - **无状态负载平衡算法。这些选项根据负载均衡器的实施而有所不同。**



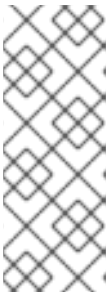
重要

不要为 API 负载均衡器配置会话持久性。为 Kubernetes API 服务器配置会话持久性可能会导致出现过量 OpenShift Container Platform 集群应用程序流量，以及过量的在集群中运行的 Kubernetes API。

在负载均衡器的前端和后端配置以下端口：

表 15.38. API 负载均衡器

port	后端机器（池成员）	internal	外部	描述
6443	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。您必须为 API 服务器健康检查探测配置 **/readyz** 端点。	X	X	Kubernetes API 服务器
22623	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。	X		机器配置服务器



注意

负载均衡器必须配置为，从 API 服务器关闭 **/readyz** 端点到从池中移除 API 服务器实例时最多需要 30 秒。在 **/readyz** 返回错误或健康后的时间范围内，端点必须被删除或添加。每 5 秒或 10 秒探测一次，有两个成功请求处于健康状态，三个成为不健康的请求是经过良好测试的值。

2. 应用程序入口负载均衡器 ：为应用程序流量从集群外部流提供入口点。OpenShift Container Platform 集群需要正确配置入口路由器。

配置以下条件：

- 仅第 4 层负载均衡.这可被称为 **Raw TCP 或 SSL Passthrough 模式。**
- 建议根据可用选项以及平台上托管的应用程序类型，使用基于连接的或基于会话的持久性。

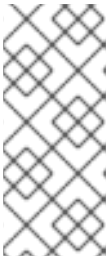
提示

如果应用程序入口负载均衡器可以看到客户端的真实 IP 地址，启用基于 IP 的会话持久性可以提高使用端到端 TLS 加密的应用程序的性能。

在负载均衡器的前端和后端配置以下端口：

表 15.39. 应用程序入口负载均衡器

port	后端机器（池成员）	internal	外部	描述
443	默认情况下，运行 Ingress Controller Pod、计算或 worker 的机器。	X	X	HTTPS 流量
80	默认情况下，运行 Ingress Controller Pod、计算或 worker 的机器。	X	X	HTTP 流量



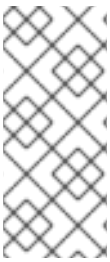
注意

如果要部署一个带有零计算节点的三节点集群，Ingress Controller Pod 在 control plane 节点上运行。在三节点集群部署中，您必须配置应用程序入口负载均衡器，将 HTTP 和 HTTPS 流量路由到 control plane 节点。

15.4.4.6.1. 用户置备的集群的负载均衡器配置示例

本节提供了一个满足用户置备集群的负载均衡要求的 API 和应用程序入口负载均衡器配置示例。示例是 HAProxy 负载均衡器的 /etc/haproxy/haproxy.cfg 配置。这个示例不是为选择一个负载平衡解决方案提供建议。

在这个示例中，将相同的负载均衡器用于 Kubernetes API 和应用入口流量。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。



注意

如果您使用 HAProxy 作为负载均衡器，并且 SELinux 设置为 enforcing，您必须通过运行 setsebool -P haproxy_connect_any=1 来确保 HAProxy 服务可以绑定到配置的 TCP 端口。

例 15.9. API 和应用程序入口负载均衡器配置示例

```
global
  log         127.0.0.1 local2
  pidfile     /var/run/haproxy.pid
  maxconn     4000
  daemon
defaults
  mode                    http
  log                     global
```

```
  option                  dontlognull
  option http-server-close
  option                  redispatch
  retries                 3
  timeout http-request    10s
  timeout queue           1m
  timeout connect         10s
  timeout client          1m
  timeout server          1m
  timeout http-keep-alive 10s
  timeout check           10s
  maxconn                 3000
listen api-server-6443 (1)
  bind *:6443
  mode tcp
  option  httpchk GET /readyz HTTP/1.0
  option  log-health-checks
  balance roundrobin
  server bootstrap bootstrap.ocp4.example.com:6443 verify none check check-ssl inter 10s
fall 2 rise 3 backup (2)
  server master0 master0.ocp4.example.com:6443 weight 1 verify none check check-ssl
inter 10s fall 2 rise 3
  server master1 master1.ocp4.example.com:6443 weight 1 verify none check check-ssl
inter 10s fall 2 rise 3
  server master2 master2.ocp4.example.com:6443 weight 1 verify none check check-ssl
inter 10s fall 2 rise 3
listen machine-config-server-22623 (3)
  bind *:22623
  mode tcp
  server bootstrap bootstrap.ocp4.example.com:22623 check inter 1s backup (4)
  server master0 master0.ocp4.example.com:22623 check inter 1s
  server master1 master1.ocp4.example.com:22623 check inter 1s
  server master2 master2.ocp4.example.com:22623 check inter 1s
listen ingress-router-443 (5)
  bind *:443
  mode tcp
  balance source
  server compute0 compute0.ocp4.example.com:443 check inter 1s
  server compute1 compute1.ocp4.example.com:443 check inter 1s
listen ingress-router-80 (6)
  bind *:80
  mode tcp
  balance source
  server compute0 compute0.ocp4.example.com:80 check inter 1s
  server compute1 compute1.ocp4.example.com:80 check inter 1s
```

(1) 端口 6443 处理 Kubernetes API 流量并指向 control plane 机器。

(2)(4) bootstrap 条目必须在 OpenShift Container Platform 集群安装前就位，且必须在 bootstrap 过程完成后删除它们。

3

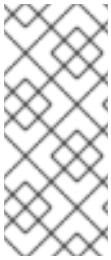
端口 22623 处理机器配置服务器流量并指向 control plane 机器。

5

端口 443 处理 HTTPS 流量，并指向运行 Ingress Controller pod 的机器。默认情况下，Ingress Controller Pod 在计算机器上运行。

6

端口 80 处理 HTTP 流量，并指向运行 Ingress Controller pod 的机器。默认情况下，Ingress Controller Pod 在计算机器上运行。



注意

如果要部署一个带有零计算节点的三节点集群，Ingress Controller Pod 在 control plane 节点上运行。在三节点集群部署中，您必须配置应用程序入口负载均衡器，将 HTTP 和 HTTPS 流量路由到 control plane 节点。

提示

如果您使用 HAProxy 作为负载均衡器，您可以通过在 HAProxy 节点上运行 netstat -nltupe 来检查 haproxy 进程是否在侦听端口 6443、22623、443 和 80。

15.4.5. 准备用户置备的基础架构

在用户置备的基础架构上安装 OpenShift Container Platform 之前，您必须准备底层基础架构。

本节详细介绍了设置集群基础架构以准备 OpenShift Container Platform 安装所需的高级别步骤。这包括为您的集群节点配置 IP 网络和网络连接，通过防火墙启用所需的端口，以及设置所需的 DNS 和负载均衡基础架构。

准备后，集群基础架构必须满足 *带有用户置备的基础架构部分的集群要求*。

先决条件

- 您已参阅 OpenShift Container Platform 4.x Tested Integrations 页面。
- 您已查看了 *具有用户置备基础架构的集群要求部分中详述的基础架构* 要求。

流程

1. 如果您使用 DHCP 向集群节点提供 IP 网络配置，请配置 DHCP 服务。

   a. 将节点的持久 IP 地址添加到您的 DHCP 服务器配置。在您的配置中，将相关网络接口的 MAC 地址与每个节点的预期 IP 地址匹配。

   b. 当您使用 DHCP 为集群机器配置 IP 寻址时，机器还通过 DHCP 获取 DNS 服务器信息。定义集群节点通过 DHCP 服务器配置使用的持久性 DNS 服务器地址。

   注意

   如果没有使用 DHCP 服务，则必须在 RHCOS 安装时为节点提供 IP 网络配置和 DNS 服务器地址。如果要从 ISO 镜像安装，这些参数可作为引导参数传递。如需有关静态 IP 置备和高级网络选项的更多信息，请参阅 *安装 RHCOS 并启动 OpenShift Container Platform bootstrap 过程* 部分。

   c. 在 DHCP 服务器配置中定义集群节点的主机名。有关 *主机名注意事项的详情，请参阅通过 DHCP 设置集群节点* 主机名部分。

   注意

   如果没有使用 DHCP 服务，集群节点可以通过反向 DNS 查找来获取其主机名。

2. 确保您的网络基础架构提供集群组件之间所需的网络连接。有关 *要求的详情，请参阅用户置备的基础架构* 的网络要求部分。

3. 将防火墙配置为启用 OpenShift Container Platform 集群组件进行通信所需的端口。*如需有关所需端口的详细信息，请参阅用户置备的基础架构* 部分的网络要求。

重要

默认情况下，OpenShift Container Platform 集群可以访问端口 1936，因为每个 control plane 节点都需要访问此端口。

避免使用 Ingress 负载均衡器公开此端口，因为这样做可能会导致公开敏感信息，如统计信息和指标（与 Ingress Controller 相关的统计信息和指标）。

4. 为集群设置所需的 DNS 基础架构。

   a. 为 Kubernetes API、应用程序通配符、bootstrap 机器、control plane 机器和计算机器配置 DNS 名称解析。

   b. 为 Kubernetes API、bootstrap 机器、control plane 机器和计算机器配置反向 DNS 解析。

   如需有关 *OpenShift Container Platform DNS 要求的更多信息，请参阅用户置备* DNS 要求部分。

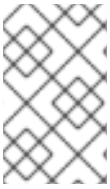
5. 验证您的 DNS 配置。

   a. 从安装节点，针对 Kubernetes API 的记录名称、通配符路由和集群节点运行 DNS 查找。验证响应中的 IP 地址是否与正确的组件对应。

   b. 从安装节点，针对负载均衡器和集群节点的 IP 地址运行反向 DNS 查找。验证响应中的记录名称是否与正确的组件对应。

   有关详细的 *DNS 验证步骤，请参阅用户置备的基础架构*验证 DNS 解析部分。

6. 置备所需的 API 和应用程序入口负载平衡基础架构。有关 *要求的更多信息，请参阅用户置备的基础架构的负载平衡* 要求部分。



注意

某些负载平衡解决方案要求在初始化负载平衡之前，对群集节点进行 DNS 名称解析。

其他资源

- 具有用户置备基础架构的集群的要求
- 安装 RHCOS 并启动 OpenShift Container Platform bootstrap 过程
- 通过 DHCP 设置集群节点主机名
- 高级 RHCOS 安装配置
- 用户置备的基础架构对网络的要求
- 用户置备的 DNS 要求
- 验证用户置备的基础架构的 DNS 解析
- 用户置备的基础架构的负载均衡要求

15.4.6. 验证用户置备的基础架构的 DNS 解析

您可以在在用户置备的基础架构上安装 OpenShift Container Platform 前验证 DNS 配置。

重要

本节中详述的验证步骤必须在安装集群前成功。

先决条件

- 已为您的用户置备的基础架构配置了所需的 DNS 记录。

流程

1. 从安装节点，针对 Kubernetes API 的记录名称、通配符路由和集群节点运行 DNS 查找。验证响应中包含的 IP 地址是否与正确的组件对应。

   a. 对 Kubernetes API 记录名称执行查询。检查结果是否指向 API 负载均衡器的 IP 地址：

   ```
   $ dig +noall +answer @<nameserver_ip> api.<cluster_name>.<base_domain> 1
   ```

   1 将 <nameserver_ip> 替换为 nameserver 的 IP 地址，<cluster_name> 替换为您的集群名称，<base_domain> 替换为您的基本域名。

   输出示例

   ```
   api.ocp4.example.com.  604800 IN A 192.168.1.5
   ```

   b. 对 Kubernetes 内部 API 记录名称执行查询。检查结果是否指向 API 负载均衡器的 IP 地址：

   ```
   $ dig +noall +answer @<nameserver_ip> api-int.<cluster_name>.<base_domain>
   ```

   输出示例

   ```
   api-int.ocp4.example.com.  604800 IN A 192.168.1.5
   ```

   c.

测试 *.apps.<cluster_name>.<base_domain> DNS 通配符查找示例。所有应用程序通配符查询都必须解析为应用程序入口负载均衡器的 IP 地址：

```
$ dig +noall +answer @<nameserver_ip> random.apps.<cluster_name>.
<base_domain>
```

输出示例

```
random.apps.ocp4.example.com.  604800 IN A 192.168.1.5
```

注意

在示例中，将相同的负载均衡器用于 Kubernetes API 和应用程序入口流量。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。

您可以使用另一个通配符值替换 random。例如，您可以查询到 OpenShift Container Platform 控制台的路由：

```
$ dig +noall +answer @<nameserver_ip> console-openshift-console.apps.
<cluster_name>.<base_domain>
```

输出示例

```
console-openshift-console.apps.ocp4.example.com. 604800 IN A 192.168.1.5
```

d.

针对 bootstrap DNS 记录名称运行查询。检查结果是否指向 bootstrap 节点的 IP 地址：

```
$ dig +noall +answer @<nameserver_ip> bootstrap.<cluster_name>.
<base_domain>
```

输出示例

```
bootstrap.ocp4.example.com.  604800 IN A 192.168.1.96
```

e. 使用此方法对 control plane 和计算节点的 DNS 记录名称执行查找。检查结果是否与每个节点的 IP 地址对应。

2. 从安装节点，针对负载均衡器和集群节点的 IP 地址运行反向 DNS 查找。验证响应中包含的记录名称是否与正确的组件对应。

a. 对 API 负载均衡器的 IP 地址执行反向查找。检查响应是否包含 Kubernetes API 和 Kubernetes 内部 API 的记录名称：

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.5
```

输出示例

```
5.1.168.192.in-addr.arpa. 604800 IN PTR api-int.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. 604800 IN PTR api.ocp4.example.com. 2
```

1 为 Kubernetes 内部 API 提供记录名称。

2 为 Kubernetes API 提供记录名称。

注意

OpenShift Container Platform 应用程序通配符不需要 PTR 记录。针对应用程序入口负载均衡器的 IP 地址解析反向 DNS 解析不需要验证步骤。

b. 对 bootstrap 节点的 IP 地址执行反向查找。检查结果是否指向 bootstrap 节点的 DNS 记录名称：

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.96
```

输出示例

```
96.1.168.192.in-addr.arpa. 604800 IN PTR bootstrap.ocp4.example.com.
```

c. 使用此方法对 control plane 和计算节点的 IP 地址执行反向查找。检查结果是否与每个节点的 DNS 记录名称对应。

其他资源

- 用户置备的 DNS 要求
- 用户置备的基础架构的负载均衡要求

15.4.7. 为集群节点 SSH 访问生成密钥对

在 OpenShift Container Platform 安装过程中，您可以为安装程序提供 SSH 公钥。密钥通过它们的 Ignition 配置文件传递给 Red Hat Enterprise Linux CoreOS(RHCOS)节点，用于验证对节点的 SSH 访问。密钥添加到每个节点上 core 用户的 ~/.ssh/authorized_keys 列表中，这将启用免密码身份验证。

将密钥传递给节点后，您可以使用密钥对作为用户 核心 通过 SSH 连接到 RHCOS 节点。若要通过 SSH 访问节点，必须由 SSH 为您的本地用户管理私钥身份。

如果要通过 SSH 连接到集群节点来执行安装调试或灾难恢复，则必须在安装过程中提供 SSH 公钥。./openshift-install gather 命令还需要在集群节点上设置 SSH 公钥。

重要

不要在生产环境中跳过这个过程，在生产环境中需要灾难恢复和调试。

注意

您必须使用本地密钥，而不是使用特定平台方法配置 的密钥，如 AWS 密钥对。

流程

1. 如果您在本地计算机上没有可用于在集群节点上进行身份验证的现有 SSH 密钥对，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 ~/.ssh/id_ed25519。如果您已有密钥对，请确保您的公钥位于 ~/.ssh 目录中。

注意

如果您计划在 x86_64、ppc64le 和 s390x 架构上安装使用 RHEL 加密库（这些加密库已提交给 NIST 用于 FIPS 140-2/140-3 验证）的 OpenShift Container Platform 集群，则不要创建使用 ed25519 算法的密钥。相反，创建一个使用 rsa 或 ecdsa 算法的密钥。

2. 查看公共 SSH 密钥：

```
$ cat <path>/<file_name>.pub
```

例如，运行以下命令来查看 ~/.ssh/id_ed25519.pub 公钥：

```
$ cat ~/.ssh/id_ed25519.pub
```

3. 将 SSH 私钥身份添加到本地用户的 SSH 代理（如果尚未添加）。在集群节点上，或者要使用 ./openshift-install gather 命令，需要对该密钥进行 SSH 代理管理，才能在集群节点上进行

免密码 SSH 身份验证。

注意

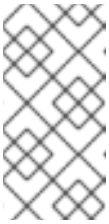
在某些发行版中，自动管理默认 SSH 私钥身份，如 ~/.ssh/id_rsa 和 ~/.ssh/id_dsa。

a. 如果 ssh-agent 进程尚未为您的本地用户运行，请将其作为后台任务启动：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```



注意

如果集群处于 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

4. 将 SSH 私钥添加到 ssh-agent ：

```
$ ssh-add <path>/<file_name> 1
```

1 指定 SSH 私钥的路径和文件名，如 ~/.ssh/id_ed25519.pub

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

后续步骤

- 安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。如果在您置备的基础架构上安装集群，则必须为安装程序提供密钥。

其他资源

- 验证节点健康状况

15.4.8. 手动创建安装配置文件

安装集群要求您手动创建安装配置文件。

先决条件

- 您在本地机器上有一个 SSH 公钥来提供给安装程序。该密钥将用于在集群节点上进行 SSH 身份验证，以进行调试和灾难恢复。
- 已获取 OpenShift Container Platform 安装程序和集群的 pull secret。
- 获取命令输出中的 imageContentSources 部分来 镜像存储库。
- 获取您的镜像 registry 的证书内容。

流程

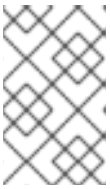
1. 创建一个安装目录来存储所需的安装资产：

```
$ mkdir <installation_directory>
```

重要

您必须创建一个目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

2. 自定义提供的 install-config.yaml 文件模板示例，并将其保存在 <installation_directory> 中。

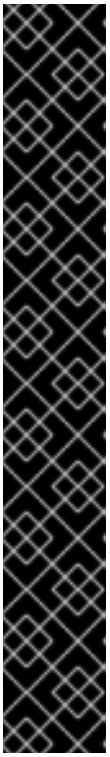


注意

您必须将此配置文件命名为 install-config.yaml。

- 除非使用 RHCOS 默认信任的 registry，如 docker.io，否则必须在 additionalTrustBundle 部分中提供镜像存储库的证书内容。在大多数情况下，您必须为您的镜像提供证书。

- 您必须包含命令输出 中的 imageContentSources 部分才能 镜像存储库。



重要

- 在镜像过程完成后，ImageContentSourcePolicy 文件作为 oc mirror 的输出生成。

- oc mirror 命令会生成 ImageContentSourcePolicy 文件，其中包含定义 ImageContentSourcePolicy 所需的 YAML。复制此文件中的文本并将其粘贴到 install-config.yaml 文件中。

- 您必须运行 'oc mirror' 命令两次。第一次运行 oc mirror 命令时，您将获得完整的 ImageContentSourcePolicy 文件。第二次运行 oc mirror 命令时，您只获得第一次运行和第二次运行之间的差别。由于此行为，您必须始终保留这些文件的备份，以防需要将这些文件合并到一个完整的 ImageContentSourcePolicy 文件中。备份这两个输出文件可确保您有完整的 ImageContentSourcePolicy 文件。

3. 备份 **install-config.yaml** 文件，以便您可以使用它安装多个集群。

重要

install-config.yaml 文件会在安装过程的下一步中使用。现在必须备份它。

其他资源

- 裸机的安装配置参数

15.4.8.1. 裸机 install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多详情，或修改所需参数的值。

```
apiVersion: v1
baseDomain: example.com (1)
compute: (2)
- hyperthreading: Enabled (3)
  name: worker
  replicas: 0 (4)
controlPlane: (5)
  hyperthreading: Enabled (6)
  name: master
  replicas: 3 (7)
metadata:
  name: test (8)
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14 (9)
    hostPrefix: 23 (10)
  networkType: OVNKubernetes (11)
  serviceNetwork: (12)
  - 172.30.0.0/16
platform:
  none: {} (13)
fips: false (14)
pullSecret: '{"auths":{"<local_registry>": {"auth": "<credentials>","email":
"you@example.com"}}}' (15)
sshKey: 'ssh-ed25519 AAAA...' (16)
additionalTrustBundle: | (17)
  -----BEGIN CERTIFICATE-----
  ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
  -----END CERTIFICATE-----
imageContentSources: (18)
```

```
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-v4.0-art-dev
```

1

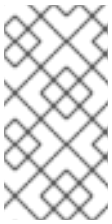
集群的基域。所有 DNS 记录都必须是这个基域的子域，并包含集群名称。

2 5

controlPlane 部分是一个单个映射，但 compute 部分是一系列映射。为满足不同数据结构的要求，compute 部分的第一行必须以连字符 - 开头，controlPlane 部分 的第一行则不以连字符开头。仅使用一个 control plane 池。

3 6

指定要启用或禁用并发多线程(SMT)还是超线程。默认情况下，启用 SMT 可提高机器中内核的性能。您可以通过将 参数值设置为 Disabled 来禁用它。如果禁用 SMT，则必须在所有集群机器中禁用它；这包括 control plane 和计算机器。



注意

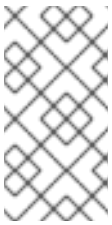
默认启用并发多线程(SMT)。如果您的 BIOS 设置中没有启用 SMT，超线程 参数无效。

重要

如果您禁用 超线程，无论是在 BIOS 中，还是在 install-config.yaml 文件中，请确保您的容量规划考虑机器性能显著降低的情况。

4

在用户置备的基础架构上安装 OpenShift Container Platform 时，必须将这个值设置为 0。在安装程序置备的安装中，参数控制集群为您创建和管理的计算机器数量。在用户置备的安装中，您必须在完成集群安装前手动部署计算机器。



注意

如果要安装一个三节点集群，在安装 Red Hat Enterprise Linux CoreOS(RHCOS)机器时不要部署任何计算机器。

7

您添加到集群的 control plane 机器数量。由于集群使用这些值作为集群中的 etcd 端点数量，所以该值必须与您部署的 control plane 机器数量匹配。

8

您在 DNS 记录中指定的集群名称。

9

从中分配 Pod IP 地址的 IP 地址块。此块不得与现有物理网络重叠。这些 IP 地址用于 pod 网络。如果需要从外部网络访问 pod，您必须配置负载均衡器和路由器来管理流量。

注意

类 E CIDR 范围被保留以供以后使用。要使用 Class E CIDR 范围，您必须确保您的网络环境接受 Class E CIDR 范围内的 IP 地址。

10

分配给每个节点的子网前缀长度。例如，如果 hostPrefix 设为 23，则每个节点从 given cidr 中分配 a /23 子网，这样就能有 510（2^(32 - 23)- 2）个 pod IP 地址。如果需要从外部网络访问节点，请配置负载均衡器和路由器来管理流量。

11

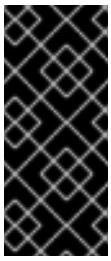
要安装的集群网络插件。默认值 OVNKubernetes 是唯一支持的值。

12

用于服务 IP 地址的 IP 地址池。您只能输入一个 IP 地址池。此块不得与现有物理网络重叠。如果您需要从外部网络访问服务，请配置负载均衡器和路由器来管理流量。

13

您必须将平台设置为 none。您无法为您的平台提供额外的平台配置变量。



重要

使用平台类型 none 安装的集群无法使用一些功能，如使用 Machine API 管理计算机器。即使附加到集群的计算机器安装在通常支持该功能的平台上，也会应用这个限制。在安装后无法更改此参数。

14

是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 Red Hat Enterprise Linux (RHEL) 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅在 FIPS 模式中安装该系统。

当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS)时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。

15

对于 <local_registry>，请指定 registry 域名，以及您的镜像 registry 用来提供内容的可选端口。例如： registry.example.com 或 registry.example.com:5000。对于 <credentials>，请为您的镜像 registry 指定 base64 编码的用户名和密码。

16

Red Hat Enterprise Linux CoreOS(RHCOS)中 core 用户的 SSH 公钥。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 ssh-agent 进程使用的 SSH 密钥。

17

提供用于镜像 registry 的证书文件内容。

18

根据您用来镜像存储库的命令输出提供 imageContentSources 部分。

重要

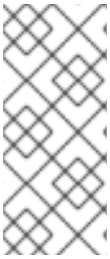
- 使用 oc adm release mirror 命令时，请使用 imageContentSources 部分中的输出。
- 使用 oc mirror 命令时，请使用运行该命令结果的 ImageContentSourcePolicy 文件的 repositoryDigestMirrors 部分。
- ImageContentSourcePolicy 已被弃用。如需更多信息，请参阅*配置镜像 registry 存储库镜像*。

其他资源

- 如需有关 API 和应用程序入口负载均衡要求的更多信息，请参阅用户置备的基础架构 的负载平衡要求。

15.4.8.2. 在安装过程中配置集群范围的代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 install-config.yaml 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。



注意

对于裸机安装，如果您没有从 install-config.yaml 文件中的 networking.machineNetwork[].cidr 字段指定的范围分配节点 IP 地址，您必须将其包括在 proxy.noProxy 字段中。

先决条件

- 您有一个现有的 install-config.yaml 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 Proxy 对象的 spec.noProxy 字段中添加站点来绕过代理。

注意

Proxy 对象 status.noProxy 字段使用安装配置中的 networking.machineNetwork[].cidr、networking.clusterNetwork[].cidr 和 networking.serviceNetwork[] 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，Proxy 对象 status.noProxy 字段也会使用实例元数据端点填充(169.254.169.254)。

流程

1. 编辑 install-config.yaml 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 http。

2 用于创建集群外 HTTPS 连接的代理 URL。

3 要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 . 以仅匹配子域。例如，.y.com 匹配 x.y.com，但不匹配 y.com。使用 * 绕过所有目的地的代理。

4 如果提供，安装程序会在 openshift-config 命名空间中生成名为 user-ca-bundle 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 trusted-ca-bundle 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， Proxy 对象的 trustedCA 字段中

也会引用此配置映射。additionalTrustBundle 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

5 可选：决定 Proxy 对象的配置以引用 trustedCA 字段中 user-ca-bundle 配置映射的策略。允许的值是 Proxyonly 和 Always。仅在配置了 http/https 代理时，使用 Proxyonly 引用 user-ca-bundle 配置映射。使用 Always 始终引用 user-ca-bundle 配置映射。默认值为 Proxyonly。

注意

安装程序不支持代理的 readinessEndpoints 字段。

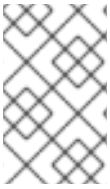
注意

如果安装程序超时，重启并使用安装程序的 wait-for 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. 保存该文件并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 cluster 的集群范围代理，该代理 使用 提供的 install-config.yaml 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 cluster Proxy 对象，但它会有一个空 spec。



注意

只支持名为 cluster 的 Proxy 对象，且无法创建额外的代理。

15.4.8.3. 配置三节点集群

另外，您可以在只由三台 control plane 机器组成的裸机集群中部署零台计算机器。这为集群管理员和开发人员提供了更小、效率更高的集群，用于测试、开发和生产。

在三节点 OpenShift Container Platform 环境中，三台 control plane 机器可以调度，这意味着应用程序工作负载被调度到它们上运行。

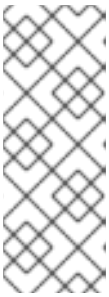
先决条件

- 您有一个现有的 install-config.yaml 文件。

流程

- 确保 install-config.yaml 文件中的计算副本数量设置为 0，如以下 计算 小节所示：

```
compute:
- name: worker
  platform: {}
  replicas: 0
```



注意

在用户置备的基础架构上安装 OpenShift Container Platform 时，无论您要部署的计算机器数量有多少，您必须将计算机器的 replicas 参数值设置为 0。在安装程序置备的安装中，参数控制集群为您创建和管理的计算机器数量。这不适用于手动部署计算机器的用户置备安装。

对于三节点集群安装，请按照以下步骤执行：

- 如果要部署一个带有零计算节点的三节点集群，Ingress Controller Pod 在 control plane 节点上运行。在三节点集群部署中，您必须配置应用程序入口负载均衡器，将 HTTP 和 HTTPS 流量路由到 control plane 节点。如需更多信息，*请参阅用户置备的基础架构的负载平衡要求* 部分。

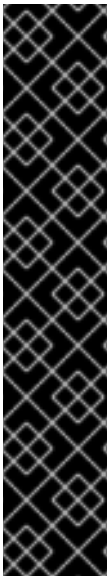
- 在以下步骤中创建 Kubernetes 清单文件时，请确保 <installation_directory>/manifests/cluster-scheduler-02-config.yml 文件中的 mastersSchedulable 参数被设置为 true。这可让应用程序工作负载在 control plane 节点上运行。

- 在创建 Red Hat Enterprise Linux CoreOS(RHCOS)机器时，不要部署任何计算节点。

15.4.9. 创建 Kubernetes 清单和 Ignition 配置文件

由于您必须修改一些集群定义文件并手动启动集群机器，因此您必须生成 Kubernetes 清单和 Ignition 配置文件来配置机器。

安装配置文件转换为 Kubernetes 清单。清单嵌套到 Ignition 配置文件中，稍后用于配置集群机器。



重要

- OpenShift Container Platform 安装程序生成的 Ignition 配置文件包含 24 小时后过期的证书，然后在该时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 node-bootstrapper 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。

- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

先决条件

- 已获得 OpenShift Container Platform 安装程序。对于受限网络安装，这些文件位于您的镜像主机上。

- 已创建 install-config.yaml 安装配置文件。

流程

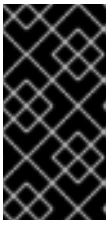
1. 进入包含 OpenShift Container Platform 安装程序的目录，并为集群生成 Kubernetes 清单：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 对于 <installation_directory>，请指定包含您创建的 install-config.yaml 文件的安装目录。

警告

如果您要安装一个三节点集群，请跳过以下步骤，以便可以调度 control plane 节点。



重要

当您将 control plane 节点从默认的不可调度配置为可以调度时，需要额外的订阅。这是因为 control plane 节点变为计算节点。

2. 检查 <installation_directory>/manifests/cluster-scheduler-02-config.yml Kubernetes 清单文件中的 mastersSchedulable 参数是否已设置为 false。此设置可防止在 control plane 机器上调度 pod：

   a. 打开 <installation_directory>/manifests/cluster-scheduler-02-config.yml 文件。

   b. 找到 mastersSchedulable 参数，并确保它被设置为 false。

   c. 保存并退出 文件。

3. 要创建 Ignition 配置文件，请从包含安装程序的目录运行以下命令：

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

1 对于 <installation_directory>，请指定相同的安装目录。

为安装目录中的 bootstrap、control plane 和计算节点创建 Ignition 配置文件。kubeadmin-password 和 kubeconfig 文件在 ./<installation_directory>/auth 目录中创建：

```
.
```

```
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

其他资源

- 如需有关 恢复 kubelet 证书的更多信息，请参阅恢复已过期的 control plane 证书。

15.4.10. 配置 chrony 时间服务

您必须通过修改 chrony .conf 文件的内容来设置 chrony 时间服务(chronyd)使用的时间服务器和相关设置，并将这些内容作为机器配置传递给节点。

流程

1. 创建一个 Butane 配置，包括 chrony.conf 文件的内容。例如，要在 worker 节点上配置 chrony，请创建一个 99-worker-chrony.bu 文件。

注意

如需有关 Butane 的信息，请参阅"使用 Butane 创建机器配置"。

```
variant: openshift
version: 4.16.0
metadata:
  name: 99-worker-chrony (1)
  labels:
    machineconfiguration.openshift.io/role: worker (2)
storage:
  files:
  - path: /etc/chrony.conf
    mode: 0644 (3)
    overwrite: true
    contents:
      inline: |
        pool 0.rhel.pool.ntp.org iburst (4)
        driftfile /var/lib/chrony/drift
        makestep 1.0 3
        rtcsync
        logdir /var/log/chrony
```

1 2 在 control plane 节点上，在这两个位置中将 master 替换为 worker。

3 为机器配置文件的 mode 字段指定数值模式。在创建文件并应用更改后，模式 将转换为十进制值。您可以使用 oc get mc <mc-name> -o yaml 命令来检查 YAML 文件。

4 指定任何有效的、可访问的时间源，如 DHCP 服务器提供的源。

2. 使用 Butane 生成 MachineConfig 对象文件 99-worker-chrony.yaml，其中包含要交付至节点的配置：

```
$ butane 99-worker-chrony.bu -o 99-worker-chrony.yaml
```

3. 使用以下两种方式之一应用配置：

- 如果集群还没有运行，在生成清单文件后，将 MachineConfig 对象文件添加到 <installation_directory>/openshift 目录中，然后继续创建集群。
- 如果集群已在运行，请应用该文件：

```
$ oc apply -f ./99-worker-chrony.yaml
```

15.4.11. 安装 RHCOS 并启动 OpenShift Container Platform bootstrap 过程

要在您置备的裸机基础架构上安装 OpenShift Container Platform，您必须在机器上安装 Red Hat Enterprise Linux CoreOS(RHCOS)。安装 RHCOS 时，您必须为您要安装的机器类型提供 OpenShift Container Platform 安装程序生成的 Ignition 配置文件。如果您配置了适当的网络、DNS 和负载均衡基础架构，OpenShift Container Platform bootstrap 过程会在 RHCOS 机器重启后自动启动。

要在机器上安装 RHCOS，请按照以下步骤使用 ISO 镜像或网络 PXE 引导。

注意

本安装文档中包括的计算节点部署步骤特定于 RHCOS。如果您选择部署基于 RHEL 的计算节点，您需要负责所有操作系统生命周期管理和维护，包括执行系统更新、应用补丁和完成所有其他必要的任务。仅支持 RHEL 8 计算机器。

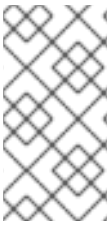
您可以使用以下方法在 ISO 和 PXE 安装过程中配置 RHCOS：

- 内核参数： 您可以使用内核参数来提供特定于安装的信息。例如，您可以指定上传到 HTTP 服务器的 RHCOS 安装文件的位置以及您要安装的节点类型的 Ignition 配置文件的位置。对于 PXE 安装，您可以使用 APPEND 参数将参数传递给 live 安装程序的内核。对于 ISO 安装，您可以中断实时安装引导过程来添加内核参数。在这两个安装情形中，您可以使用特殊的 coreos.inst.* 参数来指示实时安装程序，以及标准安装引导参数来打开或关闭标准内核服务。

- Ignition 配置：OpenShift Container Platform Ignition 配置文件(*.ign)特定于您要安装的节点类型。您可以在 RHCOS 安装过程中传递 bootstrap、control plane 或计算节点 Ignition 配置文件的位置，以便在首次启动时生效。特殊情况下，您可以创建单独的、有限的 Ignition 配置以传递给 live 系统。该 Ignition 配置可以执行特定的任务，如在安装完成后向置备系统报告成功。此特殊的 Ignition 配置由 coreos-installer 使用，以便在首次引导安装的系统时应用。不要直接为实时 ISO 提供标准 control plane 和计算节点 Ignition 配置。

- coreos-installer ：您可以将 live ISO 安装程序引导到 shell 提示符，这可让您在第一次引导前以多种方式准备持久性系统。特别是，您可以运行 coreos-installer 命令来识别要包含的各种工件、使用磁盘分区和设置网络。在某些情况下，您可以配置 live 系统上的功能并将其复制到安装的系统中。

使用 ISO 安装还是 PXE 安装取决于您的情况。PXE 安装需要可用的 DHCP 服务并进行更多准备，但可以使安装过程更加自动化。ISO 安装是一个更手动的过程，如果您设置的机器数超过几台，则可能不方便。



注意

从 OpenShift Container Platform 4.6 开始，RHCOS ISO 和其他安装工件支持在带有 4K 扇区的磁盘上安装。

15.4.11.1. 使用 ISO 镜像安装 RHCOS

您可以使用 ISO 镜像在机器上安装 RHCOS。

先决条件

- 已为集群创建 Ignition 配置文件。
- 您已配置了适当的网络、DNS 和负载平衡基础架构。
- 您有一个可以从计算机以及您创建的机器访问的 HTTP 服务器。
- 您已参阅 *Advanced RHCOS 安装配置* 部分来了解配置功能的不同方法，如网络和磁盘分区。

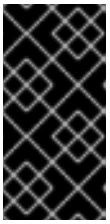
流程

1. 获取每个 Ignition 配置文件的 SHA512 摘要。例如，您可以在运行 Linux 的系统上使用以下内容来获取 bootstrap.ign Ignition 配置文件的 SHA512 摘要：

```
$ sha512sum <installation_directory>/bootstrap.ign
```

后续步骤中会向 coreos-installer 提供摘要，以验证集群节点上 Ignition 配置文件的真实性。

2. 将安装程序创建的 bootstrap、control plane 和计算节点 Ignition 配置文件上传到 HTTP 服务器。注意这些文件的 URL。



重要

您可以在 Ignition 配置中添加或更改配置设置，然后将其保存到 HTTP 服务器。如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

3. 从安装主机上，验证 Ignition 配置文件是否在 URL 上可用。以下示例获取 bootstrap 节点的 Ignition 配置文件：

```
$ curl -k http://<HTTP_server>/bootstrap.ign 1
```

输出示例

```
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0{"ignition":
{"version":"3.2.0"},"passwd":{"users":[{"name":"core","sshAuthorizedKeys":["ssh-
rsa...
```

在命令中将 bootstrap.ign 替换为 master.ign 或 worker.ign，以验证 control plane 和计算节点的 Ignition 配置文件是否可用。

4. 虽然可以从 RHCOS 镜像镜像页面获取您选择的操作系统实例安装方法所需的 RHCOS 镜像，但推荐的方法是从 openshift-install 命令的输出获取 RHCOS 镜像的正确版本：

```
$ openshift-install coreos print-stream-json | grep '\.iso[^.]'
```

输出示例

```
"location": "<url>/art/storage/releases/rhcos-4.16-aarch64/<release>/aarch64/rhcos-
<release>-live.aarch64.iso",
"location": "<url>/art/storage/releases/rhcos-4.16-ppc64le/<release>/ppc64le/rhcos-
<release>-live.ppc64le.iso",
"location": "<url>/art/storage/releases/rhcos-4.16-s390x/<release>/s390x/rhcos-
<release>-live.s390x.iso",
"location": "<url>/art/storage/releases/rhcos-4.16/<release>/x86_64/rhcos-<release>-
live.x86_64.iso",
```

重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每个发行版本而改变。您必须下载最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。如果可用，请使用与 OpenShift Container Platform 版本匹配的镜像版本。这个过程只使用 ISO 镜像。此安装类型不支持 RHCOS qcow2 镜像。

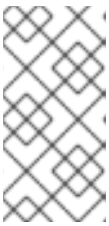
ISO 文件名类似以下示例：

```
rhcos-<version>-live.<architecture>.iso
```

5. 使用 ISO 启动 RHCOS 安装。使用以下安装选项之一：

- 将 ISO 映像刻录到磁盘并直接启动。
- 使用 light-out 管理(LOM)接口使用 ISO 重定向。

6. 在不指定任何选项或中断实时引导序列的情况下引导 RHCOS ISO 镜像。等待安装程序在 RHCOS live 环境中引导进入 shell 提示符。



注意

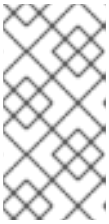
可以中断 RHCOS 安装引导过程来添加内核参数。但是，在这个 ISO 过程中，您应该使用以下步骤中所述的 coreos-installer 命令，而不是添加内核参数。

7. 运行 coreos-installer 命令并指定满足您的安装要求的选项。您至少必须指定指向节点类型的 Ignition 配置文件的 URL，以及您要安装到的设备：

```
$ sudo coreos-installer install --ignition-url=http://<HTTP_server>/<node_type>.ign
<device> --ignition-hash=sha512-<digest> 1 2
```

1 1 您必须使用 sudo 运行 coreos-installer 命令，因为 core 用户没有执行安装所需的 root 权限。

2 当 Ignition 配置文件通过 HTTP URL 获取时，需要 --ignition-hash 选项来验证集群节点上 Ignition 配置文件的真实性。<digest> 是上一步中获取的 Ignition 配置文件 SHA512 摘要。



注意

如果要通过使用 TLS 的 HTTPS 服务器提供 Ignition 配置文件，您可以在运行 coreos-installer 前将内部证书颁发机构(CA)添加到系统信任存储中。

以下示例将引导节点安装初始化到 /dev/sda 设备。bootstrap 节点的 Ignition 配置文件从 IP 地址 192.168.1.2 的 HTTP Web 服务器获取：

```
$ sudo coreos-installer install --ignition-
url=http://192.168.1.2:80/installation_directory/bootstrap.ign /dev/sda --ignition-
hash=sha512-
a5a2d43879223273c9b60af66b44202a1d1248fc01cf156c46d4a79f552b6bad47bc8cc78dd
f0116e80c59d2ea9e32ba53bc807afbca581aa059311def2c3e3b
```

8. 在机器的控制台上监控 RHCOS 安装的进度。

重要

在开始安装 OpenShift Container Platform 之前，确保每个节点中安装成功。观察安装过程可以帮助确定可能会出现 RHCOS 安装问题的原因。

9. 安装 RHCOS 后，您必须重启系统。系统重启过程中，它会应用您指定的 Ignition 配置文件。

10. 检查控制台输出，以验证 Ignition 是否运行。

示例命令

```
Ignition: ran on 2022/03/14 14:48:33 UTC (this boot)
Ignition: user-provided config was applied
```

11. 继续为集群创建其他机器。

重要

此时您必须创建 bootstrap 和 control plane 机器。如果 control plane 机器不可调度，请在安装 OpenShift Container Platform 前至少创建两台计算机器。

如果存在所需的网络、DNS 和负载均衡器基础架构，OpenShift Container Platform bootstrap 过程会在 RHCOS 节点重启后自动启动。

注意

RHCOS 节点不包含 core 用户的默认密码。您可以使用可访问 SSH 私钥的用户的身份运行 ssh core@<node>.<cluster_name>.<base_domain > 来访问节点，该私钥与您在 install_config.yaml 文件中指定的公钥配对。运行 RHCOS 的 OpenShift Container Platform 4 集群节点不可变，它依赖于 Operator 来应用集群更改。不建议使用 SSH 访问集群节点。但是，当调查安装问题时，如果 OpenShift Container Platform API 不可用，或者 kubelet 在目标节点上无法正常工作，则调试或灾难恢复可能需要 SSH 访问。

15.4.11.2. 使用 PXE 或 iPXE 启动安装 RHCOS

您可以使用 PXE 或 iPXE 启动在机器上安装 RHCOS。

先决条件

- 已为集群创建 Ignition 配置文件。
- 您已配置了适当的网络、DNS 和负载平衡基础架构。
- 您已配置了合适的 PXE 或 iPXE 基础架构。
- 您有一个可以从计算机以及您创建的机器访问的 HTTP 服务器。
- 您已参阅 *Advanced RHCOS 安装配置* 部分来了解配置功能的不同方法，如网络和磁盘分区。

流程

1. 将安装程序创建的 bootstrap、control plane 和计算节点 Ignition 配置文件上传到 HTTP 服务器。注意这些文件的 URL。

重要

您可以在 Ignition 配置中添加或更改配置设置，然后将其保存到 HTTP 服务器。如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

2. 从安装主机上，验证 Ignition 配置文件是否在 URL 上可用。以下示例获取 bootstrap 节点的 Ignition 配置文件：

```
$ curl -k http://<HTTP_server>/bootstrap.ign 1
```

输出示例

```
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0{"ignition":
{"version":"3.2.0"},"passwd":{"users":[{"name":"core","sshAuthorizedKeys":["ssh-
rsa...
```

在命令中将 bootstrap.ign 替换为 master.ign 或 worker.ign，以验证 control plane 和计算节点的 Ignition 配置文件是否可用。

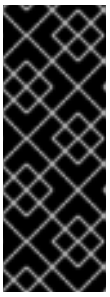
3. 虽然可以从 RHCOS image mirror 页面获取您选择的操作系统实例所需的 RHCOS kernel、initramfs 和 rootfs 文件，但推荐的方法是从 openshift-install 命令的输出中获取 RHCOS 文件的正确版本：

```
$ openshift-install coreos print-stream-json | grep -Eo '"https.*(kernel-
|initramfs.|rootfs.)\w+(\.img)?"'
```

输出示例

```
"<url>/art/storage/releases/rhcos-4.16-aarch64/<release>/aarch64/rhcos-<release>-live-
```

```
kernel-aarch64"
"<url>/art/storage/releases/rhcos-4.16-aarch64/<release>/aarch64/rhcos-<release>-live-initramfs.aarch64.img"
"<url>/art/storage/releases/rhcos-4.16-aarch64/<release>/aarch64/rhcos-<release>-live-rootfs.aarch64.img"
"<url>/art/storage/releases/rhcos-4.16-ppc64le/49.84.202110081256-0/ppc64le/rhcos-<release>-live-kernel-ppc64le"
"<url>/art/storage/releases/rhcos-4.16-ppc64le/<release>/ppc64le/rhcos-<release>-live-initramfs.ppc64le.img"
"<url>/art/storage/releases/rhcos-4.16-ppc64le/<release>/ppc64le/rhcos-<release>-live-rootfs.ppc64le.img"
"<url>/art/storage/releases/rhcos-4.16-s390x/<release>/s390x/rhcos-<release>-live-kernel-s390x"
"<url>/art/storage/releases/rhcos-4.16-s390x/<release>/s390x/rhcos-<release>-live-initramfs.s390x.img"
"<url>/art/storage/releases/rhcos-4.16-s390x/<release>/s390x/rhcos-<release>-live-rootfs.s390x.img"
"<url>/art/storage/releases/rhcos-4.16/<release>/x86_64/rhcos-<release>-live-kernel-x86_64"
"<url>/art/storage/releases/rhcos-4.16/<release>/x86_64/rhcos-<release>-live-initramfs.x86_64.img"
"<url>/art/storage/releases/rhcos-4.16/<release>/x86_64/rhcos-<release>-live-rootfs.x86_64.img"
```



重要

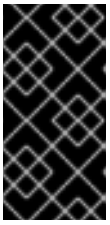
RHCOS 工件可能不会随着 OpenShift Container Platform 的每个发行版本而改变。您必须下载最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。这个过程只使用下面描述的适当 kernel、initram fs 和 rootfs 工件。此安装类型不支持 RHCOS QCOW2 镜像。

文件名包含 OpenShift Container Platform 版本号。它们类似以下示例：

- kernel:rhcos-<version>-live-kernel-<architecture>
- initramfs: rhcos-<version>-live-initramfs.<architecture>.img
- rootfs: rhcos-<version>-live-rootfs.<architecture>.img

4

4. 将 rootfs、kernel 和 initramfs 文件上传到 HTTP 服务器。



重要

如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

5. 配置网络引导基础架构，以便在安装 RHCOS 后机器从本地磁盘启动。

6. 为 RHCOS 镜像配置 PXE 或 iPXE 安装并开始安装。

为环境修改以下示例菜单条目之一，并验证能否正确访问镜像和 Ignition 文件：

- 对于 PXE(x86_64)：

```
DEFAULT pxeboot
TIMEOUT 20
PROMPT 0
LABEL pxeboot
    KERNEL http://<HTTP_server>/rhcos-<version>-live-kernel-<architecture> 1
    APPEND initrd=http://<HTTP_server>/rhcos-<version>-live-initramfs.
<architecture>.img coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-
rootfs.<architecture>.img coreos.inst.install_dev=/dev/sda
coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign 2 3
```

1 1 指定上传到 HTTP 服务器的 live kernel 文件位置。URL 必须是 HTTP、TFTP 或 FTP；不支持 HTTPS 和 NFS。

2 如果您使用多个 NIC，请在 ip 选项中指定一个接口。例如，要在名为 eno1 的 NIC 上使用 DHCP，请设置 ip=eno1:dhcp。

3 指定上传到 HTTP 服务器的 RHCOS 文件的位置。initrd 参数值是 initramfs 文件的位置，coreos.live.rootfs_url 参数值是 rootfs 文件的位置，coreos.inst.ignition_url 参数值则是 bootstrap Ignition 配置文件的位置。您还可以在 APPEND 行中添加更多内核参数来配置联网或其他引导选项。

注意

此配置不会在图形控制台的机器上启用串行控制台访问。要配置不同的控制台，请在 APPEND 行中添加一个或多个 console= 参数。例如，添加 console=tty0 console=ttyS0 以将第一个 PC 串口设置为主控制台，并将图形控制台设置为二级控制台。如需更多信息，请参阅如何在 Red Hat Enterprise Linux 中设置串行终端和/或控制台？ 和"启用 PXE 和 ISO 安装的串行控制台"部分。

- 对于 iPXE (x86_64 + aarch64) :

```
kernel http://<HTTP_server>/rhcos-<version>-live-kernel-<architecture> initrd=main
coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-rootfs.
<architecture>.img coreos.inst.install_dev=/dev/sda
coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign 1 2
initrd --name main http://<HTTP_server>/rhcos-<version>-live-initramfs.
<architecture>.img 3
boot
```

1

指定上传到 HTTP 服务器的 RHCOS 文件的位置。kernel 参数值是 kernel 文件的位置，init rd=main 参数用于在 UEFI 系统中引导，coreos.live.rootfs_url 参数值是 rootfs 文件的位置，coreos.inst.ignition_url 参数值则是 bootstrap Ignition 配置文件的位置。

2

如果您使用多个 NIC，请在 ip 选项中指定一个接口。例如，要在名为 eno1 的 NIC 上使用 DHCP，请设置 ip=eno1:dhcp。

3

指定上传到 HTTP 服务器的 initramfs 文件的位置。

注意

此配置不会在图形控制台的机器上启用串行控制台访问。要配置不同的控制台，请在 内核参数 中添加一个或多个 console= 参数。例如，添加 console=tty0 console=ttyS0 以将第一个 PC 串口设置为主控制台，并将图形控制台设置为二级控制台。如需更多信息，请参阅如何在 Red Hat Enterprise Linux 中设置串行终端和/或控制台？ 和"启用 PXE 和 ISO 安装的串行控制台"部分。

注意

要在 aarch64 架构中网络引导 CoreOS 内核，您需要使用启用了 IMAGE_GZIP 选项的 iPXE 构建版本。请参阅 iPXE 中的IMAGE_GZIP 选项。

- 对于 aarch64 中的 PXE（使用 UEFI 和 Grub 作为第二阶段）：

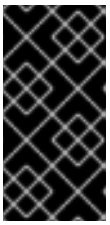
```
menuentry 'Install CoreOS' {
  linux rhcos-<version>-live-kernel-<architecture>
coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-rootfs.
<architecture>.img coreos.inst.install_dev=/dev/sda
coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign 1 2
  initrd rhcos-<version>-live-initramfs.<architecture>.img 3
}
```

1. 指定上传到 HTTP/TFTP 服务器的 RHCOS 文件的位置。kernel 参数值是 TFTP 服务器中的 kernel 文件的位置。coreos.live.rootfs_url 参数值是 rootfs 文件的位置，coreos.inst.ignition_url 参数值是 HTTP 服务器上的 bootstrap Ignition 配置文件的位置。

2. 如果您使用多个 NIC，请在 ip 选项中指定一个接口。例如，要在名为 eno1 的 NIC 上使用 DHCP，请设置 ip=eno1:dhcp。

3. 指定上传到 TFTP 服务器的 initramfs 文件的位置。

7. 在机器的控制台上监控 RHCOS 安装的进度。



重要

在开始安装 OpenShift Container Platform 之前，确保每个节点中安装成功。观察安装过程可以帮助确定可能会出现 RHCOS 安装问题的原因。

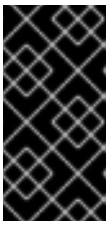
8. 安装 RHCOS 后，系统会重启。在重启过程中，系统会应用您指定的 Ignition 配置文件。

9. 检查控制台输出，以验证 Ignition 是否运行。

示例命令

```
Ignition: ran on 2022/03/14 14:48:33 UTC (this boot)
Ignition: user-provided config was applied
```

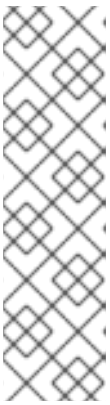
10. 继续为集群创建机器。



重要

此时您必须创建 bootstrap 和 control plane 机器。如果 control plane 机器不可调度，请在安装集群前至少创建两台计算机器。

如果存在所需的网络、DNS 和负载均衡器基础架构，OpenShift Container Platform bootstrap 过程会在 RHCOS 节点重启后自动启动。



注意

RHCOS 节点不包含 core 用户的默认密码。您可以使用可访问 SSH 私钥的用户的身份运行 ssh core@<node>.<cluster_name>.<base_domain > 来访问节点，该私钥与您在 install_config.yaml 文件中指定的公钥配对。运行 RHCOS 的 OpenShift Container Platform 4 集群节点不可变，它依赖于 Operator 来应用集群更改。不建议使用 SSH 访问集群节点。但是，当调查安装问题时，如果 OpenShift Container Platform API 不可用，或者 kubelet 在目标节点上无法正常工作，则调试或灾难恢复可能需要 SSH 访问。

15.4.11.3. 高级 RHCOS 安装配置

为 OpenShift Container Platform 手动置备 Red Hat Enterprise Linux CoreOS(RHCOS)节点的一个关键优点是能够进行通过默认的 OpenShift Container Platform 安装方法无法进行的配置。本节介绍了您可以使用的一些技术进行配置，其中包括：

- 将内核参数传递给实时安装程序

- 从 live 系统手动运行 coreos-installer
- 自定义实时 ISO 或 PXE 引导镜像

本节详述了与 Red Hat Enterprise Linux CoreOS(RHCOS)手动安装的高级配置相关的内容，如磁盘分区、网络以及使用 Ignition 配置的不同方式相关。

15.4.11.3.1. 使用高级网络选项进行 PXE 和 ISO 安装

OpenShift Container Platform 节点的网络默认使用 DHCP 来收集所有必要的配置设置。要设置静态 IP 地址或配置特殊设置，如绑定，您可以执行以下操作之一：

- 引导 live 安装程序时传递特殊内核参数。
- 使用机器配置将网络文件复制到安装的系统中。
- 从 live 安装程序 shell 提示符配置网络，然后将这些设置复制到安装的系统上，以便在安装的系统第一次引导时生效。

要配置 PXE 或 iPXE 安装，请使用以下选项之一：

- 请参阅"高级 RHCOS 安装参考"表。
- 使用机器配置将网络文件复制到安装的系统中。

要配置 ISO 安装，请使用以下步骤：

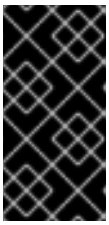
流程

1. 引导 ISO 安装程序.
2.

在 live 系统 shell 提示符下，使用可用的 RHEL 工具（如 nmcli 或 nmtui ）为 live 系统配置网络。

3. 运行 coreos-installer 命令以安装系统，添加 --copy-network 选项来复制网络配置。例如：

```
$ sudo coreos-installer install --copy-network \
    --ignition-url=http://host/worker.ign /dev/disk/by-id/scsi-<serial_number>
```



重要

copy-network 选项仅复制在 /etc/NetworkManager/system-connections 下找到的网络配置。特别是，它不会复制系统主机名。

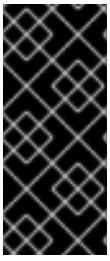
4. 重启安装的系统。

其他资源

- 有关 nmcli 和 nmtui 工具的更多信息，请参阅 RHEL 8 文档中的 Getting started with nmcli and Getting started with nmtui。

15.4.11.3.2. 磁盘分区

磁盘分区是在 Red Hat Enterprise Linux CoreOS(RHCOS)安装过程中在 OpenShift Container Platform 集群节点上创建的。特定架构的每个 RHCOS 节点使用相同的分区布局，除非覆盖默认的分区配置。在 RHCOS 安装过程中，根文件系统的大小会增加，以使用目标设备中剩余的可用空间。



重要

在节点上使用自定义分区方案可能会导致 OpenShift Container Platform 在某些节点分区上监控或警报。如果要覆盖默认分区，请参阅 了解 OpenShift 文件系统监控（驱除条件） 以了解有关 OpenShift Container Platform 如何监控主机文件系统的更多信息。

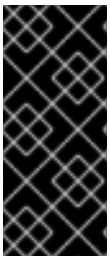
OpenShift Container Platform 监控以下两个文件系统标识符：

- nodefs，这是包含 /var/lib/kubelet 的文件系统

- imagefs，这是包含 /var/lib/containers 的文件系统

对于默认分区方案，nodefs 和 imagefs 监控相同的根文件系统 /。

要在 OpenShift Container Platform 集群节点上安装 RHCOS 时覆盖默认分区，您必须创建单独的分区。您可能想要为容器和容器镜像添加单独的存储分区。例如，通过在独立分区中挂载 /var/lib/containers，kubelet 会单独监控 /var/lib/containers 作为 imagefs 目录，以及 root 文件系统作为 nodefs 目录。



重要

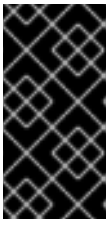
如果您已将磁盘大小调整为托管更大的文件系统，请考虑创建单独的 /var/lib/containers 分区。考虑重新定义具有 xfs 格式的磁盘大小，以减少大量分配组导致的 CPU 时间问题。

15.4.11.3.2.1. 创建独立 /var 分区

通常，您应该使用在 RHCOS 安装过程中创建的默认磁盘分区。然而，在有些情况下您可能需要为预期增长的目录创建独立分区。

OpenShift Container Platform 支持添加单个分区将存储附加到 /var 目录或 /var 的子目录中。例如：

- /var/lib/containers ：保存随着系统中添加更多镜像和容器而增长的容器相关内容。
- /var/lib/etcd ：保存您可能希望独立保留的数据，比如 etcd 存储的性能优化。
- /var ：保存您可能希望独立保留的数据，以满足审计等目的。



重要

对于大于 100GB 的磁盘大小，特别是磁盘大小大于 1TB，请创建一个独立的 /var 分区。

通过单独存储 /var 目录的内容，可以更轻松地根据需要为区域扩展存储，并在以后重新安装 OpenShift Container Platform，并保持该数据的完整性。使用这个方法，您不必再次拉取所有容器，在更新系统时也不必复制大量日志文件。

将独立分区用于 /var 目录或 /var 的子目录也会防止分区目录中的数据增加填充根文件系统。

以下流程通过添加机器配置清单来设置独立的 /var 分区，该清单会在安装准备阶段封装到节点类型的 Ignition 配置文件中。

流程

1. 在安装主机上，切换到包含 OpenShift Container Platform 安装程序的目录，并为集群生成 Kubernetes 清单：

```
$ openshift-install create manifests --dir <installation_directory>
```

2. 创建用于配置额外分区的 Butane 配置。例如，将文件命名为 $HOME/clusterconfig/98-var-partition.bu，将磁盘设备名称改为 worker 系统上存储设备的名称，并根据情况设置存储大小。这个示例将 /var 目录放在一个单独的分区中：

```
variant: openshift
version: 4.16.0
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
storage:
  disks:
  - device: /dev/disk/by-id/<device_name> 1
    partitions:
    - label: var
      start_mib: <partition_start_offset> 2
      size_mib: <partition_size> 3
      number: 5
  filesystems:
    - device: /dev/disk/by-partlabel/var
      path: /var
      format: xfs
      mount_options: [defaults, prjquota] 4
      with_mount_unit: true
```

1. 要分区的磁盘的存储设备名称。

2

当在引导磁盘中添加数据分区时，推荐最少使用偏移值 25000 兆字节。root 文件系统会自动调整大小以填充所有可用空间（最多到指定的偏移值）。如果没有指定偏移值，或者指定的值小于推荐的最小值，则生成的 root 文件系统会太小，而在以后进行的 RHCOS 重新安装可能会覆盖数据分区的开始部分。

3

以兆字节为单位的数据分区大小。

4

对于用于容器存储的文件系统，必须启用 prjquota 挂载选项。

注意

在创建单独的 /var 分区时，如果不同的实例类型没有相同的设备名称，则无法将不同的实例类型用于计算节点。

3. 从 Butane 配置创建一个清单，并将它保存到 clusterconfig/openshift 目录中。例如，运行以下命令：

```
$ butane $HOME/clusterconfig/98-var-partition.bu -o
$HOME/clusterconfig/openshift/98-var-partition.yaml
```

4. 创建 Ignition 配置文件：

```
$ openshift-install create ignition-configs --dir <installation_directory> 1
```

1

对于 <installation_directory>，请指定相同的安装目录。

为安装目录中的 bootstrap、control plane 和计算节点创建 Ignition 配置文件：

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
```

```
├── master.ign
├── metadata.json
└── worker.ign
```

<installation_directory>/manifest 和 <installation_directory>/openshift 目录中的文件被嵌套到 Ignition 配置文件中，包括包含 98-var-partition 自定义 MachineConfig 对象的文件。

后续步骤

- 您可以通过在 RHCOS 安装过程中引用 Ignition 配置文件来应用自定义磁盘分区。

15.4.11.3.2.2. 保留现有分区

对于 ISO 安装，您可以在 coreos-installer 命令中添加可让安装程序维护一个或多个现有分区的选项。对于 PXE 安装，您可以在 APPEND 参数中添加 coreos.inst.* 选项来保留分区。

保存的分区可能是来自现有 OpenShift Container Platform 系统的数据分区。您可以通过分区标签或编号识别您要保留的磁盘分区。

注意

如果您保存了现有分区，且这些分区没有为 RHCOS 留下足够空间，则安装将失败，而不影响保存的分区。

在 ISO 安装过程中保留现有分区

这个示例保留分区标签以 data开头的任何分区(data *)：

```
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign \
      --save-partlabel 'data*' /dev/disk/by-id/scsi-<serial_number>
```

以下示例演示了在运行 coreos-installer 时要保留磁盘上的第 6 个分区：

```
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign \
      --save-partindex 6 /dev/disk/by-id/scsi-<serial_number>
```

这个示例保留分区 5 及更高分区：

```
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign
      --save-partindex 5- /dev/disk/by-id/scsi-<serial_number>
```

在前面已保存分区的示例中，coreos-installer 会立即重新创建分区。

在 PXE 安装过程中保留现有分区

这个 APPEND 选项保留分区标签以 'data'('data*')开头的任何分区：

```
coreos.inst.save_partlabel=data*
```

这个 APPEND 选项保留分区 5 及更高分区：

```
coreos.inst.save_partindex=5-
```

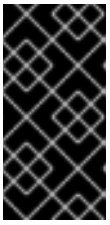
这个 APPEND 选项保留分区 6:

```
coreos.inst.save_partindex=6
```

15.4.11.3.3. 识别 Ignition 配置

在进行 RHCOS 手动安装时，您可以提供两种 Ignition 配置类型，它们有不同的原因：

- 永久安装 Ignition 配置 ：每个手动 RHCOS 安装都需要传递 openshift-installer 生成的 Ignition 配置文件之一，如 bootstrap.ign、master.ign 和 worker.ign，才能进行安装。



重要

不建议直接修改这些 Ignition 配置文件。您可以更新嵌套到 Ignition 配置文件中的清单文件，如上一节示例中所述。

对于 PXE 安装，您可以使用 coreos.inst.ignition_url= 选项在 APPEND 行上传递 Ignition 配置。对于 ISO 安装，在 ISO 引导至 shell 提示符后，您可以使用带有 --ignition-url= 选项的 coreos-installer 命令行。在这两种情况下，只支持 HTTP 和 HTTPS 协议。

- 实时安装 Ignition 配置 ：可使用 coreos-installer customize 子命令及其各种选项来创建此类型。使用此方法，Ignition 配置会传递到 live 安装介质，在引导时立即运行，并在 RHCOS

系统安装到磁盘之前或之后执行设置任务。这个方法只用于必须执行一次且之后不能再次应用的任务，比如不能使用机器配置进行的高级分区。

对于 PXE 或 ISO 引导，您可以创建 Ignition 配置，APP END ignition.config.url= 选项来标识 Ignition 配置的位置。您还需要附加 ignition.firstboot ignition.platform.id=metal 或 ignition.config.url 选项。

15.4.11.3.4. 默认控制台配置

从 OpenShift Container Platform 4.16 引导镜像安装的 Red Hat Enterprise Linux CoreOS (RHCOS) 节点使用默认控制台，旨在识别大多数虚拟化和裸机设置。不同的云和虚拟化平台可能会根据所选的架构使用不同的默认设置。裸机安装使用内核默认设置，这通常意味着图形控制台是主控制台，并且禁用串行控制台。

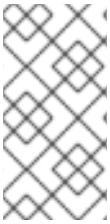
默认控制台可能与特定的硬件配置不匹配，或者您可能具有需要调整默认控制台的特定需求。例如：

- 您希望访问控制台上的紧急 shell 进行调试。
- 您的云平台没有提供到图形控制台的互动访问，但提供了一个串行控制台。
- 您需要启用多个控制台。

控制台配置继承自引导镜像。这意味着现有集群中的新节点不受默认控制台的影响。

您可以使用以下方法为裸机安装配置控制台：

- 在命令行中手动使用 coreos-installer。
- 使用带有 --dest-console 选项的 coreos-installer iso customize 或 coreos-installer pxe customize 子命令，以创建可自动执行进程的自定义镜像。



注意

对于高级自定义，请使用 coreos-installer iso 或 coreos-installer pxe 子命令而不是内核参数来执行控制台配置。

15.4.11.3.5. 为 PXE 和 ISO 安装启用串行控制台

默认情况下，Red Hat Enterprise Linux CoreOS (RHCOS) 串行控制台被禁用，所有输出都会写入图形控制台。您可以为 ISO 安装启用串行控制台并重新配置引导装载程序，以便输出同时发送到串行控制台和图形控制台。

流程

1. 引导 ISO 安装程序.

2. 运行 coreos-installer 命令来安装系统，添加 --console 选项一次来指定图形控制台，然后第二次指定串行控制台：

```
$ coreos-installer install \
  --console=tty0 \ 1
  --console=ttyS0,<options> \ 2
  --ignition-url=http://host/worker.ign /dev/disk/by-id/scsi-<serial_number>
```

1. 所需的二级控制台。在这种情况下，是图形控制台。省略这个选项将禁用图形控制台。

2. 所需的主控制台。在这种情况下，是串行控制台。options 字段定义 baud 速率和其他设置。此字段的一个常见值为 11520n8。如果没有提供选项，则使用默认内核值 9600n8。有关这个选项格式的更多信息，请参阅 Linux 内核串口控制台 文档。

3. 重启安装的系统。

注意

可以使用 coreos-installer install --append-karg 选项来获取类似的结果，并使用 console= 指定控制台。但是，这只会为内核设置控制台，而不为引导装载程序设置控制台。

要配置 PXE 安装，请确保省略 coreos.inst.install_dev 内核命令行选项，并使用 shell 提示符使用上述 ISO 安装过程手动运行 coreos-installer。

15.4.11.3.6. 自定义 live RHCOS ISO 或 PXE 安装

您可以通过将 Ignition 配置文件直接注入镜像中来使用 live ISO 镜像或 PXE 环境来安装 RHCOS。这会创建一个自定义镜像，供您用来置备系统。

对于 ISO 镜像，此操作的机制是 coreos-installer iso custom 子命令，它使用您的配置修改 .iso 文件。同样，PXE 环境的机制是 coreos-installer pxe customize 子命令，它会创建一个包含自定义的新 initramfs 文件。

custom 子命令是一个通用工具，也可以嵌入其他类型的自定义。以下任务是一些更常见的自定义示例：

- 当公司安全策略需要使用时，注入自定义的 CA 证书。
- 在不需要内核参数的情况下配置网络设置。
- 嵌入任意预安装和安装后脚本或二进制文件。

15.4.11.3.7. 自定义 live RHCOS ISO 镜像

您可以使用 coreos-installer iso custom 子命令直接自定义 live RHCOS ISO 镜像。当您引导 ISO 镜像时，会自动应用自定义。

您可以使用此功能配置 ISO 镜像来自动安装 RHCOS。

流程

1. 从 coreos-installer 镜像镜像页面下载 coreos-installer 二进制文件。

2. 从 RHCOS 镜像镜像页面和 Ignition 配置文件检索 RHCOS ISO 镜像，然后运行以下命令来直接将 Ignition 配置注入 ISO 镜像：

```
$ coreos-installer iso customize rhcos-<version>-live.x86_64.iso \
  --dest-ignition bootstrap.ign \ 1
  --dest-device /dev/disk/by-id/scsi-<serial_number> 2
```

1. 从 openshift-installer 安装程序生成的 Ignition 配置文件。

2. 当您指定这个选项时，ISO 镜像会自动运行安装。否则，镜像为安装配置，但不会自动安装，除非您指定了 coreos.inst.install_dev 内核参数。

3. 可选：要删除 ISO 镜像自定义并将镜像返回到其 pristine 状态，请运行：

```
$ coreos-installer iso reset rhcos-<version>-live.x86_64.iso
```

现在，您可以重新自定义 live ISO 镜像，或者在其 pristine 状态中使用它。

应用您的自定义会影响每个后续 RHCOS 引导。

15.4.11.3.7.1. 修改实时安装 ISO 镜像以启用串行控制台

在使用 OpenShift Container Platform 4.12 及更高版本安装的集群中，串行控制台默认被禁用，所有输出都会写入图形控制台。您可以按照以下流程启用串行控制台。

流程

1. 从 coreos-installer 镜像镜像页面下载 coreos-installer 二进制文件。

2. 从 RHCOS image mirror 页中获取 RHCOS ISO 镜像，并运行以下命令来自定义 ISO 镜像，使串行控制台能够接收输出：

```
$ coreos-installer iso customize rhcos-<version>-live.x86_64.iso \
  --dest-ignition <path> \ 1
  --dest-console tty0 \ 2
  --dest-console ttyS0,<options> \ 3
  --dest-device /dev/disk/by-id/scsi-<serial_number> 4
```

1 要安装 Ignition 配置的位置。

2 所需的二级控制台。在这种情况下，是图形控制台。省略这个选项将禁用图形控制台。

3 所需的主控制台。在这种情况下，是串行控制台。options 字段定义 baud 速率和其他设置。此字段的一个常见值为 115200n8。如果没有提供选项，则使用默认内核值 9600n8。有关这个选项格式的更多信息，请参阅 Linux 内核串口控制台文档。

4 要安装到的指定磁盘。如果省略这个选项，ISO 镜像会自动运行安装程序，除非还指定了 coreos.inst.install_dev 内核参数。

注意

--dest-console 选项会影响安装的系统，而不是实时 ISO 系统。要修改 live ISO 系统的控制台，请使用 --live-karg-append 选项并使用 console= 指定控制台。

自定义会被应用，并影响每个后续 ISO 镜像引导。

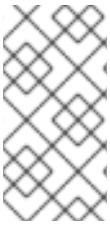
3. 可选：要删除 ISO 镜像自定义并将镜像返回到其原始状态，请运行以下命令：

```
$ coreos-installer iso reset rhcos-<version>-live.x86_64.iso
```

现在，您可以重新自定义 live ISO 镜像，或者在其原始状态中使用它。

15.4.11.3.7.2. 修改实时安装 ISO 镜像以使用自定义证书颁发机构

您可以使用 custom 子命令的 --ignition-ca 标志向 Ignition 提供证书颁发机构(CA)证书。您可以在安装过程中使用 CA 证书，并在置备安装的系统时使用 CA 证书。



注意

自定义 CA 证书会影响 Ignition 获取远程资源的方式，但它们不会影响安装到系统中的证书。

流程

1. 从 coreos-installer 镜像镜像页面下载 coreos-installer 二进制文件。

2. 从 RHCOS 镜像镜像页面检索 RHCOS ISO 镜像，并运行以下命令来自定义 ISO 镜像以用于自定义 CA：

```
$ coreos-installer iso customize rhcos-<version>-live.x86_64.iso --ignition-ca cert.pem
```

重要

coreos.inst.ignition_url 内核参数无法使用 --ignition-ca 标志。您必须使用 --dest-ignition 标志为每个集群创建自定义镜像。

应用自定义 CA 证书会影响每个后续 RHCOS 引导。

15.4.11.3.7.3. 使用自定义网络设置修改实时安装 ISO 镜像

您可以将 NetworkManager 密钥文件嵌入到 live ISO 镜像中，并使用 customize 子命令的 --network-keyfile 标志将其传递给安装的系统。

警告

在创建连接配置文件时，您必须在连接配置文件的文件名中使用 .nmconnection 文件名扩展名。如果不使用 .nmconnection 文件名扩展，集群会将连接配置集应用到 live 环境，但它不会在集群首次启动节点时应用配置，从而导致无法正常工作的设置。

流程

1. 从 coreos-installer 镜像镜像页面下载 coreos-installer 二进制文件。

2. 为绑定接口创建连接配置集。例如，在本地目录中创建 bond0.nmconnection 文件，其内容如下：

```
[connection]
id=bond0
type=bond
interface-name=bond0
multi-connect=1
permissions=

[ethernet]
mac-address-blacklist=

[bond]
miimon=100
mode=active-backup

[ipv4]
method=auto

[ipv6]
method=auto

[proxy]
```

3. 为二级接口创建连接配置集，以添加到绑定中。例如，在本地目录中创建 bond0-proxy-em1.nmconnection 文件，其内容如下：

```
[connection]
id=em1
type=ethernet
interface-name=em1
```

```
master=bond0
multi-connect=1
permissions=
slave-type=bond

[ethernet]
mac-address-blacklist=
```

4. 为二级接口创建连接配置集，以添加到绑定中。例如，在本地目录中创建 bond0-proxy-em2.nmconnection 文件，其内容如下：

```
[connection]
id=em2
type=ethernet
interface-name=em2
master=bond0
multi-connect=1
permissions=
slave-type=bond

[ethernet]
mac-address-blacklist=
```

5. 从 RHCOS 镜像镜像页面检索 RHCOS ISO 镜像，并运行以下命令来使用您配置网络自定义 ISO 镜像：

```
$ coreos-installer iso customize rhcos-<version>-live.x86_64.iso \
  --network-keyfile bond0.nmconnection \
  --network-keyfile bond0-proxy-em1.nmconnection \
  --network-keyfile bond0-proxy-em2.nmconnection
```

网络设置应用于实时系统，并传输到目标系统。

15.4.11.3.7.4. 为 iSCSI 引导设备自定义实时安装 ISO 镜像

您可以设置 iSCSI 目标和 initiator 值，以便使用自定义的 live RHCOS 镜像版本自动挂载、引导和配置。

先决条件

1. 您有一个 iSCSI 目标，您要在其中安装 RHCOS。

流程

1. 从 coreos-installer 镜像镜像页面下载 coreos-installer 二进制文件。

2. 从 RHCOS 镜像镜像页面检索 RHCOS ISO 镜像，并运行以下命令使用以下信息自定义 ISO 镜像：

```
$ coreos-installer iso customize \
  --pre-install mount-iscsi.sh \ 1
  --post-install unmount-iscsi.sh \ 2
  --dest-device /dev/disk/by-path/<IP_address>:<port>-iscsi-<target_iqn>-lun-<lun> \
3
  --dest-ignition config.ign \ 4
  --dest-karg-append rd.iscsi.initiator=<initiator_iqn> \ 5
  --dest-karg-append netroot=<target_iqn> \ 6
  -o custom.iso rhcos-<version>-live.x86_64.iso
```

1 安装之前运行的脚本。它应包含用于挂载 iSCSI 目标以及启用多路径的任何命令的 iscsiadm 命令。

2 安装后运行的脚本。它应包含命令 iscsiadm --mode node --logout=all。

3 目标系统的位置。您必须提供目标门户的 IP 地址、关联的端口号、目标 iSCSI 节点采用 IQN 格式，以及 iSCSI 逻辑单元号(LUN)。

4 目标系统的 Ignition 配置。

5 iSCSI 启动器或客户端，以 IQN 格式的名称。启动器构成连接到 iSCSI 目标的一个会话。

6 iSCSI 目标或服务器的名称（IQN 格式）。

有关 dracut 支持的 iSCSI 选项的更多信息，请参阅 dracut.cmdline 手册页。

15.4.11.3.7.5. 使用 iBFT 为 iSCSI 引导设备自定义实时安装 ISO 镜像

您可以设置 iSCSI 目标和 initiator 值，以便使用自定义的 live RHCOS 镜像版本自动挂载、引导和配置。

先决条件

1. 您有一个 iSCSI 目标，您要在其中安装 RHCOS。
2. 可选：有多路径 iSCSI 目标。

流程

1. 从 coreos-installer 镜像镜像页面下载 coreos-installer 二进制文件。
2. 从 RHCOS 镜像镜像页面检索 RHCOS ISO 镜像，并运行以下命令使用以下信息自定义 ISO 镜像：

```
$ coreos-installer iso customize \
  --pre-install mount-iscsi.sh \ 1
  --post-install unmount-iscsi.sh \ 2
  --dest-device /dev/mapper/mpatha \ 3
  --dest-ignition config.ign \ 4
  --dest-karg-append rd.iscsi.firmware=1 \ 5
  --dest-karg-append rd.multipath=default \ 6
  -o custom.iso rhcos-<version>-live.x86_64.iso
```

1. 安装之前运行的脚本。它应包含用于挂载 iSCSI 目标以及启用多路径的任何命令的 iscsiadm 命令。
2. 安装后运行的脚本。它应包含命令 iscsiadm --mode node --logout=all。
3. 设备的路径。如果连接了多个多路径设备，或者需要明确指定，您使用多路径（/dev/mapper/mpatha），您可以使用 /dev/disk/by-path 中可用的 World Wide Name (WWN) 符号链接。

4

目标系统的 Ignition 配置。

5

iSCSI 参数从 BIOS 固件读取。

6

可选：如果您要启用多路径，请包含此参数。

有关 dracut 支持的 iSCSI 选项的更多信息，请参阅 dracut.cmdline 手册页。

15.4.11.3.8. 自定义 live RHCOS PXE 环境

您可以使用 coreos-installer pxe customize 子命令直接自定义 live RHCOS PXE 环境。当您引导 PXE 环境时，会自动应用自定义。

您可以使用此功能配置 PXE 环境来自动安装 RHCOS。

流程

1. 从 coreos-installer 镜像镜像页面下载 coreos-installer 二进制文件。

2. 从 RHCOS 镜像镜像页面和 Ignition 配置文件获取 RHCOS kernel, initramfs 和 rootfs 文件，然后运行以下命令创建一个包含 Ignition 配置中的自定义的新 initramfs 文件：

```
$ coreos-installer pxe customize rhcos-<version>-live-initramfs.x86_64.img \
  --dest-ignition bootstrap.ign \ 1
  --dest-device /dev/disk/by-id/scsi-<serial_number> \ 2
  -o rhcos-<version>-custom-initramfs.x86_64.img 3
```

1

从 openshift-installer 生成的 Ignition 配置文件。

2

当您指定这个选项时，PXE 环境会自动运行安装。否则，为安装配置了镜像，除非指定了 coreos.inst.install_dev 内核参数，否则不会自动这样做。

3

在 PXE 配置中使用自定义 initramfs 文件。添加 ignition.firstboot 和 ignition.platform.id=metal 内核参数（如果它们尚不存在）。

应用您的自定义会影响每个后续 RHCOS 引导。

15.4.11.3.8.1. 修改实时安装 PXE 环境以启用串行控制台。

在使用 OpenShift Container Platform 4.12 及更高版本安装的集群中，串行控制台默认被禁用，所有输出都会写入图形控制台。您可以按照以下流程启用串行控制台。

流程

1. 从 coreos-installer 镜像镜像页面下载 coreos-installer 二进制文件。

2. 从 RHCOS image mirror 页面获取 kernel, initramfs 和 rootfs 文件，然后运行以下命令来创建新的自定义 initramfs 文件，以便串行控制台接收输出：

```
$ coreos-installer pxe customize rhcos-<version>-live-initramfs.x86_64.img \
  --dest-ignition <path> \ 1
  --dest-console tty0 \ 2
  --dest-console ttyS0,<options> \ 3
  --dest-device /dev/disk/by-id/scsi-<serial_number> \ 4
  -o rhcos-<version>-custom-initramfs.x86_64.img 5
```

1

要安装 Ignition 配置的位置。

2

所需的二级控制台。在这种情况下，是图形控制台。省略这个选项将禁用图形控制台。

3

所需的主控制台。在这种情况下，是串行控制台。options 字段定义 baud 速率和其他设置。此字段的一个常见值为 115200n8。如果没有提供选项，则使用默认内核值 9600n8。有关这个选项格式的更多信息，请参阅 Linux 内核串口控制台文档。

4

要安装到的指定磁盘。如果省略这个选项，PXE 环境会自动运行安装程序，除非还指定了 coreos.inst.install_dev 内核参数。

5

在 PXE 配置中使用自定义 initramfs 文件。添加 ignition.firstboot 和 ignition.platform.id=metal 内核参数（如果它们尚不存在）。

自定义会被应用，并影响 PXE 环境的每个后续引导。

15.4.11.3.8.2. 修改实时安装 PXE 环境以使用自定义证书颁发机构

您可以使用 custom 子命令的 --ignition-ca 标志向 Ignition 提供证书颁发机构(CA)证书。您可以在安装过程中使用 CA 证书，并在置备安装的系统时使用 CA 证书。

注意

自定义 CA 证书会影响 Ignition 获取远程资源的方式，但它们不会影响安装到系统中的证书。

流程

1. 从 coreos-installer 镜像镜像页面下载 coreos-installer 二进制文件。

2. 从 RHCOS 镜像镜像页面获取 RHCOS kernel、initramfs 和 rootfs 文件，并运行以下命令来创建一个新的自定义 initramfs 文件以用于自定义 CA：

```
$ coreos-installer pxe customize rhcos-<version>-live-initramfs.x86_64.img \
  --ignition-ca cert.pem \
  -o rhcos-<version>-custom-initramfs.x86_64.img
```

3. 在 PXE 配置中使用自定义 initramfs 文件。添加 ignition.firstboot 和 ignition.platform.id=metal 内核参数（如果它们尚不存在）。

重要

coreos.inst.ignition_url 内核参数无法使用 --ignition-ca 标志。您必须使用 --dest-ignition 标志为每个集群创建自定义镜像。

应用自定义 CA 证书会影响每个后续 RHCOS 引导。

15.4.11.3.8.3. 使用自定义网络设置修改实时安装 PXE 环境

您可以将 NetworkManager 密钥文件嵌入到 live PXE 环境中，并使用 customize 子命令的 --network-keyfile 标志将其传递给安装的系统。

警告

在创建连接配置文件时，您必须在连接配置文件的文件名中使用 .nmconnection 文件名扩展名。如果不使用 .nmconnection 文件名扩展，集群会将连接配置集应用到 live 环境，但它不会在集群首次启动节点时应用配置，从而导致无法正常工作的设置。

流程

1. 从 coreos-installer 镜像镜像页面下载 coreos-installer 二进制文件。

2. 为绑定接口创建连接配置集。例如，在本地目录中创建 bond0.nmconnection 文件，其内容如下：

```
[connection]
id=bond0
type=bond
interface-name=bond0
multi-connect=1
permissions=

[ethernet]
mac-address-blacklist=

[bond]
miimon=100
```

```
mode=active-backup

[ipv4]
method=auto

[ipv6]
method=auto

[proxy]
```

3. 为二级接口创建连接配置集，以添加到绑定中。例如，在本地目录中创建 bond0-proxy-em1.nmconnection 文件，其内容如下：

```
[connection]
id=em1
type=ethernet
interface-name=em1
master=bond0
multi-connect=1
permissions=
slave-type=bond

[ethernet]
mac-address-blacklist=
```

4. 为二级接口创建连接配置集，以添加到绑定中。例如，在本地目录中创建 bond0-proxy-em2.nmconnection 文件，其内容如下：

```
[connection]
id=em2
type=ethernet
interface-name=em2
master=bond0
multi-connect=1
permissions=
slave-type=bond

[ethernet]
mac-address-blacklist=
```

5. 从 RHCOS 镜像镜像页面获取 RHCOS kernel、initramfs 和 rootfs 文件，并运行以下命令来创建一个新的自定义 initramfs 文件，它包括您的配置网络：

```
$ coreos-installer pxe customize rhcos-<version>-live-initramfs.x86_64.img \
   --network-keyfile bond0.nmconnection \
   --network-keyfile bond0-proxy-em1.nmconnection \
   --network-keyfile bond0-proxy-em2.nmconnection \
   -o rhcos-<version>-custom-initramfs.x86_64.img
```

6. 在 PXE 配置中使用自定义 initramfs 文件。添加 ignition.firstboot 和 ignition.platform.id=metal 内核参数（如果它们尚不存在）。

网络设置应用于实时系统，并传输到目标系统。

15.4.11.3.8.4. 为 iSCSI 引导设备自定义实时安装 PXE 环境

您可以设置 iSCSI 目标和 initiator 值，以便使用自定义的 live RHCOS 镜像版本自动挂载、引导和配置。

先决条件

1. 您有一个 iSCSI 目标，您要在其中安装 RHCOS。

流程

1. 从 coreos-installer 镜像镜像页面下载 coreos-installer 二进制文件。

2. 从 RHCOS image mirror 页获取 RHCOS kernel、initramfs 和 rootfs 文件，并运行以下命令来使用以下信息创建新的自定义 initramfs 文件：

```
$ coreos-installer pxe customize \
  --pre-install mount-iscsi.sh \ 1
  --post-install unmount-iscsi.sh \ 2
  --dest-device /dev/disk/by-path/<IP_address>:<port>-iscsi-<target_iqn>-lun-<lun> \
3
  --dest-ignition config.ign \ 4
  --dest-karg-append rd.iscsi.initiator=<initiator_iqn> \ 5
  --dest-karg-append netroot=<target_iqn> \ 6
  -o custom.img rhcos-<version>-live-initramfs.x86_64.img
```

1. 安装之前运行的脚本。它应包含用于挂载 iSCSI 目标以及启用多路径的任何命令的 iscsiadm 命令。

2. 安装后运行的脚本。它应包含命令 iscsiadm --mode node --logout=all。

3.

目标系统的位置。您必须提供目标门户的 IP 地址、关联的端口号、目标 iSCSI 节点采用 IQN 格式，以及 iSCSI 逻辑单元号(LUN)。

4 目标系统的 Ignition 配置。

5 iSCSI 启动器或客户端，以 IQN 格式的名称。启动器构成连接到 iSCSI 目标的一个会话。

6 iSCSI 目标或服务器的名称（IQN 格式）。

有关 dracut 支持的 iSCSI 选项的更多信息，请参阅 dracut.cmdline 手册页。

15.4.11.3.8.5. 使用 iBFT 为 iSCSI 引导设备自定义实时安装 PXE 环境

您可以设置 iSCSI 目标和 initiator 值，以便使用自定义的 live RHCOS 镜像版本自动挂载、引导和配置。

先决条件

1. 您有一个 iSCSI 目标，您要在其中安装 RHCOS。
2. 可选：有多路径 iSCSI 目标。

流程

1. 从 coreos-installer 镜像镜像页面下载 coreos-installer 二进制文件。
2. 从 RHCOS image mirror 页获取 RHCOS kernel、initramfs 和 rootfs 文件，并运行以下命令来使用以下信息创建新的自定义 initramfs 文件：

```
$ coreos-installer pxe customize \
  --pre-install mount-iscsi.sh \ 1
```

```
--post-install unmount-iscsi.sh \ 2
--dest-device /dev/mapper/mpatha \ 3
--dest-ignition config.ign \ 4
--dest-karg-append rd.iscsi.firmware=1 \ 5
--dest-karg-append rd.multipath=default \ 6
-o custom.img rhcos-<version>-live-initramfs.x86_64.img
```

1

安装之前运行的脚本。它应包含用于挂载 iSCSI 目标的 iscsiadm 命令。

2

安装后运行的脚本。它应包含命令 iscsiadm --mode node --logout=all。

3

设备的路径。如果连接了多个多路径设备，或者需要明确指定，您使用多路径（/dev/mapper/mpatha），您可以使用 /dev/disk/by-path 中可用的 World Wide Name (WWN) 符号链接。

4

目标系统的 Ignition 配置。

5

iSCSI 参数从 BIOS 固件读取。

6

可选：如果您要启用多路径，请包含此参数。

有关 dracut 支持的 iSCSI 选项的更多信息，请参阅 dracut.cmdline 手册页。

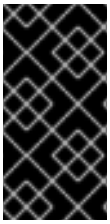
15.4.11.3.9. 高级 RHCOS 安装参考

本节演示了网络配置和其他高级选项，允许您修改 Red Hat Enterprise Linux CoreOS(RHCOS)手动安装过程。下表描述了您可以用于 RHCOS live 安装程序和 coreos-installer 命令的内核参数和命令行选项。

15.4.11.3.9.1. ISO 安装的网络和绑定选项

如果从 ISO 镜像安装 RHCOS，您可以在引导镜像时手动添加内核参数，以便为节点配置网络。如

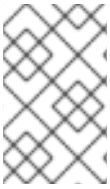
果没有指定网络参数，当 RHCOS 检测到需要网络来获取 Ignition 配置文件时，在 initramfs 中激活 DHCP。



重要

在手动添加网络参数时，还必须添加 rd.neednet=1 内核参数，以便在 initramfs 中启动网络。

以下信息提供了在 RHCOS 节点上为 ISO 安装配置网络和绑定的示例。示例描述了如何使用 ip=、name server = 和 bond= 内核参数。



注意

添加内核参数时顺序非常重要： ip=、 name server=，然后 bond=。

网络选项在系统引导过程中传递给 dracut 工具。有关 dracut 支持的网络选项的更多信息，请参阅 dracut.cmdline 手册页。

以下示例是 ISO 安装的网络选项。

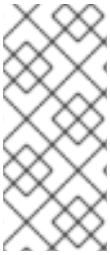
配置 DHCP 或静态 IP 地址

要配置 IP 地址，可使用 DHCP(ip=dhcp)或设置单独的静态 IP 地址(ip=<host_ip>)。如果设置静态 IP，则必须在每个节点上识别 DNS 服务器 IP 地址（名称服务器=<dns_ip>）。以下示例集：

- 节点的 IP 地址为 10.10.10.2
- 网关地址为 10.10.10.254
- 子网掩码为 255.255.255.0
- 到 core0.example.com 的主机名

- DNS 服务器地址为 4.4.4.41
- 自动配置值为 none。当以静态方式配置 IP 网络时，不需要自动配置。

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
nameserver=4.4.4.41
```



注意

当您使用 DHCP 为 RHCOS 机器配置 IP 寻址时，机器还通过 DHCP 获取 DNS 服务器信息。对于基于 DHCP 的部署，您可以通过 DHCP 服务器配置定义 RHCOS 节点使用的 DNS 服务器地址。

配置没有静态主机名的 IP 地址

您可以在不分配静态主机名的情况下配置 IP 地址。如果用户没有设置静态主机名，则会提取并通过反向 DNS 查找自动设置。要在没有静态主机名的情况下配置 IP 地址，请参考以下示例：

- 节点的 IP 地址为 10.10.10.2
- 网关地址为 10.10.10.254
- 子网掩码为 255.255.255.0
- DNS 服务器地址为 4.4.4.41
- 自动配置值为 none。当以静态方式配置 IP 网络时，不需要自动配置。

```
ip=10.10.10.2::10.10.10.254:255.255.255.0::enp1s0:none
nameserver=4.4.4.41
```

指定多个网络接口

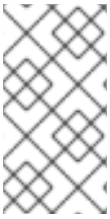
您可以通过设置多个 ip= 条目来指定多个网络接口。

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
ip=10.10.10.3::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none
```

-

配置默认网关和路由

可选：您可以通过设置 a rd.route= 值来配置到额外网络的路由。



注意

当您配置一个或多个网络时，需要一个默认网关。如果额外网络网关与主要网络网关不同，则默认网关必须是主要网络网关。

- 运行以下命令来配置默认网关：

```
ip=::10.10.10.254::::
```

- 输入以下命令为额外网络配置路由：

```
rd.route=20.20.20.0/24:20.20.20.254:enp2s0
```

在单个接口中禁用 DHCP

您可以在单一接口中禁用 DHCP，例如当有两个或者多个网络接口时，且只有一个接口被使用。在示例中，enp1s0 接口具有一个静态网络配置，而 enp2s0 禁用了 DHCP，不使用它：

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
ip=::::core0.example.com:enp2s0:none
```

合并 DHCP 和静态 IP 配置

您可以将系统上的 DHCP 和静态 IP 配置与多个网络接口合并，例如：

```
ip=enp1s0:dhcp
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none
```

在独立接口上配置 VLAN

可选： 您可以使用 vlan= 参数在单个接口上配置 VLAN。

- 要在网络接口中配置 VLAN 并使用静态 IP 地址，请运行以下命令：

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0.100:none
vlan=enp2s0.100:enp2s0
```

- 要在网络接口中配置 VLAN 并使用 DHCP，请运行以下命令：

```
ip=enp2s0.100:dhcp
vlan=enp2s0.100:enp2s0
```

提供多个 DNS 服务器

您可以通过为每个服务器添加一个 nameserver= 条目来提供多个 DNS 服务器，例如

```
nameserver=1.1.1.1
nameserver=8.8.8.8
```

将多个网络接口绑定到一个接口

可选：您可以使用 bond= 选项将多个网络接口绑定到一个接口。请参见以下示例：

- 配置绑定接口的语法为：bond=<name>[:<network_interfaces>][:options]

  <name> 是绑定设备名称 (bond0)、<network_interfaces> 代表以逗号分隔的物理（以太网）接口列表(em1,em2)，*options* 是用逗号分开的绑定选项列表。输入 modinfo bonding 查看可用选项。

- 当使用 bond= 创建绑定接口时，您必须指定如何分配 IP 地址以及绑定接口的其他信息。

  - 要将绑定接口配置为使用 DHCP，请将绑定的 IP 地址设置为 dhcp。例如：

    ```
    bond=bond0:em1,em2:mode=active-backup
    ip=bond0:dhcp
    ```

  - 要将绑定接口配置为使用静态 IP 地址，请输入您需要的特定 IP 地址和相关信息。例如：

    ```
    bond=bond0:em1,em2:mode=active-backup
    ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0:none
    ```

将多个 SR-IOV 网络接口绑定到双端口 NIC 接口

重要

支持与为 SR-IOV 设备启用 NIC 分区关联的第 1 天操作只是一个技术预览功能。技术预览功能不受红帽产品服务等级协议（SLA）支持，且功能可能并不完整。红帽不推荐在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。

有关红帽技术预览功能支持范围的更多信息，请参阅技术预览功能支持范围。

可选： 您可以使用 bond= 选项将多个 SR-IOV 网络接口绑定到双端口 NIC 接口。

在每个节点上，您必须执行以下任务：

1. 按照管理 SR-IOV 设备中的指导创建 SR-IOV 虚拟功能(VF)。按照"将 SR-IOV 网络设备附加到虚拟机"部分中的步骤操作。

2. 创建绑定，将所需的 VF 附加到绑定，并根据配置网络绑定的指导设置绑定链接状态。按照任何描述的步骤创建绑定。

以下示例演示了您必须使用的语法：

- 配置绑定接口的语法为：bond=<name>[:<network_interfaces>][:options]

  <name> 是绑定设备名称 (bond0)、<network_interfaces> 由内核中已知的名称来代表虚拟功能(VF)，并显示在 ip link 命令的输出中 (eno1f0,eno2f0)， *options* 是以逗号分隔的绑定选项列表。输入 modinfo bonding 查看可用选项。

- 当使用 bond= 创建绑定接口时，您必须指定如何分配 IP 地址以及绑定接口的其他信息。

  - 要将绑定接口配置为使用 DHCP，请将绑定的 IP 地址设置为 dhcp。例如：

    ```
    bond=bond0:eno1f0,eno2f0:mode=active-backup
    ip=bond0:dhcp
    ```

  -

要将绑定接口配置为使用静态 IP 地址，请输入您需要的特定 IP 地址和相关信息。例如：

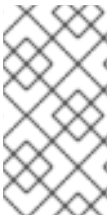
```
bond=bond0:eno1f0,eno2f0:mode=active-backup
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0:none
```

使用网络团队

可选：您可以使用 team= 参数来将网络团队用作绑定的替代选择：

- 配置组接口的语法为：team=name[:network_interfaces]

name 是组设备名称(team0)，*network_interfaces* 代表以逗号分隔的物理（以太网）接口（em1、em2）列表。



注意

当 RHCOS 切换到即将推出的 RHEL 版本时，团队(team)功能被计划弃用。如需更多信息，请参阅红帽知识库文章。

使用以下示例配置网络团队：

```
team=team0:em1,em2
ip=team0:dhcp
```

15.4.11.3.9.2. ISO 和 PXE 安装的 coreos-installer 选项

从 ISO 镜像引导 RHCOS live 环境后，您可以通过 在命令提示符下运行 coreos-installer install <options> <device > 来安装 RHCOS。

下表显示了您可以传递给 coreos-installer 命令的子命令、选项和参数。

表 15.40. coreos-installer 子命令、命令行选项和参数

coreos-installer install 子命令	
子命令	*描述*
$ coreos-installer install <options> <device>	在 ISO 镜像中嵌入 Ignition 配置。

coreos-installer install 子命令选项	
选项	***描述***
-u, --image-url <url>	手动指定镜像 URL。
-f,--image-file <path>	手动指定本地镜像文件。用于调试。
-i, --ignition-file <path>	从文件中嵌入 Ignition 配置。
-i,--ignition-url <URL>	从 URL 嵌入 Ignition 配置。
--ignition-hash <digest>	Ignition 配置 **的 type-value** 的摘要值。
-p, --platform <name>	覆盖已安装系统的 Ignition 平台 ID。
--console <spec>	为安装的系统设置内核和引导装载程序控制台。有关 **<spec>** 格式的更多信息，请参阅 Linux 内核串口控制台文档。
--append-karg <arg>...	将默认内核参数附加到安装的系统中。
--delete-karg <arg>...	从安装的系统中删除默认内核参数。
-n, --copy-network	从安装环境中复制网络配置。  **重要** **copy-network** 选项仅复制在 **/etc/NetworkManager/system-connections** 下找到的网络配置。特别是，它不会复制系统主机名。
--network-dir <path>	用于 **-n**。默认为 **/etc/NetworkManager/system-connections/**。
--save-partlabel <lx>..	使用这个标签 glob 保存分区。
--save-partindex <id>...	使用这个数值或范围保存分区。
--insecure	跳过 RHCOS 镜像签名验证。
--insecure-ignition	允许没有 HTTPS 或 hash 的 Ignition URL。
--architecture <name>	目标 CPU 架构.有效值为 **x86_64** 和 **aarch64**。
--preserve-on-error	出错时不要清除分区表。

-h,--help	打印帮助信息.
coreos-installer install 子命令参数	
参数	***描述***
<device>	目标设备.
coreos-installer ISO 子命令	
子命令	***描述***
$ coreos-installer iso customize <options> <ISO_image>	自定义 RHCOS live ISO 镜像。
coreos-installer iso reset <options> <ISO_image>	将 RHCOS live ISO 镜像恢复到默认设置。
coreos-installer iso ignition remove <options> <ISO_image>	从 ISO 镜像中删除嵌入的 Ignition 配置。
coreos-installer ISO customize 子命令选项	
选项	***描述***
--dest-ignition <path>	将指定的 Ignition 配置文件合并到目标系统的新配置片段中。
--dest-console <spec>	为目标系统指定内核和引导装载程序控制台。
--dest-device <path>	安装并覆盖指定的目标设备。
--dest-karg-append <arg>	为每个目标系统引导添加一个内核参数。
--dest-karg-delete <arg>	从目标系统的每个引导中删除内核参数。
--network-keyfile <path>	使用指定的 NetworkManager 密钥文件进行实时和目标系统配置网络。
--ignition-ca <path>	指定要被 Ignition 信任的额外 TLS 证书颁发机构。
--pre-install <path>	在安装之前运行指定的脚本。
--post-install <path>	安装后运行指定的脚本。
--installer-config <path>	应用指定的安装程序配置文件。

--live-ignition <path>	将指定的 Ignition 配置文件合并到实时环境的新配置片段中。
--live-karg-append <arg>	为每个实时环境引导添加一个内核参数。
--live-karg-delete <arg>	从实时环境每次引导时删除内核参数。
--live-karg-replace <k=o=n>	在每次启动 live 环境时替换内核参数，格式为 **key=old=new**。
-f,--force	覆盖现有的 Ignition 配置。
-o,--output <path>	将 ISO 写入到新的输出文件。
-h,--help	打印帮助信息.

coreos-installer PXE 子命令

子命令	*描述*
请注意，并非所有子命令都接受所有这些选项。	
coreos-installer pxe customize <options> <path>	自定义 RHCOS live PXE 引导配置。
coreos-installer pxe ignition wrap <options>	在镜像中嵌套 Ignition 配置。
coreos-installer pxe ignition unwrap <options> <image_name>	在镜像中显示嵌套的 Ignition 配置。

coreos-installer PXE customize 子命令选项

选项	*描述*
请注意，并非所有子命令都接受所有这些选项。	
--dest-ignition <path>	将指定的 Ignition 配置文件合并到目标系统的新配置片段中。
--dest-console <spec>	为目标系统指定内核和引导装载程序控制台。
--dest-device <path>	安装并覆盖指定的目标设备。
--network-keyfile <path>	使用指定的 NetworkManager 密钥文件进行实时和目标系统配置网络。
--ignition-ca <path>	指定要被 Ignition 信任的额外 TLS 证书颁发机构。

--pre-install <path>	在安装之前运行指定的脚本。
post-install <path>	安装后运行指定的脚本。
--installer-config <path>	应用指定的安装程序配置文件。
--live-ignition <path>	将指定的 Ignition 配置文件合并到实时环境的新配置片段中。
-o, --output <path>	将 initramfs 写入一个新输出文件。  **注意** PXE 环境需要这个选项。
-h,--help	打印帮助信息.

15.4.11.3.9.3. coreos.inst 引导选项用于 ISO 或 PXE 安装

您可以通过将 coreos.inst boot 参数传递给 RHCOS live 安装程序，在引导时自动调用 coreos-installer 选项。这些是在标准引导参数之外提供的。

- 对于 ISO 安装，可以通过在启动加载器菜单中中断自动引导来添加 coreos.inst 选项。您可以在突出显示 RHEL CoreOS(Live) 菜单选项时按 TAB 来中断自动引导。

- 对于 PXE 或 iPXE 安装，在引导 RHCOS live 安装程序前，coreos.inst 选项必须添加到 APPEND 行。

下表显示了用于 ISO 和 PXE 安装的 RHCOS live 安装程序 coreos.inst 引导选项。

表 15.41. coreos.inst 引导选项

参数	描述
coreos.inst.install_dev	必需。要安装到的系统中的块设备。建议您使用完整路径，如 **/dev/sda**，但 **允许使用**。
coreos.inst.ignition_url	可选：嵌入到安装的系统中的 Ignition 配置的 URL。如果没有指定 URL，则不会嵌入 Ignition 配置。仅支持 HTTP 和 HTTPS 协议。

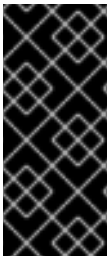
参数	描述
coreos.inst.save_partlabel	可选：在安装过程中要保留的分区分离标签。允许使用 glob 风格的通配符。指定分区不需要存在。
coreos.inst.save_partindex	可选：在安装过程中压缩要保留的分区索引。允许 **m-n** 范围，**m** 或 **n** 可以被省略。指定分区不需要存在。
coreos.inst.insecure	可选：将 **coreos.inst.image_url** 指定的 OS 镜像提交取消签名。
coreos.inst.image_url	可选：下载并安装指定的 RHCOS 镜像。 ● 这个参数不应该在生产环境中使用，而是只用于调试目的。 ● 虽然此参数可用于安装与 live 介质不匹配的 RHCOS 版本，但建议您使用与您要安装版本匹配的介质。 ● 如果您使用 **coreos.inst.image_url**，还必须使用 **coreos.inst.insecure**。这是因为，裸机介质没有为 OpenShift Container Platform 进行 GPG 签名。 ● 仅支持 HTTP 和 HTTPS 协议。
coreos.inst.skip_reboot	可选：安装后系统不会重启。安装完成后，您将收到提示，提示您检查在安装过程中发生的情况。这个参数不应该在生产环境中使用，而是只用于调试目的。
coreos.inst.platform_id	可选：安装 RHCOS 镜像的平台的 Ignition 平台 ID。默认为 **metal**。这个选项决定是否从云供应商（如 VMware）请求 Ignition 配置。例如：**coreos.inst.platform_id=vmware**。
ignition.config.url	可选：用于实时引导的 Ignition 配置的 URL。例如，这可用于自定义调用 **coreos-installer** 的方式，或者用于在安装前或安装后运行代码。这与 **coreos.inst.ignition_url**（这是已安装系统的 Ignition 配置）不同。

15.4.11.4. 在 RHCOS 上启用带有内核参数的多路径

RHCOS 支持主磁盘上的多路径，支持更强大的硬件故障弹性，以获得更高的主机可用性。

您可以在安装时为 OpenShift Container Platform 4.8 或更高版本置备的节点启用多路径。虽然安装后支持可以通过机器配置激活多路径来实现，但建议在安装过程中启用多路径。

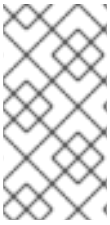
在任何 I/O 到未优化路径会导致 I/O 系统错误的设置中，您必须在安装时启用多路径。



重要

在 IBM Z® 和 IBM® LinuxONE 中，您只能在在安装过程中为它配置集群时启用多路径。如需更多信息，请参阅在 *IBM Z® 和 IBM® LinuxONE 上安装使用 z/VM 的集群"安装 RHCOS 并启动 OpenShift Container Platform bootstrap 过程"*。

以下流程在安装时启用多路径，并在 coreos-installer install 命令中附加内核参数，以便安装的系统本身将使用从第一次引导开始的多路径。



注意

OpenShift Container Platform 不支持在从 4.6 或更早版本升级的节点上启用多路径作为 2 天的活动。

流程

1. 要启用多路径并启动 multipathd 守护进程，请在安装主机上运行以下命令：

```
$ mpathconf --enable && systemctl start multipathd.service
```

- 可选：如果引导 PXE 或 ISO，则可以通过从内核命令行添加 rd.multipath=default 来启用多路径。

2. 通过调用 coreos-installer 程序附加内核参数：

- 如果只有一个多路径设备连接到计算机，则应在路径 /dev/mapper/mpatha 上可用。例如：

```
$ coreos-installer install /dev/mapper/mpatha \ 1
--append-karg rd.multipath=default \
--append-karg root=/dev/disk/by-label/dm-mpath-root \
--append-karg rw
```

1

表示单一多路径设备的路径。

- 如果有多个多路径设备连接到计算机，或者更为明确，而不是使用 /dev/mapper/mpatha，则建议使用 /dev/disk/by-id 中可用的 World Wide Name(WWN)符号链接。例如：

```
$ coreos-installer install /dev/disk/by-id/wwn-<wwn_ID> \ 1
--append-karg rd.multipath=default \
--append-karg root=/dev/disk/by-label/dm-mpath-root \
--append-karg rw
```

1

表示目标多路径设备的 WWN ID。例如： 0xx194e957fcedb4841。

当使用特殊 coreos.inst.* 参数指示 live 安装程序时，这个符号链接也可以用作 coreos.inst.install_dev 内核参数。如需更多信息，请参阅"安装 RHCOS 和启动 OpenShift Container Platform bootstrap 过程"。

3. 前往其中一个 worker 节点并列出内核命令行参数（主机上的 /proc/cmdline 中），以检查内核参数是否正常工作：

```
$ oc debug node/ip-10-0-141-105.ec2.internal
```

输出示例

```
Starting pod/ip-10-0-141-105ec2internal-debug ...
To use host binaries, run `chroot /host`

sh-4.2# cat /host/proc/cmdline
...
rd.multipath=default root=/dev/disk/by-label/dm-mpath-root
...

sh-4.2# exit
```

您应看到添加的内核参数。

15.4.11.5. 在 iSCSI 引导设备中手动安装 RHCOS

您可以在 iSCSI 目标上手动安装 RHCOS。

先决条件

1. 您在 RHCOS live 环境中。
2. 您有一个要在其上安装 RHCOS 的 iSCSI 目标。

流程

1. 运行以下命令，从 live 环境中挂载 iSCSI 目标：

```
$ iscsiadm \
    --mode discovery \
    --type sendtargets
    --portal <IP_address> \ 1
    --login
```

1. 目标门户的 IP 地址。

2. 运行以下命令并使用必要的内核参数将 RHCOS 安装到 iSCSI 目标上，例如：

```
$ coreos-installer install \
    /dev/disk/by-path/ip-<IP_address>:<port>-iscsi-<target_iqn>-lun-<lun> \ 1
    --append-karg rd.iscsi.initiator=<initiator_iqn> \ 2
    --append.karg netroot=<target_iqn> \ 3
    --console ttyS0,115200n8
    --ignition-file <path_to_file>
```

1. 您要安装到的位置。您必须提供目标门户的 IP 地址、关联的端口号、目标 iSCSI 节点采用 IQN 格式，以及 iSCSI 逻辑单元号(LUN)。

2 iSCSI 启动器或客户端，以 IQN 格式的名称。启动器构成连接到 iSCSI 目标的一个会话。

3 iSCSI 目标或服务器的名称（IQN 格式）。

有关 dracut 支持的 iSCSI 选项的更多信息，请参阅 dracut.cmdline 手册页。

3. 使用以下命令卸载 iSCSI 磁盘：

```
$ iscsiadm --mode node --logoutall=all
```

此过程也可以使用 coreos-installer iso customize 或 coreos-installer pxe customize 子命令来执行。

15.4.11.6. 使用 iBFT 在 iSCSI 引导设备中安装 RHCOS

在一个完全无盘机器上，也可以通过 iBFT. iSCSI 多路径传递 iSCSI 目标和启动器值。

先决条件

1. 您在 RHCOS live 环境中。
2. 您有一个 iSCSI 目标，您要在其中安装 RHCOS。
3. 可选：有多路径 iSCSI 目标。

流程

1. 运行以下命令，从 live 环境中挂载 iSCSI 目标：

```
$ iscsiadm \
    --mode discovery \
```

```
--type sendtargets
--portal <IP_address> \ 1
--login
```

1 目标门户的 IP 地址。

2. 可选：使用以下命令启用多路径并启动守护进程：

```
$ mpathconf --enable && systemctl start multipathd.service
```

3. 运行以下命令并使用必要的内核参数将 RHCOS 安装到 iSCSI 目标上，例如：

```
$ coreos-installer install \
 /dev/mapper/mpatha \ 1
 --append-karg rd.iscsi.firmware=1 \ 2
 --append-karg rd.multipath=default \ 3
 --console ttyS0 \
 --ignition-file <path_to_file>
```

1 单一多路径设备的路径。如果连接了多个多路径设备，或者需要明确指定，您可以使用 /dev/disk/by-path 中可用的 World Wide Name (WWN) 符号链接。

2 iSCSI 参数从 BIOS 固件读取。

3 可选：如果您要启用多路径，请包含此参数。

有关 dracut 支持的 iSCSI 选项的更多信息，请参阅 dracut.cmdline 手册页。

4. 卸载 iSCSI 磁盘：

```
$ iscsiadm --mode node --logout=all
```

此过程也可以使用 coreos-installer iso customize 或 coreos-installer pxe customize 子命令来执行。

15.4.12. 等待 bootstrap 过程完成

OpenShift Container Platform bootstrap 过程在集群节点首次引导到安装到磁盘的持久 RHCOS 环境后开始。通过 Ignition 配置文件提供的配置信息用于初始化 bootstrap 过程并在机器上安装 OpenShift Container Platform。您必须等待 bootstrap 过程完成。

先决条件

- 已为集群创建 Ignition 配置文件。
- 您已配置了适当的网络、DNS 和负载平衡基础架构。
- 已获得安装程序，并为集群生成 Ignition 配置文件。
- 已在集群机器上安装 RHCOS，并提供 OpenShift Container Platform 安装程序生成的 Ignition 配置文件。

流程

1. 监控 bootstrap 过程：

```
$ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \ 1
    --log-level=info 2
```

1. 对于 <installation_directory>，请指定安装文件保存到的目录的路径。
2. 要查看不同的安装详情，请指定 warn、debug 或 error，而不是 info。

输出示例

```
INFO Waiting up to 30m0s for the Kubernetes API at
https://api.test.example.com:6443...
INFO API v1.29.4 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
```

当 Kubernetes API 服务器提示已在 control plane 机器上引导它时，该命令会成功。

2. bootstrap 过程完成后，从负载均衡器中删除 bootstrap 机器。

重要

此时您必须从负载均衡器中删除 bootstrap 机器。您还可以删除或重新格式化 bootstrap 机器本身。

其他资源

- 如需有关 监控安装日志的更多信息，请参阅监控安装进度，并在出现安装问题时检索诊断数据。

15.4.13. 使用 CLI 登录集群

您可以通过导出集群 kubeconfig 文件，以默认系统用户身份登录集群。kubeconfig 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。
- 已安装 oc CLI。

流程

1. 导出 kubeadmin 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 <installation_directory>，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 oc 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

15.4.14. 批准机器的证书签名请求

当您将机器添加到集群时，会为您添加的每台机器生成两个待处理证书签名请求(CSR)。您必须确认这些 CSR 已获得批准，或根据需要自行批准。必须首先批准客户端请求，然后批准服务器请求。

先决条件

- 您已将机器添加到集群中。

流程

1. 确认集群可以识别这些机器：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0  Ready    master  63m  v1.29.4
master-1  Ready    master  63m  v1.29.4
master-2  Ready    master  64m  v1.29.4
```

输出中列出了您创建的所有机器。

注意

在有些 CSR 被批准前，前面的输出可能不包括计算节点（也称为 worker 节点）。

2. 检查待处理的 CSR，并确保添加到集群中的每台机器都有 Pending 或 Approved 状态的客户端请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE    REQUESTOR                                                   CONDITION
csr-8b2br   15m    system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
csr-8vnps   15m    system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
...
```

在本例中，两台机器加入集群。您可能会在列表中看到更多已批准的 CSR。

3. 如果 CSR 没有获得批准，在您添加的机器的所有待处理 CSR 都处于 Pending 状态后，请批准集群机器的 CSR：

注意

由于 CSR 会自动轮转，因此请在将机器添加到集群后一小时内批准您的 CSR。如果没有在一小时内批准它们，证书将会轮转，每个节点会存在多个证书。您必须批准所有这些证书。批准客户端 CSR 后，Kubelet 为服务证书创建一个二级 CSR，这需要手动批准。然后，如果 Kubelet 请求具有相同参数的新证书，则后续提供证书续订请求由 machine-approver 自动批准。

注意

对于在未启用机器 API 的平台上运行的集群，如裸机和其他用户置备的基础架构，您必须实施一种方法来自动批准 kubelet 提供证书请求(CSR)。如果没有批准请求，则 oc exec、ocrsh 和 oc logs 命令将无法成功，因为 API 服务器连接到 kubelet 时需要服务证书。与 Kubelet 端点联系的任何操作都需要此证书批准。该方法必须监视新的 CSR，确认 CSR 由 system: node 或 system:admin 组中的 node-bootstrapper 服务帐户提交，并确认节点的身份。

- 要单独批准，请对每个有效的 CSR 运行以下命令：

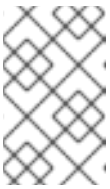
```
$ oc adm certificate approve <csr_name> 1
```

1

<csr_name> 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}
{{"\n"}}{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```



注意

在有些 CSR 被批准前，一些 Operator 可能无法使用。

4. 现在，您的客户端请求已被批准，您必须查看添加到集群中的每台机器的服务器请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE     REQUESTOR                                                   CONDITION
csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal       Pending
csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal       Pending
...
```

5. 如果剩余的 CSR 没有被批准，且处于 Pending 状态，请批准集群机器的 CSR：

- 要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 <csr_name> 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}
{{"\n"}}{{end}}{{end}}' | xargs oc adm certificate approve
```

6. 批准所有客户端和服务器 CSR 后，机器将 处于 Ready 状态。运行以下命令验证：

```
$ oc get nodes
```

输出示例

```
NAME       STATUS   ROLES   AGE  VERSION
master-0   Ready    master  73m  v1.29.4
master-1   Ready    master  73m  v1.29.4
master-2   Ready    master  74m  v1.29.4
worker-0   Ready    worker  11m  v1.29.4
worker-1   Ready    worker  11m  v1.29.4
```

注意

批准服务器 CSR 后可能需要几分钟时间让机器过渡到 Ready 状态。

其他信息

- 如需有关 CSR 的更多信息，请参阅 证书签名请求。

15.4.15. 初始 Operator 配置

在 control plane 初始化后，您必须立即配置一些 Operator，以便它们都可用。

先决条件

- 您的 control plane 已初始化。

流程

1. 观察集群组件上线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION   AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.16.0    True        False         False      19m
baremetal                                  4.16.0    True        False         False      37m
cloud-credential                           4.16.0    True        False         False      40m
cluster-autoscaler                         4.16.0    True        False         False      37m
config-operator                            4.16.0    True        False         False      38m
console                                    4.16.0    True        False         False      26m
csi-snapshot-controller                    4.16.0    True        False         False      37m
dns                                        4.16.0    True        False         False      37m
etcd                                       4.16.0    True        False         False      36m
image-registry                             4.16.0    True        False         False      31m
ingress                                    4.16.0    True        False         False      30m
```

```
insights                                   4.16.0    True        False         False      31m
kube-apiserver                             4.16.0    True        False         False      26m
kube-controller-manager                    4.16.0    True        False         False      36m
kube-scheduler                             4.16.0    True        False         False      36m
kube-storage-version-migrator              4.16.0    True        False         False      37m
machine-api                                4.16.0    True        False         False      29m
machine-approver                           4.16.0    True        False         False      37m
machine-config                             4.16.0    True        False         False      36m
marketplace                                4.16.0    True        False         False      37m
monitoring                                 4.16.0    True        False         False      29m
network                                    4.16.0    True        False         False      38m
node-tuning                                4.16.0    True        False         False      37m
openshift-apiserver                        4.16.0    True        False         False      32m
openshift-controller-manager               4.16.0    True        False         False      30m
openshift-samples                          4.16.0    True        False         False      32m
operator-lifecycle-manager                 4.16.0    True        False         False      37m
operator-lifecycle-manager-catalog         4.16.0    True        False         False      37m
operator-lifecycle-manager-packageserver   4.16.0    True        False         False      32m
service-ca                                 4.16.0    True        False         False      38m
storage                                    4.16.0    True        False         False      37m
```

2. 配置不可用的 Operator。

其他资源

- 如需了解 在 OpenShift Container Platform 安装失败时收集数据的详细信息，请参阅从失败安装收集日志。

- 如需了解在集群中检查 Operator pod 健康状况的步骤，并收集 Operator 日志以进行诊断，请参阅故障排除 Operator 问题。

15.4.15.1. 禁用默认的 OperatorHub 目录源

在 OpenShift Container Platform 安装过程中，默认为 OperatorHub 配置由红帽和社区项目提供的源内容的 operator 目录。在受限网络环境中，必须以集群管理员身份禁用默认目录。

流程

- 通过在 OperatorHub 对象中添加 disableAllDefaultSources: true 来 禁用默认目录的源：

```
$ oc patch OperatorHub cluster --type json \
    -p '[{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]'
```

提示

或者，您可以使用 Web 控制台管理目录源。在 Administration → Cluster Settings → Configuration → OperatorHub 页面中，点 Sources 选项卡，您可以在其中创建、更新、删除、禁用和启用单独的源。

15.4.15.2. 镜像 registry 存储配置

对于不提供默认存储的平台，Image Registry Operator 最初不可用。安装后，您必须将 registry 配置为使用存储，以便 Registry Operator 可用。

显示配置生产集群所需的持久性卷的说明。如果适用，显示有关将空目录配置为存储位置的说明，这仅适用于非生产集群。

提供了在升级过程中使用 Recreate rollout 策略来允许镜像 registry 使用块存储类型的说明。

15.4.15.2.1. 更改镜像 registry 的管理状态

要启动镜像 registry，您必须将 Image Registry Operator 配置的 managementState 从 Removed 改为 Managed。

流程

- 将 managementState Image Registry Operator 配置从 Removed 改为 Managed。例如：

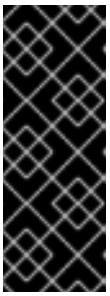
```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch
'{"spec":{"managementState":"Managed"}}'
```

15.4.15.2.2. 为裸机和其他手动安装配置 registry 存储

作为集群管理员，在安装后需要配置 registry 来使用存储。

先决条件

- 您可以使用具有 cluster-admin 角色的用户访问集群。
- 您有一个使用手动置备的 Red Hat Enterprise Linux CoreOS(RHCOS)节点（如裸机）的集群。
- 您已为集群置备持久性存储，如 Red Hat OpenShift Data Foundation。



重要

当您只有一个副本时，OpenShift Container Platform 支持对镜像 registry 存储的 ReadWriteOnce 访问。ReadWriteOnce 访问还要求 registry 使用 Recreate rollout 策略。要部署支持高可用性的镜像 registry，需要两个或多个副本，ReadWriteMany 访问。

- 必须具有 100Gi 容量。

流程

1. 要将 registry 配置为使用存储，修改 configs.imageregistry/cluster 资源中的 spec.storage.pvc。

注意

使用共享存储时，请查看您的安全设置以防止外部访问。

2. 验证您没有 registry pod:

```
$ oc get pod -n openshift-image-registry -l docker-registry=default
```

输出示例

```
No resources found in openshift-image-registry namespace
```

注意

如果您的输出中有一个 registry pod，则不需要继续这个过程。

3. 检查 **registry** 配置：

```
$ oc edit configs.imageregistry.operator.openshift.io
```

输出示例

```
storage:
  pvc:
    claim:
```

将 claim 字段留空以允许自动创建 image-registry-storage PVC。

4. 检查 **clusteroperator** 状态：

```
$ oc get clusteroperator image-registry
```

输出示例

```
NAME             VERSION         AVAILABLE   PROGRESSING   DEGRADED   SINCE
MESSAGE
image-registry   4.16            True        False         False      6h50m
```

5. 确保 registry 设置为 managed，以启用镜像的构建和推送。

- 运行：

```
$ oc edit configs.imageregistry/cluster
```

然后，更改行

```
managementState: Removed
```

至

```
managementState: Managed
```

15.4.15.2.3. 在非生产集群中为镜像 registry 配置存储

您必须为 Image Registry Operator 配置存储。对于非生产集群，您可以将镜像 registry 设置为空目录。如果您这样做，重启 registry 时会丢失所有镜像。

流程

- 将镜像 registry 存储设置为空目录：

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec":{"storage":{"emptyDir":{}}}}'
```

> **警告**
>
> 仅为非生产集群配置这个选项。

如果在 Image Registry Operator 初始化其组件前运行这个命令，oc patch 命令会失败并显示以下错误：

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster"
not found
```

等待几分钟，然后再次运行 命令。

15.4.15.2.4. 为裸机配置块 registry 存储

要允许镜像 registry 在作为集群管理员升级过程中使用块存储类型，您可以使用 Recreate rollout 策略。

重要

支持块存储卷或块持久性卷，但不建议在生产环境中使用镜像 registry。在块存储上配置 registry 的安装不具有高可用性，因为 registry 无法具有多个副本。

如果您选择将块存储卷与镜像 registry 搭配使用，则必须使用文件系统持久性卷声明(PVC)。

流程

1. 输入以下命令将镜像 registry 存储设置为块存储类型，对 registry 进行补丁，使其使用 Recreate rollout 策略，并只使用一个副本运行：

```
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec":
{"rolloutStrategy":"Recreate","replicas":1}}'
```

2. 为块存储设备置备 PV，并为该卷创建 PVC。请求的块卷使用 ReadWriteOnce(RWO)访问模式。

   a. 创建包含以下内容的 pvc.yaml 文件以定义 VMware vSphere PersistentVolumeClaim 对象：

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: image-registry-storage 1
  namespace: openshift-image-registry 2
spec:
  accessModes:
  - ReadWriteOnce 3
```

```
  resources:
    requests:
      storage: 100Gi 4
```

1

代表 PersistentVolumeClaim 对象的唯一名称。

2

PersistentVolumeClaim 对象的命名空间，即 openshift-image-registry。

3

持久性卷声明的访问模式。使用 ReadWriteOnce 时，单个节点可以通过读写权限挂载该卷。

4

持久性卷声明的大小。

b. 输入以下命令从文件创建 PersistentVolumeClaim 对象：

```
$ oc create -f pvc.yaml -n openshift-image-registry
```

3. 输入以下命令编辑 registry 配置，使其引用正确的 PVC：

```
$ oc edit config.imageregistry.operator.openshift.io -o yaml
```

输出示例

```
storage:
  pvc:
    claim: 1
```

1

通过创建自定义 PVC，您可以将 claim 字段留空，以便默认自动创建 image-registry-storage PVC。

15.4.16. 在用户置备的基础架构上完成安装

完成 Operator 配置后，可以在您提供的基础架构上完成集群安装。

先决条件

- 您的 control plane 已初始化。
- 已完成初始 Operator 配置。

流程

1. 使用以下命令确认所有集群组件都在线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION   AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.16.0    True        False         False      19m
baremetal                                  4.16.0    True        False         False      37m
cloud-credential                           4.16.0    True        False         False      40m
cluster-autoscaler                         4.16.0    True        False         False      37m
config-operator                            4.16.0    True        False         False      38m
console                                    4.16.0    True        False         False      26m
csi-snapshot-controller                    4.16.0    True        False         False      37m
dns                                        4.16.0    True        False         False      37m
etcd                                       4.16.0    True        False         False      36m
image-registry                             4.16.0    True        False         False      31m
ingress                                    4.16.0    True        False         False      30m
insights                                   4.16.0    True        False         False      31m
kube-apiserver                             4.16.0    True        False         False      26m
kube-controller-manager                    4.16.0    True        False         False      36m
kube-scheduler                             4.16.0    True        False         False      36m
kube-storage-version-migrator              4.16.0    True        False         False      37m
machine-api                                4.16.0    True        False         False      29m
machine-approver                           4.16.0    True        False         False      37m
machine-config                             4.16.0    True        False         False      36m
marketplace                                4.16.0    True        False         False      37m
monitoring                                 4.16.0    True        False         False      29m
```

```
network                                    4.16.0    True        False         False      38m
node-tuning                                4.16.0    True        False         False      37m
openshift-apiserver                        4.16.0    True        False         False      32m
openshift-controller-manager               4.16.0    True        False         False      30m
openshift-samples                          4.16.0    True        False         False      32m
operator-lifecycle-manager                 4.16.0    True        False         False      37m
operator-lifecycle-manager-catalog         4.16.0    True        False         False      37m
operator-lifecycle-manager-packageserver   4.16.0    True        False         False      32m
service-ca                                 4.16.0    True        False         False      38m
storage                                    4.16.0    True        False         False      37m
```

另外，当所有集群都可用时，以下命令会通知您。它还检索并显示凭证：

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

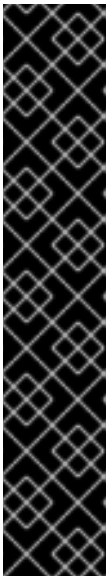
1

对于 <installation_directory>，请指定安装文件保存到的目录的路径。

输出示例

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

Cluster Version Operator 完成从 Kubernetes API 服务器部署 OpenShift Container Platform 集群时，该命令会成功。



重要

- 安装程序生成的 Ignition 配置文件包含 24 小时后过期的证书，然后在该时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 node-bootstrapper 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。

- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

2. 确认 Kubernetes API 服务器正在与 pod 通信。

   a. 要查看所有 pod 的列表，请使用以下命令：

   ```
   $ oc get pods --all-namespaces
   ```

   输出示例

   ```
   NAMESPACE                       NAME                                         READY   STATUS
   RESTARTS   AGE
   openshift-apiserver-operator    openshift-apiserver-operator-85cb746d55-zqhs8
   1/1     Running   1          9m
   openshift-apiserver             apiserver-67b9g                              1/1     Running   0
   3m
   openshift-apiserver             apiserver-ljcmx                              1/1     Running   0
   1m
   openshift-apiserver             apiserver-z25h4                              1/1     Running   0
   2m
   openshift-authentication-operator authentication-operator-69d5d8bf84-vh2n8
   1/1     Running   0          5m
   ...
   ```

   b. 使用以下命令，查看上一命令的输出中所列 pod 的日志：

   ```
   $ oc logs <pod_name> -n <namespace> 1
   ```

1

指定 pod 名称和命名空间，如上一命令的输出中所示。

如果 pod 日志显示，Kubernetes API 服务器可以与集群机器通信。

3. 对于使用光纤通道协议(FCP)的安装，还需要额外的步骤才能启用多路径。不要在安装过程中启用多路径。

如需更多信息，请参阅 *安装后机器配置任务* 文档中的"使用 RHCOS 上使用内核参数启用多路径"。

4. 在 Cluster registration 页面注册 您的集群。

15.4.17. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于 远程健康监控

15.4.18. 后续步骤

- 验证安装.
- 自定义集群。
- 为 Cluster Samples Operator 和 must-gather 工具 配置镜像流。

- 了解如何在 受限网络中使用 Operator Lifecycle Manager(OLM)。
- 如果您用来安装集群的镜像 registry 具有可信任的 CA，请通过 配置额外的信任存储将其添加到集群中。
- 如果需要，您可以选择 不使用远程健康报告。
- 如果需要，请参阅 注册断开连接的集群

15.5. 使用 BARE METAL OPERATOR 扩展用户置备的集群

部署用户置备的基础架构集群后，您可以使用 Bare Metal Operator (BMO) 和其他 metal3 组件来扩展集群中的裸机主机。这种方法可帮助您以更自动化的方式扩展用户置备的集群。

15.5.1. 关于使用 Bare Metal Operator 扩展用户置备的集群

您可以使用 Bare Metal Operator (BMO) 和其他 metal3 组件扩展用户置备的基础架构集群。用户置备的基础架构安装不支持 Machine API Operator。Machine API Operator 通常管理集群中的裸机主机的生命周期。但是，可以使用 BMO 和其他 metal3 组件在用户置备的集群中扩展节点，而无需 Machine API Operator。

15.5.1.1. 扩展用户置备的集群的先决条件

- 您在裸机上安装了用户置备的基础架构集群。
- 您有到主机的基板管理控制器 (BMC) 访问权限。

15.5.1.2. 扩展用户置备的集群的限制

- 您不能使用 provisioning 网络来扩展用户置备的基础架构集群，使用 Bare Metal Operator (BMO)。
  - 因此，您只能使用支持虚拟介质网络的裸机主机驱动程序，如 redfish-virtualmedia 和 idrac-virtualmedia。

- 您不能使用 BMO 在用户置备的基础架构集群中扩展 MachineSet 对象。

15.5.2. 配置置备资源以扩展用户置备的集群

创建 Provisioning 自定义资源 (CR)，以便在用户置备的基础架构集群中启用裸机平台组件。

先决条件

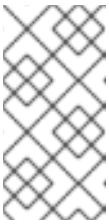
- 您在裸机上安装了用户置备的基础架构集群。

流程

1. 创建 Provisioning CR。

   a. 将以下 YAML 保存到 provisioning.yaml 文件中：

```
apiVersion: metal3.io/v1alpha1
kind: Provisioning
metadata:
  name: provisioning-configuration
spec:
  provisioningNetwork: "Disabled"
  watchAllNamespaces: false
```



注意

在使用 Bare Metal Operator 扩展用户置备的集群时，OpenShift Container Platform 4.16 不支持启用 provisioning 网络。

2. 运行以下命令来创建 Provisioning CR：

```
$ oc create -f provisioning.yaml
```

输出示例

```
provisioning.metal3.io/provisioning-configuration created
```

验证

- 运行以下命令验证置备服务是否正在运行：

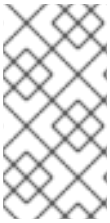
```
$ oc get pods -n openshift-machine-api
```

输出示例

```
NAME                                           READY   STATUS    RESTARTS       AGE
cluster-autoscaler-operator-678c476f4c-jjdn5   2/2     Running   0              5d21h
cluster-baremetal-operator-6866f7b976-gmvgh    2/2     Running   0              5d21h
control-plane-machine-set-operator-7d8566696c-bh4jz   1/1     Running   0
5d21h
ironic-proxy-64bdw                             1/1     Running   0              5d21h
ironic-proxy-rbggf                             1/1     Running   0              5d21h
ironic-proxy-vj54c                             1/1     Running   0              5d21h
machine-api-controllers-544d6849d5-tgj9l       7/7     Running   1 (5d21h ago)
5d21h
machine-api-operator-5c4ff4b86d-6fjmq          2/2     Running   0              5d21h
metal3-6d98f84cc8-zn2mx                        5/5     Running   0              5d21h
metal3-image-customization-59d745768d-bhrp7    1/1     Running   0              5d21h
```

15.5.3. 使用 BMO 在用户置备的集群中置备新主机

您可以通过创建一个 BareMetalHost 自定义资源 (CR)，使用 Bare Metal Operator (BMO) 在用户置备的集群中置备裸机主机。



注意

要使用 BMO 为集群置备裸机主机，您必须将 BareMetalHost 自定义资源中的 spec.externallyProvisioned 规格设置为 false。

先决条件

- 您创建了用户置备的裸机集群。

- 您有到主机的基板管理控制器 (BMC) 访问权限。
- 通过创建一个 Provisioning CR，在集群中部署了置备服务。

流程

1. 创建 Secret CR 和 BareMetalHost CR。

   a. 将以下 YAML 保存到 bmh.yaml 文件中：

```
---
apiVersion: v1
kind: Secret
metadata:
  name: worker1-bmc
  namespace: openshift-machine-api
type: Opaque
data:
  username: <base64_of_uid>
  password: <base64_of_pwd>
---
apiVersion: metal3.io/v1alpha1
kind: BareMetalHost
metadata:
  name: worker1
  namespace: openshift-machine-api
spec:
  bmc:
    address: <protocol>://<bmc_url> (1)
    credentialsName: "worker1-bmc"
  bootMACAddress: <nic1_mac_address>
  externallyProvisioned: false (2)
  customDeploy:
    method: install_coreos
  online: true
  userData:
    name: worker-user-data-managed
    namespace: openshift-machine-api
```

(1) 您只能使用支持虚拟介质网络引导的裸机主机驱动程序，如 redfish-virtualmedia 和 idrac-virtualmedia。

(2)

2. 运行以下命令来创建裸机主机对象：

```
$ oc create -f bmh.yaml
```

输出示例

```
secret/worker1-bmc created
baremetalhost.metal3.io/worker1 created
```

3. 批准所有证书签名请求 (CSR)。

a. 运行以下命令验证主机的置备状态是否为 provisioned：

```
$ oc get bmh -A
```

输出示例

```
NAMESPACE               NAME        STATE                  CONSUMER   ONLINE
ERROR   AGE
openshift-machine-api   controller1   externally provisioned             true
5m25s
openshift-machine-api   worker1       provisioned                        true          4m45s
```

b. 运行以下命令，获取待处理的 CSR 列表：

```
$ oc get csr
```

输出示例

```
NAME        AGE   SIGNERNAME                                    REQUESTOR
REQUESTEDDURATION CONDITION
csr-gfm9f   33s   kubernetes.io/kube-apiserver-client-kubelet
system:serviceaccount:openshift-machine-config-o
perator:node-bootstrapper   <none>              Pending
```

c. 运行以下命令来批准 CSR：

```
$ oc adm certificate approve <csr_name>
```

输出示例

```
certificatesigningrequest.certificates.k8s.io/<csr_name> approved
```

验证

- 运行以下命令验证节点是否已就绪：

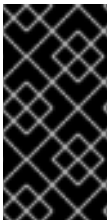
```
$ oc get nodes
```

输出示例

```
NAME       STATUS   ROLES          AGE     VERSION
app1       Ready    worker         47s     v1.24.0+dc5a2fd
controller1 Ready    master,worker   2d22h   v1.24.0+dc5a2fd
```

15.5.4. 可选：使用 BMO 管理用户置备的集群中的现有主机

另外，您可以通过为现有主机创建一个 BareMetalHost 对象来管理用户置备的集群中的现有裸机控制器主机，来使用 Bare Metal Operator (BMO) 来管理用户置备集群中的现有裸机控制器主机。它不是必须的来管理现有用户置备的主机；但是，您可以将它们注册为外部置备主机，以实现清单目的。



重要

要使用 BMO 管理现有主机，您必须将 BareMetalHost 自定义资源中的 spec.externallyProvisioned 规格设置为 true，以防止 BMO 重新置备主机。

先决条件

- 您创建了用户置备的裸机集群。
- 您有到主机的基板管理控制器 (BMC) 访问权限。
- 通过创建一个 Provisioning CR，在集群中部署了置备服务。

流程

1. 创建 Secret CR 和 BareMetalHost CR。

   a. 将以下 YAML 保存到 controller.yaml 文件中：

```
---
apiVersion: v1
kind: Secret
metadata:
  name: controller1-bmc
  namespace: openshift-machine-api
type: Opaque
data:
  username: <base64_of_uid>
  password: <base64_of_pwd>
---
apiVersion: metal3.io/v1alpha1
kind: BareMetalHost
metadata:
  name: controller1
  namespace: openshift-machine-api
spec:
  bmc:
    address: <protocol>://<bmc_url> (1)
```

```
  credentialsName: "controller1-bmc"
bootMACAddress: <nic1_mac_address>
customDeploy:
  method: install_coreos
externallyProvisioned: true 2
online: true
userData:
  name: controller-user-data-managed
  namespace: openshift-machine-api
```

1 您只能使用支持虚拟介质网络引导的裸机主机驱动程序，如 redfish-virtualmedia 和 idrac-virtualmedia。

2 您必须将值设为 true，以防止 BMO 重新置备裸机控制器主机。

2. 运行以下命令来创建裸机主机对象：

```
$ oc create -f controller.yaml
```

输出示例

```
secret/controller1-bmc created
baremetalhost.metal3.io/controller1 created
```

验证

- 运行以下命令，验证 BMO 创建了裸机主机对象：

```
$ oc get bmh -A
```

输出示例

```
NAMESPACE               NAME        STATE                 CONSUMER   ONLINE   ERROR
AGE
openshift-machine-api   controller1   externally provisioned             true
13s
```

15.5.5. 使用 BMO 从用户置备的集群中删除主机

您可以使用 Bare Metal Operator (BMO)从用户置备的集群中删除裸机主机。

先决条件

- 您创建了用户置备的裸机集群。
- 您有到主机的基板管理控制器 (BMC) 访问权限。
- 通过创建一个 Provisioning CR，在集群中部署了置备服务。

流程

1. 运行以下命令进行 cordon 和 drain 主机：

```
$ oc adm drain app1 --force --ignore-daemonsets=true
```

输出示例

```
node/app1 cordoned
WARNING: ignoring DaemonSet-managed Pods: openshift-cluster-node-tuning-
operator/tuned-tvthg, openshift-dns/dns-
default-9q6rz, openshift-dns/node-resolver-zvt42, openshift-image-registry/node-ca-
mzxth, openshift-ingress-cana
ry/ingress-canary-qq5lf, openshift-machine-config-operator/machine-config-daemon-
v79dm, openshift-monitoring/nod
e-exporter-2vn59, openshift-multus/multus-additional-cni-plugins-wssvj, openshift-
multus/multus-fn8tg, openshift
-multus/network-metrics-daemon-5qv55, openshift-network-diagnostics/network-
check-target-jqxn2, openshift-ovn-ku
bernetes/ovnkube-node-rsvqg
evicting pod openshift-operator-lifecycle-manager/collect-profiles-27766965-258vp
```

```
evicting pod openshift-operator-lifecycle-manager/collect-profiles-27766950-kg5mk
evicting pod openshift-operator-lifecycle-manager/collect-profiles-27766935-stf4s
pod/collect-profiles-27766965-258vp evicted
pod/collect-profiles-27766950-kg5mk evicted
pod/collect-profiles-27766935-stf4s evicted
node/app1 drained
```

2. 从 BareMetalHost CR 中删除 customDeploy 规格。

   a. 运行以下命令，为主机编辑 BareMetalHost CR：

   ```
   $ oc edit bmh -n openshift-machine-api <host_name>
   ```

   b. 删除 spec.customDeploy 和 spec.customDeploy.method 行：

   ```
   ...
     customDeploy:
       method: install_coreos
   ```

   c. 运行以下命令，验证主机的置备状态是否更改为 deprovisioning ：

   ```
   $ oc get bmh -A
   ```

   输出示例

   ```
   NAMESPACE               NAME        STATE                  CONSUMER   ONLINE
   ERROR   AGE
   openshift-machine-api   controller1   externally provisioned             true          58m
   openshift-machine-api   worker1       deprovisioning                     true          57m
   ```

3. 运行以下命令来删除节点：

   ```
   $ oc delete node <node_name>
   ```

验证

- 运行以下命令验证节点是否已删除：

```
$ oc get nodes
```

输出示例

```
NAME          STATUS   ROLES          AGE     VERSION
controller1   Ready    master,worker   2d23h   v1.24.0+dc5a2fd
```

15.6. 裸机的安装配置参数

在部署 OpenShift Container Platform 集群前，您可以提供一个自定义 的 install-config.yaml 安装配置文件，该文件描述了您的环境的详情。

15.6.1. 裸机的可用安装配置参数

下表指定您可以在安装过程中设置所需的、可选和特定于裸机的安装配置参数。

注意

安装后，您无法在 install-config.yaml 文件中修改这些参数。

15.6.1.1. 所需的配置参数

下表描述了所需的安装配置参数：

表 15.42. 所需的参数

参数	描述	值
apiVersion:	**install-config.yaml** 内容的 API 版本。当前版本为 **v1**。安装程序可能还支持旧的 API 版本。	字符串

参数	描述	值
baseDomain:	云供应商的基域。基域用于创建到 OpenShift Container Platform 集群组件的路由。集群的完整 DNS 名称是 **baseDomain** 和 **metadata.name** 参数值的组合，其格式为 **<metadata.name>.<baseDomain>**。	完全限定域名或子域名，如 **example.com**。
metadata:	Kubernetes 资源 **ObjectMeta**，其中只消耗 **name** 参数。	对象
metadata: name:	集群的名称。集群的 DNS 记录是 **{{.metadata.name}}.{{.baseDomain}}** 的子域。	小写字母和连字符 (**-**) 的字符串，如 **dev**。
platform:	对于特定平台的配置取决于执行安装的环境： **aws**, **baremetal**, **azure**, **gcp**, **ibmcloud**, **nutanix**, **openstack**, **powervs**, **vsphere**, 或 **{}**.有关 **platform.<platform>** 参数的更多信息，请参考下表中您的特定平台。	对象

pullSecret:	从 Red Hat OpenShift Cluster Manager 获取 pull secret，验证从 Quay.io 等服务中下载 OpenShift Container Platform 组件的容器镜像。	<pre>{
		  "auths":{
		  "cloud.openshift.com":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  },
		  "quay.io":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  }
		  }
		}</pre>

15.6.1.2. 网络配置参数

您可以根据现有网络基础架构的要求自定义安装配置。例如，您可以扩展集群网络的 IP 地址块，或者提供不同于默认值的不同 IP 地址块。

-

如果使用 Red Hat OpenShift Networking OVN-Kubernetes 网络插件，则支持 IPv4 和 IPv6 地址系列。

如果将集群配置为使用两个 IP 地址系列，请查看以下要求：

- 两个 IP 系列都必须将相同的网络接口用于默认网关。
- 两个 IP 系列都必须具有默认网关。
- 您必须为所有网络配置参数指定 IPv4 和 IPv6 地址。例如，以下配置 IPv4 地址列在 IPv6 地址的前面。

```
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  - cidr: fd00:10:128::/56
    hostPrefix: 64
  serviceNetwork:
  - 172.30.0.0/16
  - fd00:172:16::/112
```

注意

Red Hat OpenShift Data Foundation 灾难恢复解决方案不支持 Globalnet。对于区域灾难恢复场景，请确保为每个集群中的集群和服务网络使用非重叠的专用 IP 地址。

表 15.43. 网络参数

参数	描述	值
networking:	集群网络的配置。	对象  **注意** 您无法在安装后修改 **网络** 对象指定的参数。

参数	描述	值
networking: networkType:	要安装的 Red Hat OpenShift Networking 网络插件。	**OVNKubernetes**。**OVNKubernetes** 是 Linux 网络和包含 Linux 和 Windows 服务器的混合网络的 CNI 插件。默认值为 **OVNKubernetes**。
networking: clusterNetwork:	pod 的 IP 地址块。 默认值为 **10.128.0.0/14**，主机前缀为 **/23**。 如果您指定了多个 IP 地址块，块不得重叠。	对象数组。例如： networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23 - cidr: fd01::/48 hostPrefix: 64
networking: clusterNetwork: cidr:	使用 **networking.clusterNetwork** 时需要此项。IP 地址块。 如果使用 OVN-Kubernetes 网络插件，您可以指定 IPv4 和 IPv6 网络。	使用 CIDR 形式的 IP 地址块。IPv4 块的前缀长度介于 **0** 到 **32** 之间。IPv6 块的前缀长度介于 **0** 到 **128** 之间。例如，**10.128.0.0/14** 或 **fd01::/48**。
networking: clusterNetwork: hostPrefix:	分配给每个节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从 given **cidr** 中分配 a/**23** 子网。**hostPrefix** 值 **23** 提供 510（2^(32 - 23)- 2）pod IP 地址。	子网前缀。 对于 IPv4 网络，默认值为 **23**。对于 IPv6 网络，默认值为 **64**。默认值也是 IPv6 的最小值。
networking: serviceNetwork:	服务的 IP 地址块。默认值为 **172.30.0.0/16**。 OVN-Kubernetes 网络插件只支持服务网络的一个 IP 地址块。 如果使用 OVN-Kubernetes 网络插件，您可以为 IPv4 和 IPv6 地址系列指定一个 IP 地址块。	CIDR 格式具有 IP 地址块的数组。例如： networking: serviceNetwork: - 172.30.0.0/16 - fd02::/112
networking: machineNetwork:	机器的 IP 地址块。 如果您指定了多个 IP 地址块，块不得重叠。	对象数组。例如： networking: machineNetwork: - cidr: 10.0.0.0/16

参数	描述	值
networking: machineNetwork: cidr:	使用 **networking.machineNetwork** 时需要此项。IP 地址块。对于 libvirt 和 IBM Power® Virtual Server 以外的所有平台，默认值为 **10.0.0.0/16**。对于 libvirt，默认值 为 **192.168.126.0/24**。对于 IBM Power® Virtual Server，默认值为 **192.168.0.0/24**。	CIDR 表示法中的 IP 网络块。 例如。**10.0.0.0/16** 或 **fd00::/48**。  **注意** 将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

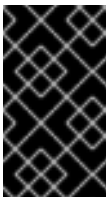
15.6.1.3. 可选的配置参数

下表描述了可选的安装配置参数：

表 15.44. 可选参数

参数	描述	值
additionalTrustBundle:	添加到节点可信证书存储中的 PEM 编码 X.509 证书捆绑包。配置了代理时，也可以使用此信任捆绑包。	字符串
capabilities:	控制可选核心组件的安装。您可以通过禁用可选组件来减少 OpenShift Container Platform 集群的空间。如需更多信息，请参阅 *安装*中的"集群功能"页面。	字符串数组
capabilities: baselineCapabilitySet:	选择要启用的一组初始可选功能。有效值为 **None**、**v4.11**、**v4.12** 和 **vCurrent**。默认值为 **vCurrent**。	字符串
capabilities: additionalEnabledCapabilities:	将可选功能集合扩展到您在 **baselineCapabilitySet** 中指定的范围。您可以在此参数中指定多个功能。	字符串数组

参数	描述	值
cpuPartitioningMode:	启用工作负载分区，它会隔离 OpenShift Container Platform 服务、集群管理工作负载和基础架构 pod，以便在保留的一组 CPU 上运行。工作负载分区只能在安装过程中启用，且在安装后无法禁用。虽然此字段启用工作负载分区，但它不会将工作负载配置为使用特定的 CPU。如需更多信息，请参阅 *Scalability and Performance* 部分中的 *Workload partitioning* 页面。	**None** 或 **AllNodes**。**None** 是默认值。
compute:	组成计算节点的机器的配置。	**MachinePool** 对象的数组。
compute: architecture:	决定池中机器的指令集合架构。目前，不支持具有不同架构的集群。所有池都必须指定相同的架构。有效值为 **amd64** 和 **arm64**。	字符串
compute: hyperthreading:	是否在计算机器上启用或禁用并发多**线程或超线程**。默认情况下，启用并发多线程以提高机器内核的性能。 **重要** 如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。	**enabled** 或 **Disabled**
compute: name:	使用 **compute** 时需要此项。机器池的名称。	**worker**
compute: platform:	使用 **compute** 时需要此项。使用此参数指定托管 worker 机器的云供应商。此参数值必须与 **controlPlane.platform** 参数值匹配。	**aws**, **azure**, **gcp**, **ibmcloud**, **nutanix**, **openstack**, **powervs**, **vsphere**, 或 **{}**
compute: replicas:	要置备的计算机器数量，也称为 worker 机器。	大于或等于 **2** 的正整数。默认值为 **3**。

参数	描述	值
featureSet:	为功能集启用集群。功能集是 OpenShift Container Platform 功能的集合，默认情况下不启用。有关在安装过程中启用功能集的更多信息，请参阅"使用功能门启用功能"。	字符串.要启用的功能集的名称，如 **TechPreviewNoUpgrade**。
controlPlane:	组成 control plane 的机器的配置。	**MachinePool** 对象的数组。
controlPlane: architecture:	决定池中机器的指令集合架构。目前，不支持具有不同架构的集群。所有池都必须指定相同的架构。有效值为 **amd64** 和 **arm64**。	字符串
controlPlane: hyperthreading:	是否在 control plane 机器上启用或禁用并发多 **线程或超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。	**enabled** 或 **Disabled**
controlPlane: name:	使用 **controlPlane** 时需要此项。机器池的名称。	**master**
controlPlane: platform:	使用 **controlPlane** 时需要此项。使用此参数指定托管 control plane 机器的云供应商。此参数值必须与 **compute.platform** 参数值匹配。	**aws**, **azure**, **gcp**, **ibmcloud**, **nutanix**, **openstack**, **powervs**, **vsphere**, 或 **{}**
controlPlane: replicas:	要置备的 control plane 机器数量。	部署单节点 OpenShift 时支持的值为 **3** 或 **1**。
credentialsMode:	Cloud Credential Operator(CCO)模式。如果没有指定模式，CCO 会动态尝试决定提供的凭证的功能，在支持多个模式的平台上首选 mint 模式。	**Mint**、**Passthrough**、**Manual** 或空字符串(**""**)。[1]

参数	描述	值
fips:	启用或禁用 FIPS 模式。默认值为 **false**（禁用）。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。  **重要** 要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 Red Hat Enterprise Linux (RHEL) 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅在 FIPS 模式中安装该系统。 当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS) 时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。  **注意** 如果使用 Azure File 存储，则无法启用 FIPS 模式。	**false** 或 **true**

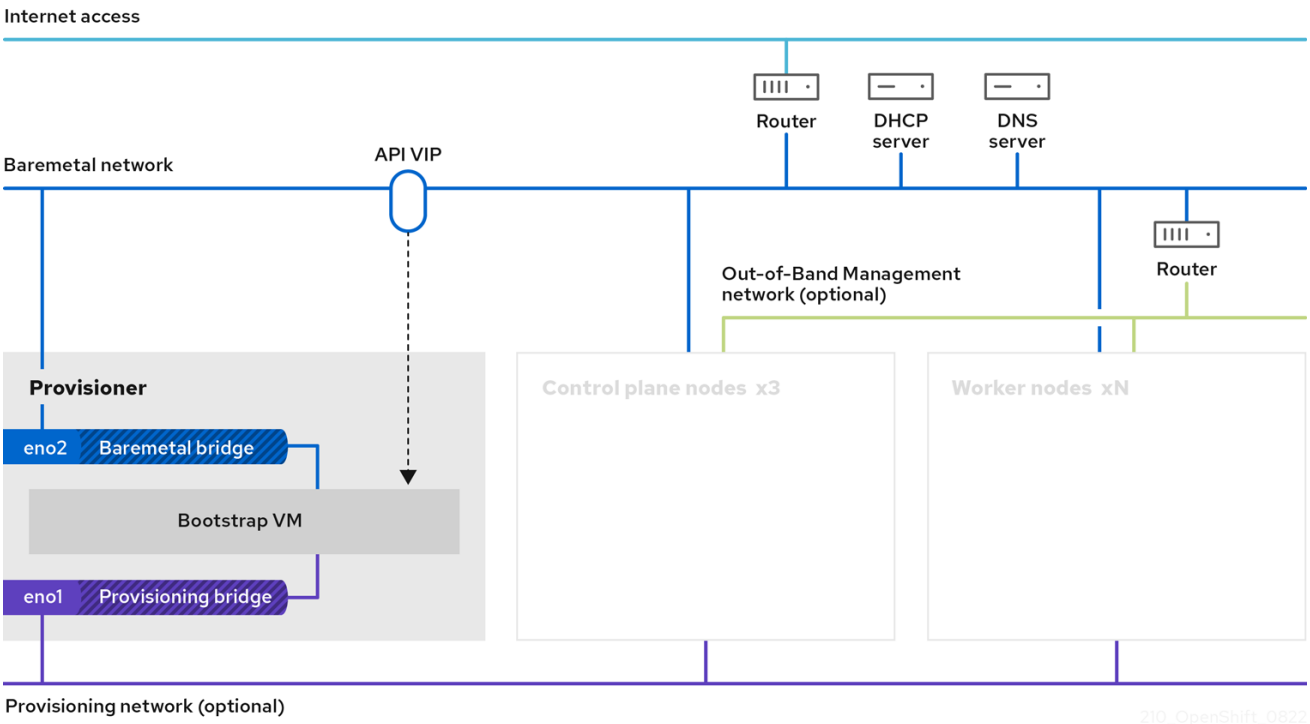
参数	描述	值
imageContentSources:	release-image 内容的源和存储库。	对象数组。包括一个 **source** 以及可选的 **mirrors**，如本表的以下行所述。
imageContentSources: source:	使用 **imageContentSources** 时需要此项。指定用户在镜像拉取规格中引用的存储库。	字符串
imageContentSources: mirrors:	指定可能还包含同一镜像的一个或多个仓库。	字符串数组
publish:	如何发布或公开集群的面向用户的端点，如 Kubernetes API、OpenShift 路由。	**内部或外部 .**默认值为 **External**。 在非云平台上不支持将此字段设置为 **Internal**。 **重要** 如果将字段的值设为 **Internal**，集群将无法运行。如需更多信息，请参阅 BZ#1953035。
sshKey:	用于验证对集群机器的访问的 SSH 密钥。 **注意** 对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。	例如，**sshKey: ssh-ed25519 AAAA..**。

1. **不是所有 CCO 模式都支持所有云供应商。有关 CCO 模式的更多信息，请参阅*身份验证和授权*内容中的"管理云供应商凭证"条目。**

第 16 章 在裸机上部署安装程序置备的集群

16.1. 概述

裸机节点上的安装程序置备安装会部署并配置运行 OpenShift Container Platform 集群的基础架构。本指南提供了一种成功实现安装程序置备的裸机安装的方法。下图演示了部署阶段 1 中的安装环境：



对于安装，上图中的关键元素是：

- Provisioner：运行安装程序的物理计算机，并托管部署新 OpenShift Container Platform 集群的控制平面的 bootstrap 虚拟机。
- Bootstrap 虚拟机 ：部署 OpenShift Container Platform 集群过程中使用的虚拟机。
- 网络桥接 ：bootstrap 虚拟机连接到裸机网络，如果存在，通过网桥 eno1 和 eno2 连接到 provisioning 网络。
- API VIP ：API 虚拟 IP 地址 (VIP) 用于在 control plane 节点上提供 API 服务器的故障切换。API VIP 首先位于 bootstrap 虚拟机上。在启动服务前，脚本会生成 keepalived.conf 配置文

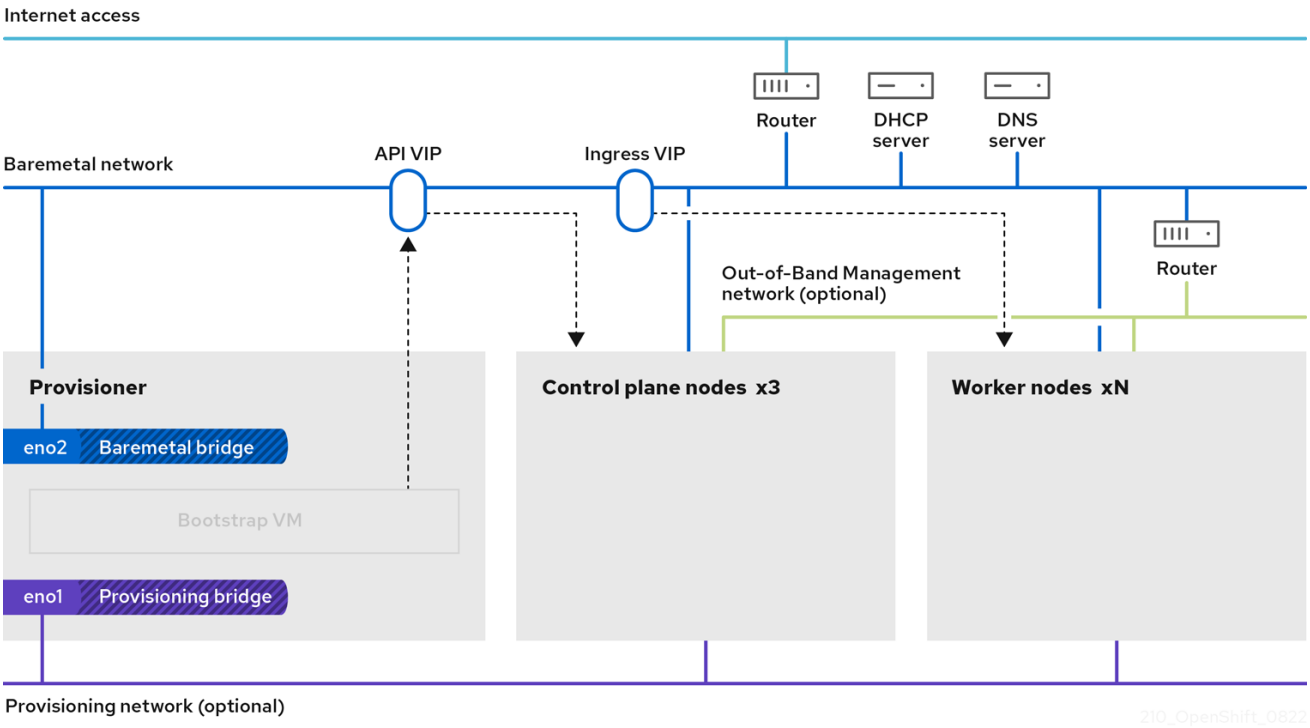
件。在 bootstrap 过程完成后，VIP 被移到一个 control plane 节点，bootstrap 虚拟机会停止。

在部署阶段 2 中，置备程序会自动销毁 bootstrap 虚拟机，并将虚拟 IP 地址 (VIP) 移到适当的节点。

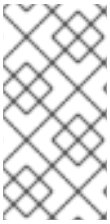
keepalived.conf 文件会将 control plane 机器设置为其虚拟路由器冗余协议 (VRRP 优先级比 bootstrap 虚拟机的设置低，这样可确保在 API VIP 从 bootstrap 虚拟机移到控制平面机器前，控制平面上的 API 可以完全正常工作。当 API VIP 移动到其中一个 control plane 节点后，从外部客户端发送到 API VIP 路由的流量发送到该 control plane 节点上运行的 haproxy 负载均衡器。此 haproxy 实例在 control plane 节点之间负载均衡 API VIP 流量。

Ingress VIP 移到计算节点。keepalived 实例还管理 Ingress VIP。

下图演示了部署的阶段 2：



此时，置备程序使用的节点可以被移除或重新使用。在这里，所有额外的置备任务都由 control plane 执行。

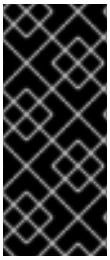


注意

对于安装程序置备的基础架构安装，CoreDNS 在节点级别公开端口 53，使其可以从其他可路由网络访问。

其他资源

- 使用 DNS 转发



重要

provisioning 网络是可选的，但 PXE 启动需要它。如果在没有 provisioning 网络的情况下部署，则必须使用虚拟介质管理控制器 (BMC) 寻址选项，如 redfish-virtualmedia 或 idrac-virtualmedia。

16.2. 先决条件

OpenShift Container Platform 安装程序置备的安装需要：

1. 安装了 Red Hat Enterprise Linux(RHEL) 9.x 的一个置备程序节点。在安装后可以删除置备程序。
2. 三个 control plane 节点
3. 对每个节点的 Baseboard 管理控制器 (BMC) 访问
4. 至少一个网络：
    a. 一个必需的可路由网络
    b. 一个可选的 provisioning 网络
    c. 一个可选的管理网络

在开始 OpenShift Container Platform 安装程序置备的安装前，请确保硬件环境满足以下要求。

16.2.1. 节点要求

安装程序置备的安装涉及多个硬件节点要求：

- CPU 架构： 所有节点都必须使用 x86_64 或 aarch64 CPU 架构。
- 类似的节点： 红帽建议每个角色都有相同的配置。也就是说，红帽建议节点具有相同的品牌和型号，它们具有相同的 CPU、内存和存储配置。
- Baseboard Management Controller： provisioner 节点必须能够访问每个 OpenShift Container Platform 集群节点的基板管理控制器(BMC)。您可以使用 IPMI、Redfish 或专有协议。
- 最新一代： 节点必须是最新一代。安装程序置备的安装依赖于 BMC 协议，这些协议必须在节点间兼容。另外，RHEL 9.x 附带了 RAID 控制器的最新驱动程序。确保节点足以支持 provisioner 节点的 RHEL 9.x，支持 control plane 和 worker 节点的 RHCOS 9.x。
- registry 节点：（可选）如果设置了一个断开连接的镜像 registry，建议 registry 驻留在自己的节点上。
- provisioner 节点： 安装程序置备的安装需要一个 provisioner 节点。
- control plane： 安装程序置备的安装需要三个 control plane 节点才能实现高可用性。您可以部署仅具有三个 control plane 节点的 OpenShift Container Platform 集群，使 control plane 节点可以作为 worker 节点调度。较小的集群在开发、生产和测试过程中为管理员和开发人员提供更多资源。
- Worker 节点： 虽然不是必须的的，但典型的生产环境集群一般会有两个或者多个 worker 节点。

重要

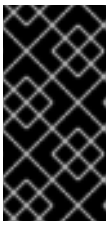
不要只部署一个 worker 节点的集群，因为集群将使用降级状态的路由器和入口流量进行部署。

- **网络接口：每个节点必须至少有一个网络接口用于可路由的 baremetal 网络。在使用 provisioning 网络进行部署时，每个节点都必须有一个网络接口用于 provisioning 网络。使用 provisioning 网络是默认配置。**

注意

在同一子网上只能有一个网卡 (NIC) 可以通过网关路由流量。默认情况下，地址解析协议 (ARP) 使用编号最低的 NIC。对同一子网中的每个节点都使用一个单独的 NIC，以确保网络负载均衡按预期工作。当为同一子网中的一个节点使用多个 NIC 时，需要使用一个绑定或组接口。然后，以一个别名 IP 地址的形式将其他 IP 地址添加到该接口。如果您需要在网络接口级别进行容错或负载均衡，请在绑定或团队接口上使用别名 IP 地址。或者，您可以在同一子网中禁用二级 NIC，或者确保它没有 IP 地址。

- **统一可扩展固件接口(UEFI)：安装程序置备的安装需要在 置备 网络中使用 IPv6 时在所有 OpenShift Container Platform 节点上进行 UEFI 引导。另外，UEFI Device PXE 设置必须设置为在 provisioning 网络 NIC 中使用 IPv6 协议，但忽略 provisioning 网络会删除这个要求。**



重要

从 ISO 镜像启动安装时，删除所有旧的 UEFI 引导条目。如果引导条目包含不是固件提供通用条目的条目，则安装可能会失败。

- **安全引导：许多生产环境中需要启用安全引导的节点来验证节点仅使用可信软件引导，如 UEFI 固件驱动程序、EFI 应用程序和操作系统。您可以使用安全引导手动或管理进行部署。**

1. **手动：要使用安全引导机制手动部署 OpenShift Container Platform 集群，您必须在每个 control plane 节点上和每个 worker 节点上启用 UEFI 引导模式和安全引导。红帽支持仅在安装程序置备的安装使用 Redfish 虚拟介质时手动启用 UEFI 和安全引导。如需了解更多详细信息，请参阅"配置节点"一节中的"手动配置安全引导节点"。**

2. **Managed: 要使用受管安全引导机制部署 OpenShift Container Platform 集群，您必须在 install-config.yaml 文件中将 bootMode 值设置为 UEFISecureBoot。红帽仅在第 10 代 HPE 硬件和 13 代运行固件版本 2.75.75.75 或更高版本的 Dell 硬件中支持带受管安全引**

导的安装程序置备安装。使用受管安全引导进行部署不需要 Redfish 虚拟介质。详情请参阅"为 OpenShift 安装设置环境"一节中的"配置受管安全引导"。

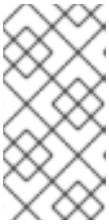
注意

红帽不支持使用自生成的密钥进行安全引导。

16.2.2. 为 OpenShift Virtualization 规划裸机集群

如果使用 OpenShift Virtualization，必须在安装裸机集群前了解一些要求。

- 如果要使用实时迁移功能，*在集群安装时*需要有多个 worker 节点。这是因为实时迁移需要集群级别的高可用性 (HA) 标记设置为 true。当安装集群时，会设置 HA 标志，之后无法更改。如果在安装集群时定义少于两个 worker 节点，则集群生命周期中的 HA 标记被设置为 false。



注意

您可以在单节点集群中安装 OpenShift Virtualization，但单节点 OpenShift 不支持高可用性。

- 实时迁移需要共享存储。OpenShift Virtualization 的存储必须支持并使用 ReadWriteMany (RWX) 访问模式。

- 如果您计划使用单根 I/O 虚拟化 (SR-IOV)，请确保 OpenShift Container Platform 支持网络接口控制器 (NIC)。

其他资源

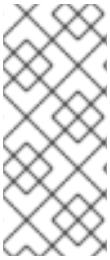
- 为 OpenShift Virtualization 准备集群

- 关于单根 I/O 虚拟化（SR-IOV）硬件网络

- 将虚拟机连接到 SR-IOV 网络

16.2.3. 使用虚拟介质安装的固件要求

安装程序置备的 OpenShift Container Platform 集群的安装程序会验证与 Redfish 虚拟介质的硬件和固件兼容性。如果节点固件不兼容，安装程序不会在节点上开始安装。下表列出了经过测试并验证以使用 Redfish 虚拟介质部署的安装程序置备的 OpenShift Container Platform 集群的最低固件版本。



注意

红帽不测试固件、硬件或其他第三方组件的组合。有关第三方支持的更多信息，请参阅红帽第三方支持政策。有关更新固件的详情，请查看硬件文档了解节点或联系硬件厂商。

表 16.1. HP 硬件与 Redfish 虚拟介质的固件兼容性

model	管理	固件版本
第 10 代	iLO5	2.63 或更高版本

表 16.2. Dell 硬件与 Redfish 虚拟介质的固件兼容性

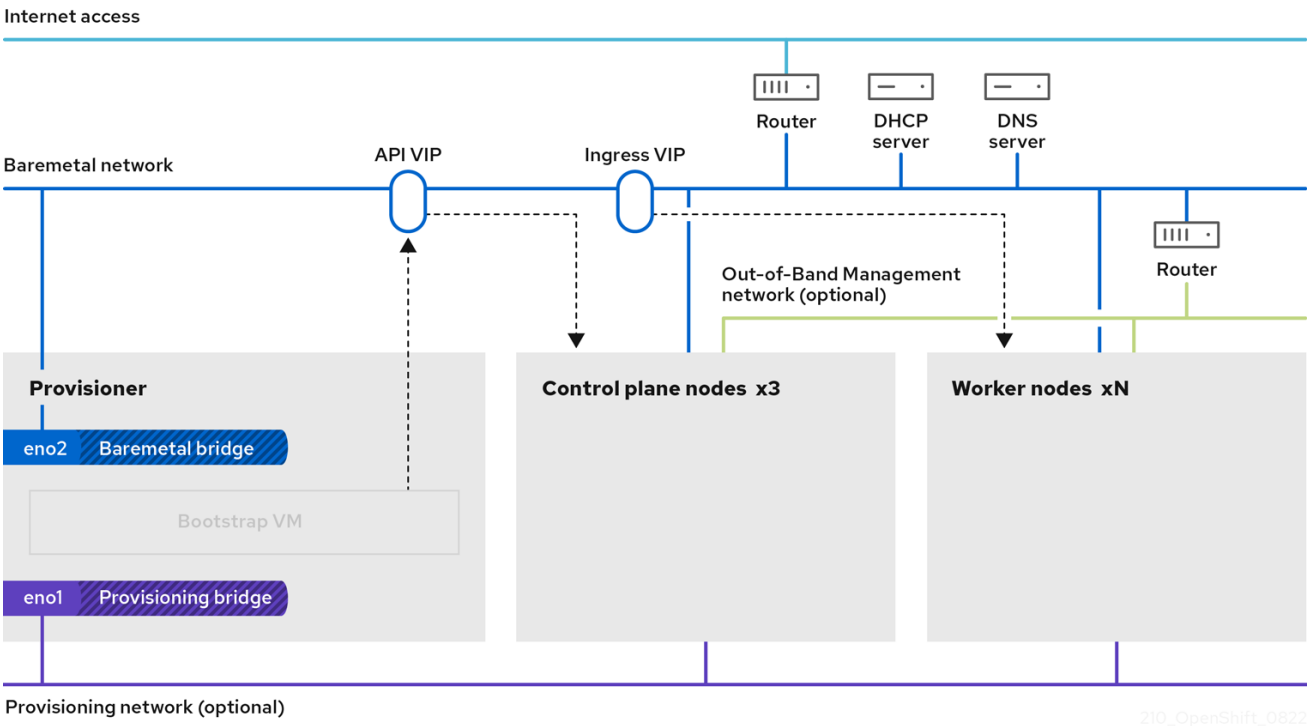
model	管理	固件版本
第 15 代	iDRAC 9	v6.10.30.00 和 v7.00.60.00
第 14 代	iDRAC 9	v6.10.30.00
第 13 代	iDRAC 8	v2.75.75.75 或更高版本

其他资源

- 无法使用 BMC 发现新的裸机主机

16.2.4. 网络要求

OpenShift Container Platform 安装程序置备的安装涉及几个网络要求。首先，安装程序置备的安装涉及一个可选的不可路由 置备 网络，用于在每个裸机节点上置备操作系统。其次，安装程序置备的安装涉及一个可路由的 baremetal 网络。



16.2.4.1. 确保打开所需的端口

某些端口必须在集群节点之间打开，才能成功完成安装程序置备的安装。在某些情况下，比如在边缘 worker 节点中使用单独的子网，您必须确保这些子网中的节点可以与以下所需端口上其他子网中的节点通信。

表 16.3. 所需端口

port	描述
67,68	在使用 provisioning 网络时，集群节点使用端口 **67** 和 **68** 访问 **dnsmasq** DHCP 服务器。
69	在使用 provisioning 网络时，集群节点使用它们的置备网络接口与端口 **69** 上的 TFTP 服务器通信。TFTP 服务器在 bootstrap 虚拟机上运行。bootstrap 虚拟机在 provisioner 节点上运行。
80	如果没有使用镜像缓存选项或使用虚拟介质，则 provisioner 节点必须在 **baremetal** 机器网络接口上打开端口 **80**，才能将来自 provisioner 节点的 Red Hat Enterprise Linux CoreOS (RHCOS)镜像流传输到集群节点。
123	集群节点必须使用 **baremetal** 机器网络访问端口 **123** 上的 NTP 服务器。

port	描述
5050	Ironic Inspector API 在 control plane 节点上运行，并侦听端口 **5050**。Inspector API 负责硬件内省，它收集有关裸机节点的硬件特征的信息。
5051	端口 **5050** 使用端口 **5051** 作为代理。
6180	当使用虚拟介质而不是 TLS 部署时，provisioner 节点和 control plane 节点必须在 **baremetal** 机器网络接口上打开端口 **6180**，以便 worker 节点的基板管理控制器(BMC)可以访问 RHCOS 镜像。从 OpenShift Container Platform 4.13 开始，默认 HTTP 端口为 **6180**。
6183	当使用虚拟介质并使用 TLS 部署时，provisioner 节点和 control plane 节点必须在 **baremetal** 机器网络接口上打开端口 **6183**，以便 worker 节点的 BMC 可以访问 RHCOS 镜像。
6385	Ironic API 服务器最初在 bootstrap 虚拟机上运行，稍后在 control plane 节点上运行，并侦听端口 **6385**。Ironic API 允许客户端与 Ironic 交互以进行裸机节点置备和管理，包括注册新节点、管理电源状态、部署镜像和清理硬件等操作。
6388	端口 **6385** 使用端口 **6388** 作为代理。
8080	在不使用 TLS 的情况下使用镜像缓存时，必须在 provisioner 节点上打开端口 **8080**，并可以被集群节点的 BMC 接口访问。
8083	当将镜像缓存选项与 TLS 搭配使用时，必须在 provisioner 节点上打开端口 **8083**，并可以被集群节点的 BMC 接口访问。
9999	默认情况下，Ironic Python 代理(IPA)侦听来自 Ironic 编排器服务的 API 调用的 TCP 端口 **9999**。此端口用于运行 IPA 的裸机节点和 Ironic 编排器服务之间的通信。

16.2.4.2. 增加网络 MTU

在部署 OpenShift Container Platform 前，将网络最大传输单元 (MTU) 增加到 1500 或更多。如果 MTU 小于 1500，用于引导节点的 Ironic 镜像可能无法与 Ironic 检查 pod 通信，检查将失败。如果发生了这种情况，安装会停止，因为节点无法用于安装。

16.2.4.3. 配置 NIC

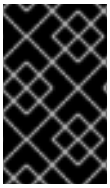
OpenShift Container Platform 使用两个网络部署：

- provisioning ： 置备 网络是一个可选的不可路由的网络，用于在作为 OpenShift Container Platform 集群一部分的每个节点上置备底层操作系统。每个集群节点上 provisioning 网络的网络接口必须将 BIOS 或 UEFI 配置为 PXE 引导。

  provisioningNetworkInterface 配置设置指定 control plane 节点上的 provisioning 网络 NIC 名称，它在 control plane 节点上必须相同。bootMACAddress 配置设置提供了一种方法，用于在每个节点上为 provisioning 网络指定特定的 NIC。

  调配 网络是可选的，但 PXE 引导需要该网络。如果您在没有 provisioning 网络的情况下部署，则必须使用虚拟介质 BMC 寻址选项，如 redfish-virtualmedia 或 idrac-virtualmedia。

- baremetal ： baremetal 网络是一个可路由的网络。您可以使用任何 NIC 与 baremetal 网络 进行接口，只要 NIC 没有配置为使用 provisioning 网络。



重要

在使用 VLAN 时，每个 NIC 必须位于与相应网络对应的独立 VLAN 中。

16.2.4.4. DNS 要求

客户端通过 baremetal 网络访问 OpenShift Container Platform 集群节点。网络管理员必须配置子域或子区，其中规范名称扩展是集群名称。

```
<cluster_name>.<base_domain>
```

例如：

```
test-cluster.example.com
```

OpenShift Container Platform 包含使用集群成员资格信息来生成 A/AAAA 记录的功能。这会将节点名称解析为其 IP 地址。使用 API 注册节点后，集群可以在不使用 CoreDNS-mDNS 的情况下分散节点信息。这可消除与多播 DNS 关联的网络流量。

在 OpenShift Container Platform 部署中，以下组件需要 DNS 名称解析：

- The Kubernetes API
- OpenShift Container Platform 应用程序通配符入口 API

A/AAAA 记录用于名称解析，而 PTR 记录用于反向名称解析。Red Hat Enterprise Linux CoreOS(RHCOS)使用反向记录或 DHCP 为所有节点设置主机名。

安装程序置备的安装包括使用集群成员资格信息生成 A/AAAA 记录的功能。这会将节点名称解析为其 IP 地址。在每个记录中，<cluster_name> 是集群名称，<base_domain> 是您在 install-config.yaml 文件中指定的基域。完整的 DNS 记录采用以下形式： <component>.<cluster_name>.<base_domain>.。

表 16.4. 所需的 DNS 记录

组件	记录	描述
Kubernetes API	**api.<cluster_name>.<base_domain>.**	A/AAAA 记录和 PTR 记录可识别 API 负载均衡器。这些记录必须由集群外的客户端和集群中的所有节点解析。
Routes	***.apps.<cluster_name>.<base_domain>.**	通配符 A/AAAA 记录指的是应用程序入口负载均衡器。应用程序入口负载均衡器以运行 Ingress Controller pod 的节点为目标。默认情况下，Ingress Controller pod 在 worker 节点上运行。这些记录必须由集群外的客户端和集群中的所有节点解析。 例如，console **-openshift-console.apps.<cluster_name>.<base_domain>** 用作到 OpenShift Container Platform 控制台的通配符路由。

提示

您可以使用 dig 命令验证 DNS 解析。

16.2.4.5. 动态主机配置协议(DHCP)要求

默认情况下，安装程序置备的安装会在 provisioning 网络启用了 DHCP 的情况下部署 ironic-dnsmasq。当 provisioningNetwork 配置设置为 managed 时（默认值）,不能有其他 DHCP 服务器在 provisioning 网络中运行。如果您在 provisioning 网络上运行 DHCP 服务器，则必须在 install-config.yaml 文件中将 provisioningNetwork 配置设置设置为 非受管。

网络管理员必须为 OpenShift Container Platform 集群中的各个节点为外部 DHCP 服务器上的

baremetal 网络 保留 IP 地址。

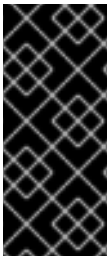
16.2.4.6. 使用 DHCP 服务器为节点保留 IP 地址

对于 baremetal 网络，网络管理员必须保留多个 IP 地址，包括：

1. 两个唯一的虚拟 IP 地址。
   - API 端点的一个虚拟 IP 地址。
   - 一个用于通配符入口端点的虚拟 IP 地址。
2. 一个用于 provisioner 节点的 IP 地址。
3. 每个 control plane 节点有一个 IP 地址。
4. 每个 worker 节点有一个 IP 地址（如果适用）。

保留 IP 地址，以便其成为静态 IP 地址

有些管理员更喜欢使用静态 IP 地址，以便在没有 DHCP 服务器时每个节点的 IP 地址保持恒定状态。要使用 NMState 配置静态 IP 地址，请参阅"设置 OpenShift 安装环境"部分中的"（可选）配置节点网络接口"。



外部负载均衡器和 CONTROL PLANE 节点间的网络

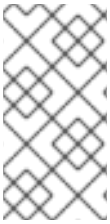
当使用 VLAN 在负载均衡服务和 control plane 节点之间路由流量时，外部负载平衡服务和 control plane 节点必须在同一个 L2 网络上运行，并使用 VLAN 来路由负载平衡服务和 control plane 节点之间的流量。

重要

存储接口需要 DHCP 保留或静态 IP。

下表提供了完全限定域名的实例化。API 和 Nameserver 地址以规范名称扩展开头。control plane 和 worker 节点的主机名是示例，您可以使用您喜欢的任何主机命名规则。

用法	主机名	IP
API	**api.<cluster_name>.<base_domain>**	**<ip>**
Ingress LB(apps)	***.apps.<cluster_name>.<base_domain>**	**<ip>**
provisioner 节点	**provisioner.<cluster_name>.<base_domain>**	**<ip>**
control-plane-0	**openshift-control-plane-0.<cluster_name>.<base_domain>**	**<ip>**
Control-plane-1	**openshift-control-plane-1.<cluster_name>-.<base_domain>**	**<ip>**
Control-plane-2	**openshift-control-plane-2.<cluster_name>.<base_domain>**	**<ip>**
Worker-0	**openshift-worker-0.<cluster_name>.<base_domain>**	**<ip>**
Worker-1	**openshift-worker-1.<cluster_name>.<base_domain>**	**<ip>**
Worker-n	**openshift-worker-n.<cluster_name>.<base_domain>**	**<ip>**



注意

如果您不创建 DHCP 保留，安装程序需要反向 DNS 解析来为 Kubernetes API 节点、provisioner 节点、control plane 节点和 worker 节点设置主机名。

16.2.4.7. provisioner 节点要求

您必须在安装配置中为 provisioner 节点指定 MAC 地址。bootMacAddress 规范通常与 PXE 网络引导关联。但是，Ironic 置备服务还需要 bootMacAddress 规格，以便在检查集群期间或集群中重新部署节点时识别节点。

provisioner 节点需要第 2 层连接才能进行网络引导、DHCP 和 DNS 解析，以及本地网络通信。provisioner 节点需要第 3 层连接才能进行虚拟介质引导。

16.2.4.8. 网络时间协议(NTP)

集群中的每个 OpenShift Container Platform 节点都必须有权访问 NTP 服务器。OpenShift Container Platform 节点使用 NTP 来同步其时钟。例如，集群节点使用需要验证的 SSL 证书，如果节点之间的日期和时间未同步，则可能会失败。

重要

在每个群集节点的 BIOS 设置中定义一致的时钟日期和时间格式，或者安装可能会失败。

您可以重新配置 control plane 节点，作为断开连接的集群中的 NTP 服务器，并重新配置 worker 节点以从 control plane 节点检索时间。

16.2.4.9. 带外管理 IP 地址的端口访问

带外管理 IP 地址位于与节点分开的网络中。为确保带外管理可以在安装过程中与 provisioner 节点通信，带外管理 IP 地址必须被授予对 provisioner 节点和 OpenShift Container Platform control plane 节点上端口 6180 的访问权限。虚拟介质安装需要 TLS 端口 6183，例如使用 Redfish。

其他资源

- 使用 DNS 转发

16.2.5. 配置节点

在使用 provisioning 网络时配置节点

集群中的每个节点都需要以下配置才能正确安装。

警告

在节点间不匹配将导致安装失败。

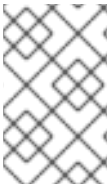
虽然集群节点可以包含多于 2 个 NIC，但安装过程只关注于前两个 NIC:在下表中，NIC1 是一个不可

路由的网络（置备），它仅用于安装 OpenShift Container Platform 集群。

NIC	网络	VLAN
NIC1	**provisioning**	**<provisioning_vlan>**
NIC2	**baremetal**	**<baremetal_vlan>**

provisioner 节点上的 Red Hat Enterprise Linux(RHEL) 9.x 安装过程可能会有所不同。要使用本地 Satellite 服务器或 PXE 服务器安装 Red Hat Enterprise Linux(RHEL) 9.x，PXE 启用 NIC2。

PXE	引导顺序
NIC1 PXE-enabled **provisioning** 网络	1
NIC2 **baremetal** 网络.启用 PXE 是可选的。	2



注意

确保在所有其他 NIC 上禁用 PXE。

配置 control plane 和 worker 节点，如下所示：

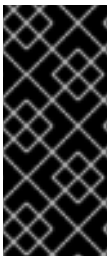
PXE	引导顺序
NIC1 PXE-enabled（调配网络）	1

在没有 provisioning 网络的情况下配置节点

安装过程需要一个 NIC：

NIC	网络	VLAN
NICx	**baremetal**	**<baremetal_vlan>**

NICx 是一个可路由的网络(baremetal)，用于安装 OpenShift Container Platform 集群，并可路由到互联网。



重要

调配网络是可选的，但 PXE 引导需要该网络。如果您在没有 provisioning 网络的情况下部署，则必须使用虚拟介质 BMC 寻址选项，如 redfish-virtualmedia 或 idrac-virtualmedia。

为安全引导手动配置节点

安全引导可防止节点引导，除非它只验证节点只使用可信软件，如 UEFI 固件驱动程序、EFI 应用程序和操作系统。

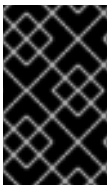
注意

红帽仅在使用 Redfish 虚拟介质部署时支持手动配置安全引导机制。

要手动启用安全引导，请参阅节点的硬件指南并执行以下内容：

流程

1. 引导节点并进入 BIOS 菜单。
2. 将节点的引导模式设置为 UEFI Enabled。
3. 启用安全引导.



重要

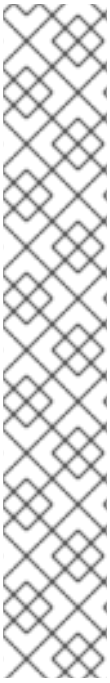
红帽不支持使用自生成的密钥进行安全引导。

16.2.6. 带外管理

节点通常有一个额外的 NIC，供 Baseboard Management Controller(BMC)使用。这些 BMC 必须可以从 provisioner 节点访问。

每个节点必须通过带外管理进行访问。在使用带外管理网络时，provisioner 节点需要访问带外管理网络才能成功安装 OpenShift Container Platform。

带外管理设置超出了本文档的范围。使用单独的管理网络进行带外管理可能会提高性能并提高安全性。但是，使用 provisioning 网络或 baremetal 网络是有效的选项。



注意

bootstrap 虚拟机具有最多两个网络接口。如果您为带外管理配置一个单独的管理网络，并且您使用 provisioning 网络，bootstrap 虚拟机需要通过其中一个网络接口路由到管理网络。在这种情况下，bootstrap 虚拟机可以访问三个网络：

- 裸机网络
- provisioning 网络
- 通过其中一个网络接口的管理网络

16.2.7. 安装所需的数据

在安装 OpenShift Container Platform 集群前，从所有集群节点收集以下信息：

- 带外管理 IP
  - 示例
    - Dell(iDRAC)IP
    - HP(iLO)IP
    - Fujitsu(iRMC)IP

在使用 provisioning 网络时

- NIC（置备）MAC 地址
- NIC（裸机）MAC 地址

在省略 provisioning 网络时

- NIC（裸机）MAC 地址

16.2.8. 节点验证清单

在使用 provisioning 网络时

❑ 为 provisioning 网络配置了 NIC1 VLAN。

❑ 置备网络的 NIC1 在 provisioner、control plane 和 worker 节点上启用了 PXE。

❑ 为 baremetal 网络 配置了 NIC2 VLAN。

❑ PXE 在所有其他 NIC 上都被禁用。

❑ DNS 被配置为使用 API 和 Ingress 端点。

❑ 已配置 control plane 和 worker 节点。

❑ 所有节点都可通过带外管理访问。

❑ （可选）已创建一个单独的管理网络。

❑ 安装所需的数据。

在省略 provisioning 网络时

❑ 为 baremetal 网络配置了 NIC1 VLAN。

❑ DNS 被配置为使用 API 和 Ingress 端点。

❑ 已配置 control plane 和 worker 节点。

❑ 所有节点都可通过带外管理访问。

❑ （可选）已创建一个单独的管理网络。

❑ 安装所需的数据。

16.3. 为 OPENSHIFT 安装设置环境

16.3.1. 在置备程序节点上安装 RHEL

完成先决条件后，下一步是在置备程序节点上安装 RHEL 9.x。安装 OpenShift Container Platform 集群时，安装程序使用 provisioner 节点作为编配器。在本文档中，在 provisioner 节点上安装 RHEL 超出了范围。但是，选项包括但不限于使用 RHEL Satellite 服务器、PXE 或安装介质。

16.3.2. 为 OpenShift Container Platform 安装准备 provisioner 节点

执行以下步骤准备环境。

流程

1. 通过 ssh 登录到 provisioner 节点。
2. 创建非 root 用户(kni)并为该用户提供 sudo 权限：

```
# useradd kni
# passwd kni
```

```
# echo "kni ALL=(root) NOPASSWD:ALL" | tee -a /etc/sudoers.d/kni
```

```
# chmod 0440 /etc/sudoers.d/kni
```

3. 为新用户创建 ssh 密钥：

```
# su - kni -c "ssh-keygen -t ed25519 -f /home/kni/.ssh/id_rsa -N ''"
```

4. 以新用户身份登录到 provisioner 节点：

```
# su - kni
```

5. 使用 Red Hat Subscription Manager 注册 provisioner 节点：

```
$ sudo subscription-manager register --username=<user> --password=<pass> --auto-attach
```

```
$ sudo subscription-manager repos --enable=rhel-9-for-<architecture>-appstream-rpms --enable=rhel-9-for-<architecture>-baseos-rpms
```

注意

有关 Red Hat Subscription Manager 的详情，请参考 使用和配置 Red Hat Subscription Manager。

6. 安装以下软件包：

```
$ sudo dnf install -y libvirt qemu-kvm mkisofs python3-devel jq ipmitool
```

7. 修改用户，将 libvirt 组添加到新创建的用户：

```
$ sudo usermod --append --groups libvirt <user>
```

8. 重启 firewalld 并启用 http 服务：

```
$ sudo systemctl start firewalld
```

```
$ sudo firewall-cmd --zone=public --add-service=http --permanent
```

```
$ sudo firewall-cmd --reload
```

9. 启动并启用 libvirtd 服务：

```
$ sudo systemctl enable libvirtd --now
```

10. 创建 默认 存储池并启动它：

```
$ sudo virsh pool-define-as --name default --type dir --target /var/lib/libvirt/images
```

```
$ sudo virsh pool-start default
```

```
$ sudo virsh pool-autostart default
```

11. 创建 pull-secret.txt 文件：

```
$ vim pull-secret.txt
```

在 Web 浏览器中，进入 Install OpenShift on Bare Metal with installer-provisioned infrastructure。点 Copy pull secret。将内容粘贴到 pull-secret.txt 文件中，并将内容保存到 kni 用户的主目录中。

16.3.3. 检查 NTP 服务器同步

OpenShift Container Platform 安装程序在集群节点上安装 chrony 网络时间协议(NTP)服务。要完成安装，每个节点都必须有权访问 NTP 时间服务器。您可以使用 chrony 服务验证 NTP 服务器同步。

对于断开连接的集群，您必须在 control plane 节点上配置 NTP 服务器。如需更多信息，请参阅*附加资源部分*。

先决条件

- 在目标节点上安装了 chrony 软件包。

流程

1. 使用 ssh 命令登录节点。

2. 运行以下命令，查看节点可用的 NTP 服务器：

```
$ chronyc sources
```

输出示例

```
MS Name/IP address         Stratum Poll Reach LastRx Last sample
===============================================================================
^+ time.cloudflare.com           3  10   377   187   -209us[ -209us] +/-   32ms
^+ t1.time.ir2.yahoo.com         2  10   377   185  -4382us[-4382us] +/-   23ms
^+ time.cloudflare.com           3  10   377   198   -996us[-1220us] +/-   33ms
^* brenbox.westnet.ie            1  10   377   193  -9538us[-9761us] +/-   24ms
```

3. 使用 ping 命令确保节点可以访问 NTP 服务器，例如：

```
$ ping time.cloudflare.com
```

输出示例

```
PING time.cloudflare.com (162.159.200.123) 56(84) bytes of data.
64 bytes from time.cloudflare.com (162.159.200.123): icmp_seq=1 ttl=54 time=32.3 ms
64 bytes from time.cloudflare.com (162.159.200.123): icmp_seq=2 ttl=54 time=30.9 ms
64 bytes from time.cloudflare.com (162.159.200.123): icmp_seq=3 ttl=54 time=36.7 ms
...
```

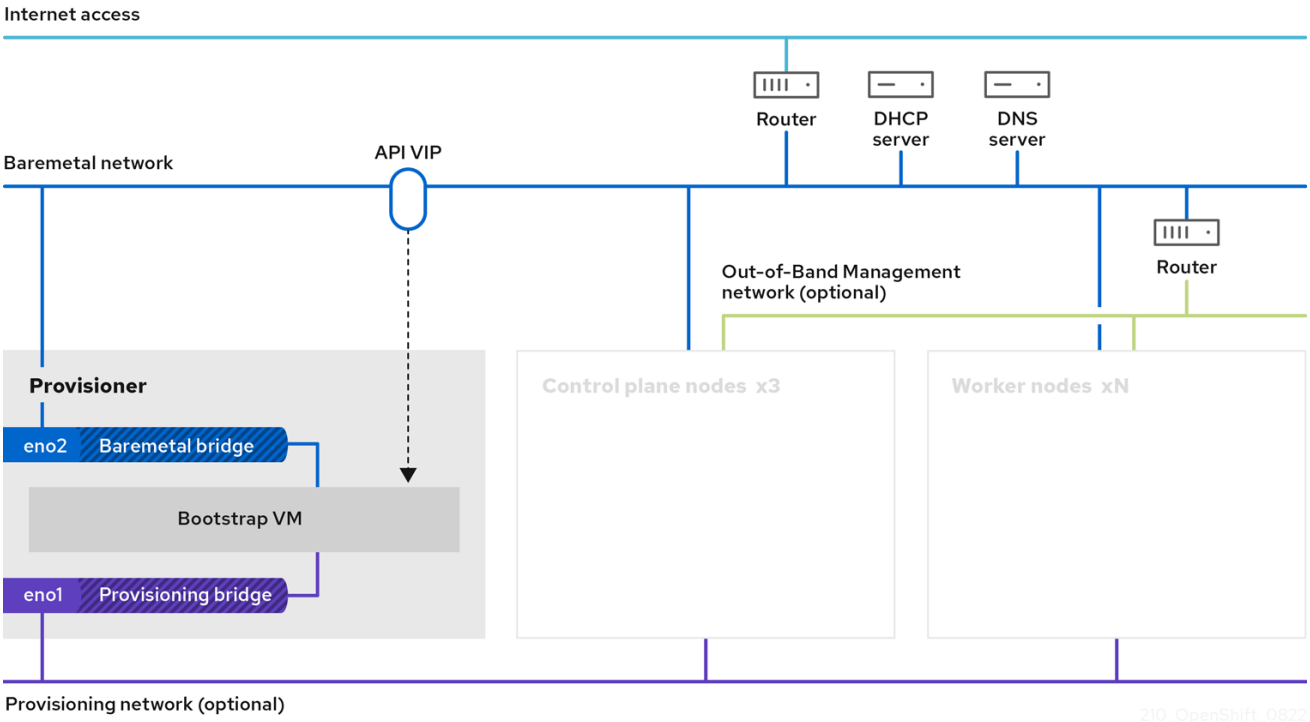
其他资源

- 可选：为断开连接的集群配置 NTP

- 网络时间协议(NTP)

16.3.4. 配置网络

在安装前，您必须在 provisioner 节点上配置网络。安装程序置备的集群使用裸机网桥和网络部署，以及可选的 provisioning 网桥和网络。



注意

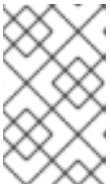
您还可以从 Web 控制台配置网络。

流程

1. 导出裸机网络 NIC 名称：

```
$ export PUB_CONN=<baremetal_nic_name>
```

2. 配置裸机网络：



注意

执行这些步骤后 SSH 连接可能会断开。

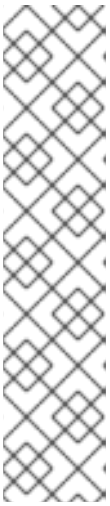
```
$ sudo nohup bash -c "
    nmcli con down \"$PUB_CONN\"
    nmcli con delete \"$PUB_CONN\"
    # RHEL 8.1 appends the word \"System\" in front of the connection, delete in case it
exists
    nmcli con down \"System $PUB_CONN\"
    nmcli con delete \"System $PUB_CONN\"
    nmcli connection add ifname baremetal type bridge con-name baremetal bridge.stp
no
    nmcli con add type bridge-slave ifname \"$PUB_CONN\" master baremetal
    pkill dhclient;dhclient baremetal
"
```

3. 可选：如果您使用 provisioning 网络部署，请导出 provisioning 网络 NIC 名称：

```
$ export PROV_CONN=<prov_nic_name>
```

4. 可选： 如果要使用 provisioning 网络部署，请配置 provisioning 网络：

```
$ sudo nohup bash -c "
    nmcli con down \"$PROV_CONN\"
    nmcli con delete \"$PROV_CONN\"
    nmcli connection add ifname provisioning type bridge con-name provisioning
    nmcli con add type bridge-slave ifname \"$PROV_CONN\" master provisioning
    nmcli connection modify provisioning ipv6.addresses fd00:1101::1/64 ipv6.method
manual
    nmcli con down provisioning
    nmcli con up provisioning
"
```



注意

执行这些步骤后 ssh 连接可能会断开。

只要无法通过 bare-meta 网络路由，IPv6 地址可以是任何地址。

在使用 IPv6 地址时，请确保启用了 UEFI，并将 UEFI PXE 设置设置为 IPv6 协议。

5. 可选：如果您使用 provisioning 网络部署，请在 provisioning 网络连接中配置 IPv4 地址：

```
$ nmcli connection modify provisioning ipv4.addresses 172.22.0.254/24 ipv4.method
manual
```

6. ssh 到 provisioner 节点（如果需要）。

```
# ssh kni@provisioner.<cluster-name>.<domain>
```

7. 验证连接网桥是否已正确创建：

```
$ sudo nmcli con show
```

```
NAME               UUID                                  TYPE      DEVICE
baremetal          4d5133a5-8351-4bb9-bfd4-3af264801530  bridge    baremetal
provisioning       43942805-017f-4d7d-a2c2-7cb3324482ed  bridge    provisioning
virbr0             d9bca40f-eee1-410b-8879-a2d4bb0465e7  bridge    virbr0
bridge-slave-eno1  76a8ed50-c7e5-4999-b4f6-6d9014dd0812  ethernet  eno1
bridge-slave-eno2  f31c3353-54b7-48de-893a-02d2b34c4736  ethernet  eno2
```

16.3.5. 在子网间建立通信

在典型的 OpenShift Container Platform 集群设置中，所有节点（包括 control plane 和计算节点）都驻留在同一网络中。但是，对于边缘计算场景，定位接近边缘的计算节点会很有用。这通常涉及将不同的网络段或子网用于与 control plane 和本地计算节点使用的子网不同的网络段或子网。此类设置可以降低边缘的延迟并允许增强的可扩展性。

在安装 OpenShift Container Platform 前，您必须正确配置网络，以确保包含远程节点的边缘子网可以访问包含 control plane 节点的子网，并从 control plane 接收流量。

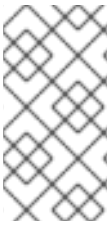
您可以通过配置用户管理的负载均衡器来代替默认负载均衡器，在同一子网或多个子网中运行 control plane 节点。使用多个子网环境，您可以降低 OpenShift Container Platform 集群因为硬件故障或网络中断而失败的问题。如需更多信息，请参阅"配置用户管理的负载均衡器"和"配置用户管理的负载均衡器"。

在多个子网环境中运行 control plane 节点需要完成以下关键任务：

- 通过在 install-config.yaml 文件的 loadBalancer.type 参数中指定 UserManaged 来配置用户管理的负载均衡器，而不是默认的负载均衡器。

- 在 install-config.yaml 文件的 ingressVIPs 和 apiVIPs 参数中配置用户管理的负载均衡器地址。

- 将多个子网无类别域间路由 (CIDR) 和用户管理的负载均衡器 IP 地址添加到 install-config.yaml 文件中的 networking.machineNetworks 参数。



注意

使用多个子网部署集群需要使用虚拟介质，如 redfish-virtualmedia 和 idrac-virtualmedia。

此流程详细介绍了允许第二个子网中的远程计算节点与第一个子网中的 control plane 节点有效通信所需的网络配置，并允许第一个子网中的 control plane 节点与第二个子网中的远程 worker 节点有效通信。

在此过程中，集群跨越两个子网：

- 第一个子网 (10.0.0.0) 包含 control plane 和本地计算节点。

- 第二个子网 (192.168.0.0) 包含边缘计算节点。

流程

1. 配置第一个子网与第二个子网通信：

   a. 运行以下命令，以 root 用户身份登录 control plane 节点：

   ```
   $ sudo su -
   ```

   b. 运行以下命令，获取网络接口的名称：

   ```
   # nmcli dev status
   ```

c. 运行以下命令，通过网关向第二个子网(192.168.0.0)添加路由：

```
# nmcli connection modify <interface_name> +ipv4.routes "192.168.0.0/24 via
<gateway>"
```

将 <interface_name> 替换为接口名称。使用实际网关的 IP 地址替换 <gateway>。

示例

```
# nmcli connection modify eth0 +ipv4.routes "192.168.0.0/24 via 192.168.0.1"
```

d. 运行以下命令来应用更改：

```
# nmcli connection up <interface_name>
```

将 <interface_name> 替换为接口名称。

e. 验证路由表以确保路由已被成功添加：

```
# ip route
```

f. 对第一个子网中的每个 control plane 节点重复前面的步骤。

注意

调整命令以匹配您的实际接口名称和网关。

2. 将第二个子网配置为与第一个子网通信：

a. 运行以下命令，以 root 用户身份登录远程计算节点：

```
$ sudo su -
```

b. 运行以下命令，获取网络接口的名称：

```
# nmcli dev status
```

c. 运行以下命令，通过网关向第一个子网(10.0.0.0)添加路由：

```
# nmcli connection modify <interface_name> +ipv4.routes "10.0.0.0/24 via
<gateway>"
```

将 <interface_name> 替换为接口名称。使用实际网关的 IP 地址替换 <gateway>。

示例

```
# nmcli connection modify eth0 +ipv4.routes "10.0.0.0/24 via 10.0.0.1"
```

d. 运行以下命令来应用更改：

```
# nmcli connection up <interface_name>
```

将 <interface_name> 替换为接口名称。

e. 运行以下命令，验证路由表以确保路由已被成功添加：

```
# ip route
```

f. 对第二个子网中的每一计算节点重复前面的步骤。

注意

调整命令以匹配您的实际接口名称和网关。

3. 配置网络后，测试连接以确保远程节点可以访问 control plane 节点，control plane 节点可以访问远程节点。

a. 从第一个子网中的 control plane 节点，运行以下命令来 ping 第二个子网中的远程节点：

```
$ ping <remote_node_ip_address>
```

如果 ping 成功，则意味着第一个子网中的 control plane 节点可以访问第二个子网中的远程节点。如果您没有收到响应，请检查网络配置并重复该节点的步骤。

b. 在第二个子网中的远程节点中，运行以下命令来 ping 第一个子网中的 control plane 节点：

```
$ ping <control_plane_node_ip_address>
```

如果 ping 成功，则意味着第二个子网中的远程计算节点可以访问第一个子网中的 control plane。如果您没有收到响应，请检查网络配置并重复该节点的步骤。

16.3.6. 检索 OpenShift Container Platform 安装程序

使用安装程序的 stable-4.x 版本和您选择的架构来部署 OpenShift Container Platform 的一般稳定版本：

```
$ export VERSION=stable-4.16
```

```
$ export RELEASE_ARCH=<architecture>
```

```
$ export RELEASE_IMAGE=$(curl -s https://mirror.openshift.com/pub/openshift-v4/$RELEASE_ARCH/clients/ocp/$VERSION/release.txt | grep 'Pull From: quay.io' | awk -F ' ' '{print $3}')
```

16.3.7. 提取 OpenShift Container Platform 安装程序

在获取安装程序后，下一步是提取它。

流程

1. 设置环境变量：

```
$ export cmd=openshift-baremetal-install
```

```
$ export pullsecret_file=~/pull-secret.txt
```

```
$ export extract_dir=$(pwd)
```

2. 获取 oc 二进制文件：

```
$ curl -s https://mirror.openshift.com/pub/openshift-
v4/clients/ocp/$VERSION/openshift-client-linux.tar.gz | tar zxvf - oc
```

3. 解压安装程序：

```
$ sudo cp oc /usr/local/bin
```

```
$ oc adm release extract --registry-config "${pullsecret_file}" --command=$cmd --to
"${extract_dir}" ${RELEASE_IMAGE}
```

```
$ sudo cp openshift-baremetal-install /usr/local/bin
```

16.3.8. 可选：创建 RHCOS 镜像缓存

要使用镜像缓存，您必须下载 bootstrap 虚拟机使用的 Red Hat Enterprise Linux CoreOS(RHCOS) 镜像，以置备集群节点。镜像缓存是可选的，但在有限带宽的网络中运行安装程序时特别有用。

注意

安装程序不再需要 clusterOSImage RHCOS 镜像，因为正确的镜像位于发行版本有效负载中。

如果您在带有有限带宽的网络中运行安装程序，且 RHCOS 镜像下载时间超过 15 到 20 分钟，安装程序会超时。在这种情况下，将映像缓存到 Web 服务器上将有所帮助。

警告

如果为 HTTPD 服务器启用 TLS，您必须确认 root 证书由客户端信任的颁发机构签名，并验证 OpenShift Container Platform hub 和 spoke 集群和 HTTPD 服务器之间的可信证书链。使用配置了不受信任的证书的服务器可防止将镜像下载到创建镜像中。不支持使用不受信任的 HTTPS 服务器。

安装包含镜像的容器。

流程

1. 安装 podman：

```
$ sudo dnf install -y podman
```

2. 打开防火墙端口 8080 以用于 RHCOS 镜像缓存：

```
$ sudo firewall-cmd --add-port=8080/tcp --zone=public --permanent
```

```
$ sudo firewall-cmd --reload
```

3. 创建用于存储 bootstraposimage 的目录：

```
$ mkdir /home/kni/rhcos_image_cache
```

4. 为新创建的目录设置适当的 SELinux 上下文：

```
$ sudo semanage fcontext -a -t httpd_sys_content_t
"/home/kni/rhcos_image_cache(/.*)?"
```

```
$ sudo restorecon -Rv /home/kni/rhcos_image_cache/
```

5. 获取安装程序要在 bootstrap 虚拟机上部署的 RHCOS 镜像的 URI：

```
$ export RHCOS_QEMU_URI=$(/usr/local/bin/openshift-baremetal-install coreos print-stream-json | jq -r --arg ARCH "$(arch)" '.architectures[$ARCH].artifacts.qemu.formats["qcow2.gz"].disk.location')
```

6. 获取安装程序要在 bootstrap 虚拟机上部署的镜像名称：

```
$ export RHCOS_QEMU_NAME=${RHCOS_QEMU_URI##*/}
```

7. 获取要在 bootstrap 虚拟机上部署的 RHCOS 镜像的 SHA 哈希：

```
$ export RHCOS_QEMU_UNCOMPRESSED_SHA256=$(/usr/local/bin/openshift-baremetal-install coreos print-stream-json | jq -r --arg ARCH "$(arch)" '.architectures[$ARCH].artifacts.qemu.formats["qcow2.gz"].disk["uncompressed-sha256"]')
```

8. 下载镜像并将其放在 /home/kni/rhcos_image_cache 目录中：

```
$ curl -L ${RHCOS_QEMU_URI} -o /home/kni/rhcos_image_cache/${RHCOS_QEMU_NAME}
```

9. 为新文件确认 SELinux 类型为 httpd_sys_content_t ：

```
$ ls -Z /home/kni/rhcos_image_cache
```

10. 创建 pod：

```
$ podman run -d --name rhcos_image_cache \ (1)
-v /home/kni/rhcos_image_cache:/var/www/html \
-p 8080:8080/tcp \
registry.access.redhat.com/ubi9/httpd-24
```

(1) 创建名为 rhcos_image_cache 的缓存 webserver。此 pod 在 install-config.yaml 文件中为部署提供 bootstrapOSImage 镜像。

11. 生成 bootstrapOSImage 配置：

```
$ export BAREMETAL_IP=$(ip addr show dev baremetal | awk '/inet /{print $2}' | cut -d"/" -f1)
```

```
$ export BOOTSTRAP_OS_IMAGE="http://${BAREMETAL_IP}:8080/${RHCOS_QEMU_NAME}?sha256=${RHCOS_QEMU_UNCOMPRESSED_SHA256}"
```

```
$ echo "    bootstrapOSImage=${BOOTSTRAP_OS_IMAGE}"
```

12. 在 platform. baremetal 下将所需的配置添加到 install-config.yaml 文件中：

```
platform:
  baremetal:
    bootstrapOSImage: <bootstrap_os_image> 1
```

1 将 <bootstrap_os_image> 替换为 $BOOTSTRAP_OS_IMAGE 的值。

如需了解更多详细信息，请参阅"配置 install-config.yaml 文件"部分。

16.3.9. 用户管理的负载均衡器的服务

您可以将 OpenShift Container Platform 集群配置为使用用户管理的负载均衡器来代替默认负载均衡器。

重要

配置用户管理的负载均衡器取决于您的厂商的负载均衡器。

本节中的信息和示例仅用于指导目的。有关供应商负载均衡器的更多信息，请参阅供应商文档。

红帽支持用户管理的负载均衡器的以下服务：

- Ingress Controller
- OpenShift API
- OpenShift MachineConfig API

您可以选择是否要为用户管理的负载均衡器配置一个或多个所有服务。仅配置 Ingress Controller 服务是一个通用的配置选项。要更好地了解每个服务，请查看以下图表：

图 16.1. 显示 OpenShift Container Platform 环境中运行的 Ingress Controller 的网络工作流示例

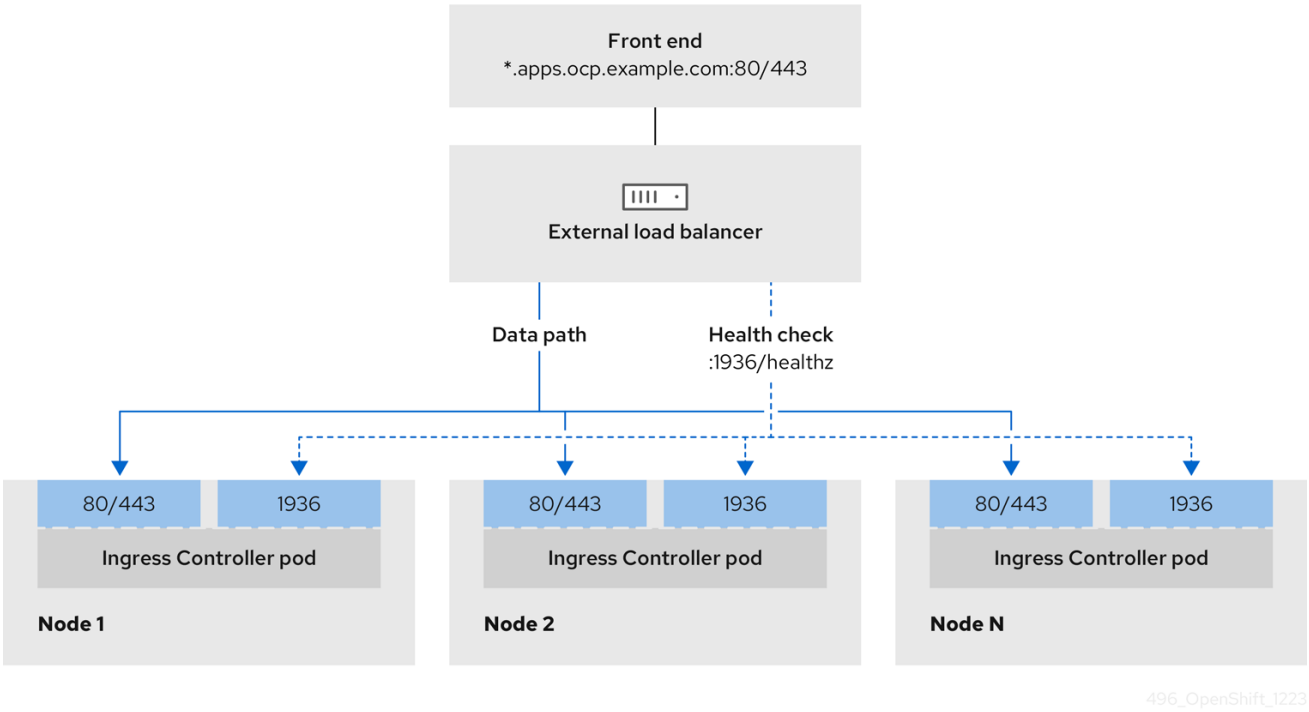


图 16.2. 显示 OpenShift Container Platform 环境中运行的 OpenShift API 的网络工作流示例

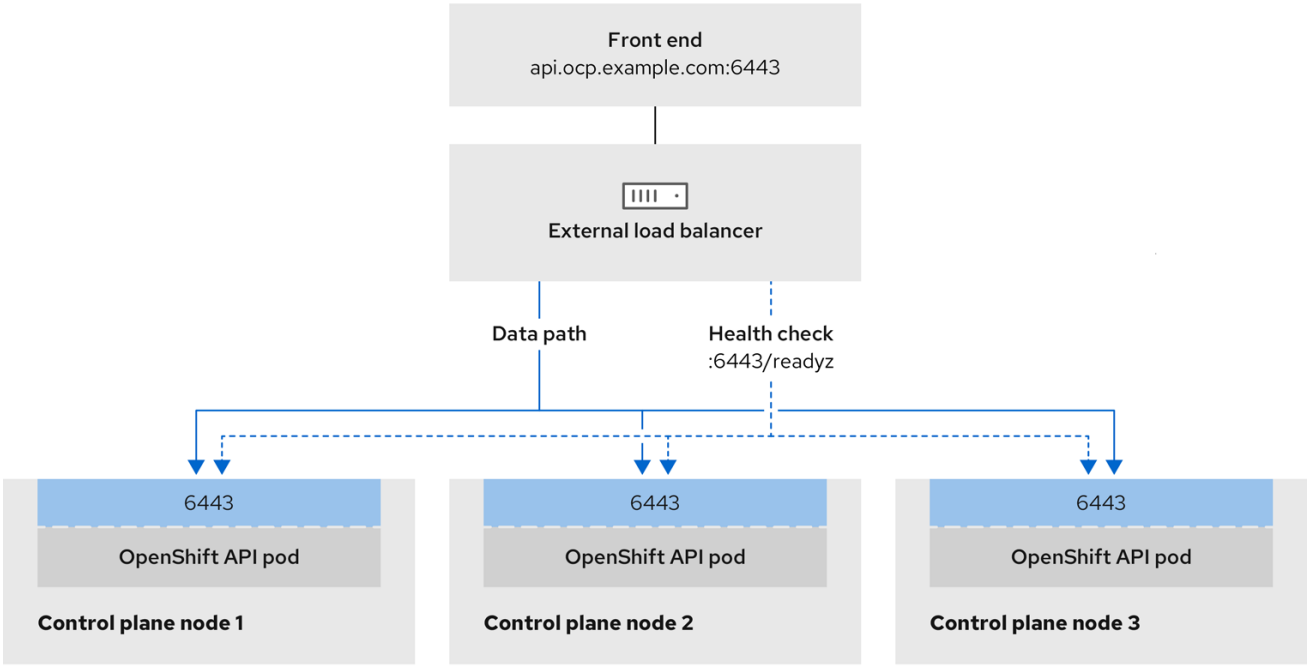
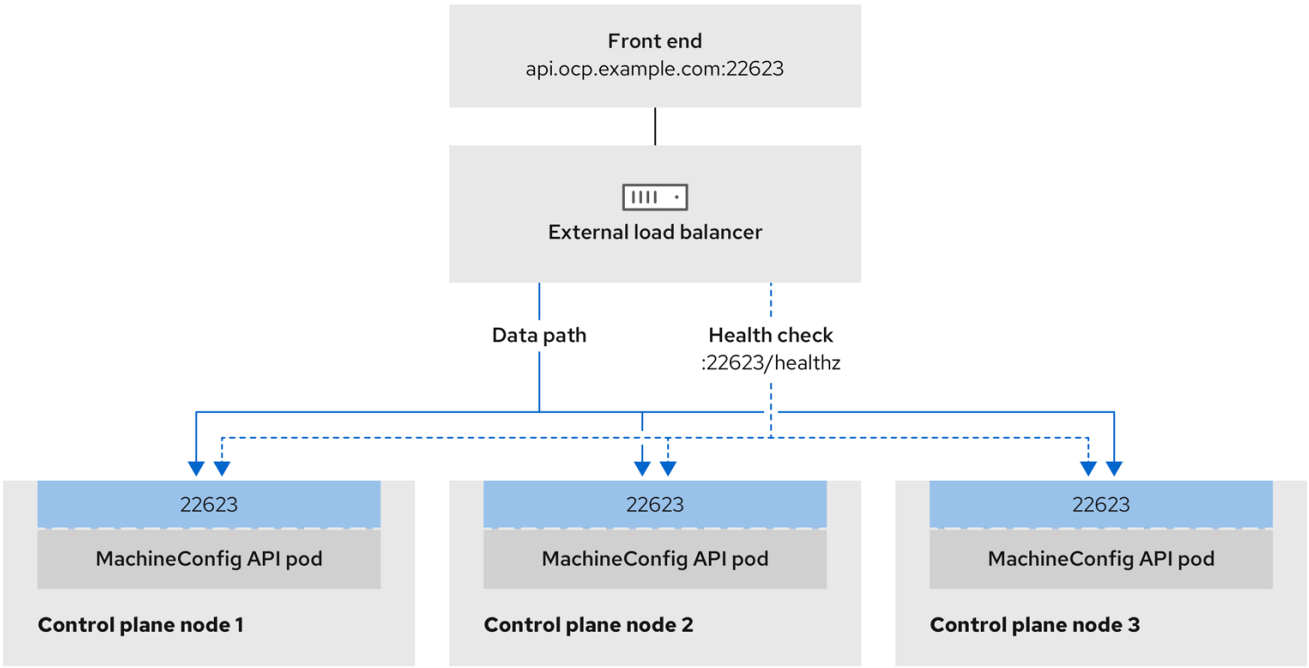


图 16.3. 显示 OpenShift Container Platform 环境中运行的 OpenShift MachineConfig API 的网络工作流示例



用户管理的负载均衡器支持以下配置选项：

- 使用节点选择器将 Ingress Controller 映射到一组特定的节点。您必须为这个集合中的每个节点分配一个静态 IP 地址，或者将每个节点配置为从动态主机配置协议(DHCP)接收相同的 IP 地

址。基础架构节点通常接收这种类型的配置。

- 以子网上的所有 IP 地址为目标。此配置可减少维护开销，因为您可以在这些网络中创建和销毁节点，而无需重新配置负载均衡器目标。如果您使用较小的网络上的机器集来部署入口 pod，如 /27 或 /28，您可以简化负载均衡器目标。

  提示

  您可以通过检查机器配置池的资源来列出网络中存在的所有 IP 地址。

在为 OpenShift Container Platform 集群配置用户管理的负载均衡器前，请考虑以下信息：

- 对于前端 IP 地址，您可以对前端 IP 地址、Ingress Controller 的负载均衡器和 API 负载均衡器使用相同的 IP 地址。查看厂商的文档以获取此功能的相关信息。

- 对于后端 IP 地址，请确保 OpenShift Container Platform control plane 节点的 IP 地址在用户管理的负载均衡器生命周期内不会改变。您可以通过完成以下操作之一来实现此目的：

  - 为每个 control plane 节点分配一个静态 IP 地址。

  - 将每个节点配置为在每次节点请求 DHCP 租期时从 DHCP 接收相同的 IP 地址。根据供应商，DHCP 租期可能采用 IP 保留或静态 DHCP 分配的形式。

- 在 Ingress Controller 后端服务的用户管理的负载均衡器中手动定义运行 Ingress Controller 的每个节点。例如，如果 Ingress Controller 移到未定义节点，则可能会出现连接中断。

16.3.9.1. 配置用户管理的负载均衡器

您可以将 OpenShift Container Platform 集群配置为使用用户管理的负载均衡器来代替默认负载均衡器。

重要

在配置用户管理的负载均衡器前，请确保阅读用户管理的负载均衡器部分。

阅读适用于您要为用户管理的负载均衡器配置的服务的以下先决条件。

注意

MetalLB，在集群中运行，充当用户管理的负载均衡器。

OpenShift API 的先决条件

- 您定义了前端 IP 地址。
- TCP 端口 6443 和 22623 在负载均衡器的前端 IP 地址上公开。检查以下项：
  - 端口 6443 提供对 OpenShift API 服务的访问。
  - 端口 22623 可以为节点提供 ignition 启动配置。
- 前端 IP 地址和端口 6443 可以被您的系统的所有用户访问，其位置为 OpenShift Container Platform 集群外部。
- 前端 IP 地址和端口 22623 只能被 OpenShift Container Platform 节点访问。
- 负载均衡器后端可以在端口 6443 和 22623 上与 OpenShift Container Platform control plane 节点通信。

Ingress Controller 的先决条件

- 您定义了前端 IP 地址。

- TCP 端口 443 和 80 在负载均衡器的前端 IP 地址上公开。

- 前端 IP 地址、端口 80 和端口 443 可以被您的系统所有用户访问，以及 OpenShift Container Platform 集群外部的位置。

- 前端 IP 地址、端口 80 和端口 443 可被 OpenShift Container Platform 集群中运行的所有节点访问。

- 负载均衡器后端可以在端口 80、443 和 1936 上与运行 Ingress Controller 的 OpenShift Container Platform 节点通信。

健康检查 URL 规格的先决条件

您可以通过设置健康检查 URL 来配置大多数负载均衡器，以确定服务是否可用或不可用。OpenShift Container Platform 为 OpenShift API、Machine Configuration API 和 Ingress Controller 后端服务提供这些健康检查。

以下示例显示了之前列出的后端服务的健康检查规格：

Kubernetes API 健康检查规格示例

```
Path: HTTPS:6443/readyz
Healthy threshold: 2
Unhealthy threshold: 2
Timeout: 10
Interval: 10
```

Machine Config API 健康检查规格示例

```
Path: HTTPS:22623/healthz
Healthy threshold: 2
Unhealthy threshold: 2
Timeout: 10
Interval: 10
```

Ingress Controller 健康检查规格示例

```
Path: HTTP:1936/healthz/ready
Healthy threshold: 2
Unhealthy threshold: 2
Timeout: 5
Interval: 10
```

流程

1. **配置 HAProxy Ingress Controller，以便您可以在端口 6443、22623、443 和 80 上从负载均衡器访问集群。根据您的需要，您可以在 HAProxy 配置中指定来自多个子网的单个子网或 IP 地址的 IP 地址。**

   带有列出子网的 HAProxy 配置示例

```
# ...
listen my-cluster-api-6443
    bind 192.168.1.100:6443
    mode tcp
    balance roundrobin
  option httpchk
  http-check connect
  http-check send meth GET uri /readyz
  http-check expect status 200
    server my-cluster-master-2 192.168.1.101:6443 check inter 10s rise 2 fall 2
    server my-cluster-master-0 192.168.1.102:6443 check inter 10s rise 2 fall 2
    server my-cluster-master-1 192.168.1.103:6443 check inter 10s rise 2 fall 2

listen my-cluster-machine-config-api-22623
    bind 192.168.1.100:22623
    mode tcp
    balance roundrobin
  option httpchk
  http-check connect
  http-check send meth GET uri /healthz
  http-check expect status 200
    server my-cluster-master-2 192.168.1.101:22623 check inter 10s rise 2 fall 2
    server my-cluster-master-0 192.168.1.102:22623 check inter 10s rise 2 fall 2
    server my-cluster-master-1 192.168.1.103:22623 check inter 10s rise 2 fall 2
```

```
listen my-cluster-apps-443
    bind 192.168.1.100:443
    mode tcp
    balance roundrobin
  option httpchk
  http-check connect
  http-check send meth GET uri /healthz/ready
  http-check expect status 200
    server my-cluster-worker-0 192.168.1.111:443 check port 1936 inter 10s rise 2 fall 2
    server my-cluster-worker-1 192.168.1.112:443 check port 1936 inter 10s rise 2 fall 2
    server my-cluster-worker-2 192.168.1.113:443 check port 1936 inter 10s rise 2 fall 2

listen my-cluster-apps-80
   bind 192.168.1.100:80
   mode tcp
   balance roundrobin
  option httpchk
  http-check connect
  http-check send meth GET uri /healthz/ready
  http-check expect status 200
    server my-cluster-worker-0 192.168.1.111:80 check port 1936 inter 10s rise 2 fall 2
    server my-cluster-worker-1 192.168.1.112:80 check port 1936 inter 10s rise 2 fall 2
    server my-cluster-worker-2 192.168.1.113:80 check port 1936 inter 10s rise 2 fall 2
# ...
```

带有多个列出子网的 HAProxy 配置示例

```
# ...
listen api-server-6443
    bind *:6443
    mode tcp
      server master-00 192.168.83.89:6443 check inter 1s
      server master-01 192.168.84.90:6443 check inter 1s
      server master-02 192.168.85.99:6443 check inter 1s
      server bootstrap 192.168.80.89:6443 check inter 1s

listen machine-config-server-22623
    bind *:22623
    mode tcp
      server master-00 192.168.83.89:22623 check inter 1s
      server master-01 192.168.84.90:22623 check inter 1s
      server master-02 192.168.85.99:22623 check inter 1s
      server bootstrap 192.168.80.89:22623 check inter 1s

listen ingress-router-80
    bind *:80
    mode tcp
    balance source
```

```
    server worker-00 192.168.83.100:80 check inter 1s
    server worker-01 192.168.83.101:80 check inter 1s

listen ingress-router-443
  bind *:443
  mode tcp
  balance source
    server worker-00 192.168.83.100:443 check inter 1s
    server worker-01 192.168.83.101:443 check inter 1s

listen ironic-api-6385
  bind *:6385
  mode tcp
  balance source
    server master-00 192.168.83.89:6385 check inter 1s
    server master-01 192.168.84.90:6385 check inter 1s
    server master-02 192.168.85.99:6385 check inter 1s
    server bootstrap 192.168.80.89:6385 check inter 1s

listen inspector-api-5050
  bind *:5050
  mode tcp
  balance source
    server master-00 192.168.83.89:5050 check inter 1s
    server master-01 192.168.84.90:5050 check inter 1s
    server master-02 192.168.85.99:5050 check inter 1s
    server bootstrap 192.168.80.89:5050 check inter 1s
# ...
```

2. 使用 curl CLI 命令验证用户管理的负载均衡器及其资源是否正常运行：

   a. 运行以下命令并查看响应，验证集群机器配置 API 是否可以被 Kubernetes API 服务器资源访问：

   ```
   $ curl https://<loadbalancer_ip_address>:6443/version --insecure
   ```

   如果配置正确，您会收到 JSON 对象的响应：

   ```
   {
     "major": "1",
     "minor": "11+",
     "gitVersion": "v1.11.0+ad103ed",
     "gitCommit": "ad103ed",
     "gitTreeState": "clean",
     "buildDate": "2019-01-09T06:44:10Z",
     "goVersion": "go1.10.3",
   ```

```
  "compiler": "gc",
  "platform": "linux/amd64"
}
```

b. 运行以下命令并观察输出，验证集群机器配置 API 是否可以被 Machine 配置服务器资源访问：

```
$ curl -v https://<loadbalancer_ip_address>:22623/healthz --insecure
```

如果配置正确，命令的输出会显示以下响应：

```
HTTP/1.1 200 OK
Content-Length: 0
```

c. 运行以下命令并观察输出，验证控制器是否可以被端口 80 上的 Ingress Controller 资源访问：

```
$ curl -I -L -H "Host: console-openshift-console.apps.<cluster_name>.
<base_domain>" http://<load_balancer_front_end_IP_address>
```

如果配置正确，命令的输出会显示以下响应：

```
HTTP/1.1 302 Found
content-length: 0
location: https://console-openshift-console.apps.ocp4.private.opequon.net/
cache-control: no-cache
```

d. 运行以下命令并观察输出，验证控制器是否可以被端口 443 上的 Ingress Controller 资源访问：

```
$ curl -I -L --insecure --resolve console-openshift-console.apps.<cluster_name>.
<base_domain>:443:<Load Balancer Front End IP Address> https://console-
openshift-console.apps.<cluster_name>.<base_domain>
```

如果配置正确，命令的输出会显示以下响应：

```
HTTP/1.1 200 OK
referrer-policy: strict-origin-when-cross-origin
set-cookie: csrf-
token=UlYWOyQ62LWjw2h003xtYSKlh1a0Py2hhctw0WmV2YEdhJjFyQwWcGBsja2
61dGLgaYO0nxzVErhiXt6QepA7g==; Path=/; Secure; SameSite=Lax
x-content-type-options: nosniff
```

```
x-dns-prefetch-control: off
x-frame-options: DENY
x-xss-protection: 1; mode=block
date: Wed, 04 Oct 2023 16:29:38 GMT
content-type: text/html; charset=utf-8
set-cookie:
1e2670d92730b515ce3a1bb65da45062=1bf5e9573c9a2760c964ed1659cc1673;
path=/; HttpOnly; Secure; SameSite=None
cache-control: private
```

3. 配置集群的 DNS 记录，使其以用户管理的负载均衡器的前端 IP 地址为目标。您必须在负载均衡器上将记录更新为集群 API 和应用程序的 DNS 服务器。

修改 DNS 记录示例

```
<load_balancer_ip_address>  A  api.<cluster_name>.<base_domain>
A record pointing to Load Balancer Front End
```

```
<load_balancer_ip_address>   A apps.<cluster_name>.<base_domain>
A record pointing to Load Balancer Front End
```

重要

DNS 传播可能需要一些时间才能获得每个 DNS 记录。在验证每个记录前，请确保每个 DNS 记录传播。

4. 要使 OpenShift Container Platform 集群使用用户管理的负载均衡器，您必须在集群的 install-config.yaml 文件中指定以下配置：

```
# ...
platform:
  baremetal:
    loadBalancer:
      type: UserManaged 1
      apiVIPs:
      - <api_ip> 2
      ingressVIPs:
      - <ingress_ip> 3
# ...
```

1

为 type 参数设置 UserManaged，为集群指定用户管理的负载均衡器。参数默认为 OpenShiftManagedDefault，它表示默认的内部负载均衡器。对于 openshift-kni-infra 命名空间中定义的服务，用户管理的负载均衡器可将 coredns 服务部署到集群中的 pod，但忽略 keepalived 和 haproxy 服务。

2

指定用户管理的负载均衡器时所需的参数。指定用户管理的负载均衡器的公共 IP 地址，以便 Kubernetes API 可以与用户管理的负载均衡器通信。

3

指定用户管理的负载均衡器时所需的参数。指定用户管理的负载均衡器的公共 IP 地址，以便用户管理的负载均衡器可以管理集群的入口流量。

验证

1. 使用 curl CLI 命令验证用户管理的负载均衡器和 DNS 记录配置是否正常工作：

   a. 运行以下命令并查看输出，验证您可以访问集群 API：

   ```
   $ curl https://api.<cluster_name>.<base_domain>:6443/version --insecure
   ```

   如果配置正确，您会收到 JSON 对象的响应：

   ```
   {
    "major": "1",
    "minor": "11+",
    "gitVersion": "v1.11.0+ad103ed",
    "gitCommit": "ad103ed",
    "gitTreeState": "clean",
    "buildDate": "2019-01-09T06:44:10Z",
    "goVersion": "go1.10.3",
    "compiler": "gc",
    "platform": "linux/amd64"
    }
   ```

   b. 运行以下命令并查看输出，验证您可以访问集群机器配置：

   ```
   $ curl -v https://api.<cluster_name>.<base_domain>:22623/healthz --insecure
   ```

如果配置正确，命令的输出会显示以下响应：

```
HTTP/1.1 200 OK
Content-Length: 0
```

c. 运行以下命令并查看输出，验证您可以在端口上访问每个集群应用程序：

```
$ curl http://console-openshift-console.apps.<cluster_name>.<base_domain> -I -L
--insecure
```

如果配置正确，命令的输出会显示以下响应：

```
HTTP/1.1 302 Found
content-length: 0
location: https://console-openshift-console.apps.<cluster-name>.<base domain>/
cache-control: no-cacheHTTP/1.1 200 OK
referrer-policy: strict-origin-when-cross-origin
set-cookie: csrf-
token=39HoZgztDnzjJkq/JuLJMeoKNXlfiVv2YgZc09c3TBOBU4Nl6kDXaJH1LdicNh
N1UsQWzon4Dor9GWGfopaTEQ==; Path=/; Secure
x-content-type-options: nosniff
x-dns-prefetch-control: off
x-frame-options: DENY
x-xss-protection: 1; mode=block
date: Tue, 17 Nov 2020 08:42:10 GMT
content-type: text/html; charset=utf-8
set-cookie:
1e2670d92730b515ce3a1bb65da45062=9b714eb87e93cf34853e87a92d6894be;
path=/; HttpOnly; Secure; SameSite=None
cache-control: private
```

d. 运行以下命令并查看输出，验证您可以在端口 443 上访问每个集群应用程序：

```
$ curl https://console-openshift-console.apps.<cluster_name>.<base_domain> -I -
L --insecure
```

如果配置正确，命令的输出会显示以下响应：

```
HTTP/1.1 200 OK
referrer-policy: strict-origin-when-cross-origin
set-cookie: csrf-
token=UlYWOyQ62LWjw2h003xtYSKlh1a0Py2hhctw0WmV2YEdhJjFyQwWcGBsja2
61dGLgaYO0nxzVErhiXt6QepA7g==; Path=/; Secure; SameSite=Lax
x-content-type-options: nosniff
```

```
x-dns-prefetch-control: off
x-frame-options: DENY
x-xss-protection: 1; mode=block
date: Wed, 04 Oct 2023 16:29:38 GMT
content-type: text/html; charset=utf-8
set-cookie:
1e2670d92730b515ce3a1bb65da45062=1bf5e9573c9a2760c964ed1659cc1673;
path=/; HttpOnly; Secure; SameSite=None
cache-control: private
```

16.3.10. 通过 DHCP 设置集群节点主机名

在 Red Hat Enterprise Linux CoreOS (RHCOS)机器上，主机名通过 NetworkManager 设置。默认情况下，机器通过 DHCP 获取其主机名。如果 DHCP 没有提供主机名，请通过内核参数或者其它方法静态设置，它们通过反向 DNS 查找来获取。反向 DNS 查找在网络初始化后进行，可能需要一些时间来解决。其他系统服务可以在此之前启动，并将主机名检测为 localhost 或类似的内容。您可以使用 DHCP 为每个集群节点提供主机名，以避免在分配主机名时避免此延迟。

另外，通过 DHCP 设置主机名可以绕过实施 DNS split-horizon 的环境中的手动 DNS 记录名称配置错误。

16.3.11. 配置 install-config.yaml 文件

16.3.11.1. 配置 install-config.yaml 文件

install-config.yaml 文件需要一些额外的详情。大多数信息都教授安装程序，生成的集群有足够的集群来完全管理它。

注意

安装程序不再需要 clusterOSImage RHCOS 镜像，因为正确的镜像位于发行版本有效负载中。

1. 配置 install-config.yaml。更改适当的变量以匹配环境，包括 pullSecret 和 sshKey ：

```
apiVersion: v1
baseDomain: <domain>
metadata:
  name: <cluster_name>
networking:
  machineNetwork:
  - cidr: <public_cidr>
  networkType: OVNKubernetes
```

```
compute:
- name: worker
  replicas: 2 (1)
controlPlane:
  name: master
  replicas: 3
  platform:
    baremetal: {}
platform:
  baremetal:
    apiVIPs:
      - <api_ip>
    ingressVIPs:
      - <wildcard_ip>
    provisioningNetworkCIDR: <CIDR>
    bootstrapExternalStaticIP: <bootstrap_static_ip_address> (2)
    bootstrapExternalStaticGateway: <bootstrap_static_gateway> (3)
    bootstrapExternalStaticDNS: <bootstrap_static_dns> (4)
    hosts:
      - name: openshift-master-0
        role: master
        bmc:
          address: ipmi://<out_of_band_ip> (5)
          username: <user>
          password: <password>
        bootMACAddress: <NIC1_mac_address>
        rootDeviceHints:
         deviceName: "<installation_disk_drive_path>" (6)
      - name: <openshift_master_1>
        role: master
        bmc:
          address: ipmi://<out_of_band_ip>
          username: <user>
          password: <password>
        bootMACAddress: <NIC1_mac_address>
        rootDeviceHints:
         deviceName: "<installation_disk_drive_path>"
      - name: <openshift_master_2>
        role: master
        bmc:
          address: ipmi://<out_of_band_ip>
          username: <user>
          password: <password>
        bootMACAddress: <NIC1_mac_address>
        rootDeviceHints:
         deviceName: "<installation_disk_drive_path>"
      - name: <openshift_worker_0>
        role: worker
        bmc:
          address: ipmi://<out_of_band_ip>
          username: <user>
          password: <password>
        bootMACAddress: <NIC1_mac_address>
      - name: <openshift_worker_1>
        role: worker
```

```
      bmc:
        address: ipmi://<out_of_band_ip>
        username: <user>
        password: <password>
      bootMACAddress: <NIC1_mac_address>
      rootDeviceHints:
       deviceName: "<installation_disk_drive_path>"
pullSecret: '<pull_secret>'
sshKey: '<ssh_pub_key>'
```

1 根据作为 OpenShift Container Platform 集群一部分的计算节点数量扩展计算机器。replicas 值可以是 0，以及大于或等于 2 的整数。将副本数设置为 0 以部署一个三节点集群，该集群仅包含三个 control plane 机器。三节点集群是一个较小的、效率更高的集群，可用于测试、开发和生产。您不能只使用一个计算节点安装集群。

2 当使用静态 IP 地址部署集群时，您必须设置 bootstrapExternalStaticIP 配置设置，以便在裸机网络上没有 DHCP 服务器时可以指定 bootstrap 虚拟机的静态 IP 地址。

3 当使用静态 IP 地址部署集群时，您必须设置 bootstrapExternalStaticGateway 配置设置，以便当裸机网络上没有 DHCP 服务器时可以为 bootstrap 虚拟机指定网关 IP 地址。

4 当使用静态 IP 地址部署集群时，您必须设置 bootstrapExternalStaticDNS 配置设置，以便在裸机网络上没有 DHCP 服务器时指定 bootstrap 虚拟机的 DNS 地址。

5 如需了解更多选项，请参阅 BMC 寻址部分。

6 要设置安装磁盘驱动器的路径，请输入磁盘的内核名称。例如：/dev/sda。

重要

由于磁盘发现顺序无法保证，因此磁盘的内核名称可以在具有多个磁盘的机器的引导选项之间更改。例如： /dev/sda 变为 /dev/sdb，反之亦然。要避免这个问题，您必须使用持久性磁盘属性，如 disk World Wide Name (WWN) 或 /dev/disk/by-path/。建议您使用 /dev/disk/by-path/<device_path> 链接到存储位置。要使用磁盘 WWN，请将 deviceName 参数替换为 wwnWithExtension 参数。根据您使用的参数，输入以下值之一：

- **磁盘名称。例如： /dev/sda 或 /dev/disk/by-path/。**
- **磁盘 WWN。例如，"0x64cd98f04fde100024684cf3034da5c2"。确保您在引号中输入磁盘 WWN 值，使其用作字符串值，而不是十六进制值。**

无法满足 rootDeviceHints 参数的这些要求可能会导致以下错误：

```
ironic-inspector inspection failed: No disks satisfied root device hints
```

注意

在 OpenShift Container Platform 4.12 之前，集群安装程序只接受 apiVIP 和 ingressVIP 配置设置的 IPv4 地址或 IPv6 地址。在 OpenShift Container Platform 4.12 及更新的版本中，这些配置设置已弃用。反之，使用 apiVIPs 和 ingressVIPs 配置设置中的列表格式来指定 IPv4 地址、IPv6 地址或两个 IP 地址格式。

2. **创建用于存储集群配置的目录：**

```
$ mkdir ~/clusterconfigs
```

3. **将 install-config.yaml 文件复制到新目录中：**

```
$ cp install-config.yaml ~/clusterconfigs
```

4. 在安装 OpenShift Container Platform 集群前，请确保关闭所有裸机节点：

```
$ ipmitool -I lanplus -U <user> -P <password> -H <management-server-ip> power off
```

5. 如果以前的部署尝试中保留了旧的 bootstrap 资源，请删除旧的 bootstrap 资源：

```
for i in $(sudo virsh list | tail -n +3 | grep bootstrap | awk {'print $2'});
do
  sudo virsh destroy $i;
  sudo virsh undefine $i;
  sudo virsh vol-delete $i --pool $i;
  sudo virsh vol-delete $i.ign --pool $i;
  sudo virsh pool-destroy $i;
  sudo virsh pool-undefine $i;
done
```

16.3.11.2. 其他 install-config 参数

下表列出了 install-config.yaml 文件所需的 参数、hosts 参数和 bmc 参数。

表 16.5. 所需的参数

参数	default	描述
baseDomain		集群的域名。例如：**example.com**。
bootMode	**UEFI**	节点的引导模式。选项为 **legacy**、**U** EFI 和 **UEFISecureBoot**。如果没有设置 **bootMode**，Ironic 在检查节点时设置它。
bootstrapExternalStaticDNS		bootstrap 节点的静态网络 DNS。当裸机网络上没有动态主机配置协议 (DHCP) 服务器时，您必须使用静态 IP 地址部署集群时设置这个值。如果没有设置这个值，安装程序将使用 **bootstrapExternalStaticGateway** 的值，这会导致在网关和 DNS 的 IP 地址值不同时出现问题。
bootstrapExternalStaticIP		bootstrap 虚拟机的静态 IP 地址。当 bare-metal 网络中没有 DHCP 服务器时部署使用静态 IP 地址的集群时，必须设置这个值。
bootstrapExternalStaticGateway		bootstrap 虚拟机网关的静态 IP 地址。当 bare-metal 网络中没有 DHCP 服务器时部署使用静态 IP 地址的集群时，必须设置这个值。
sshKey		**sshKey** 配置设置包含 **~/.ssh/id_rsa.pub** 文件中访问 control plane 节点和计算节点所需的密钥。通常，这个密钥来自 **provisioner** 节点。

参数	default	描述
pullSecret		**pullSecret** 配置设置包含准备 provisioner 节点时从Install OpenShift on Bare Metal 页面下载的 pull secret 的副本。
metadata: name:		提供给 OpenShift Container Platform 集群的名称。例如：**openshift**。
networking: machineNetwork: - cidr:		外部网络的公共 CIDR(Classless Inter-Domain Routing)。例如：**10.0.0.0/24**。
compute: - name: worker		OpenShift Container Platform 集群需要为计算节点提供一个名称，即使没有节点也是如此。
compute: replicas: 2		replicas 设置 OpenShift Container Platform 集群中的计算节点数量。
controlPlane: name: master		OpenShift Container Platform 集群需要一个 control plane 节点的名称。
controlPlane: replicas: 3		replicas 设置作为 OpenShift Container Platform 集群一部分的 control plane 节点数量。
provisioningNetworkInterface		连接到 provisioning 网络的节点上的网络接口名称。对于 OpenShift Container Platform 4.9 及更新的版本，使用 **bootMACAddress** 配置设置来启用 Ironic 标识 NIC 的 IP 地址，**而不使用 provisioningNetworkInterface** 配置设置来标识 NIC 的名称。
defaultMachinePlatform		用于没有平台配置的机器池的默认配置。

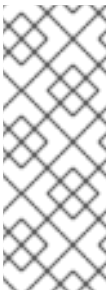
参数	default	描述
apiVIPs		（可选）用于 Kubernetes API 通信的虚拟 IP 地址。 此设置必须在 **install-config.yaml** 文件中作为保留的 IP，或从 DNS 中预先配置到 DNS 中，以便正确解析默认名称。在 **install-config.yaml** 文件中的 **apiVIPs** 配置设置中添加值时，请使用虚拟 IP 地址而不是 FQDN。在使用双栈网络时，主 IP 地址必须来自 IPv4 网络。如果没有设置，安装程序将使用 **api.<cluster_name>.<base_domain>** 从 DNS 派生 IP 地址。  **注意** 在 OpenShift Container Platform 4.12 之前，集群安装程序只接受 **apiVIP** 配置设置的 IPv4 地址或 IPv6 地址。在 OpenShift Container Platform 4.12 或更高版本中，**apiVIP** 配置设置已弃用。反之，使用 **apiVIPs** 配置设置的列表格式来指定 IPv4 地址、IPv6 地址或两个 IP 地址格式。
disableCertificateVerification	**False**	**RedFish** 和 **redfish-virtualmedia** 需要这个参数来管理 BMC 地址。当 BMC 地址使用自签名证书时，这个值应该是 **True**。
ingressVIPs		（可选）入口流量的虚拟 IP 地址。 此设置必须在 **install-config.yaml** 文件中作为保留的 IP，或从 DNS 中预先配置到 DNS 中，以便正确解析默认名称。在 **install-config.yaml** 文件中的 **ingressVIPs** 配置设置中添加值时，请使用虚拟 IP 地址而不是 FQDN。在使用双栈网络时，主 IP 地址必须来自 IPv4 网络。如果没有设置，安装程序将使用 **test.apps.<cluster_name>.<base_domain>** 从 DNS 派生 IP 地址。  **注意** 在 OpenShift Container Platform 4.12 之前，集群安装程序只接受 **ingressVIP** 配置设置的 IPv4 地址或 IPv6 地址。在 OpenShift Container Platform 4.12 及更新的版本中，**ingressVIP** 配置设置已弃用。反之，使用 **ingressVIPs** 配置设置的列表格式来指定 IPv4 地址、IPv6 地址或两个 IP 地址格式。

表 16.6. 可选参数

参数	default	描述
provisioningDHCPRange	**172.22.0.10,172.22.0.100**	定义 provisioning 网络上节点的 IP 范围。
provisioningNetworkCIDR	**172.22.0.0/24**	用于置备的网络的 CIDR。在 provisioning 网络中不使用默认地址范围时需要这个选项。
clusterProvisioningIP	**provisioningNetworkCIDR** 的第三个 IP 地址。	运行置备服务的集群中的 IP 地址。默认为 provisioning 子网的第三个 IP 地址。例如： **172.22.0.3**。
bootstrapProvisioningIP	**provisioningNetworkCIDR** 的第二个 IP 地址。	在安装程序部署 control plane(master)节点时运行置备服务的 bootstrap 虚拟机上的 IP 地址。默认为 provisioning 子网的第二个 IP 地址。例如： **172.22.0.2 或 2620:52:0:1307::2**。
externalBridge	**baremetal**	附加到裸机网络的虚拟机监控程序的裸机网桥名称。
provisioningBridge	**provisioning**	附加到 provisioning 网络的 **provisioner** 主机上的 provisioning 网桥的名称。
架构		定义集群的主机架构。有效值为 **amd64** 或 **arm64**。
defaultMachinePlatform		用于没有平台配置的机器池的默认配置。
bootstrapOSImage		用于覆盖 bootstrap 节点的默认操作系统镜像的 URL。URL 必须包含镜像的 SHA-256 哈希。例如： **https://mirror.openshift.com/rhcos-<version>-qemu.qcow2.gz?sha256=<uncompressed_sha256>**。
provisioningNetwork		**provisioningNetwork** 配置设置决定了集群是否使用 provisioning 网络。如果存在，配置设置也会决定集群是否管理网络。 **disabled** ：将此参数设置为 **Disabled**，以禁用对 provisioning 网络的要求。当设置为 **Disabled** 时，您必须只使用基于虚拟介质的置备，或使用支持的安装程序启动集群。如果 **Disabled** 并使用电源管理，则必须可以从 bare-metal 网络访问 BMC。如果 **Disabled**，则必须在 bare-metal 网络上提供两个用于置备服务的 IP 地址。 **Managed** ：将此参数设置为 **Managed**（默认设置），以全面管理调配网络，包括 DHCP、TFTP 等。 **Unmanaged** ：将此参数 **设置为 Unmanaged** 以启用调配网络，但需要手动配置 DHCP。建议进行虚拟介质调配，但在需要时仍可使用 PXE。
httpProxy		将此参数设置为环境中使用的适当 HTTP 代理。

参数	default	描述
httpsProxy		将此参数设置为环境中使用的适当 HTTPS 代理。
noProxy		将此参数设置为适合环境中代理使用的排除项。

主机

hosts 参数是用于构建集群的独立裸机资产列表。

表 16.7. 主机

名称	default	描述
名称		与详细信息 **关联的 BareMetalHost** 资源的名称。例如： **openshift-master-0**。
role		裸机节点的角色。**master** (control plane 节点) 或 **worker** （计算节点）。
bmc		基板管理控制器的连接详情。如需了解更多详细信息，请参阅 BMC 寻址部分。
bootMACAddress		主机用于 provisioning 网络的 NIC 的 MAC 地址。Ironic 使用 **bootMACAddress** 配置设置检索 IP 地址。然后，它将绑定到 主机。 **注意** 如果您禁用了 provisioning 网络，则必须从主机提供有效的 MAC 地址。
networkConfig		设置此可选参数来配置主机的网络接口。如需了解更多详细信息，请参阅"（可选）配置主机网络接口"。

16.3.11.3. BMC 地址

大多数供应商支持使用智能平台管理接口(IPMI)寻址的基板管理控制器(BMC)。IPMI 不加密通信。它适用于通过安全或专用管理网络在数据中心内使用。检查您的供应商，了解它们是否支持 Redfish 网络引导。RedFish 为融合、混合型 IT 和软件定义型数据中心(SDDC)提供简单而安全的管理。RedFish 是人类可读的，能够使用通用的互联网和 Web 服务标准将信息直接公开给现代工具链。如果您的硬件不支持 Redfish 网络引导，请使用 IPMI。

如果节点处于 Registering 状态，则可以在安装过程中修改 BMC 地址。当节点已不再为 Registering 状态时，如果需要修改 BMC 地址，您需要首先断开节点与 Ironic 的连接，编辑 BareMetalHost 资源，

并将节点重新连接到 Ironic。详情请参阅*编辑 BareMetalHost 资源*部分。

IPMI

使用 IPMI 的主机使用 ipmi://<out-of-band-ip>:<port> 地址格式，如果未指定则默认为端口 623。以下示例演示了 install-config.yaml 文件中的 IPMI 配置。

```
platform:
  baremetal:
    hosts:
      - name: openshift-master-0
        role: master
        bmc:
          address: ipmi://<out-of-band-ip>
          username: <user>
          password: <password>
```

重要

当 PXE 引导使用 IPMI 进行 BMC 寻址时，需要 provisioning 网络。没有 provisioning 网络，就无法 PXE 引导主机。如果您在没有 provisioning 网络的情况下部署，则必须使用虚拟介质 BMC 寻址选项，如 redfish-virtualmedia 或 idrac-virtualmedia。详情请查看 "Redfish 虚拟介质 for HPE iLO" 部分的"适用于 HPE iLO 的 BMC 寻址"部分或"适用于戴尔 iDRAC 的"红帽虚拟媒体"部分中的"红帽虚拟介质"。

RedFish 网络引导

要启用 Redfish，请使用 redfish:// 或 redfish+http:// 禁用 TLS。安装程序需要主机名或 IP 地址，以及系统 ID 的路径。以下示例演示了 install-config.yaml 文件中的 Redfish 配置。

```
platform:
  baremetal:
    hosts:
      - name: openshift-master-0
        role: master
        bmc:
          address: redfish://<out-of-band-ip>/redfish/v1/Systems/1
          username: <user>
          password: <password>
```

虽然建议为带外管理地址提供颁发机构证书，但在使用自签名证书时，您必须在 bmc 配置中包含 disableCertificateVerification: True。以下示例演示了在 install-config.yaml 文件中使用 disableCertificateVerification: True 配置参数的 Redfish 配置。

```
platform:
  baremetal:
    hosts:
```

```
  - name: openshift-master-0
    role: master
    bmc:
      address: redfish://<out-of-band-ip>/redfish/v1/Systems/1
      username: <user>
      password: <password>
      disableCertificateVerification: True
```

其他资源

- 编辑 BareMetalHost 资源

16.3.11.4. 验证对 Redfish API 的支持

当使用 Redfish API 安装时，安装程序会在裸机上使用安装程序置备的基础架构时，调用基板管理控制器 (BMC) 上的几个 Redfish 端点。如果使用 Redfish，请确保 BMC 在安装前支持所有 Redfish API。

流程

1. 运行以下命令设置 BMC 的 IP 地址或主机名：

```
$ export SERVER=<ip_address> 1
```

1 将 <ip_address> 替换为 BMC 的 IP 地址或主机名。

2. 运行以下命令设定系统的 ID：

```
$ export SystemID=<system_id> 1
```

1 将 <system_id> 替换为系统 ID。例如：System.Embedded.1 或 1。详情请查看以下特定于供应商的 BMC 部分。

Redfish API 列表：

1. 运行以下命令检查 power on 支持：

```
$ curl -u $USER:$PASS -X POST -H'Content-Type: application/json' -H'Accept:
application/json' -d '{"ResetType": "On"}'
https://$SERVER/redfish/v1/Systems/$SystemID/Actions/ComputerSystem.Reset
```

2. 运行以下命令检查 **power off** 支持：

```
$ curl -u $USER:$PASS -X POST -H'Content-Type: application/json' -H'Accept:
application/json' -d '{"ResetType": "ForceOff"}'
https://$SERVER/redfish/v1/Systems/$SystemID/Actions/ComputerSystem.Reset
```

3. 运行以下命令，检查使用 **pxe** 的临时引导实现：

```
$ curl -u $USER:$PASS -X PATCH -H "Content-Type: application/json"
https://$Server/redfish/v1/Systems/$SystemID/ -d '{"Boot":
{"BootSourceOverrideTarget": "pxe", "BootSourceOverrideEnabled": "Once"}}
```

4. 运行以下命令，检查使用 **Legacy** 或 **UEFI** 的 **BIOS** 引导模式的状态：

```
$ curl -u $USER:$PASS -X PATCH -H "Content-Type: application/json"
https://$Server/redfish/v1/Systems/$SystemID/ -d '{"Boot":
{"BootSourceOverrideMode":"UEFI"}}
```

Redfish 虚拟介质 **API** 列表：

1. 运行以下命令，检查使用 **cd** 或 **dvd** 的临时引导设备的功能：

```
$ curl -u $USER:$PASS -X PATCH -H "Content-Type: application/json"
https://$Server/redfish/v1/Systems/$SystemID/ -d '{"Boot":
{"BootSourceOverrideTarget": "cd", "BootSourceOverrideEnabled": "Once"}}'
```

2. 运行以下命令，检查挂载虚拟介质的功能：

```
$ curl -u $USER:$PASS -X PATCH -H "Content-Type: application/json" -H "If-Match: *"
https://$Server/redfish/v1/Managers/$ManagerID/VirtualMedia/$VmediaId -d '{"Image":
"https://example.com/test.iso", "TransferProtocolType": "HTTPS", "UserName": "",
"Password":""}'
```

注意

Redfish API 的 **PowerOn** 和 **PowerOff** 命令与 **Redfish** 虚拟介质 **API** 相同。

重要

HTTPS 和 HTTP 是 TransferProtocolTypes 唯一支持的参数类型。

16.3.11.5. 适用于 Dell iDRAC 的 BMC 寻址

每个 bmc 条目 的地址 字段是连接到 OpenShift Container Platform 集群节点的 URL，包括 URL 方案中的控制器类型及其网络中的位置。

```
platform:
  baremetal:
    hosts:
      - name: <hostname>
        role: <master | worker>
        bmc:
          address: <address> 1
          username: <user>
          password: <password>
```

1

地址 配置设置指定协议。

对于 Dell 硬件，红帽支持集成 Dell Remote Access Controller(iDRAC)虚拟介质、Redfish 网络引导和 IPMI。

Dell iDRAC 的 BMC 地址格式

协议	地址格式
iDRAC 虚拟介质	idrac-virtualmedia://<out-of-band-ip>/redfish/v1/Systems/System.Embedded.1
RedFish 网络引导	redfish://<out-of-band-ip>/redfish/v1/Systems/System.Embedded.1
IPMI	ipmi://<out-of-band-ip>

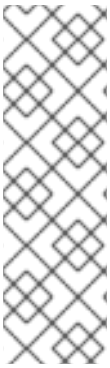
重要

使用 idrac-virtualmedia 作为 Redfish 虚拟介质的协议。RedFish-virtualmedia 无法在 Dell 硬件上运行。Dell 的 idrac-virtualmedia 使用带 Dell OEM 扩展的 Redfish 标准。

详情请查看以下部分。

Dell iDRAC 的 RedFish 虚拟介质

对于 Dell 服务器上的 Redfish 虚拟介质，在 address 设置中使用 idrac-virtualmedia://。使用 redfish-virtualmedia:// 无法正常工作。



注意

使用 idrac-virtualmedia:// 作为 Redfish 虚拟介质的协议。使用 redfish-virtualmedia:// 无法在 Dell 硬件中工作，因为 idrac-virtualmedia:// 协议与 Ironic 中的 idrac 硬件类型和 Redfish 协议对应。Dell 的 idrac-virtualmedia:// 协议使用带有 Dell OEM 扩展的 Redfish 标准。Ironic 还支持带有 WSMAN 协议的 idrac 类型。因此，您必须指定 idrac-virtualmedia://，以避免在 Dell 硬件上选择使用 Redfish 和虚拟介质时出现意外行为。

以下示例演示了在 install-config.yaml 文件中使用 iDRAC 虚拟介质。

```
platform:
  baremetal:
    hosts:
      - name: openshift-master-0
        role: master
        bmc:
          address: idrac-virtualmedia://<out-of-band-ip>/redfish/v1/Systems/System.Embedded.1
          username: <user>
          password: <password>
```

虽然建议为带外管理地址提供颁发机构证书，但在使用自签名证书时，您必须在 bmc 配置中包含 disableCertificateVerification: True。

注意

通过 iDRAC 控制台，确保 OpenShift Container Platform 集群节点启用了 AutoAttach。菜单路径为： Configuration → Virtual Media → Attach Mode → AutoAttach。

以下示例演示了在 install-config.yaml 文件中使用 disableCertificateVerification: True 配置参数的 Redfish 配置。

```
platform:
```

```
  baremetal:
    hosts:
      - name: openshift-master-0
        role: master
        bmc:
          address: idrac-virtualmedia://<out-of-band-ip>/redfish/v1/Systems/System.Embedded.1
          username: <user>
          password: <password>
          disableCertificateVerification: True
```

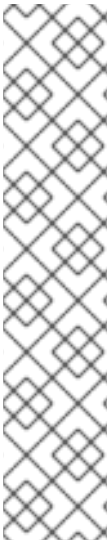
iDRAC 的 RedFish 网络引导

要启用 Redfish，请使用 redfish:// 或 redfish+http:// 禁用传输层安全(TLS)。安装程序需要主机名或 IP 地址，以及系统 ID 的路径。以下示例演示了 install-config.yaml 文件中的 Redfish 配置。

```
platform:
  baremetal:
    hosts:
      - name: openshift-master-0
        role: master
        bmc:
          address: redfish://<out-of-band-ip>/redfish/v1/Systems/System.Embedded.1
          username: <user>
          password: <password>
```

虽然建议为带外管理地址提供颁发机构证书，但在使用自签名证书时，您必须在 bmc 配置中包含 disableCertificateVerification: True。以下示例演示了在 install-config.yaml 文件中使用 disableCertificateVerification: True 配置参数的 Redfish 配置。

```
platform:
  baremetal:
    hosts:
      - name: openshift-master-0
        role: master
        bmc:
          address: redfish://<out-of-band-ip>/redfish/v1/Systems/System.Embedded.1
          username: <user>
          password: <password>
          disableCertificateVerification: True
```



注意

Dell iDRAC 9 中存在一个已知问题，带有固件版本 04.40.00.00，所有版本都包括 5xx 系列，用于裸机部署中的安装程序置备安装。虚拟控制台插件默认为 eHTML5，它是 HTML5 的增强版本，这会导致 InsertVirtualMedia 工作流出现问题。将插件设置为使用 HTML5 以避免出现这个问题。菜单路径为 Configuration → Virtual console → Plug-in Type → HTML5。

通过 iDRAC 控制台，确保 OpenShift Container Platform 集群节点启用了 AutoAttach。菜单路径为： Configuration → Virtual Media → Attach Mode → AutoAttach。

16.3.11.6. HPE iLO 的 BMC 寻址

每个 bmc 条目 的地址 字段是连接到 OpenShift Container Platform 集群节点的 URL，包括 URL 方案中的控制器类型及其网络中的位置。

```
platform:
  baremetal:
    hosts:
      - name: <hostname>
        role: <master | worker>
        bmc:
          address: <address> 1
          username: <user>
          password: <password>
```

1

地址 配置设置指定协议。

对于 HPE 集成 Lights Out(iLO)，红帽支持红帽虚拟媒体、Redfish 网络引导和 IPMI。

表 16.8. HPE iLO 的 BMC 地址格式

协议	地址格式
RedFish 虚拟介质	**redfish-virtualmedia://<out-of-band-ip>/redfish/v1/Systems/1**
RedFish 网络引导	**redfish://<out-of-band-ip>/redfish/v1/Systems/1**
IPMI	**ipmi://<out-of-band-ip>**

详情请查看以下部分。

HPE iLO 的 RedFish 虚拟介质

要为 HPE 服务器启用 Redfish 虚拟介质，请在 address 设置中使用 redfish-virtualmedia://。以下示例演示了在 install-config.yaml 文件中使用 Redfish 虚拟介质。

```
platform:
  baremetal:
    hosts:
      - name: openshift-master-0
        role: master
        bmc:
          address: redfish-virtualmedia://<out-of-band-ip>/redfish/v1/Systems/1
          username: <user>
          password: <password>
```

虽然建议为带外管理地址提供颁发机构证书，但在使用自签名证书时，您必须在 bmc 配置中包含 disableCertificateVerification: True。以下示例演示了在 install-config.yaml 文件中使用 disableCertificateVerification: True 配置参数的 Redfish 配置。

```
platform:
  baremetal:
    hosts:
      - name: openshift-master-0
        role: master
        bmc:
          address: redfish-virtualmedia://<out-of-band-ip>/redfish/v1/Systems/1
          username: <user>
          password: <password>
          disableCertificateVerification: True
```

注意

运行 iLO4 的 9 代系统上不支持 RedFish 虚拟介质，因为 Ironic 不支持使用虚拟介质的 iLO4。

HPE iLO 的 RedFish 网络引导

要启用 Redfish，请使用 redfish:// 或 redfish+http:// 禁用 TLS。安装程序需要主机名或 IP 地址，以及系统 ID 的路径。以下示例演示了 install-config.yaml 文件中的 Redfish 配置。

```
platform:
  baremetal:
    hosts:
      - name: openshift-master-0
```

```
      role: master
      bmc:
        address: redfish://<out-of-band-ip>/redfish/v1/Systems/1
        username: <user>
        password: <password>
```

虽然建议为带外管理地址提供颁发机构证书，但在使用自签名证书时，您必须在 bmc 配置中包含 disableCertificateVerification: True。以下示例演示了在 install-config.yaml 文件中使用 disableCertificateVerification: True 配置参数的 Redfish 配置。

```
platform:
  baremetal:
    hosts:
      - name: openshift-master-0
        role: master
        bmc:
          address: redfish://<out-of-band-ip>/redfish/v1/Systems/1
          username: <user>
          password: <password>
          disableCertificateVerification: True
```

16.3.11.7. Fujitsu iRMC 的 BMC 寻址

每个 bmc 条目 的地址 字段是连接到 OpenShift Container Platform 集群节点的 URL，包括 URL 方案中的控制器类型及其网络中的位置。

```
platform:
  baremetal:
    hosts:
      - name: <hostname>
        role: <master | worker>
        bmc:
          address: <address> 1
          username: <user>
          password: <password>
```

1

地址 配置设置指定协议。

对于富士通硬件，红帽支持集成远程管理控制器(iRMC)和 IPMI。

表 16.9. Fujitsu iRMC 的 BMC 地址格式

协议	地址格式
iRMC	**irmc://<out-of-band-ip>**
IPMI	**ipmi://<out-of-band-ip>**

iRMC

Fujitsu 节点可以使用 irmc://<out-of-band-ip>，默认为端口 443。以下示例演示了 install-config.yaml 文件中的 iRMC 配置。

```
platform:
  baremetal:
    hosts:
      - name: openshift-master-0
        role: master
        bmc:
          address: irmc://<out-of-band-ip>
          username: <user>
          password: <password>
```

注意

目前，Fujitsu 支持 iRMC S5 固件版本 3.05 及更高版本用于裸机上的安装程序置备安装。

16.3.11.8. Root 设备提示

rootDeviceHints 参数可让安装程序将 Red Hat Enterprise Linux CoreOS(RHCOS)镜像置备到特定的设备。安装程序会按照发现设备的顺序检查设备，并将发现的值与 hint 值进行比较。安装程序使用第一个与 hint 值匹配的发现设备。配置可以组合多个 hint，但设备必须与所有提示匹配，以便安装程序进行选择。

表 16.10. 子字段

子字段	描述
deviceName	包含 Linux 设备名称的字符串（如 **/dev/vda** 或 **/dev/disk/by-path/** ）。建议您使用 **/dev/disk/by-path/<device_path>** 链接到存储位置。hint 必须与实际值完全匹配。
hctl	包含类似 **0:0:0:0:0 的 SCSI 总线地址的字符串**。hint 必须与实际值完全匹配。

子字段	描述
model	包含特定厂商的设备标识符的字符串。hint 可以是实际值的子字符串。
vendor	包含该设备厂商或制造商名称的字符串。hint 可以是实际值的子字符串。
serialNumber	包含设备序列号的字符串。hint 必须与实际值完全匹配。
minSizeGigabytes	以 GB 为单位代表设备的最小大小的整数。
wwn	包含唯一存储标识符的字符串。hint 必须与实际值完全匹配。
wwnWithExtension	包含唯一存储标识符的字符串，并附加厂商扩展。hint 必须与实际值完全匹配。
wwnVendorExtension	包含唯一厂商存储标识符的字符串。hint 必须与实际值完全匹配。
rotational	指明该设备为旋转磁盘(true)还是非旋转磁盘(false)的布尔值。

用法示例

```
- name: master-0
  role: master
  bmc:
    address: ipmi://10.10.0.3:6203
    username: admin
    password: redhat
  bootMACAddress: de:ad:be:ef:00:40
  rootDeviceHints:
    deviceName: "/dev/sda"
```

16.3.11.9. 可选：设置代理设置

要使用代理部署 OpenShift Container Platform 集群，请对 **install-config.yaml** 文件进行以下更改。

```
apiVersion: v1
baseDomain: <domain>
proxy:
  httpProxy: http://USERNAME:PASSWORD@proxy.example.com:PORT
  httpsProxy: https://USERNAME:PASSWORD@proxy.example.com:PORT
  noProxy: <WILDCARD_OF_DOMAIN>,<PROVISIONING_NETWORK/CIDR>,
<BMC_ADDRESS_RANGE/CIDR>
```

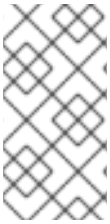
以下是带有值的 noProxy 示例。

```
noProxy: .example.com,172.22.0.0/24,10.10.0.0/24
```

启用代理后，在对应的键/值对中设置代理的适当值。

主要考虑：

- 如果代理没有 HTTPS 代理，请将 httpsProxy 的值从 https :// 改为 http://。
- 如果使用 provisioning 网络，将其包含在 noProxy 设置中，否则安装程序将失败。
- 将所有代理设置设置为 provisioner 节点中的环境变量。例如：HTTP_PROXY、HTTPS_PROXY 和 NO_PROXY。



注意

使用 IPv6 置备时，您无法在 noProxy 设置中定义 CIDR 地址块。您必须单独定义每个地址。

16.3.11.10. 可选：在没有 provisioning 网络的情况下进行部署

要在没有 provisioning 网络的情况下部署 OpenShift Container Platform 集群，请对 install-config.yaml 文件进行以下更改。

```
platform:
  baremetal:
    apiVIPs:
      - <api_VIP>
```

```
ingressVIPs:
  - <ingress_VIP>
provisioningNetwork: "Disabled" 1
```

1

如果需要，添加 provisioningNetwork 配置设置并将其设置为 Disabled。

重要

PXE 引导需要 provisioning 网络。如果您在没有 provisioning 网络的情况下部署，则必须使用虚拟介质 BMC 寻址选项，如 redfish-virtualmedia 或 idrac-virtualmedia。详情请查看 "Redfish 虚拟介质 for HPE iLO" 部分的"适用于 HPE iLO 的 BMC 寻址"部分或"适用于戴尔 iDRAC 的"红帽虚拟媒体"部分中的"红帽虚拟介质"。

16.3.11.11. 可选：使用双栈网络部署

对于 OpenShift Container Platform 集群中的双栈网络，您可以为集群节点配置 IPv4 和 IPv6 地址端点。要为集群节点配置 IPv4 和 IPv6 地址端点，请编辑 install-config.yaml 文件中的 machineNetwork、clusterNetwork 和 serviceNetwork 配置设置。每个设置必须分别有两个 CIDR 条目。对于将 IPv4 系列用作主地址系列的集群，请首先指定 IPv4 设置。对于将 IPv6 系列用作主地址系列的集群，请首先指定 IPv6 设置。

```
machineNetwork:
- cidr: {{ extcidrnet }}
- cidr: {{ extcidrnet6 }}
clusterNetwork:
- cidr: 10.128.0.0/14
  hostPrefix: 23
- cidr: fd02::/48
  hostPrefix: 64
serviceNetwork:
- 172.30.0.0/16
- fd03::/112
```

重要

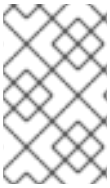
在裸机平台上，如果您在 install-config.yaml 文件的 networkConfig 部分中指定了 NMState 配置，请将 interfaces.wait-ip: ipv4+ipv6 添加到 NMState YAML 文件中，以解决阻止集群在双栈网络上部署的问题。

包含 wait-ip 参数的 NMState YAML 配置文件示例

```
networkConfig:
  nmstate:
    interfaces:
    - name: <interface_name>
# ...
      wait-ip: ipv4+ipv6
# ...
```

要为使用 IPv4 和 IPv6 地址的应用程序提供接口，请为 Ingress VIP 和 API VIP 服务配置 IPv4 和 IPv6 虚拟 IP (VIP) 地址端点。要配置 IPv4 和 IPv6 地址端点，请编辑 install-config.yaml 文件中的 apiVIPs 和 ingressVIPs 配置设置。apiVIPs 和 ingressVIPs 配置设置使用列表格式。列表的顺序决定了每个服务的主 VIP 地址和次 VIP 地址。

```
platform:
  baremetal:
    apiVIPs:
      - <api_ipv4>
      - <api_ipv6>
    ingressVIPs:
      - <wildcard_ipv4>
      - <wildcard_ipv6>
```



注意

对于具有双栈网络配置的集群，您必须将 IPv4 和 IPv6 地址分配到同一接口。

16.3.11.12. 可选：配置主机网络接口

在安装前，您可以在 install-config.yaml 文件中设置 networkConfig 配置设置，以使用 NMState 配置主机网络接口。

此功能的最常见用例是在 bare-metal 网络中指定一个静态 IP 地址，但您也可以配置其他网络，如存储网络。此功能支持 VLAN、VXLAN、网桥、绑定、路由、MTU 和 DNS 解析器设置等其他 NMState 功能。

先决条件

- 使用静态 IP 地址为每个节点配置带有有效主机名的 PTR DNS 记录。
- 安装 NMState CLI (nmstate)。

流程

1. 可选：在 install-config.yaml 文件中包括 nmstatectl gc 前测试 NMState 语法，因为安装程序不会检查 NMState YAML 语法。

注意

YAML 语法中的错误可能会导致无法应用网络配置。另外，在部署后或在扩展集群时应用 Kubernetes NMState 更改时，维护所验证的 YAML 语法会很有用。

a. 创建 NMState YAML 文件：

```
interfaces:
- name: <nic1_name> 1
  type: ethernet
  state: up
  ipv4:
    address:
    - ip: <ip_address> 2
      prefix-length: 24
    enabled: true
dns-resolver:
  config:
    server:
    - <dns_ip_address> 3
routes:
  config:
  - destination: 0.0.0.0/0
    next-hop-address: <next_hop_ip_address> 4
    next-hop-interface: <next_hop_nic1_name> 5
```

1 2 3 4 5

使用适当的值替换 <nic1_name>, <ip_address>, <dns_ip_address>, <next_hop_ip_address> 和 <next_hop_nic1_name>。

b. 运行以下命令来测试配置文件：

```
$ nmstatectl gc <nmstate_yaml_file>
```

将 <nmstate_yaml_file> 替换为配置文件名称。

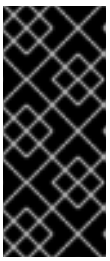
2. 通过在 install-config.yaml 文件中的主机中添加 NMState 配置，使用 networkConfig 配置设置：

```
hosts:
  - name: openshift-master-0
    role: master
    bmc:
      address: redfish+http://<out_of_band_ip>/redfish/v1/Systems/
      username: <user>
      password: <password>
      disableCertificateVerification: null
    bootMACAddress: <NIC1_mac_address>
    bootMode: UEFI
    rootDeviceHints:
      deviceName: "/dev/sda"
    networkConfig: 1
      interfaces:
      - name: <nic1_name> 2
        type: ethernet
        state: up
        ipv4:
          address:
          - ip: <ip_address> 3
            prefix-length: 24
          enabled: true
      dns-resolver:
        config:
          server:
          - <dns_ip_address> 4
      routes:
        config:
        - destination: 0.0.0.0/0
          next-hop-address: <next_hop_ip_address> 5
          next-hop-interface: <next_hop_nic1_name> 6
```

1

2 3 4 5 6

使用适当的值替换 <nic1_name>, <ip_address>, <dns_ip_address>, <next_hop_ip_address> 和 <next_hop_nic1_name>。

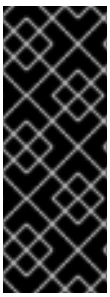


重要

部署集群后，您无法修改 install-config.yaml 文件的 networkConfig 配置设置，以更改主机网络接口。在部署后，使用 Kubernetes NMState Operator 更改主机网络接口。

16.3.11.13. 为子网配置主机网络接口

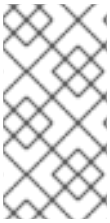
对于边缘计算场景，定位接近边缘的计算节点会很有用。要在子网中定位远程节点，您可以对远程节点使用不同的网络片段或子网，而不是用于 control plane 子网和本地计算节点。您可以通过为边缘计算场景设置子网来减少边缘延迟并允许增强可扩展性。



重要

当使用默认负载均衡器时，OpenShiftManagedDefault 并将远程节点添加到 OpenShift Container Platform 集群中，所有 control plane 节点必须在同一子网中运行。当使用多个子网时，您还可以使用清单将 Ingress VIP 配置为在 control plane 节点上运行。详情请参阅"配置要在 control plane 上运行的网络组件"。

如果您为远程节点创建了不同的网络段或子网，如 "Establishing communication" 部分所述，如果 worker 使用静态 IP 地址、绑定或其他高级网络，您必须在 machineNetwork 配置设置中指定子网。当为每个远程节点在 networkConfig 参数中设置节点 IP 地址时，还必须在使用静态 IP 地址时为包含 control plane 节点的子网指定网关和 DNS 服务器。这样可确保远程节点可以访问包含 control plane 节点的子网，它们可以从 control plane 接收网络流量。



注意

使用多个子网部署集群需要使用虚拟介质，如 redfish-virtualmedia 和 idrac-virtualmedia，因为远程节点无法访问本地置备网络。

流程

1. 在使用静态 IP 地址时，将子网添加到 install-config.yaml 文件中的 machineNetwork 中：

```
networking:
  machineNetwork:
  - cidr: 10.0.0.0/24
  - cidr: 192.168.0.0/24
  networkType: OVNKubernetes
```

2. 在使用静态 IP 地址或高级网络（如绑定）时，使用 NMState 语法将网关和 DNS 配置添加到每个边缘计算节点的 networkConfig 参数中：

```
networkConfig:
  interfaces:
  - name: <interface_name> 1
    type: ethernet
    state: up
    ipv4:
      enabled: true
      dhcp: false
      address:
      - ip: <node_ip> 2
        prefix-length: 24
      gateway: <gateway_ip> 3
      dns-resolver:
        config:
          server:
          - <dns_ip> 4
```

1 将 <interface_name> 替换为接口名称。

2 将 <node_ip> 替换为节点的 IP 地址。

3 使用网关的 IP 地址替换 <gateway_ip>。

4 将 <dns_ip> 替换为 DNS 服务器的 IP 地址。

16.3.11.14. 可选：在双栈网络中为 SLAAC 配置地址生成模式

对于使用 Stateless Address AutoConfiguration (SLAAC)的双栈集群，您必须为 ipv6.addr-gen-mode 网络设置指定一个全局值。您可以使用 NMState 设置这个值来配置 RAM 磁盘和集群配置文件。如

果您没有在这些位置配置一致的 ipv6.addr-gen-mode，则集群中的 CSR 资源和 BareMetalHost 资源之间可能会发生 IPv6 地址不匹配。

先决条件

- 安装 NMState CLI (nmstate)。

流程

1. 可选：在 install-config.yaml 文件中包括 nmstatectl gc 命令前使用 nmstatectl gc 命令测试 NMState YAML 语法，因为安装程序不会检查 NMState YAML 语法。

   a. 创建 NMState YAML 文件：

```
interfaces:
- name: eth0
  ipv6:
    addr-gen-mode: <address_mode> 1
```

   1 将 <address_mode> 替换为集群中 IPv6 地址所需的地址生成模式类型。有效值为 eui64、 stable-privacy 或 random。

   b. 运行以下命令来测试配置文件：

```
$ nmstatectl gc <nmstate_yaml_file> 1
```

   1 将 <nmstate_yaml_file> 替换为测试配置文件的名称。

2. 将 NMState 配置添加到 install-config.yaml 文件中的 hosts.networkConfig 部分：

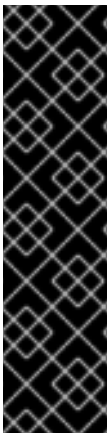
```
hosts:
  - name: openshift-master-0
    role: master
    bmc:
      address: redfish+http://<out_of_band_ip>/redfish/v1/Systems/
      username: <user>
```

```
      password: <password>
      disableCertificateVerification: null
    bootMACAddress: <NIC1_mac_address>
    bootMode: UEFI
    rootDeviceHints:
      deviceName: "/dev/sda"
    networkConfig:
      interfaces:
      - name: eth0
        ipv6:
          addr-gen-mode: <address_mode> 1
...
```

1

将 <address_mode> 替换为集群中 IPv6 地址所需的地址生成模式类型。有效值为 eui64、 stable-privacy 或 random。

16.3.11.15. 可选：为双端口 NIC 配置主机网络接口



重要

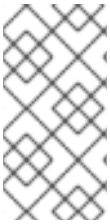
支持与为 SR-IOV 设备启用 NIC 分区关联的第 1 天操作只是一个技术预览功能。技术预览功能不受红帽产品服务等级协议（SLA）支持，且功能可能并不完整。红帽不推荐在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。

有关红帽技术预览功能支持范围的更多信息，请参阅技术预览功能支持范围。

在安装前，您可以在 install-config.yaml 文件中设置 networkConfig 配置设置，以使用 NMState 配置主机网络接口来支持双端口 NIC。

先决条件

- 使用静态 IP 地址为每个节点配置带有有效主机名的 PTR DNS 记录。
- 安装 NMState CLI (nmstate)。



注意

YAML 语法中的错误可能会导致无法应用网络配置。另外，在部署后或在扩展集群时应用 Kubernetes NMState 更改时，维护所验证的 YAML 语法会很有用。

流程

1. 将 NMState 配置添加到 install-config.yaml 文件中的 networkConfig 主机中：

```
hosts:
  - name: worker-0
    role: worker
    bmc:
      address: redfish+http://<out_of_band_ip>/redfish/v1/Systems/
      username: <user>
      password: <password>
      disableCertificateVerification: false
    bootMACAddress: <NIC1_mac_address>
    bootMode: UEFI
    networkConfig: 1
      interfaces: 2
       - name: eno1 3
         type: ethernet 4
         state: up
         mac-address: 0c:42:a1:55:f3:06
         ipv4:
           enabled: true
           dhcp: false 5
         ethernet:
           sr-iov:
             total-vfs: 2 6
         ipv6:
           enabled: false
           dhcp: false
       - name: sriov:eno1:0
         type: ethernet
         state: up 7
         ipv4:
           enabled: false 8
         ipv6:
           enabled: false
       - name: sriov:eno1:1
         type: ethernet
         state: down
       - name: eno2
         type: ethernet
         state: up
         mac-address: 0c:42:a1:55:f3:07
         ipv4:
           enabled: true
         ethernet:
```

```
      sr-iov:
        total-vfs: 2
      ipv6:
        enabled: false
    - name: sriov:eno2:0
      type: ethernet
      state: up
      ipv4:
        enabled: false
      ipv6:
        enabled: false
    - name: sriov:eno2:1
      type: ethernet
      state: down
    - name: bond0
      type: bond
      state: up
      min-tx-rate: 100 9
      max-tx-rate: 200 10
      link-aggregation:
        mode: active-backup 11
        options:
          primary: sriov:eno1:0 12
        port:
          - sriov:eno1:0
          - sriov:eno2:0
      ipv4:
        address:
          - ip: 10.19.16.57 13
            prefix-length: 23
        dhcp: false
        enabled: true
      ipv6:
        enabled: false
      dns-resolver:
        config:
          server:
            - 10.11.5.160
            - 10.2.70.215
      routes:
        config:
          - destination: 0.0.0.0/0
            next-hop-address: 10.19.17.254
            next-hop-interface: bond0 14
            table-id: 254
```

1

networkConfig 字段包含有关主机的网络配置的信息，子字段包括 接口、dns-resolver 和 routes。

2

interfaces 字段是为主机定义的网络接口数组。

3

接口的名称。

4

接口的类型。这个示例创建了一个以太网接口。

5

如果物理功能 (PF) 没有被严格要求，则将其设置为 'false 以禁用 DHCP。

6

设置为要实例化的 SR-IOV 虚拟功能 (VF) 的数量。

7

把它设置为 up。

8

把它设置为 false，以禁用附加到绑定的 VF 的 IPv4 寻址。

9

为 VF 设置最小传输率（以 Mbps 为单位）。这个示例值设置 100 Mbps 的速度。

- 这个值必须小于或等于最大传输率。
- Intel NIC 不支持 min-tx-rate 参数。如需更多信息，请参阅 BZ#1772847。

10

为 VF 设置最大传输率（以 Mbps 为单位）。此示例值设置 200 Mbps 的速度。

11

设置所需的绑定模式。

12

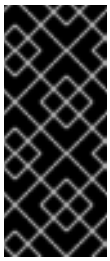
设置绑定接口的首选端口。主设备是要使用的绑定接口的第一个，除非失败，否则不会被取消。当绑定接口中的一个 NIC 速度更快时，此设置特别有用，因此可以处理较大的负载。只有在绑定接口处于 active-backup 模式（模式 1）和 balance-tlb （模式 5）时，此设置才有效。

13

为绑定接口设置静态 IP 地址。这是节点 IP 地址。

14

将 bond0 设置为默认路由的网关。



重要

部署集群后，您无法修改 install-config.yaml 文件的 networkConfig 配置设置，以更改主机网络接口。在部署后，使用 Kubernetes NMState Operator 更改主机网络接口。

其他资源

- 配置网络绑定

16.3.11.16. 配置多个集群节点

您可以使用相同的设置同时配置 OpenShift Container Platform 集群节点。配置多个集群节点可避免在 install-config.yaml 文件中添加每个节点的冗余信息。这个文件包含特定的参数，可将相同的配置应用到集群中的多个节点。

Compute 节点与控制器节点独立配置。但是，两个节点类型的配置都使用 install-config.yaml 文件中突出显示的参数启用多节点配置。将 networkConfig 参数设置为 BOND，如下例所示：

```
hosts:
- name: ostest-master-0
 [...]
 networkConfig: &BOND
   interfaces:
   - name: bond0
     type: bond
     state: up
     ipv4:
```

```
      dhcp: true
      enabled: true
    link-aggregation:
      mode: active-backup
      port:
      - enp2s0
      - enp3s0
- name: ostest-master-1
 [...]
 networkConfig: *BOND
- name: ostest-master-2
 [...]
 networkConfig: *BOND
```

注意

配置多个集群节点仅适用于安装程序置备的基础架构上的初始部署。

16.3.11.17. 可选：配置管理的安全引导

在使用 Redfish BMC 寻址（如 redfish, redfish-virtualmedia, 或 idrac-virtualmedia）部署安装程序置备的集群时，您可以启用受管安全引导。要启用受管安全引导，请在每个节点中添加 bootMode 配置设置：

示例

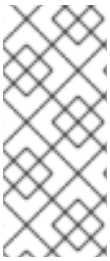
```
hosts:
  - name: openshift-master-0
    role: master
    bmc:
      address: redfish://<out_of_band_ip> 1
      username: <username>
      password: <password>
    bootMACAddress: <NIC1_mac_address>
    rootDeviceHints:
     deviceName: "/dev/sda"
    bootMode: UEFISecureBoot 2
```

1

确保 bmc.address 设置使用 redfish、redfish-virtualmedia 或 idrac-virtualmedia 作为协议。如需了解更多详细信息，请参阅"HPE iLO"或"BMC 寻址用于 Dell iDRAC"的"BMC 寻址"。

2

bootMode 设置默认为 UEFI。将它更改为 UEFISecureBoot 以启用受管安全引导。



注意

请参阅"先决条件"中的"配置节点"，以确保节点能够支持受管安全引导。如果节点不支持受管安全引导，请参阅"配置节点"部分中的"手动配置安全引导节点"。手动配置安全引导机制需要 Redfish 虚拟介质。

注意

红帽不支持使用 IPMI 进行安全引导，因为 IPMI 不提供安全引导管理功能。

16.3.12. 清单配置文件

16.3.12.1. 创建 OpenShift Container Platform 清单

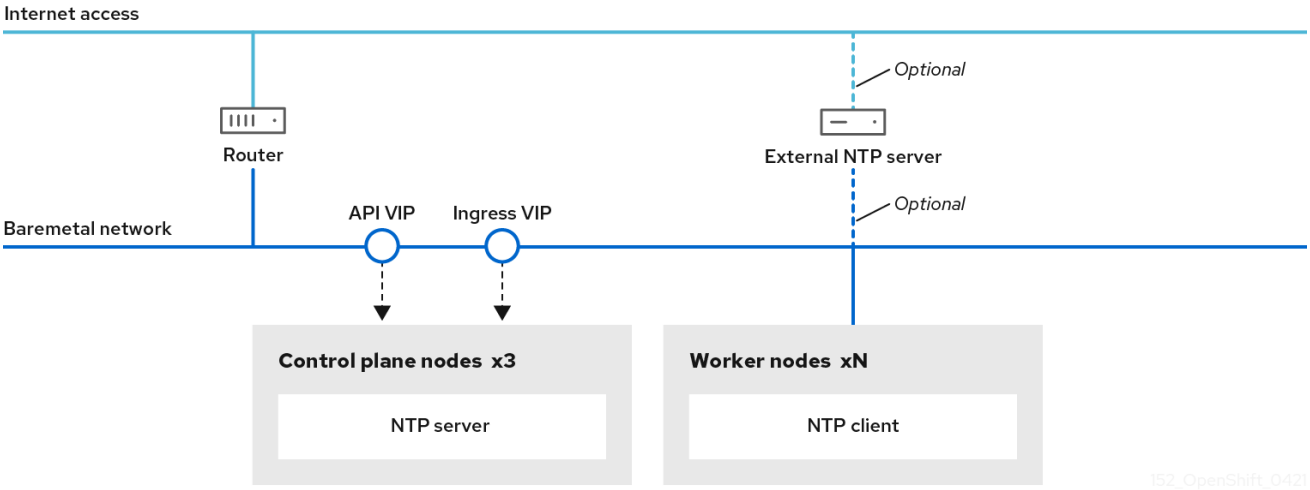
1. 创建 OpenShift Container Platform 清单。

```
$ ./openshift-baremetal-install --dir ~/clusterconfigs create manifests
```

```
INFO Consuming Install Config from target directory
WARNING Making control-plane schedulable by setting MastersSchedulable to true
for Scheduler cluster settings
WARNING Discarding the OpenShift Manifest that was provided in the target directory
because its dependencies are dirty and it needs to be regenerated
```

16.3.12.2. 可选：为断开连接的集群配置 NTP

OpenShift Container Platform 在集群节点上安装 chrony 网络时间协议(NTP)服务。



OpenShift Container Platform 节点必须在日期和时间上达成一致才能正确运行。当计算节点从 control plane 节点上的 NTP 服务器检索日期和时间时，它会启用未连接到可路由网络的集群的安装和操作，因此无法访问更高的 stratum NTP 服务器。

流程

1. **为 control plane 节点创建一个 Butane 配置 99-master-chrony-conf-override.bu，包括 chrony.conf 文件的内容。**

注意

如需有关 Butane 的信息，请参阅"使用 Butane 创建机器配置"。

但ane 配置示例

```
variant: openshift
version: 4.16.0
metadata:
  name: 99-master-chrony-conf-override
  labels:
    machineconfiguration.openshift.io/role: master
storage:
  files:
    - path: /etc/chrony.conf
      mode: 0644
      overwrite: true
      contents:
        inline: |
          # Use public servers from the pool.ntp.org project.
          # Please consider joining the pool (https://www.pool.ntp.org/join.html).
```

```
# The Machine Config Operator manages this file
server openshift-master-0.<cluster-name>.<domain> iburst 1
server openshift-master-1.<cluster-name>.<domain> iburst
server openshift-master-2.<cluster-name>.<domain> iburst

stratumweight 0
driftfile /var/lib/chrony/drift
rtcsync
makestep 10 3
bindcmdaddress 127.0.0.1
bindcmdaddress ::1
keyfile /etc/chrony.keys
commandkey 1
generatecommandkey
noclientlog
logchange 0.5
logdir /var/log/chrony

# Configure the control plane nodes to serve as local NTP servers
# for all compute nodes, even if they are not in sync with an
# upstream NTP server.

# Allow NTP client access from the local network.
allow all
# Serve time even if not synchronized to a time source.
local stratum 3 orphan
```

1

您必须将 <cluster-name> 替换为集群名称，并将 <domain> 替换为完全限定域名。

2. 使用 Butane 生成 MachineConfig 对象文件 99-master-chrony-conf-override.yaml，其中包含要发送到 control plane 节点的配置：

```
$ butane 99-master-chrony-conf-override.bu -o 99-master-chrony-conf-override.yaml
```

3. 为引用 control plane 节点上的 NTP 服务器的计算节点创建 Butane 配置 99-worker-chrony-conf-override.bu，包括 chrony.conf 文件的内容。

但ane 配置示例

```
variant: openshift
```

```
version: 4.16.0
metadata:
  name: 99-worker-chrony-conf-override
  labels:
    machineconfiguration.openshift.io/role: worker
storage:
  files:
    - path: /etc/chrony.conf
      mode: 0644
      overwrite: true
      contents:
        inline: |
          # The Machine Config Operator manages this file.
          server openshift-master-0.<cluster-name>.<domain> iburst 1
          server openshift-master-1.<cluster-name>.<domain> iburst
          server openshift-master-2.<cluster-name>.<domain> iburst

          stratumweight 0
          driftfile /var/lib/chrony/drift
          rtcsync
          makestep 10 3
          bindcmdaddress 127.0.0.1
          bindcmdaddress ::1
          keyfile /etc/chrony.keys
          commandkey 1
          generatecommandkey
          noclientlog
          logchange 0.5
          logdir /var/log/chrony
```

1

您必须将 <cluster-name> 替换为集群名称，并将 <domain> 替换为完全限定域名。

4. 使用 Butane 生成 MachineConfig 对象文件 99-worker-chrony-conf-override.yaml，其中包含要交付至 worker 节点的配置：

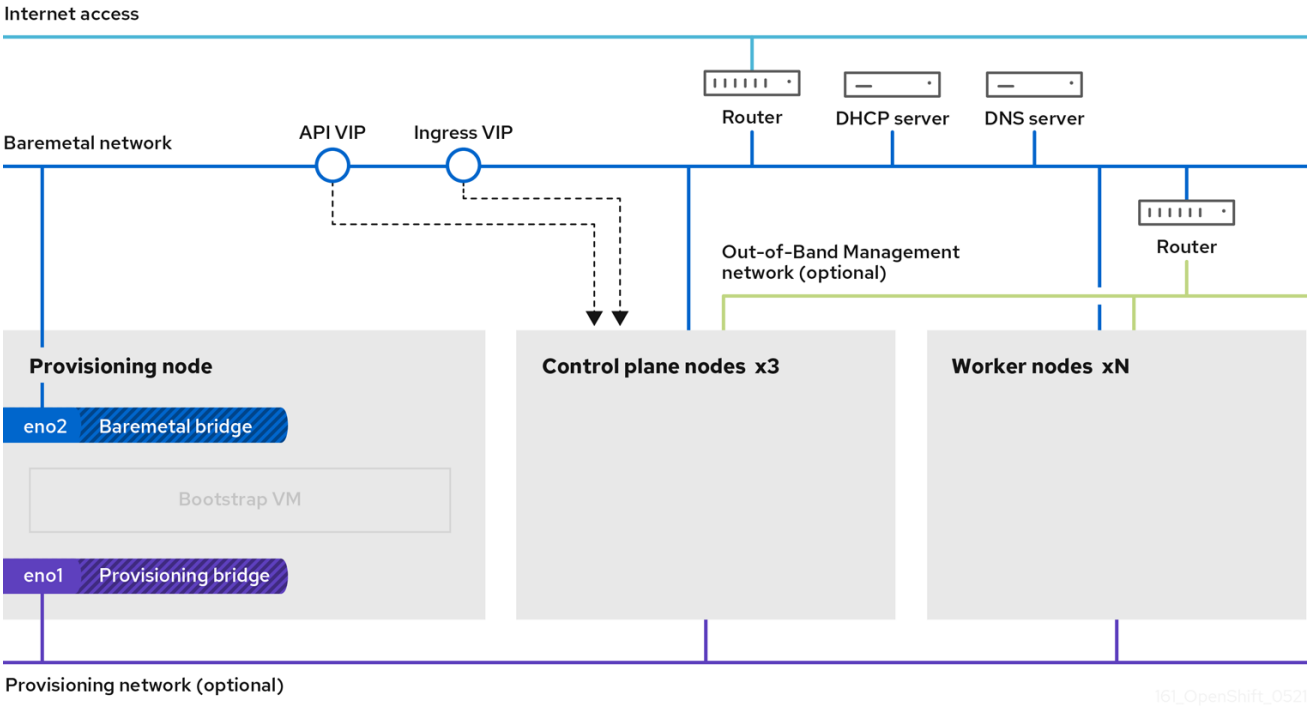
```
$ butane 99-worker-chrony-conf-override.bu -o 99-worker-chrony-conf-override.yaml
```

16.3.12.3. 配置要在 control plane 上运行的网络组件

您可以配置网络组件，使其仅在 control plane 节点上运行。默认情况下，OpenShift Container Platform 允许机器配置池中的任何节点托管 ingressVIP 虚拟 IP 地址。但是，有些环境在与 control plane 节点独立的子网中部署计算节点，这需要将 ingressVIP 虚拟 IP 地址配置为在 control plane 节点上运行。

重要

在单独的子网中部署远程节点时，您必须将 ingressVIP 虚拟 IP 地址专门用于 control plane 节点。



流程

1. 进入存储 install-config.yaml 文件的目录：

```
$ cd ~/clusterconfigs
```

2. 切换到 manifests 子目录：

```
$ cd manifests
```

3. 创建名为 cluster-network-avoid-workers-99-config.yaml 的文件：

```
$ touch cluster-network-avoid-workers-99-config.yaml
```

4. 在编辑器中打开 cluster-network-avoid-workers-99-config.yaml 文件，并输入描述 Operator 配置的自定义资源(CR)：

```
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  name: 50-worker-fix-ipi-rwn
  labels:
    machineconfiguration.openshift.io/role: worker
spec:
  config:
    ignition:
      version: 3.2.0
    storage:
      files:
        - path: /etc/kubernetes/manifests/keepalived.yaml
          mode: 0644
          contents:
            source: data:,
```

此清单将 ingressVIP 虚拟 IP 地址放在 control plane 节点上。另外，此清单仅在 control plane 节点上部署以下进程：

- openshift-ingress-operator
- keepalived

5. 保存 cluster-network-avoid-workers-99-config.yaml 文件。

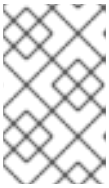
6. 创建 manifests/cluster-ingress-default-ingresscontroller.yaml 文件：

```
apiVersion: operator.openshift.io/v1
kind: IngressController
metadata:
  name: default
  namespace: openshift-ingress-operator
spec:
  nodePlacement:
    nodeSelector:
      matchLabels:
        node-role.kubernetes.io/master: ""
```

7. 考虑备份 manifests 目录。在创建集群时，安装程序会删除 manifests/ 目录。

8. 通过将 mastersSchedulable 字段设置为 true 来修改 cluster-scheduler-02-config.yml 清单，使 control plane 节点可以调度。默认情况下，control plane 节点不可调度。例如：

```
$ sed -i "s;mastersSchedulable: false;mastersSchedulable: true;g"
clusterconfigs/manifests/cluster-scheduler-02-config.yml
```

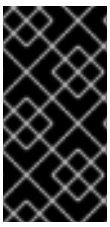


注意

如果在完成此步骤后 control plane 节点不可调度，则部署集群将失败。

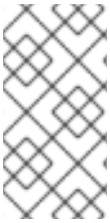
16.3.12.4. 可选：在计算节点上部署路由器

在安装过程中，安装程序会在计算节点上部署路由器 pod。默认情况下，安装程序会安装两个路由器 pod。如果部署的集群需要额外的路由器来处理用于 OpenShift Container Platform 集群中服务的外部流量负载，您可以创建一个 yaml 文件来设置适当数量的路由器副本。



重要

不支持只使用一个计算节点部署集群。虽然在使用一个计算节点时修改路由器副本数量会解决降级状态的问题，但集群丢失了入口 API 的高可用性，它不适用于生产环境。



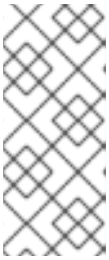
注意

默认情况下，安装程序会部署两个路由器。如果集群没有计算节点，安装程序会默认在 control plane 节点上部署两个路由器。

流程

1. **创建 router-replicas.yaml 文件：**

```
apiVersion: operator.openshift.io/v1
kind: IngressController
metadata:
  name: default
  namespace: openshift-ingress-operator
spec:
  replicas: <num-of-router-pods>
  endpointPublishingStrategy:
    type: HostNetwork
  nodePlacement:
    nodeSelector:
      matchLabels:
        node-role.kubernetes.io/worker: ""
```



注意

将 <num-of-router-pods> 替换为适当的值。如果只使用一个计算节点，请将 replicas: 设置为 1。如果使用超过 3 个计算节点，您可以根据情况增加 replicas: 的值（默认值为 2）。

2. 将 router-replicas.yaml 文件复制到 clusterconfigs/openshift 目录中：

```
$ cp ~/router-replicas.yaml clusterconfigs/openshift/99_router-replicas.yaml
```

16.3.12.5. 可选：配置 BIOS

以下流程在安装过程中配置 BIOS。

流程

1. 创建清单。

2. 修改与节点对应的 BareMetalHost 资源文件：

```
$ vim clusterconfigs/openshift/99_openshift-cluster-api_hosts-*.yaml
```

3. 将 BIOS 配置添加到 BareMetalHost 资源的 spec 部分：

```
spec:
  firmware:
    simultaneousMultithreadingEnabled: true
    sriovEnabled: true
    virtualizationEnabled: true
```

注意

红帽支持三种 BIOS 配置：仅支持 BMC 类型 irmc 的服务器。目前不支持其他类型的服务器。

4. 创建集群。

其他资源

- 裸机配置

16.3.12.6. 可选：配置 RAID

以下流程在安装过程中使用基板管理控制器 (BMC) 配置独立磁盘的冗余阵列 (RAID)。

注意

如果要为节点配置硬件 RAID，请验证节点是否具有支持的 RAID 控制器。OpenShift Container Platform 4.16 不支持软件 RAID。

表 16.11. 厂商的硬件 RAID 支持

Vendor	BMC 和协议	固件版本	RAID 级别
Fujitsu	iRMC	N/A	0、1、5、6 和 10
Dell	使用 Redfish 的 iDRAC	版本 6.10.30.20 或更高版本	0、1 和 5

流程

1. 创建清单。

2. 修改与节点对应的 BareMetalHost 资源：

```
$ vim clusterconfigs/openshift/99_openshift-cluster-api_hosts-*.yaml
```

注意

以下示例使用硬件 RAID 配置，因为 OpenShift Container Platform 4.16 不支持软件 RAID。

a. 如果您在 spec 部分添加了特定的 RAID 配置，这会导致节点在 preparing 阶段删除原始 RAID 配置，并在 RAID 上执行指定的配置。例如：

```
spec:
  raid:
    hardwareRAIDVolumes:
    - level: "0" (1)
      name: "sda"
      numberOfPhysicalDisks: 1
      rotational: true
      sizeGibibytes: 0
```

1

level 是必填字段，另一个是可选字段。

b.

如果您在 spec 部分添加了空的 RAID 配置，空配置会导致节点在 preparing 阶段删除原始 RAID 配置，但不执行新配置。例如：

```
spec:
  raid:
    hardwareRAIDVolumes: []
```

c.

如果您没有在 spec 部分添加 raid 字段，则原始 RAID 配置不会被删除，且不会执行新的配置。

3.

创建集群。

16.3.12.7. 可选：在节点上配置存储

您可以通过创建由 Machine Config Operator (MCO) 管理的 MachineConfig 对象来更改 OpenShift Container Platform 节点上的操作系统。

MachineConfig 规格包括一个 ignition 配置，用于在第一次引导时配置机器。此配置对象可用于修改 OpenShift Container Platform 机器上运行的文件、systemd 服务和其他操作系统功能。

流程

使用 ignition 配置在节点上配置存储。以下 MachineSet 清单示例演示了如何将分区添加到主节点上的设备中。在本例中，在安装前应用清单，其名为 restore 的分区，大小为 16 GiB。

1.

创建 custom-partitions.yaml 文件，并包含一个包含分区布局的 MachineConfig 对象：

-

```
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: primary
  name: 10_primary_storage_config
spec:
  config:
    ignition:
      version: 3.2.0
    storage:
      disks:
        - device: </dev/xxyN>
          partitions:
            - label: recovery
              startMiB: 32768
              sizeMiB: 16384
      filesystems:
        - device: /dev/disk/by-partlabel/recovery
          label: recovery
          format: xfs
```

2. 将 custom-partitions.yaml 文件复制到 clusterconfigs/openshift 目录中：

```
$ cp ~/<MachineConfig_manifest> ~/clusterconfigs/openshift
```

其他资源

- 裸机配置
- 分区命名方案

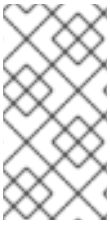
16.3.13. 创建断开连接的 registry

在某些情况下，您可能想要使用安装 registry 的本地副本安装 OpenShift Container Platform 集群。这可能是为了提高网络效率，因为集群节点位于无法访问互联网的网络中。

一个本地的或被镜像的 registry 副本需要以下内容：

- registry 节点的证书。这可以是自签名证书。

- 系统中的容器将服务的 Web 服务器。
- 包含证书和本地存储库信息的更新的 pull secret。



注意

在 registry 节点上创建断开连接的 registry 是可选的。如果需要在 registry 节点上创建断开连接的 registry，您必须完成以下所有子章节。

先决条件

- 如果您已经为断开连接的安装准备了镜像 registry，您可以跳过修改 install-config.yaml 文件以使用断开连接的 registry。

16.3.13.1. 准备 registry 节点以托管已镜像的 registry

在裸机上托管镜像的 registry 之前，必须完成以下步骤。

流程

1. 打开 registry 节点上的防火墙端口：

```
$ sudo firewall-cmd --add-port=5000/tcp --zone=libvirt  --permanent
$ sudo firewall-cmd --add-port=5000/tcp --zone=public   --permanent
$ sudo firewall-cmd --reload
```

2. 为 registry 节点安装所需的软件包：

```
$ sudo yum -y install python3 podman httpd httpd-tools jq
```

3. 创建保存存储库信息的目录结构：

```
$ sudo mkdir -p /opt/registry/{auth,certs,data}
```

16.3.13.2. 为断开连接的 registry 镜像 OpenShift Container Platform 镜像存储库

完成以下步骤，为断开连接的 registry 镜像 OpenShift Container Platform 镜像存储库。

先决条件

- 您的镜像主机可访问互联网。
- 您已将镜像 registry 配置为在受限网络中使用，并可访问您配置的证书和凭证。
- 您已从 Red Hat OpenShift Cluster Manager 下载了 pull secret，并已修改为包含镜像存储库的身份验证。

流程

1. 查看 OpenShift Container Platform 下载页面，以确定您要安装的 OpenShift Container Platform 版本，并决定 Repository Tags 页中的相应标签（tag）。
2. 设置所需的环境变量：
   a. 导出发行版本信息：

```
$ OCP_RELEASE=<release_version>
```

   对于 <release_version>，请指定与 OpenShift Container Platform 版本对应的标签，用于您的架构，如 4.5.4。

   b. 导出本地 registry 名称和主机端口：

```
$ LOCAL_REGISTRY='<local_registry_host_name>:<local_registry_host_port>'
```

   对于 <local_registry_host_name>，请指定镜像存储库的 registry 域名；对于 <local_registry_host_port>，请指定用于提供内容的端口。

c. 导出本地存储库名称：

```
$ LOCAL_REPOSITORY='<local_repository_name>'
```

对于 <local_repository_name>，请指定要在 registry 中创建的仓库名称，如 ocp4/openshift4。

d. 导出要进行镜像的存储库名称：

```
$ PRODUCT_REPO='openshift-release-dev'
```

对于生产环境版本，必须指定 openshift-release-dev。

e. 导出 registry pull secret 的路径：

```
$ LOCAL_SECRET_JSON='<path_to_pull_secret>'
```

对于 <path_to_pull_secret>，请指定您创建的镜像 registry 的 pull secret 的绝对路径和文件名。

f. 导出发行版本镜像：

```
$ RELEASE_NAME="ocp-release"
```

对于生产环境版本，您必须指定 ocp-release。

g. 为您的集群导出构架类型：

```
$ ARCHITECTURE=<cluster_architecture> 1
```

1 指定集群的构架，如 x86_64, aarch64, s390x, 获 ppc64le。

h. 导出托管镜像的目录的路径：

```
$ REMOVABLE_MEDIA_PATH=<path> 1
```

1 指定完整路径，包括开始的前斜杠(/)字符。

3. 将版本镜像(mirror)到镜像 registry：

- 如果您的镜像主机无法访问互联网，请执行以下操作：

i. 将可移动介质连接到连接到互联网的系统。

ii. 查看要镜像的镜像和配置清单：

```
$ oc adm release mirror -a ${LOCAL_SECRET_JSON} \
    --from=quay.io/${PRODUCT_REPO}/${RELEASE_NAME}:${OCP_RELEASE}-
${ARCHITECTURE} \
    --to=${LOCAL_REGISTRY}/${LOCAL_REPOSITORY} \
    --to-release-
image=${LOCAL_REGISTRY}/${LOCAL_REPOSITORY}:${OCP_RELEASE}-
${ARCHITECTURE} --dry-run
```

iii. 记录上一命令输出中的 imageContentSources 部分。您的镜像信息与您的镜像存储库相对应，您必须在安装过程中将 imageContentSources 部分添加到 install-config.yaml 文件中。

iv. 将镜像镜像到可移动介质的目录中：

```
$ oc adm release mirror -a ${LOCAL_SECRET_JSON} --to-
dir=${REMOVABLE_MEDIA_PATH}/mirror
quay.io/${PRODUCT_REPO}/${RELEASE_NAME}:${OCP_RELEASE}-
${ARCHITECTURE}
```

v. 将介质上传到受限网络环境中，并将镜像上传到本地容器 registry。

```
$ oc image mirror -a ${LOCAL_SECRET_JSON} --from-
```

```
dir=${REMOVABLE_MEDIA_PATH}/mirror
"file://openshift/release:${OCP_RELEASE}*"
${LOCAL_REGISTRY}/${LOCAL_REPOSITORY} 1
```

1

对于 REMOVABLE_MEDIA_PATH，您必须使用与镜像镜像时指定的同一路径。

- 如果本地容器 registry 连接到镜像主机，请执行以下操作：

i. 使用以下命令直接将发行版镜像推送到本地 registry:

```
$ oc adm release mirror -a ${LOCAL_SECRET_JSON} \
    --from=quay.io/${PRODUCT_REPO}/${RELEASE_NAME}:${OCP_RELEASE}-
${ARCHITECTURE} \
    --to=${LOCAL_REGISTRY}/${LOCAL_REPOSITORY} \
    --to-release-
image=${LOCAL_REGISTRY}/${LOCAL_REPOSITORY}:${OCP_RELEASE}-
${ARCHITECTURE}
```

该命令将发行信息提取为摘要，其输出包括安装集群时所需的 imageContentSources 数据。

ii. 记录上一命令输出中的 imageContentSources 部分。您的镜像信息与您的镜像存储库相对应，您必须在安装过程中将 imageContentSources 部分添加到 install-config.yaml 文件中。

注意

镜像名称在镜像过程中被修补到 Quay.io， podman 镜像将在 bootstrap 虚拟机的 registry 中显示 Quay.io。

4. 要创建基于您镜像内容的安装程序，请提取内容并将其固定到发行版中：

- 如果您的镜像主机无法访问互联网，请运行以下命令：

```
$ oc adm release extract -a ${LOCAL_SECRET_JSON} --command=openshift-
baremetal-install
"${LOCAL_REGISTRY}/${LOCAL_REPOSITORY}:${OCP_RELEASE}"
```

- 如果本地容器 registry 连接到镜像主机，请运行以下命令：

```
$ oc adm release extract -a ${LOCAL_SECRET_JSON} --command=openshift-
baremetal-install
"${LOCAL_REGISTRY}/${LOCAL_REPOSITORY}:${OCP_RELEASE}-
${ARCHITECTURE}"
```

重要

要确保将正确的镜像用于您选择的 OpenShift Container Platform 版本，您必须从镜像内容中提取安装程序。

您必须在有活跃互联网连接的机器上执行这个步骤。

如果您位于断开连接的环境中,请使用 --image 标志作为 must-gather 的一部分，指向有效负载镜像。

5. 对于使用安装程序置备的基础架构的集群，运行以下命令：

```
$ openshift-baremetal-install
```

16.3.13.3. 修改 install-config.yaml 文件以使用断开连接的 registry

在 provisioner 节点上，install -config.yaml 文件应该使用从 pull-secret- update.txt 文件中新创建的 pull-secret。install-config.yaml 文件还必须包含断开连接的 registry 节点的证书和 registry 信息。

流程

1. 将断开连接的 registry 节点的证书添加到 install-config.yaml 文件中：

```
$ echo "additionalTrustBundle: |" >> install-config.yaml
```

证书应跟在 "additionalTrustBundle: |" 行后面，并正确缩进（通常为两个空格）。

```
$ sed -e 's/^/  /' /opt/registry/certs/domain.crt >> install-config.yaml
```

2. 将 registry 的镜像信息添加到 install-config.yaml 文件中：

```
$ echo "imageContentSources:" >> install-config.yaml
```

```
$ echo "- mirrors:" >> install-config.yaml
```

```
$ echo "  - registry.example.com:5000/ocp4/openshift4" >> install-config.yaml
```

将 registry.example.com 替换为 registry 的完全限定域名。

```
$ echo "  source: quay.io/openshift-release-dev/ocp-release" >> install-config.yaml
```

```
$ echo "- mirrors:" >> install-config.yaml
```

```
$ echo "  - registry.example.com:5000/ocp4/openshift4" >> install-config.yaml
```

将 registry.example.com 替换为 registry 的完全限定域名。

```
$ echo "  source: quay.io/openshift-release-dev/ocp-v4.0-art-dev" >> install-
config.yaml
```

16.3.14. 安装的验证清单

❑ 已检索到 OpenShift Container Platform 安装程序。

❑ 已提取 OpenShift Container Platform 安装程序。

❑ 已配置了 install-config.yaml 的必要参数。

❑ 已配置了 install-config.yaml 的 hosts 参数。

❑ 已配置了 install-config.yaml 的 bmc 参数。

❑ 在 bmc address 字段中配置的值已被应用。

❑ 创建 OpenShift Container Platform 清单。

❑ (Optional) 可选）在计算节点上部署路由器。

❑ （可选）创建断开连接的 registry。

❑ （可选）如果使用，验证断开连接的 registry 设置。

16.3.15. 通过 OpenShift Container Platform 安装程序部署集群

运行 OpenShift Container Platform 安装程序：

```
$ ./openshift-baremetal-install --dir ~/clusterconfigs --log-level debug create cluster
```

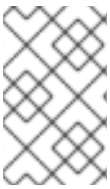
16.3.16. 安装后

在部署过程中，您可以通过向安装目录文件夹中的 .openshift_install.log 日志文件发出 tail 命令来检查安装的整体状态：

```
$ tail -f /path/to/install-dir/.openshift_install.log
```

16.3.17. 验证静态 IP 地址配置

如果集群节点的 DHCP 保留指定了无限租期，安装程序成功置备该节点后，分配程序脚本会检查节点的网络配置。如果脚本确定网络配置包含无限 DHCP 租期，它将 DHCP 租期的 IP 地址用作静态 IP 地址来创建新连接。



注意

分配程序脚本可能会在成功置备的节点上运行，同时持续置备集群中的其他节点。

验证网络配置是否正常工作。

流程

1. 检查节点上的网络接口配置。

2. 关闭 DHCP 服务器并重启 OpenShift Container Platform 节点，并确保网络配置可以正常工作。

16.3.18. 准备在裸机上重新安装集群

在裸机上重新安装集群前，您必须执行清理操作。

流程

1. 删除或重新格式化 bootstrap、control plane 节点和计算节点的磁盘。如果您在虚拟机监控程序环境中工作，您必须添加您要删除的任何磁盘。

2. 删除之前生成的工件：

```
$ cd ; /bin/rm -rf auth/ bootstrap.ign master.ign worker.ign metadata.json \
.openshift_install.log .openshift_install_state.json
```

3. 生成新清单和 Ignition 配置文件。如需更多信息，请参阅"创建 Kubernetes 清单和 Ignition 配置文件"。

4. 将安装程序创建的新 bootstrap、control plane 和计算节点 Ignition 配置文件上传到 HTTP 服务器。这将覆盖以前的 Ignition 文件。

16.3.19. 其他资源

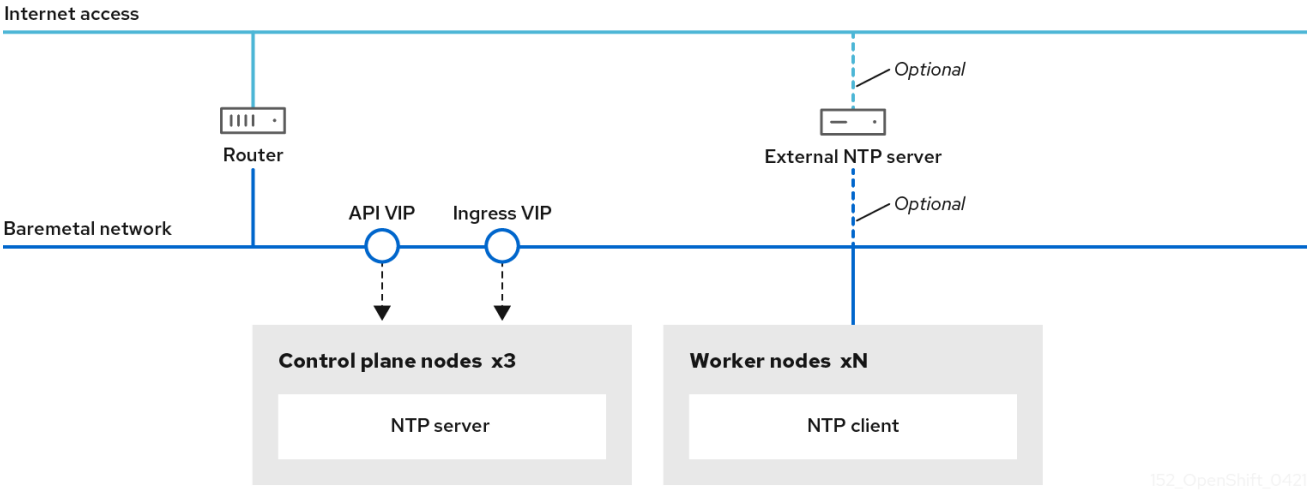
- OpenShift Container Platform 创建 Kubernetes 清单和 Ignition 配置文件
- 了解更新频道和发行版本

16.4. 安装程序置备的安装后配置

成功部署安装程序置备的集群后，请考虑以下安装后流程。

16.4.1. 可选：为断开连接的集群配置 NTP

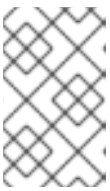
OpenShift Container Platform 在集群节点上安装 chrony 网络时间协议(NTP)服务。使用以下步骤在 control plane 节点上配置 NTP 服务器，并将计算节点配置为成功部署后，将 worker 节点配置为 control plane 节点的 NTP 客户端。



OpenShift Container Platform 节点必须在日期和时间上达成一致才能正确运行。当计算节点从 control plane 节点上的 NTP 服务器检索日期和时间时，它会启用未连接到可路由网络的集群的安装和操作，因此无法访问更高的 stratum NTP 服务器。

流程

1. 为 control plane 节点创建一个 Butane 配置 99-master-chrony-conf-override.bu，包括 chrony.conf 文件的内容。



注意

如需有关 Butane 的信息，请参阅"使用 Butane 创建机器配置"。

但ane 配置示例

```
variant: openshift
version: 4.16.0
metadata:
  name: 99-master-chrony-conf-override
  labels:
    machineconfiguration.openshift.io/role: master
storage:
  files:
    - path: /etc/chrony.conf
      mode: 0644
      overwrite: true
      contents:
        inline: |
          # Use public servers from the pool.ntp.org project.
          # Please consider joining the pool (https://www.pool.ntp.org/join.html).

          # The Machine Config Operator manages this file
          server openshift-master-0.<cluster-name>.<domain> iburst 1
          server openshift-master-1.<cluster-name>.<domain> iburst
          server openshift-master-2.<cluster-name>.<domain> iburst

          stratumweight 0
          driftfile /var/lib/chrony/drift
          rtcsync
          makestep 10 3
          bindcmdaddress 127.0.0.1
          bindcmdaddress ::1
          keyfile /etc/chrony.keys
          commandkey 1
          generatecommandkey
          noclientlog
          logchange 0.5
          logdir /var/log/chrony

          # Configure the control plane nodes to serve as local NTP servers
          # for all compute nodes, even if they are not in sync with an
          # upstream NTP server.

          # Allow NTP client access from the local network.
          allow all
          # Serve time even if not synchronized to a time source.
          local stratum 3 orphan
```

1 您必须将 <cluster-name> 替换为集群名称，并将 <domain> 替换为完全限定域名。

2. 使用 Butane 生成 MachineConfig 对象文件 99-master-chrony-conf-override.yaml，其中包含要发送到 control plane 节点的配置：

```
$ butane 99-master-chrony-conf-override.bu -o 99-master-chrony-conf-override.yaml
```

3. 为引用 control plane 节点上的 NTP 服务器的计算节点创建 Butane 配置 99-worker-chrony-conf-override.bu，包括 chrony.conf 文件的内容。

但ane 配置示例

```
variant: openshift
version: 4.16.0
metadata:
  name: 99-worker-chrony-conf-override
  labels:
    machineconfiguration.openshift.io/role: worker
storage:
  files:
    - path: /etc/chrony.conf
      mode: 0644
      overwrite: true
      contents:
        inline: |
          # The Machine Config Operator manages this file.
          server openshift-master-0.<cluster-name>.<domain> iburst 1
          server openshift-master-1.<cluster-name>.<domain> iburst
          server openshift-master-2.<cluster-name>.<domain> iburst

          stratumweight 0
          driftfile /var/lib/chrony/drift
          rtcsync
          makestep 10 3
          bindcmdaddress 127.0.0.1
          bindcmdaddress ::1
          keyfile /etc/chrony.keys
          commandkey 1
          generatecommandkey
          noclientlog
          logchange 0.5
          logdir /var/log/chrony
```

1 您必须将 <cluster-name> 替换为集群名称，并将 <domain> 替换为完全限定域名。

4.

使用 Butane 生成 MachineConfig 对象文件 99-worker-chrony-conf-override.yaml，其中包含要交付至 worker 节点的配置：

```
$ butane 99-worker-chrony-conf-override.bu -o 99-worker-chrony-conf-override.yaml
```

5. 将 99-master-chrony-conf-override.yaml 策略应用到 control plane 节点。

```
$ oc apply -f 99-master-chrony-conf-override.yaml
```

输出示例

```
machineconfig.machineconfiguration.openshift.io/99-master-chrony-conf-override
created
```

6. 将 99-worker-chrony-conf-override.yaml 策略应用到计算节点。

```
$ oc apply -f 99-worker-chrony-conf-override.yaml
```

输出示例

```
machineconfig.machineconfiguration.openshift.io/99-worker-chrony-conf-override
created
```

7. 检查应用的 NTP 设置的状态。

```
$ oc describe machineconfigpool
```

16.4.2. 安装后启用置备网络

通过为裸机集群提供支持的安装程序和安装程序置备安装，可以在没有 provisioning 网络的情况下部署集群。当每个节点的基板管理控制器可以通过 baremetal 网络 路由时，此功能适用于概念验证集群或

仅使用 Redfish 虚拟介质单独部署的情况。

您可在安装后使用 Cluster Baremetal Operator(CBO)启用 置备 网络。

先决条件

- 必须存在专用物理网络，连接到所有 worker 和 control plane 节点。
- 您必须隔离原生、未标记的物理网络。
- 当 provisioningNetwork 配置设置为 Managed 时，网络无法有一个 DHCP 服务器。
- 您可以省略 OpenShift Container Platform 4.10 中的 provisioningInterface 设置，以使用 bootMACAddress 配置设置。

流程

1. 设置 provisioningInterface 设置时，首先确定集群节点的调配接口名称。例如： eth0 or eno1。
2. 在集群节点的 调配 网络接口上启用预引导执行环境(PXE)。
3. 检索 provisioning 网络的当前状态，并将其保存到 provisioning 自定义资源(CR)文件中：

```
$ oc get provisioning -o yaml > enable-provisioning-nw.yaml
```

4. 修改 provisioning CR 文件：

```
$ vim ~/enable-provisioning-nw.yaml
```

向下滚动到 provisioningNetwork 配置设置，并将它从 Disabled 改为 Managed。然后，在 provisioningNetwork 设置后添加 provisioningIP、provisioningNetworkCIDR、provisioningDHCPRange、provisioningInterface 和 watchAllNameSpaces 配置设置。为每项设置提供适当的值。

```
apiVersion: v1
items:
- apiVersion: metal3.io/v1alpha1
  kind: Provisioning
  metadata:
    name: provisioning-configuration
  spec:
    provisioningNetwork: 1
    provisioningIP: 2
    provisioningNetworkCIDR: 3
    provisioningDHCPRange: 4
    provisioningInterface: 5
    watchAllNameSpaces: 6
```

1

provisioningNetwork 是 Managed、Unmanaged 或 Disabled 之一。当设置为 Managed 时，Metal3 管理调配网络，CBO 使用配置的 DHCP 服务器部署 Metal3 pod。当设置为 Unmanaged 时，系统管理员手动配置 DHCP 服务器。

2

provisioningIP 是 DHCP 服务器和 ironic 用于调配网络的静态 IP 地址。这个静态 IP 地址必须在 provisioning 子网内，且不在 DHCP 范围内。如果配置这个设置，它必须具有有效的 IP 地址，即使 provisioning 网络是 Disabled。静态 IP 地址绑定到 metal3 pod。如果 metal3 pod 失败并移动到其他服务器，静态 IP 地址也会移到新服务器。

3

无类别域间路由(CIDR)地址。如果配置这个设置，它必须具有有效的 CIDR 地址，即使 provisioning 网络是 Disabled。例如：192.168.0.1/24。

4

DHCP 范围。此设置仅适用于 受管 置备网络。如果 provisioning 网络为 Disabled，则省略此配置设置。例如：192.168.0.64, 192.168.0.253。

5

集群节点上 置备 接口的 NIC 名称。provisioningInterface 设置仅适用于 受管和非受管 置备 网络。如果 provisioning 网络为 Disabled，忽略 provisioningInterface 配置设置。省略 provisioningInterface 配置设置，以使用 bootMACAddress 配置设置。

6

如果您希望 metal3 监视默认 openshift-machine-api 命名空间以外的命名空间，请将此设置设置为 true。默认值为 false。

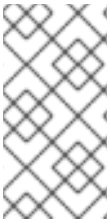
5. 保存对 provisioning CR 文件的更改。

6. 将 provisioning CR 文件应用到集群：

```
$ oc apply -f enable-provisioning-nw.yaml
```

16.5. 扩展集群

部署安装程序置备的 OpenShift Container Platform 集群后，您可以使用以下步骤扩展 worker 节点的数量。确保每个 worker 节点都满足先决条件。



注意

使用 RedFish Virtual Media 扩展集群需要满足最低固件要求。有关使用 RedFish Virtual Media 扩展集群的详情，请参阅先决条件部分中的使用虚拟介质安装的固件要求。

16.5.1. 准备裸机节点

要扩展集群，必须为节点提供相关 IP 地址。这可以通过静态配置，或使用 DHCP （动态主机配置协议）服务器来完成。在使用 DHCP 服务器扩展集群时，每个节点都必须有 DHCP 保留。

保留 IP 地址，以便其成为静态 IP 地址

有些管理员更喜欢使用静态 IP 地址，以便在没有 DHCP 服务器时每个节点的 IP 地址保持恒定状态。要使用 NMState 配置静态 IP 地址，请参阅 install-config.yaml 文件中的 "可选：配置主机网络接口，以了解更多详细信息。

准备裸机节点需要从 provisioner 节点执行以下步骤。

流程

1. 获取 oc 二进制文件：

```
$ curl -s https://mirror.openshift.com/pub/openshift-
v4/clients/ocp/$VERSION/openshift-client-linux-$VERSION.tar.gz | tar zxvf - oc
```

```
$ sudo cp oc /usr/local/bin
```

2. 使用基板管理控制器 (BMC) 关闭裸机节点，并确保它已关闭。

3. 检索裸机节点基板管理控制器的用户名和密码。然后，从用户名和密码创建 base64 字符串：

```
$ echo -ne "root" | base64
```

```
$ echo -ne "password" | base64
```

4. 为裸机节点创建配置文件。根据您是否使用静态配置还是 DHCP 服务器，请使用以下示例 bmh.yaml 文件，替换 YAML 中的值以匹配您的环境：

```
$ vim bmh.yaml
```

- 静态配置 bmh.yaml ：

```
---
apiVersion: v1 (1)
kind: Secret
metadata:
 name: openshift-worker-<num>-network-config-secret (2)
 namespace: openshift-machine-api
type: Opaque
stringData:
 nmstate: | (3)
  interfaces: (4)
  - name: <nic1_name> (5)
    type: ethernet
    state: up
    ipv4:
      address:
      - ip: <ip_address> (6)
        prefix-length: 24
      enabled: true
  dns-resolver:
    config:
      server:
      - <dns_ip_address> (7)
  routes:
    config:
    - destination: 0.0.0.0/0
      next-hop-address: <next_hop_ip_address> (8)
      next-hop-interface: <next_hop_nic1_name> (9)
```

```
---
apiVersion: v1
kind: Secret
metadata:
  name: openshift-worker-<num>-bmc-secret (10)
  namespace: openshift-machine-api
type: Opaque
data:
  username: <base64_of_uid> (11)
  password: <base64_of_pwd> (12)
---
apiVersion: metal3.io/v1alpha1
kind: BareMetalHost
metadata:
  name: openshift-worker-<num> (13)
  namespace: openshift-machine-api
spec:
  online: True
  bootMACAddress: <nic1_mac_address> (14)
  bmc:
    address: <protocol>://<bmc_url> (15)
    credentialsName: openshift-worker-<num>-bmc-secret (16)
    disableCertificateVerification: True (17)
    username: <bmc_username> (18)
    password: <bmc_password> (19)
  rootDeviceHints:
    deviceName: <root_device_hint> (20)
  preprovisioningNetworkDataName: openshift-worker-<num>-network-config-
secret (21)
```

(1) 要为新创建的节点配置网络接口，请指定包含网络配置的 secret 的名称。按照 nmstate 语法为节点定义网络配置。有关配置 NMState 语法的详情，请参阅"可选：在 install-config.yaml 文件中配置主机网络接口"。

(2) (10) (13) (16) 在 name、credentialsName 字段和 preprovisioningNetworkDataName 字段中，使用裸机节点的 worker 数量替换 <num>。

(3) 添加 NMState YAML 语法来配置主机接口。

(4) 可选：如果您使用 nmstate 配置网络接口，并且您要禁用接口，请将 state: 设置为 enabled: false，如下所示：

```
---
  interfaces:
  - name: <nic_name>
    type: ethernet
    state: up
    ipv4:
      enabled: false
    ipv6:
      enabled: false
```

5 6 7 8 9

使用适当的值替换 <nic1_name>, <ip_address>, <dns_ip_address>, <next_hop_ip_address> 和 <next_hop_nic1_name>。

11 12

将 <base64_of_uid> 和 <base64_of_pwd> 替换为用户名和密码的 base64 字符串。

14

将 <nic1_mac_address> 替换为裸机节点第一个 NIC 的 MAC 地址。如需了解更多 BMC 配置选项，请参阅"BMC 寻址"部分。

15

将 <protocol> 替换为 BMC 协议，如 IPMI、RedFish 或其他协议。将 <bmc_url> 替换为裸机节点基板管理控制器的 URL。

17

要跳过证书验证，将 disableCertificateVerification 设为 true。

18 19

将 <bmc_username> 和 <bmc_password> 替换为 BMC 用户名和密码的字符串。

20

可选：如果您指定了 root 设备提示，将 <root_device_hint> 替换为设备路径。

21

可选： 如果您为新创建的节点配置了网络接口，请在 BareMetalHost CR 的 preprovisioningNetworkDataName 中提供网络配置 secret 名称。

- DHCP 配置 bmh.yaml ：

```
---
apiVersion: v1
kind: Secret
metadata:
  name: openshift-worker-<num>-bmc-secret (1)
  namespace: openshift-machine-api
type: Opaque
data:
  username: <base64_of_uid> (2)
  password: <base64_of_pwd> (3)
---
apiVersion: metal3.io/v1alpha1
kind: BareMetalHost
metadata:
  name: openshift-worker-<num> (4)
  namespace: openshift-machine-api
spec:
  online: True
  bootMACAddress: <nic1_mac_address> (5)
  bmc:
    address: <protocol>://<bmc_url> (6)
    credentialsName: openshift-worker-<num>-bmc-secret (7)
    disableCertificateVerification: True (8)
    username: <bmc_username> (9)
    password: <bmc_password> (10)
  rootDeviceHints:
    deviceName: <root_device_hint> (11)
  preprovisioningNetworkDataName: openshift-worker-<num>-network-config-
secret (12)
```

(1) (4) (7)

在 name、credentialsName 字段和 preprovisioningNetworkDataName 字段中，使用裸机节点的 worker 数量替换 <num>。

(2) (3)

将 <base64_of_uid> 和 <base64_of_pwd> 替换为用户名和密码的 base64 字符串。

(5)

将 <nic1_mac_address> 替换为裸机节点第一个 NIC 的 MAC 地址。如需了解更多 BMC 配置选项，请参阅"BMC 寻址"部分。

(6)

8

要跳过证书验证，将 disableCertificateVerification 设为 true。

9 10

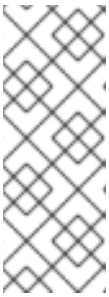
将 <bmc_username> 和 <bmc_password> 替换为 BMC 用户名和密码的字符串。

11

可选：如果您指定了 root 设备提示，将 <root_device_hint> 替换为设备路径。

12

可选： 如果您为新创建的节点配置了网络接口，请在 BareMetalHost CR 的 preprovisioningNetworkDataName 中提供网络配置 secret 名称。



注意

如果现有裸机节点的 MAC 地址与您试图置备的裸机主机的 MAC 地址匹配，则 Ironic 安装将失败。如果主机注册、检查、清理或其他 Ironic 步骤失败，Bare Metal Operator 会持续重试安装。如需更多信息，请参阅"诊断主机重复的 MAC 地址"。

5. 创建裸机节点：

```
$ oc -n openshift-machine-api create -f bmh.yaml
```

输出示例

```
secret/openshift-worker-<num>-network-config-secret created
secret/openshift-worker-<num>-bmc-secret created
baremetalhost.metal3.io/openshift-worker-<num> created
```

其中 <num> 是 worker 号。

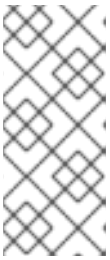
6. 启动并检查裸机节点：

```
$ oc -n openshift-machine-api get bmh openshift-worker-<num>
```

其中 <num> 是 worker 节点号。

输出示例

```
NAME                 STATE      CONSUMER   ONLINE   ERROR
openshift-worker-<num>  available           true
```



注意

要允许 worker 节点加入集群，请将 machineset 对象扩展到 BareMetalHost 对象的数量。您可以手动或自动缩放节点。要自动扩展节点，请为 machineset 使用 metal3.io/autoscale-to-hosts 注解。

其他资源

- 有关配置 NMState 语法的详情，请参阅可选：在 install-config.yaml 文件中配置主机网络接口。
- 如需了解有关自动扩展机器的详细信息，请参阅自动缩放机器到可用的裸机主机数量。

16.5.2. 替换裸机 control plane 节点

使用以下步骤替换安装程序置备的 OpenShift Container Platform control plane 节点。

重要

如果您从现有 control plane 主机重复使用 BareMetalHost 对象定义，请不要将 externallyProvisioned 字段保留为 true。

如果 OpenShift Container Platform 安装程序置备，现有 control plane BareMetalHost 对象可能会将 externallyProvisioned 标记设为 true。

先决条件

- 您可以使用具有 cluster-admin 角色的用户访问集群。
- 已进行 etcd 备份。

重要

执行此流程前进行 etcd 备份，以便在遇到任何问题时可以恢复集群。有关获取 etcd 备份的更多信息，请参阅*附加资源*部分。

流程

1. 确保 Bare Metal Operator 可用：

```
$ oc get clusteroperator baremetal
```

输出示例

```
NAME      VERSION   AVAILABLE   PROGRESSING   DEGRADED   SINCE   MESSAGE
baremetal   4.16    True        False         False      3d15h
```

2. 删除旧的 BareMetalHost 和 Machine 对象：

```
$ oc delete bmh -n openshift-machine-api <host_name>
$ oc delete machine -n openshift-machine-api <machine_name>
```

将 <host_name> 替换为主机名，<machine_name> 替换为机器的名称。机器名称会出现在 CONSUMER 字段下。

删除 BareMetalHost 和 Machine 对象后，机器控制器会自动删除 Node 对象。

3. 创建新的 BareMetalHost 对象和 secret，以存储 BMC 凭证：

```
$ cat <<EOF | oc apply -f -
apiVersion: v1
kind: Secret
metadata:
  name: control-plane-<num>-bmc-secret 1
  namespace: openshift-machine-api
data:
  username: <base64_of_uid> 2
  password: <base64_of_pwd> 3
type: Opaque
---
apiVersion: metal3.io/v1alpha1
kind: BareMetalHost
metadata:
  name: control-plane-<num> 4
  namespace: openshift-machine-api
spec:
  automatedCleaningMode: disabled
  bmc:
    address: <protocol>://<bmc_ip> 5
    credentialsName: control-plane-<num>-bmc-secret 6
  bootMACAddress: <NIC1_mac_address> 7
  bootMode: UEFI
  externallyProvisioned: false
  online: true
EOF
```

1 4 6 在 name 字段和 credentialsName 字段中，使用裸机节点的 control plane 数量替换 <num>。

2 将 <base64_of_uid> 替换为用户名的 base64 格式的字符串。

3 将 <base64_of_pwd> 替换为密码的 base64 格式的字符串。

5

将 <protocol> 替换为 BMC 协议，如 redfish、redfish-virtualmedia、idrac-virtualmedia 或其他。将 <bmc_ip> 替换为裸机节点基板管理控制器的 IP 地址。如需了解更多 BMC 配置选项，请参阅*附加资源*部分中的 "BMC 寻址"。

7

将 <NIC1_mac_address> 替换为裸机节点第一个 NIC 的 MAC 地址。

检查完成后，BareMetalHost 对象会被创建并可用置备。

4. 查看可用的 BareMetalHost 对象：

```
$ oc get bmh -n openshift-machine-api
```

输出示例

```
NAME                          STATE                  CONSUMER              ONLINE   ERROR   AGE
control-plane-1.example.com   available              control-plane-1       true             1h10m
control-plane-2.example.com   externally provisioned control-plane-2       true             4h53m
control-plane-3.example.com   externally provisioned control-plane-3       true             4h53m
compute-1.example.com         provisioned            compute-1-ktmmx       true             4h53m
compute-1.example.com         provisioned            compute-2-l2zmb       true             4h53m
```

control plane 节点没有 MachineSet 对象，因此您必须创建 Machine 对象。您可以从另一个 control plane Machine 对象复制 providerSpec。

5. 创建 Machine 对象：

```
$ cat <<EOF | oc apply -f -
apiVersion: machine.openshift.io/v1beta1
kind: Machine
```

```
metadata:
  annotations:
    metal3.io/BareMetalHost: openshift-machine-api/control-plane-<num> 1
  labels:
    machine.openshift.io/cluster-api-cluster: control-plane-<num> 2
    machine.openshift.io/cluster-api-machine-role: master
    machine.openshift.io/cluster-api-machine-type: master
  name: control-plane-<num> 3
  namespace: openshift-machine-api
spec:
  metadata: {}
  providerSpec:
    value:
      apiVersion: baremetal.cluster.k8s.io/v1alpha1
      customDeploy:
        method: install_coreos
      hostSelector: {}
      image:
        checksum: ""
        url: ""
      kind: BareMetalMachineProviderSpec
      metadata:
        creationTimestamp: null
      userData:
        name: master-user-data-managed
EOF
```

1 2 3 在 name, labels 和 annotations 字段中，使用裸机节点的 control plane 数量替换 <num>。

6. 要查看 BareMetalHost 对象，请运行以下命令：

```
$ oc get bmh -A
```

输出示例

```
NAME                 STATE                  CONSUMER             ONLINE   ERROR   AGE
control-plane-1.example.com   provisioned            control-plane-1      true
2h53m
control-plane-2.example.com   externally provisioned   control-plane-2        true
5h53m
control-plane-3.example.com   externally provisioned   control-plane-3        true
5h53m
compute-1.example.com         provisioned            compute-1-ktmmx        true
```

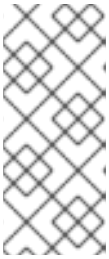
```
5h53m
compute-2.example.com      provisioned          compute-2-l2zmb          true
5h53m
```

7. 在 RHCOS 安装后，验证 BareMetalHost 是否已添加到集群中：

```
$ oc get nodes
```

输出示例

```
NAME                        STATUS     ROLES    AGE   VERSION
control-plane-1.example.com   available   master   4m2s  v1.29.4
control-plane-2.example.com   available   master   141m  v1.29.4
control-plane-3.example.com   available   master   141m  v1.29.4
compute-1.example.com         available   worker   87m   v1.29.4
compute-2.example.com         available   worker   87m   v1.29.4
```



注意

替换新的 control plane 节点后，在新节点上运行的 etcd pod 处于 crashloopback 状态。如需更多信息，请参阅*附加资源*部分中的 "替换不健康的 etcd 成员"。

其他资源

- 替换不健康的 etcd 成员
- 备份 etcd
- 裸机配置
- BMC 地址

16.5.3. 准备在 baremetal 网络中使用 Virtual Media 进行部署

如果启用了 provisioning 网络，且您要使用 baremetal 网络中的 Virtual Media 扩展集群，请使用以下步骤。

先决条件

- 有一个带有 a baremetal 网络和 provisioning 网络的现有集群。

流程

1. 编辑 置备 自定义资源(CR)，以在 baremetal 网络中使用 Virtual Media 启用部署：

```
oc edit provisioning
```

```
apiVersion: metal3.io/v1alpha1
kind: Provisioning
metadata:
  creationTimestamp: "2021-08-05T18:51:50Z"
  finalizers:
  - provisioning.metal3.io
  generation: 8
  name: provisioning-configuration
  resourceVersion: "551591"
  uid: f76e956f-24c6-4361-aa5b-feaf72c5b526
spec:
  provisioningDHCPRange: 172.22.0.10,172.22.0.254
  provisioningIP: 172.22.0.3
  provisioningInterface: enp1s0
  provisioningNetwork: Managed
  provisioningNetworkCIDR: 172.22.0.0/24
  virtualMediaViaExternalNetwork: true (1)
status:
  generations:
  - group: apps
    hash: ""
    lastGeneration: 7
    name: metal3
    namespace: openshift-machine-api
    resource: deployments
  - group: apps
    hash: ""
    lastGeneration: 1
    name: metal3-image-cache
    namespace: openshift-machine-api
    resource: daemonsets
  observedGeneration: 8
  readyReplicas: 0
```

1

将 virtualMediaViaExternalNetwork: true 添加到 provisioning CR。

2. 如果镜像 URL 存在，请编辑 machineset 以使用 API VIP 地址。此步骤只适用于在 4.9 或更早版本安装的集群。

```
oc edit machineset
```

```
apiVersion: machine.openshift.io/v1beta1
kind: MachineSet
metadata:
  creationTimestamp: "2021-08-05T18:51:52Z"
  generation: 11
  labels:
    machine.openshift.io/cluster-api-cluster: ostest-hwmdt
    machine.openshift.io/cluster-api-machine-role: worker
    machine.openshift.io/cluster-api-machine-type: worker
  name: ostest-hwmdt-worker-0
  namespace: openshift-machine-api
  resourceVersion: "551513"
  uid: fad1c6e0-b9da-4d4a-8d73-286f78788931
spec:
  replicas: 2
  selector:
    matchLabels:
      machine.openshift.io/cluster-api-cluster: ostest-hwmdt
      machine.openshift.io/cluster-api-machineset: ostest-hwmdt-worker-0
  template:
    metadata:
      labels:
        machine.openshift.io/cluster-api-cluster: ostest-hwmdt
        machine.openshift.io/cluster-api-machine-role: worker
        machine.openshift.io/cluster-api-machine-type: worker
        machine.openshift.io/cluster-api-machineset: ostest-hwmdt-worker-0
    spec:
      metadata: {}
      providerSpec:
        value:
          apiVersion: baremetal.cluster.k8s.io/v1alpha1
          hostSelector: {}
          image:
            checksum: http:/172.22.0.3:6181/images/rhcos-<version>.
<architecture>.qcow2.<md5sum> 1
            url: http://172.22.0.3:6181/images/rhcos-<version>.<architecture>.qcow2 2
          kind: BareMetalMachineProviderSpec
          metadata:
            creationTimestamp: null
          userData:
            name: worker-user-data
  status:
```

```
availableReplicas: 2
fullyLabeledReplicas: 2
observedGeneration: 11
readyReplicas: 2
replicas: 2
```

1 编辑 校验和 URL，以使用 API VIP 地址。

2 编辑 URL 以使用 API VIP 地址。

16.5.4. 在集群中置备新主机时诊断重复的 MAC 地址

如果集群中现有裸机节点的 MAC 地址与您试图添加到集群的裸机主机的 MAC 地址匹配，Bare Metal Operator 将主机与现有节点关联。如果主机注册、检查、清理或其他 Ironic 步骤失败，Bare Metal Operator 会持续重试安装。失败的裸机主机会显示注册错误。

您可以通过检查在 openshift-machine-api 命名空间中运行的裸机主机来诊断重复的 MAC 地址。

先决条件

- 在裸机上安装 OpenShift Container Platform 集群。
- 安装 OpenShift Container Platform CLI oc。
- 以具有 cluster-admin 权限的用户身份登录。

流程

要确定置备失败的裸机主机是否具有与现有节点相同的 MAC 地址，请执行以下操作：

1. 获取在 openshift-machine-api 命名空间中运行的裸机主机：

```
$ oc get bmh -n openshift-machine-api
```

输出示例

```
NAME                 STATUS   PROVISIONING STATUS      CONSUMER
openshift-master-0   OK       externally provisioned   openshift-zpwpq-master-0
openshift-master-1   OK       externally provisioned   openshift-zpwpq-master-1
openshift-master-2   OK       externally provisioned   openshift-zpwpq-master-2
openshift-worker-0   OK       provisioned              openshift-zpwpq-worker-0-lv84n
openshift-worker-1   OK       provisioned              openshift-zpwpq-worker-0-zd8lm
openshift-worker-2   error    registering
```

2. 要查看失败主机状态的更多详细信息，请运行以下命令将 <bare_metal_host_name> 替换为主机名称：

```
$ oc get -n openshift-machine-api bmh <bare_metal_host_name> -o yaml
```

输出示例

```
...
status:
  errorCount: 12
  errorMessage: MAC address b4:96:91:1d:7c:20 conflicts with existing node
openshift-worker-1
  errorType: registration error
...
```

16.5.5. 置备裸机节点

置备裸机节点需要从 provisioner 节点执行以下步骤。

流程

1. 在置备裸机节点前，请确保 STATE 为 available。

```
$  oc -n openshift-machine-api get bmh openshift-worker-<num>
```

其中 <num> 是 worker 节点号。

```
NAME            STATE     ONLINE ERROR  AGE
openshift-worker  available true         34h
```

2. 获取 worker 节点数量。

```
$ oc get nodes
```

```
NAME                                        STATUS   ROLES        AGE     VERSION
openshift-master-1.openshift.example.com    Ready    master       30h     v1.29.4
openshift-master-2.openshift.example.com    Ready    master       30h     v1.29.4
openshift-master-3.openshift.example.com    Ready    master       30h     v1.29.4
openshift-worker-0.openshift.example.com    Ready    worker       30h     v1.29.4
openshift-worker-1.openshift.example.com    Ready    worker       30h     v1.29.4
```

3. 获取计算机器集。

```
$ oc get machinesets -n openshift-machine-api
```

```
NAME                        DESIRED   CURRENT   READY   AVAILABLE   AGE
...
openshift-worker-0.example.com    1         1         1       1           55m
openshift-worker-1.example.com    1         1         1       1           55m
```

4. 将 worker 节点数量增加一个。

```
$ oc scale --replicas=<num> machineset <machineset> -n openshift-machine-api
```

将 <num> 替换为新的 worker 节点数。将 <machineset> 替换为上一步中计算机器设置的名称。

5. 检查裸机节点的状态。

```
$ oc -n openshift-machine-api get bmh openshift-worker-<num>
```

其中 <num> 是 worker 节点号。STATE 从 ready 变为 provisioning。

```
NAME                 STATE          CONSUMER                    ONLINE   ERROR
openshift-worker-<num>  provisioning     openshift-worker-<num>-65tjz     true
```

provisioning 状态会保持，直到 OpenShift Container Platform 集群置备节点。这可能需要 30 分钟或更长时间。在调配节点后，其状态将更改为 provisioned。

```
NAME                 STATE          CONSUMER                    ONLINE   ERROR
openshift-worker-<num>  provisioned      openshift-worker-<num>-65tjz     true
```

6. 在置备完成后，确保裸机节点就绪。

```
$ oc get nodes
```

```
NAME                                STATUS   ROLES   AGE    VERSION
openshift-master-1.openshift.example.com     Ready    master  30h    v1.29.4
openshift-master-2.openshift.example.com     Ready    master  30h    v1.29.4
openshift-master-3.openshift.example.com     Ready    master  30h    v1.29.4
openshift-worker-0.openshift.example.com     Ready    worker  30h    v1.29.4
openshift-worker-1.openshift.example.com     Ready    worker  30h    v1.29.4
openshift-worker-<num>.openshift.example.com  Ready    worker  3m27s   v1.29.4
```

您还可以检查 kubelet。

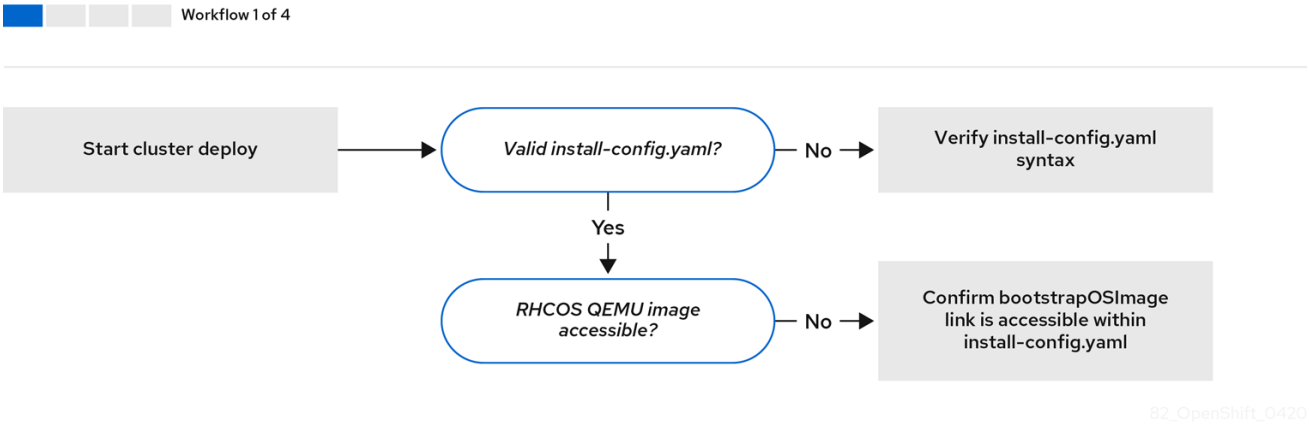
```
$ ssh openshift-worker-<num>
```

```
[kni@openshift-worker-<num>]$ journalctl -fu kubelet
```

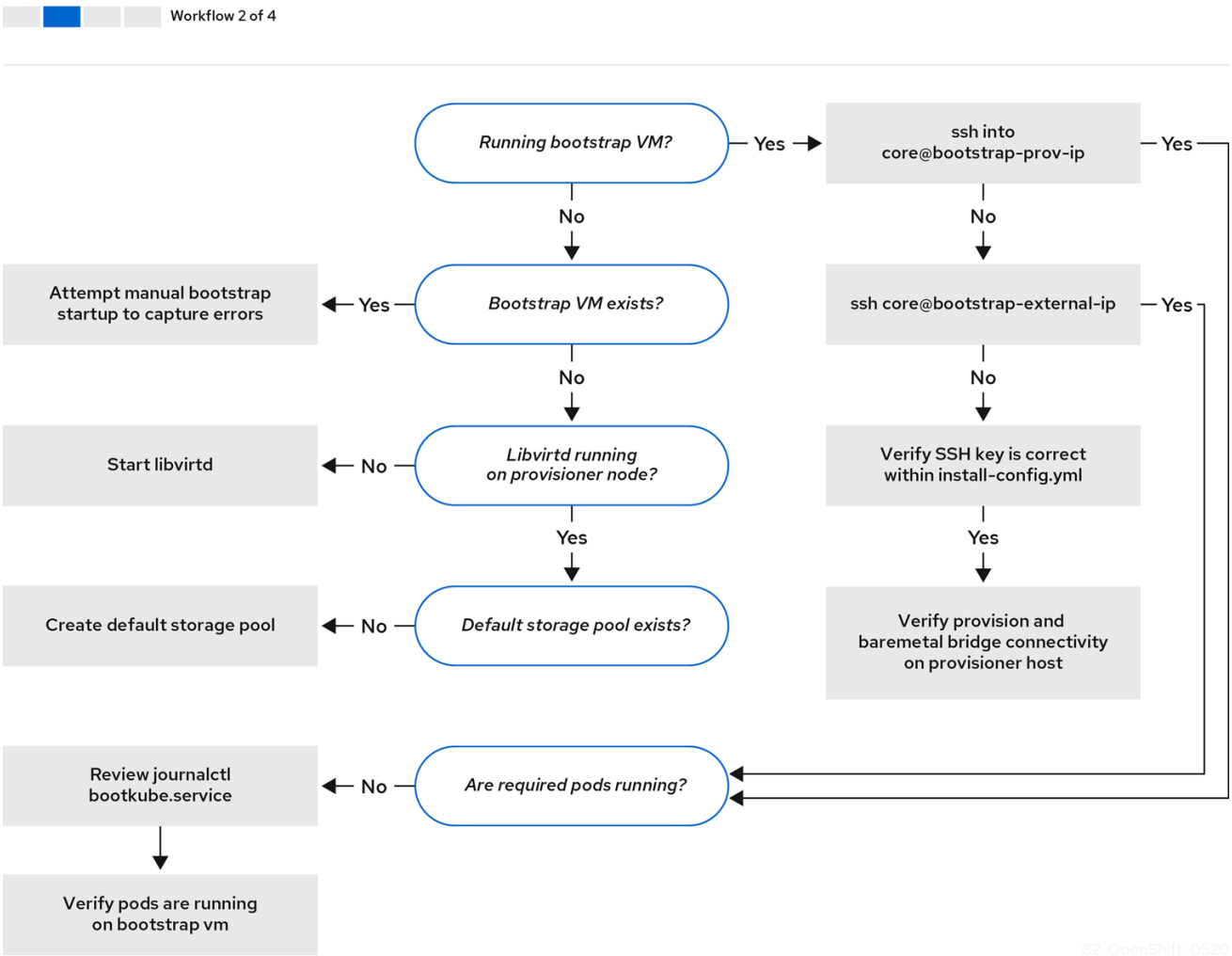
16.6. 故障排除

16.6.1. 安装程序工作流故障排除

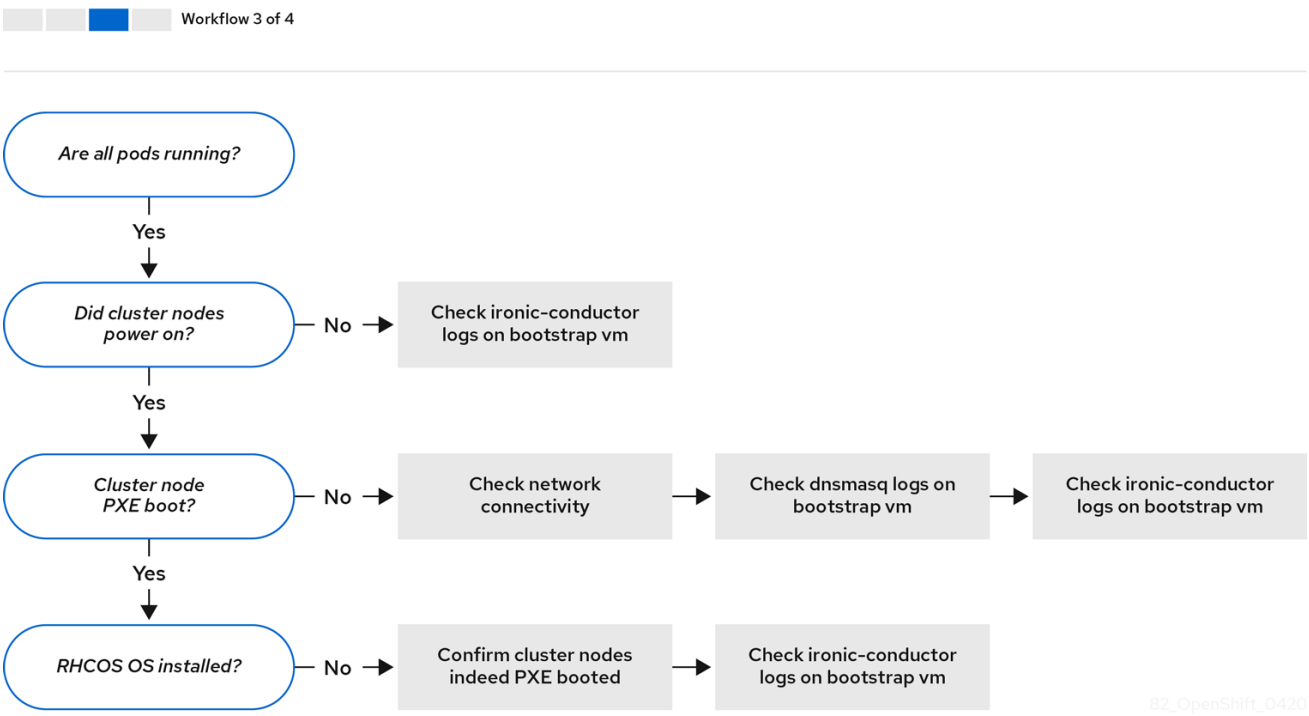
在对安装环境进行故障排除之前，了解裸机上安装程序置备安装的整体流至关重要。下图提供了故障排除流程，并按步骤划分环境。



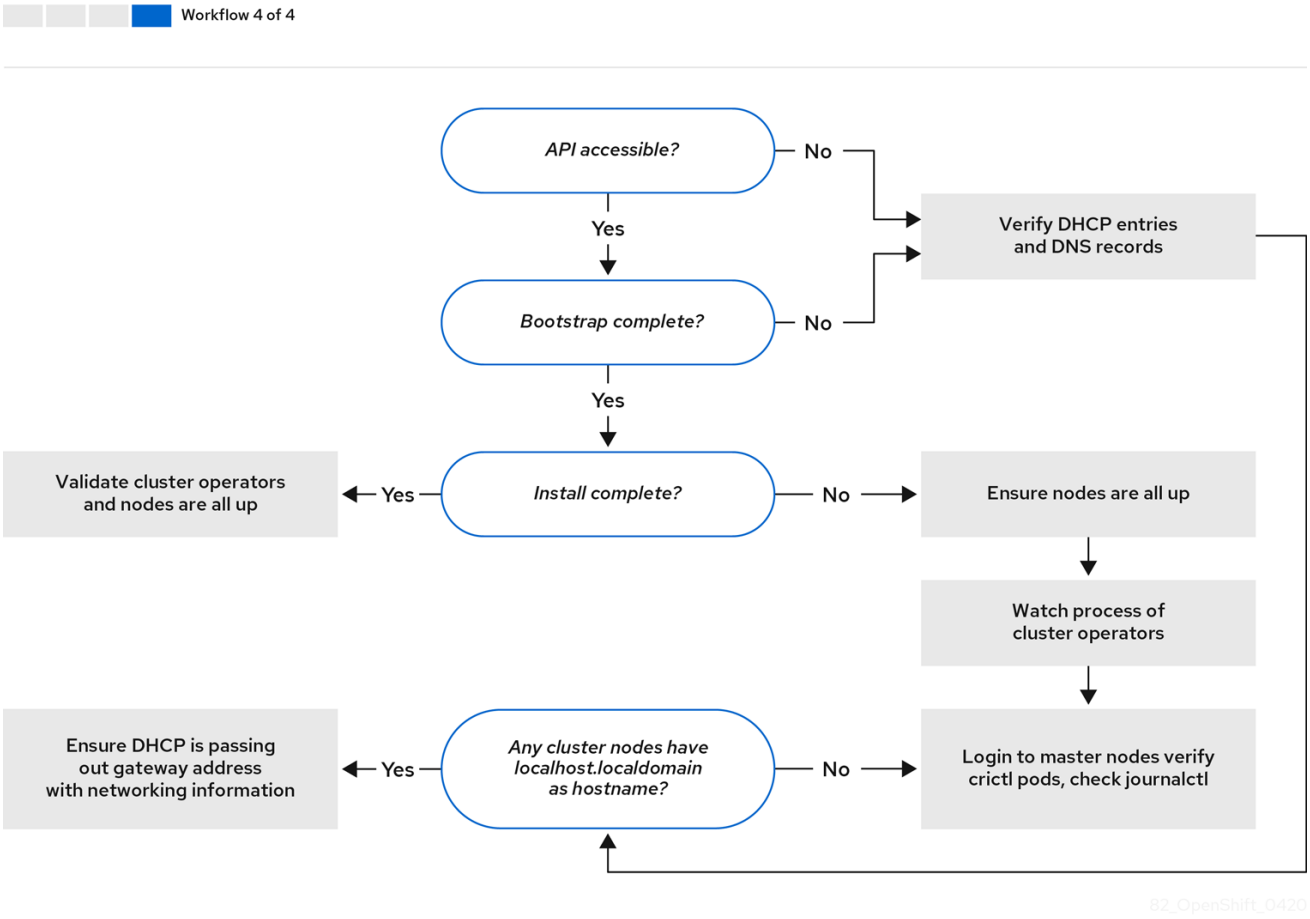
当 install-config.yaml 文件出错或者无法访问 Red Hat Enterprise Linux CoreOS(RHCOS)镜像时，工作流 *1（共 4 步）* 的工作流演示了故障排除工作流。故障排除建议可在 故障排除 install-config.yaml 中找到。



*工作流 2（共 4 步）*描述了对 bootstrap 虚拟机问题、 无法引导集群节点的 bootstrap 虚拟机 以及检查日志 的故障排除工作流。当在没有 provisioning 网络的情况下安装 OpenShift Container Platform 集群时，此工作流将不应用。



*工作流 3（共 4 步）*描述了 不能 PXE 引导的集群节点 的故障排除工作流。如果使用 RedFish Virtual Media 安装，每个节点都必须满足安装程序部署节点的最低固件要求。如需了解更多详细信息，请参阅先决条件部分中的使用虚拟介质安装的固件要求。



工作流 4（共 4 步） 演示了 无法访问 API 的故障排除工作流，以及 经过验证的安装。

16.6.2. install-config.yaml故障排除

install-config.yaml 配置文件代表属于 OpenShift Container Platform 集群的所有节点。该文件包含由 apiVersion、baseDomain、imageContentSources 和虚拟 IP 地址组成的必要选项。如果在 OpenShift Container Platform 集群部署早期发生错误，install-config.yaml 配置文件中可能会出现错误。

流程

1. 使用 YAML-tips 中的指南。
2. 使用 syntax-check 验证 YAML 语法是否正确。
3. 验证 Red Hat Enterprise Linux CoreOS(RHCOS)QEMU 镜像是否已正确定义，并可通过 install-config.yaml 提供的 URL 访问。例如：

```
$ curl -s -o /dev/null -I -w "%{http_code}\n" http://webserver.example.com:8080/rhcos-44.81.202004250133-0-qemu.<architecture>.qcow2.gz?sha256=7d884b46ee54fe87bbc3893bf2aa99af3b2d31f2e19ab5529c60636fbd0f1ce7
```

如果输出为 200，则会从存储 bootstrap 虚拟机镜像的 webserver 获得有效的响应。

16.6.3. bootstrap 虚拟机问题故障排除

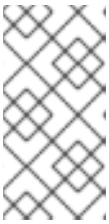
OpenShift Container Platform 安装程序生成 bootstrap 节点虚拟机，该虚拟机处理置备 OpenShift Container Platform 集群节点。

流程

1. 触发安装程序后大约 10 到 15 分钟，使用 virsh 命令检查 bootstrap 虚拟机是否正常工作：

```
$ sudo virsh list
```

```
 Id    Name                           State
----------------------------------------------------
 12    openshift-xf6fq-bootstrap      running
```



注意

bootstrap 虚拟机的名称始终是集群名称，后跟一组随机字符，并以"bootstrap"结尾。

2. 如果 bootstrap 虚拟机在 10-15 分钟后没有运行，请执行以下命令来验证 libvirtd 是否在系统中运行：

```
$ systemctl status libvirtd
```

```
● libvirtd.service - Virtualization daemon
   Loaded: loaded (/usr/lib/systemd/system/libvirtd.service; enabled; vendor preset: enabled)
   Active: active (running) since Tue 2020-03-03 21:21:07 UTC; 3 weeks 5 days ago
     Docs: man:libvirtd(8)
           https://libvirt.org
 Main PID: 9850 (libvirtd)
    Tasks: 20 (limit: 32768)
```

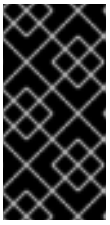
```
Memory: 74.8M
CGroup: /system.slice/libvirtd.service
        ├─ 9850 /usr/sbin/libvirtd
```

如果 bootstrap 虚拟机可以正常工作，请登录它。

3. 使用 virsh console 命令查找 bootstrap 虚拟机的 IP 地址：

```
$ sudo virsh console example.com
```

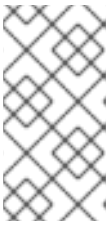
```
Connected to domain example.com
Escape character is ^]
Red Hat Enterprise Linux CoreOS 43.81.202001142154.0 (Ootpa) 4.3
SSH host key: SHA256:BRWJktXZgQQRY5zjuAV0IKZ4WM7i4TiUyMVanqu9Pqg
(ED25519)
SSH host key: SHA256:7+iKGA7VtG5szmk2jB5gl/5EZ+SNcJ3a2g23o0lnlio (ECDSA)
SSH host key: SHA256:DH5VWhvhvagOTaLsYiVNse9ca+ZSW/30OOMed8rIGOc (RSA)
ens3:  fd35:919d:4042:2:c7ed:9a9f:a9ec:7
ens4: 172.22.0.2 fe80::1d05:e52e:be5d:263f
localhost login:
```



重要

当在没有 provisioning 网络的情况下部署 OpenShift Container Platform 集群时，您必须使用公共 IP 地址，而不是专用 IP 地址，如 172.22.0.2。

4. 获取 IP 地址后，使用 ssh 命令登录到 bootstrap 虚拟机：



注意

在上一步的控制台输出中，您可以使用 ens3 提供的 IPv6 IP 地址或 ens 4 提供的 IPv4 IP 地址 。

```
$ ssh core@172.22.0.2
```

如果您无法成功登录 bootstrap 虚拟机，您可能会遇到以下情况之一：

- 您无法访问 172.22.0.0/24 网络。验证 provisioner 和 provisioning 网桥之间的网络连接。如果您使用 provisioning 网络，则可能会出现此问题。

- 您无法通过公共网络访问 bootstrap 虚拟机。当尝试 SSH via baremetal 网络 时，验证 provisioner 主机上的连接情况，特别是 baremetal 网桥的连接。

- 您遇到了 Permission denied(publickey,password,keyboard-interactive) 的问题。当尝试访问 bootstrap 虚拟机时，可能会出现 Permission denied 错误。验证尝试登录到虚拟机的用户的 SSH 密钥是否在 install-config.yaml 文件中设置。

16.6.3.1. Bootstrap 虚拟机无法引导集群节点

在部署过程中，bootstrap 虚拟机可能无法引导集群节点，这会阻止虚拟机使用 RHCOS 镜像置备节点。这可能是因为以下原因：

- install-config.yaml 文件存在问题。

- 使用 baremetal 网络时出现带外网络访问的问题。

要验证这个问题，有三个与 ironic 相关的容器：

- ironic

- ironic-inspector

流程

1. 登录到 bootstrap 虚拟机：

```
$ ssh core@172.22.0.2
```

2. 要检查容器日志，请执行以下操作：

```
[core@localhost ~]$ sudo podman logs -f <container_name>
```

将 <container_name> 替换为 ironic 或 ironic-inspector 之一。如果您遇到 control plane 节点没有从 PXE 引导的问题，请检查 ironic pod。ironic pod 包含有关尝试引导集群节点的信

息，因为它尝试通过 IPMI 登录节点。

潜在原因

集群节点在部署启动时可能处于 ON 状态。

解决方案

在通过 IPMI 开始安装前关闭 OpenShift Container Platform 集群节点：

```
$ ipmitool -I lanplus -U root -P <password> -H <out_of_band_ip> power off
```

16.6.3.2. 检查日志

在下载或访问 RHCOS 镜像时，首先验证 install-config.yaml 配置文件中的 URL 是否正确。

内部 webserver 托管 RHCOS 镜像的示例

```
bootstrapOSImage: http://<ip:port>/rhcos-43.81.202001142154.0-qemu.
<architecture>.qcow2.gz?
sha256=9d999f55ff1d44f7ed7c106508e5deecd04dc3c06095d34d36bf1cd127837e0c
clusterOSImage: http://<ip:port>/rhcos-43.81.202001142154.0-openstack.
<architecture>.qcow2.gz?
sha256=a1bda656fa0892f7b936fdc6b6a6086bddaed5dafacedcd7a1e811abb78fe3b0
```

coreos-downloader 容器从 webserver 或外部 quay.io registry 下载资源（由 install-config.yaml 配置文件指定）。验证 coreos-downloader 容器是否正在运行，并根据需要检查其日志。

流程

1. 登录到 bootstrap 虚拟机：

```
$ ssh core@172.22.0.2
```

2. 运行以下命令，检查 bootstrap 虚拟机中的 coreos-downloader 容器的状态：

```
[core@localhost ~]$ sudo podman logs -f coreos-downloader
```

如果 bootstrap 虚拟机无法访问镜像的 URL，请使用 curl 命令验证虚拟机是否可以访问镜像。

3. 要检查 bootkube 日志，以指示部署阶段是否启动了所有容器，请执行以下操作：

```
[core@localhost ~]$ journalctl -xe
```

```
[core@localhost ~]$ journalctl -b -f -u bootkube.service
```

4. 验证包括 dnsmasq、mariadb、httpd 和 ironic 等所有 pod 都在运行：

```
[core@localhost ~]$ sudo podman ps
```

5. 如果 pod 存在问题，请检查有问题的容器日志。要检查 ironic 服务的日志，请运行以下命令：

```
[core@localhost ~]$ sudo podman logs ironic
```

16.6.4. 调查不可用的 Kubernetes API

当 Kubernetes API 不可用时，检查 control plane 节点以确保它们运行正确的组件。另外，检查主机名解析。

流程

1. 运行以下命令，确保 etcd 在每个 control plane 节点上运行：

```
$ sudo crictl logs $(sudo crictl ps --pod=$(sudo crictl pods --name=etcd-member --quiet) --quiet)
```

2. 如果上一命令失败，请运行以下命令确保 Kubelet 创建 etcd pod：

```
$ sudo crictl pods --name=etcd-member
```

如果没有 pod，请调查 etcd。

3. 使用以下命令检查集群节点，以确保它们具有完全限定域名，而不只是 localhost.localdomain ：

```
$ hostname
```

如果没有设置主机名，请设置正确的主机名。例如：

```
$ sudo hostnamectl set-hostname <hostname>
```

4. 使用 dig 命令，确保每个节点在 DNS 服务器中具有正确的名称解析：

```
$ dig api.<cluster_name>.example.com
```

```
; <<>> DiG 9.11.4-P2-RedHat-9.11.4-26.P2.el8 <<>> api.<cluster_name>.example.com
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NOERROR, id: 37551
;; flags: qr aa rd ra; QUERY: 1, ANSWER: 1, AUTHORITY: 1, ADDITIONAL: 2

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
; COOKIE: 866929d2f8e8563582af23f05ec44203d313e50948d43f60 (good)
;; QUESTION SECTION:
;api.<cluster_name>.example.com. IN A

;; ANSWER SECTION:
api.<cluster_name>.example.com. 10800 IN A 10.19.13.86

;; AUTHORITY SECTION:
<cluster_name>.example.com. 10800 IN NS <cluster_name>.example.com.

;; ADDITIONAL SECTION:
<cluster_name>.example.com. 10800 IN A 10.19.14.247

;; Query time: 0 msec
;; SERVER: 10.19.14.247#53(10.19.14.247)
;; WHEN: Tue May 19 20:30:59 UTC 2020
;; MSG SIZE  rcvd: 140
```

示例中的输出显示 api.<cluster_name>.example.com VIP 的适当 IP 地址为 10.19.13.86。此 IP 地址应当位于 baremetal 网络中。

16.6.5. 对初始化集群失败进行故障排除

安装程序使用 Cluster Version Operator 创建 OpenShift Container Platform 集群的所有组件。当安装程序无法初始化集群时，您可以从 ClusterVersion 和 ClusterOperator 对象检索最重要的信息。

流程

1. 运行以下命令检查 ClusterVersion 对象：

```
$ oc --kubeconfig=${INSTALL_DIR}/auth/kubeconfig get clusterversion -o yaml
```

输出示例

```
apiVersion: config.openshift.io/v1
kind: ClusterVersion
metadata:
  creationTimestamp: 2019-02-27T22:24:21Z
  generation: 1
  name: version
  resourceVersion: "19927"
  selfLink: /apis/config.openshift.io/v1/clusterversions/version
  uid: 6e0f4cf8-3ade-11e9-9034-0a923b47ded4
spec:
  channel: stable-4.1
  clusterID: 5ec312f9-f729-429d-a454-61d4906896ca
status:
  availableUpdates: null
  conditions:
  - lastTransitionTime: 2019-02-27T22:50:30Z
    message: Done applying 4.1.1
    status: "True"
    type: Available
  - lastTransitionTime: 2019-02-27T22:50:30Z
    status: "False"
    type: Failing
  - lastTransitionTime: 2019-02-27T22:50:30Z
    message: Cluster version is 4.1.1
    status: "False"
    type: Progressing
  - lastTransitionTime: 2019-02-27T22:24:31Z
    message: 'Unable to retrieve available updates: unknown version 4.1.1
    reason: RemoteFailed
    status: "False"
    type: RetrievedUpdates
  desired:
    image: registry.svc.ci.openshift.org/openshift/origin-
release@sha256:91e6f754975963e7db1a9958075eb609ad226968623939d262d1cf45e9db
c39a
```

```
    version: 4.1.1
  history:
  - completionTime: 2019-02-27T22:50:30Z
    image: registry.svc.ci.openshift.org/openshift/origin-
release@sha256:91e6f754975963e7db1a9958075eb609ad226968623939d262d1cf45e9db
c39a
    startedTime: 2019-02-27T22:24:31Z
    state: Completed
    version: 4.1.1
  observedGeneration: 1
  versionHash: Wa7as_ik1qE=
```

2. 运行以下命令来查看条件：

```
$ oc --kubeconfig=${INSTALL_DIR}/auth/kubeconfig get clusterversion version \
    -o=jsonpath='{range .status.conditions[*]}{.type}{" "}{.status}{" "}{.message}{"\n"}
{end}'
```

某些最重要的条件包括 Failing、Available 和 Progressing。

输出示例

```
Available True Done applying 4.1.1
Failing False
Progressing False Cluster version is 4.0.0-0.alpha-2019-02-26-194020
RetrievedUpdates False Unable to retrieve available updates: unknown version 4.1.1
```

3. 运行以下命令检查 ClusterOperator 对象：

```
$ oc --kubeconfig=${INSTALL_DIR}/auth/kubeconfig get clusteroperator
```

该命令返回集群 Operator 的状态。

输出示例

```
NAME                                       VERSION   AVAILABLE   PROGRESSING   FAILING   SINCE
cluster-baremetal-operator                             True        False         False     17m
cluster-autoscaler                                     True        False         False     17m
cluster-storage-operator                               True        False         False     10m
console                                                True        False         False     7m21s
dns                                                    True        False         False     31m
image-registry                                         True        False         False     9m58s
ingress                                                True        False         False     10m
kube-apiserver                                         True        False         False     28m
kube-controller-manager                                True        False         False     21m
kube-scheduler                                         True        False         False     25m
machine-api                                            True        False         False     17m
machine-config                                         True        False         False     17m
marketplace-operator                                   True        False         False     10m
monitoring                                             True        False         False     8m23s
network                                                True        False         False     13m
node-tuning                                            True        False         False     11m
openshift-apiserver                                    True        False         False     15m
openshift-authentication                               True        False         False     20m
openshift-cloud-credential-operator                    True        False         False     18m
openshift-controller-manager                           True        False         False     10m
openshift-samples                                      True        False         False     8m42s
operator-lifecycle-manager                             True        False         False     17m
service-ca                                             True        False         False     30m
```

4. 运行以下命令检查单个集群 Operator：

```
$ oc --kubeconfig=${INSTALL_DIR}/auth/kubeconfig get clusteroperator <operator> -oyaml 1
```

1 将 <operator> 替换为集群 Operator 的名称。此命令可用于识别集群 Operator 没有达到 Available 状态的原因或处于 Failed 状态。

输出示例

```
apiVersion: config.openshift.io/v1
kind: ClusterOperator
metadata:
  creationTimestamp: 2019-02-27T22:47:04Z
  generation: 1
  name: monitoring
  resourceVersion: "24677"
```

```
  selfLink: /apis/config.openshift.io/v1/clusteroperators/monitoring
  uid: 9a6a5ef9-3ae1-11e9-bad4-0a97b6ba9358
spec: {}
status:
  conditions:
  - lastTransitionTime: 2019-02-27T22:49:10Z
    message: Successfully rolled out the stack.
    status: "True"
    type: Available
  - lastTransitionTime: 2019-02-27T22:49:10Z
    status: "False"
    type: Progressing
  - lastTransitionTime: 2019-02-27T22:49:10Z
    status: "False"
    type: Failing
  extension: null
  relatedObjects: null
  version: ""
```

5. 要获取集群 Operator 的状态条件，请运行以下命令：

```
$ oc --kubeconfig=${INSTALL_DIR}/auth/kubeconfig get clusteroperator <operator> \
    -o=jsonpath='{range .status.conditions[*]}{.type}{" "}{.status}{" "}{.message}{"\n"}
{end}'
```

将 <operator> 替换为以上其中一个 Operator 的名称。

输出示例

```
Available True Successfully rolled out the stack
Progressing False
Failing False
```

6. 要检索集群 Operator 拥有的对象列表，请执行以下命令：

```
oc --kubeconfig=${INSTALL_DIR}/auth/kubeconfig get clusteroperator kube-apiserver
\
  -o=jsonpath='{.status.relatedObjects}'
```

输出示例

```
[map[resource:kubeapiservers group:operator.openshift.io name:cluster] map[group:
name:openshift-config resource:namespaces] map[group: name:openshift-config-
managed resource:namespaces] map[group: name:openshift-kube-apiserver-operator
resource:namespaces] map[group: name:openshift-kube-apiserver
resource:namespaces]]
```

16.6.6. 对获取控制台 URL 失败进行故障排除

安装程序通过在 openshift-console 命名空间中使用 [route][route-object] 检索 OpenShift Container Platform 控制台的 URL。如果安装程序失败，请检索控制台的 URL，请使用以下步骤。

流程

1. 运行以下命令，检查控制台路由器是否处于 Available 或 Failing 状态：

```
$ oc --kubeconfig=${INSTALL_DIR}/auth/kubeconfig get clusteroperator console -
oyaml
```

```
apiVersion: config.openshift.io/v1
kind: ClusterOperator
metadata:
  creationTimestamp: 2019-02-27T22:46:57Z
  generation: 1
  name: console
  resourceVersion: "19682"
  selfLink: /apis/config.openshift.io/v1/clusteroperators/console
  uid: 960364aa-3ae1-11e9-bad4-0a97b6ba9358
spec: {}
status:
  conditions:
  - lastTransitionTime: 2019-02-27T22:46:58Z
    status: "False"
    type: Failing
  - lastTransitionTime: 2019-02-27T22:50:12Z
    status: "False"
    type: Progressing
  - lastTransitionTime: 2019-02-27T22:50:12Z
    status: "True"
    type: Available
  - lastTransitionTime: 2019-02-27T22:46:57Z
    status: "True"
    type: Upgradeable
```

```
  extension: null
  relatedObjects:
  - group: operator.openshift.io
    name: cluster
    resource: consoles
  - group: config.openshift.io
    name: cluster
    resource: consoles
  - group: oauth.openshift.io
    name: console
    resource: oauthclients
  - group: ""
    name: openshift-console-operator
    resource: namespaces
  - group: ""
    name: openshift-console
    resource: namespaces
  versions: null
```

2. 执行以下命令手动检索控制台 URL：

```
$ oc --kubeconfig=${INSTALL_DIR}/auth/kubeconfig get route console -n openshift-
console \
    -o=jsonpath='{.spec.host}' console-openshift-console.apps.adahiya-
1.devcluster.openshift.com
```

16.6.7. 对将入口证书添加到 kubeconfig 失败的问题

安装程序将默认入口证书添加到 ${INSTALL_DIR}/auth/kubeconfig 中的可信客户端证书颁发机构列表中。如果安装程序无法将入口证书添加到 kubeconfig 文件中，您可以从集群中检索证书并添加它。

流程

1. 使用以下命令从集群中删除证书：

```
$ oc --kubeconfig=${INSTALL_DIR}/auth/kubeconfig get configmaps default-ingress-
cert \
    -n openshift-config-managed -o=jsonpath='{.data.ca-bundle\.crt}'
```

```
-----BEGIN CERTIFICATE-----
MIIC/TCCAeWgAwIBAgIBATANBgkqhkiG9w0BAQsFADAuMSwwKgYDVQQDDCNjbHV
z
dGVyLWluZ3Jlc3Mtb3BlcmF0b3JAMTU1MTMwNzU4OTAeFw0xOTAyMjcyMjQ2Mjha
Fw0yMTAyMjYyMjQ2MjlaMC4xLDAqBgNVBAMMI2NsdXN0ZXItaW5ncmVzcy1vcGVy
YXRvckAxNTUxMzA3NTg5MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA
uCA4fQ+2YXoXSUL4h/mcvJfrgpBfKBW5hfB8NcgXeCYiQPnCKblH1sEQnl3VC5Pk
2OfNCF3PUlfm4i8CHC95a7nCkRjmJNg1gVrWCvS/ohLgnO0BvszSiRLxlpuo3C4S
EVqqvxValHcbdAXWgZLQoYZXV7RMz8yZjl5CfhDaaltyBFj3GtlJkXgUwp/5sUfl
```

```
LDXW8MM6AXfuG+kweLdLCMm3g8WLLfLBLvVBKB+4IhIH7Il0buOz04RKhnYN+Ebw
tcvFi55vwuUCWMnGhWHGEQ8sWm/wLnNlOwsUz7S1/sW8nj87GFHzgkaVM9EOnoNl
gKhMBK9ltNzjrP6dgiKBCQIDAQABoyYwJDAOBgNVHQ8BAf8EBAMCAqQwEgYDVR0T
AQH/BAgwBgEB/wIBADANBgkqhkiG9w0BAQsFAAOCAQEAq+vi0sFKudaZ9aUQMMha
CeWx9CZvZBblnAWT/61UdpZKpFi4eJ2d33lGcfKwHOi2NP/iSKQBebfG0iNLVVPz
vwLbSG1i9R9GLdAbnHpPT9UG6fLaDloKpnKiBfGENfxeiq5vTln2bAgivxrVlyiq
+MdDXFAWb6V4u2xh6RChI7akNsS3oU9PZ9YOs5e8vJp2YAEphht05X0swA+X8V8T
C278FFifpo0h3Q0Dbv8Rfn4UpBEtN4KkLeS+JeT+0o2XOsFZp7Uhr9yFlodRsnNo
H/Uwmab28ocNrGNiEVaVH6eTTQeeZuOdoQzUbClElpVmkrNGY0M42K0PvOQ/e7+y
AQ==
-----END CERTIFICATE-----
```

2. 将证书添加到 ${INSTALL_DIR}/auth/kubeconfig 文件中的 client-certificate-authority-data 字段中。

16.6.8. 对集群节点的 SSH 访问故障排除

为提高安全性，您无法默认从集群外部通过 SSH 连接到集群。但是，您可以从 provisioner 节点访问 control plane 和 worker 节点。如果无法从 provisioner 节点通过 SSH 连接到集群节点，则节点可能会在 bootstrap 虚拟机上等待。control plane 节点从 bootstrap 虚拟机检索其引导配置，如果它们没有检索引导配置，则无法成功引导。

流程

1. 如果您有对节点的物理访问权限，请检查其控制台输出以确定它们是否已成功引导。如果节点仍然检索其引导配置，则 bootstrap 虚拟机可能会出现问题。

2. 确保您在 install-config.yaml 文件中配置了 sshKey: '<ssh_pub_key>' 设置，其中 <ssh_pub_key> 是 provisioner 节点上的 kni 用户的公钥。

16.6.9. 集群节点不能 PXE 引导

当 OpenShift Container Platform 集群节点无法 PXE 引导时，请在不能 PXE 引导的集群节点上执行以下检查。在没有 provisioning 网络安装 OpenShift Container Platform 集群时，此流程不适用。

流程

1. 检查与 provisioning 网络的网络连接。

2. 确保 provisioning 网络的 NIC 上启用了 PXE，并为所有其他 NIC 禁用 PXE。

3. 验证 **install-config.yaml** 配置文件是否包含 **rootDeviceHints** 参数，以及连接到 **provisioning** 网络的 **NIC** 的引导 **MAC** 地址。例如：

control plane 节点设置

```
bootMACAddress: 24:6E:96:1B:96:90 # MAC of bootable provisioning NIC
```

Worker 节点设置

```
bootMACAddress: 24:6E:96:1B:96:90 # MAC of bootable provisioning NIC
```

16.6.10. 安装不会创建 worker 节点

安装程序不会直接置备 **worker** 节点。相反，**Machine API Operator** 会在受支持的平台上扩展节点。如果 **worker** 节点在 **15** 到 **20** 分钟后没有创建，具体取决于集群互联网连接的速度，请调查 **Machine API Operator**。

流程

1. 运行以下命令检查 **Machine API Operator**：

```
$ oc --kubeconfig=${INSTALL_DIR}/auth/kubeconfig \
  --namespace=openshift-machine-api get deployments
```

如果环境中没有设置 **${INSTALL_DIR}**，请将该值替换为安装目录的名称。

输出示例

```
NAME                          READY   UP-TO-DATE   AVAILABLE   AGE
cluster-autoscaler-operator   1/1     1            1           86m
```

```
cluster-baremetal-operator    1/1     1          1          86m
machine-api-controllers       1/1     1          1          85m
machine-api-operator          1/1     1          1          86m
```

2. 运行以下命令检查机器控制器日志：

```
$ oc --kubeconfig=${INSTALL_DIR}/auth/kubeconfig \
    --namespace=openshift-machine-api logs deployments/machine-api-controllers \
    --container=machine-controller
```

16.6.11. Cluster Network Operator 故障排除

Cluster Network Operator 负责部署网络组件。它在安装过程中的早期运行，在 control plane 节点启动后，但安装程序删除 bootstrap control plane 前。此 Operator 的问题可能表示安装程序问题。

流程

1. 运行以下命令确保网络配置存在：

```
$ oc get network -o yaml cluster
```

如果不存在，安装程序不会创建它。要找出原因，请运行以下命令：

```
$ openshift-install create manifests
```

查看清单，以确定安装程序没有创建网络配置的原因。

2. 输入以下命令来确保网络正在运行：

```
$ oc get po -n openshift-network-operator
```

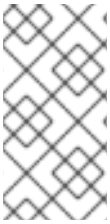
16.6.12. 无法使用 BMC 发现新的裸机主机

在某些情况下，安装程序将无法发现新的裸机主机并发出错误，因为它无法挂载远程虚拟介质共享。

例如：

```
ProvisioningError 51s metal3-baremetal-controller Image provisioning failed: Deploy step
deploy.deploy failed with BadRequestError: HTTP POST
https://<bmc_address>/redfish/v1/Managers/iDRAC.Embedded.1/VirtualMedia/CD/Actions/Virt
ualMedia.InsertMedia
returned code 400.
Base.1.8.GeneralError: A general error has occurred. See ExtendedInfo for more information
Extended information: [
  {
    "Message": "Unable to mount remote share https://<ironic_address>/redfish/boot-
<uuid>.iso.",
    "MessageArgs": [
      "https://<ironic_address>/redfish/boot-<uuid>.iso"
    ],
    "MessageArgs@odata.count": 1,
    "MessageId": "IDRAC.2.5.RAC0720",
    "RelatedProperties": [
      "#/Image"
    ],
    "RelatedProperties@odata.count": 1,
    "Resolution": "Retry the operation.",
    "Severity": "Informational"
  }
].
```

在这种情况下，如果您使用带有未知证书颁发机构的虚拟介质，您可以配置基板管理控制器 (BMC) 远程文件共享设置，以信任未知证书颁发机构以避免这个错误。



注意

这个解析已在带有 Dell iDRAC 9 的 OpenShift Container Platform 4.11 上测试，固件版本 5.10.50。

16.6.13. 无法加入集群的 worker 节点故障排除

安装程序置备的集群使用 DNS 服务器部署，其中包含 api-int.<cluster_name>.<base_domain> URL 的 DNS 条目。如果集群中的节点使用外部或上游 DNS 服务器解析 api-int.<cluster_name>.<base_domain> URL，且没有这样的条目，则 worker 节点可能无法加入集群。确保集群中的所有节点都可以解析域名。

流程

1. 添加 DNS A/AAAA 或 CNAME 记录，以内部标识 API 负载均衡器。例如，在使用 dnsmasq 时，修改 dnsmasq.conf 配置文件：

```
$ sudo nano /etc/dnsmasq.conf
```

```
address=/api-int.<cluster_name>.<base_domain>/<IP_address>
address=/api-int.mycluster.example.com/192.168.1.10
address=/api-int.mycluster.example.com/2001:0db8:85a3:0000:0000:8a2e:0370:7334
```

2. 添加 DNS PTR 记录以内部标识 API 负载均衡器。例如，在使用 dnsmasq 时，修改 dnsmasq.conf 配置文件：

```
$ sudo nano /etc/dnsmasq.conf
```

```
ptr-record=<IP_address>.in-addr.arpa,api-int.<cluster_name>.<base_domain>
ptr-record=10.1.168.192.in-addr.arpa,api-int.mycluster.example.com
```

3. 重启 DNS 服务器。例如，在使用 dnsmasq 时执行以下命令：

```
$ sudo systemctl restart dnsmasq
```

这些记录必须可以从集群中的所有节点解析。

16.6.14. 清理以前的安装

如果以前部署失败，在尝试再次部署 OpenShift Container Platform 前从失败的尝试中删除工件。

流程

1. 在安装 OpenShift Container Platform 集群前关闭所有裸机节点：

```
$ ipmitool -I lanplus -U <user> -P <password> -H <management_server_ip> power off
```

2. 删除所有旧的 bootstrap 资源（如果以前的部署尝试中保留）：

```
for i in $(sudo virsh list | tail -n +3 | grep bootstrap | awk {'print $2'});
do
  sudo virsh destroy $i;
  sudo virsh undefine $i;
  sudo virsh vol-delete $i --pool $i;
  sudo virsh vol-delete $i.ign --pool $i;
```

```
  sudo virsh pool-destroy $i;
  sudo virsh pool-undefine $i;
done
```

16.6.15. 创建 registry 的问题

在创建断开连接的 registry 时，在尝试镜像 registry 时，可能会遇到 "User Not Authorized" 错误。如果您没有将新身份验证附加到现有的 pull-secret.txt 文件中，可能会发生这个错误。

流程

1. 检查以确保身份验证成功：

```
$ /usr/local/bin/oc adm release mirror \
  -a pull-secret-update.json
  --from=$UPSTREAM_REPO \
  --to-release-image=$LOCAL_REG/$LOCAL_REPO:${VERSION} \
  --to=$LOCAL_REG/$LOCAL_REPO
```

注意

用于镜像安装镜像的变量输出示例：

```
UPSTREAM_REPO=${RELEASE_IMAGE}
LOCAL_REG=<registry_FQDN>:<registry_port>
LOCAL_REPO='ocp4/openshift4'
```

在为 OpenShift 安装设置环境 部分的 获取 OpenShift 安装程序 步骤中，设置了 RELEASE_IMAGE 和 VERSION 的值。

2. 镜像 registry 后，确认您可以在断开连接的环境中访问它：

```
$ curl -k -u <user>:<password> https://registry.example.com:
<registry_port>/v2/_catalog
{"repositories":["<Repo_Name>"]}
```

16.6.16. 其它问题

16.6.16.1. 解决 运行时网络未就绪 错误

部署集群后，您可能会收到以下错误：

> `runtime network not ready: NetworkReady=false reason:NetworkPluginNotReady message:Network plugin returns error: Missing CNI default network`

Cluster Network Operator 负责部署网络组件以响应安装程序创建的特殊对象。它会在安装过程的早期阶段运行（在 control plane(master)节点启动后，bootstrap control plane 被停止前运行。它可能会显示更细微的安装程序问题，比如启动 control plane (master)节点时延迟时间过长，或者 apiserver 通讯的问题。

流程

1. **检查 openshift-network-operator 命名空间中的 pod：**

```
$ oc get all -n openshift-network-operator
```

```
NAME                                READY STATUS          RESTARTS   AGE
pod/network-operator-69dfd7b577-bg89v   0/1   ContainerCreating 0          149m
```

2. **在 provisioner 节点上，确定存在网络配置：**

```
$ kubectl get network.config.openshift.io cluster -oyaml
```

```
apiVersion: config.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  serviceNetwork:
  - 172.30.0.0/16
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  networkType: OVNKubernetes
```

如果不存在，安装程序不会创建它。要确定安装程序没有创建它的原因，请执行以下操作：

```
$ openshift-install create manifests
```

3. **检查 network-operator 是否正在运行：**

```
$ kubectl -n openshift-network-operator get pods
```

4. 检索日志：

```
$ kubectl -n openshift-network-operator logs -l "name=network-operator"
```

在具有三个或更多 control plane 节点的高可用性集群中，Operator 将执行领导选举机制，所有其他 Operator 将会休眠。如需了解更多详细信息，请参阅 故障排除。

16.6.16.2. 解决 "No disk found with matching rootDeviceHints" 错误消息

部署集群后，您可能会收到以下出错信息：

```
No disk found with matching rootDeviceHints
```

要解决 No disk found with matching rootDeviceHints 错误，一个临时解决方法是将 rootDeviceHints 更改为 minSizeGigabytes: 300。

更改 rootDeviceHints 设置后，使用以下命令引导 CoreOS，然后使用以下命令验证磁盘信息：

```
$ udevadm info /dev/sda
```

如果您使用 DL360 Gen 10 服务器，请注意它们有一个 SD-card 插槽，它可能会被分配 /dev/sda 设备名称。如果服务器中没有 SD 卡，可能会导致冲突。确定在服务器的 BIOS 设置中禁用 SD 卡插槽。

如果 minSizeGigabytes 临时解决方案没有满足要求，您可能需要将 rootDeviceHints 恢复为 /dev/sda。此更改允许 ironic 镜像成功引导。

修复此问题的另一种方法是使用磁盘的串行 ID。但是，请注意，发现串行 ID 可能具有挑战性，并且可能会导致配置文件无法读取。如果选择此路径，请确保使用之前记录的命令收集串行 ID，并将其合并到您的配置中。

16.6.16.3. 集群节点没有通过 DHCP 获得正确的 IPv6 地址

如果集群节点没有通过 DHCP 获得正确的 IPv6 地址，请检查以下内容：

1. 确保保留的 IPv6 地址不在 DHCP 范围内。

2. 在 DHCP 服务器上的 IP 地址保留中，确保保留指定了正确的 DHCP 唯一标识符(DUID)。例如：

```
# This is a dnsmasq dhcp reservation, 'id:00:03:00:01' is the client id and
'18:db:f2:8c:d5:9f' is the MAC Address for the NIC
id:00:03:00:01:18:db:f2:8c:d5:9f,openshift-master-1,[2620:52:0:1302::6]
```

3. 确保路由声明正常工作。

4. 确保 DHCP 服务器正在侦听提供 IP 地址范围所需的接口。

16.6.16.4. 集群节点没有通过 DHCP 获得正确的主机名

在 IPv6 部署过程中，集群节点必须通过 DHCP 获得其主机名。有时 NetworkManager 不会立即分配主机名。control plane(master)节点可能会报告错误，例如：

```
Failed Units: 2
  NetworkManager-wait-online.service
  nodeip-configuration.service
```

这个错误表示集群节点可能在没有从 DHCP 服务器接收主机名的情况下引导，这会导致 kubelet 使用 localhost.localdomain 主机名引导。要解决错误，请强制节点更新主机名。

流程

1. 检索 主机名 ：

```
[core@master-X ~]$ hostname
```

如果主机名是 localhost，请执行以下步骤。

注意

其中 X 是 control plane 节点号。

2. 强制集群节点续订 DHCP 租期：

```
[core@master-X ~]$ sudo nmcli con up "<bare_metal_nic>"
```

将 <bare_metal_nic> 替换为与 baremetal 网络对应的有线连接。

3. 再次检查 主机名 ：

```
[core@master-X ~]$ hostname
```

4. 如果主机名仍然是 localhost.localdomain，重启 NetworkManager ：

```
[core@master-X ~]$ sudo systemctl restart NetworkManager
```

5. 如果主机名仍然是 localhost.localdomain，请等待几分钟并再次检查。如果主机名仍为 localhost.localdomain，请重复前面的步骤。

6. 重启 nodeip-configuration 服务：

```
[core@master-X ~]$ sudo systemctl restart nodeip-configuration.service
```

此服务将使用正确的主机名引用来重新配置 kubelet 服务。

7. 因为 kubelet 在上一步中更改，所以重新载入单元文件定义：

```
[core@master-X ~]$ sudo systemctl daemon-reload
```

8. 重启 kubelet 服务：

```
[core@master-X ~]$ sudo systemctl restart kubelet.service
```

9. 确保 kubelet 使用正确的主机名引导：

```
[core@master-X ~]$ sudo journalctl -fu kubelet.service
```

如果集群节点在集群启动并运行后没有通过 DHCP 获得正确的主机名，比如在重启过程中，集群会有一个待处理的 csr。不要 批准 csr，否则可能会出现其他问题。

处理 csr

1. 在集群上获取 CSR：

```
$ oc get csr
```

2. 验证待处理的 csr 是否包含 Subject Name: localhost.localdomain:

```
$ oc get csr <pending_csr> -o jsonpath='{.spec.request}' | base64 --decode | openssl req -noout -text
```

3. 删除包含 Subject Name: localhost.localdomain 的任何 csr ：

```
$ oc delete csr <wrong_csr>
```

16.6.16.5. 路由无法访问端点

在安装过程中，可能会出现虚拟路由器冗余协议(VRRP)冲突。如果之前使用特定集群名称部署的 OpenShift Container Platform 节点仍在运行，但不是使用相同集群名称部署的当前 OpenShift Container Platform 集群的一部分，则可能会出现冲突。例如，一个集群使用集群名称 openshift 部署，它 部署了三个 control plane(master)节点和三个 worker 节点。之后，一个单独的安装使用相同的集群名称 openshift，但这个重新部署只安装了三个 control plane(master)节点，使以前部署的三个 worker 节点处于 ON 状态。这可能导致 Virtual Router Identifier(VRID)冲突和 VRRP 冲突。

1. 获取路由：

```
$ oc get route oauth-openshift
```

2. 检查服务端点：

```
$ oc get svc oauth-openshift
```

```
NAME              TYPE      CLUSTER-IP      EXTERNAL-IP   PORT(S)   AGE
oauth-openshift   ClusterIP   172.30.19.162   <none>        443/TCP   59m
```

3. 尝试从 control plane(master)节点访问该服务：

```
[core@master0 ~]$ curl -k https://172.30.19.162
```

```
{
  "kind": "Status",
  "apiVersion": "v1",
  "metadata": {
  },
  "status": "Failure",
  "message": "forbidden: User \"system:anonymous\" cannot get path \"/\"",
  "reason": "Forbidden",
  "details": {
  },
  "code": 403
```

4. 识别 provisioner 节点的 authentication-operator 错误：

```
$ oc logs deployment/authentication-operator -n openshift-authentication-operator
```

```
Event(v1.ObjectReference{Kind:"Deployment", Namespace:"openshift-authentication-
operator", Name:"authentication-operator", UID:"225c5bd5-b368-439b-9155-
5fd3c0459d98", APIVersion:"apps/v1", ResourceVersion:"", FieldPath:""}): type:
'Normal' reason: 'OperatorStatusChanged' Status for clusteroperator/authentication
changed: Degraded message changed from "IngressStateEndpointsDegraded: All 2
endpoints for oauth-server are reporting"
```

解决方案

1. 确保每个部署的集群名称都是唯一的，确保没有冲突。

2. 关闭所有不是使用相同集群名称的集群部署的一部分的节点。否则，OpenShift Container Platform 集群的身份验证 pod 可能无法成功启动。

16.6.16.6. 在 Firstboot 过程中 Ignition 失败

在 Firstboot 过程中，Ignition 配置可能会失败。

流程

1. 连接到 Ignition 配置失败的节点：

```
Failed Units: 1
  machine-config-daemon-firstboot.service
```

2. 重启 machine-config-daemon-firstboot 服务：

```
[core@worker-X ~]$ sudo systemctl restart machine-config-daemon-firstboot.service
```

16.6.16.7. NTP 不同步

OpenShift Container Platform 集群的部署需要集群节点间的 NTP 同步时钟。如果没有同步时钟，如果时间差大于 2 秒，则部署可能会因为时钟偏移而失败。

流程

1. 检查集群节点的 AGE 的不同。例如：

```
$ oc get nodes
```

```
NAME                         STATUS   ROLES    AGE   VERSION
master-0.cloud.example.com   Ready    master   145m   v1.29.4
master-1.cloud.example.com   Ready    master   135m   v1.29.4
master-2.cloud.example.com   Ready    master   145m   v1.29.4
worker-2.cloud.example.com   Ready    worker   100m   v1.29.4
```

2. 检查时钟偏移导致的时间延迟。例如：

```
$ oc get bmh -n openshift-machine-api
```

```
master-1   error registering master-1  ipmi://<out_of_band_ip>
```

```
$ sudo timedatectl
```

```
               Local time: Tue 2020-03-10 18:20:02 UTC
           Universal time: Tue 2020-03-10 18:20:02 UTC
                 RTC time: Tue 2020-03-10 18:36:53
                Time zone: UTC (UTC, +0000)
System clock synchronized: no
              NTP service: active
          RTC in local TZ: no
```

处理现有集群中的时钟偏移

1.

创建一个 Butane 配置文件，其中包含要发送到节点的 chrony.conf 文件的内容。在以下示例中，创建 99-master-chrony.bu 将文件添加到 control plane 节点。您可以修改 worker 节点的 文件，或者对 worker 角色重复此步骤。

注意

有关 Butane 的信息，请参阅"使用 Butane 创建机器配置"。

```
variant: openshift
version: 4.16.0
metadata:
  name: 99-master-chrony
  labels:
    machineconfiguration.openshift.io/role: master
storage:
  files:
  - path: /etc/chrony.conf
    mode: 0644
    overwrite: true
    contents:
      inline: |
        server <NTP_server> iburst 1
        stratumweight 0
        driftfile /var/lib/chrony/drift
        rtcsync
        makestep 10 3
        bindcmdaddress 127.0.0.1
        bindcmdaddress ::1
        keyfile /etc/chrony.keys
        commandkey 1
        generatecommandkey
        noclientlog
        logchange 0.5
        logdir /var/log/chrony
```

1

将 <NTP_server> 替换为 NTP 服务器的 IP 地址。

2. 使用 Butane 生成 MachineConfig 对象文件 99-master-chrony.yaml，其中包含要交付至节点的配置：

```
$ butane 99-master-chrony.bu -o 99-master-chrony.yaml
```

3. 应用 MachineConfig 对象文件：

-

```
$ oc apply -f 99-master-chrony.yaml
```

4. 确保 系统时钟同步 值为 yes ：

```
$ sudo timedatectl
```

```
               Local time: Tue 2020-03-10 19:10:02 UTC
           Universal time: Tue 2020-03-10 19:10:02 UTC
                 RTC time: Tue 2020-03-10 19:36:53
                Time zone: UTC (UTC, +0000)
System clock synchronized: yes
              NTP service: active
          RTC in local TZ: no
```

要在部署前设置时钟同步，请生成清单文件并将此文件添加到 openshift 目录中。例如：

```
$ cp chrony-masters.yaml ~/clusterconfigs/openshift/99_masters-chrony-
configuration.yaml
```

然后，继续创建集群。

16.6.17. 检查安装

安装后，请确保安装程序成功部署节点和 pod。

流程

1. 当正确安装 OpenShift Container Platform 集群节点时，在 STATUS 列中会显示以下 Ready 状态 ：

```
$ oc get nodes
```

```
NAME                 STATUS   ROLES          AGE  VERSION
master-0.example.com   Ready    master,worker   4h   v1.29.4
master-1.example.com   Ready    master,worker   4h   v1.29.4
master-2.example.com   Ready    master,worker   4h   v1.29.4
```

2. 确认安装程序成功部署了所有 pod。以下命令将移除仍在运行或已完成的 pod 作为输出的一部分。

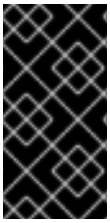
```
$ oc get pods --all-namespaces | grep -iv running | grep -iv complete
```

-

第 17 章 安装 IBM CLOUD BARE METAL (CLASSIC)

17.1. 先决条件

您可以使用安装程序置备的安装在 IBM Cloud® Bare Metal (Classic) 节点上安装 OpenShift Container Platform。本文档描述了在 IBM Cloud® 节点上安装 OpenShift Container Platform 时的先决条件和步骤。



重要

红帽仅在置备网络上支持 IPMI 和 PXE。红帽尚未测试 Red Hat Fish、虚拟介质或其他补充技术，如 IBM Cloud® 部署的安全引导。需要 provisioning 网络。

OpenShift Container Platform 安装程序置备的安装需要：

- 安装了 Red Hat Enterprise Linux CoreOS (RHCOS) 8.x 的一个节点，用于运行置备程序
- 三个 control plane 节点
- 一个可路由的网络
- 一个 provisioning 网络

在 IBM Cloud® Bare Metal (Classic) 上启动 OpenShift Container Platform 安装程序置备的安装前，需要满足以下先决条件和要求。

17.1.1. 设置 IBM Cloud Bare Metal (Classic) 基础架构

要在 IBM Cloud® Bare Metal (Classic) 基础架构上部署 OpenShift Container Platform 集群，您必须首先置备 IBM Cloud® 节点。

重要

红帽支持 provisioning 网络上的 IPMI 和 PXE。红帽尚未测试 Red Hat Fish、虚拟介质或其他补充技术，如 IBM Cloud® 部署的安全引导。provisioning 网络是必需的。

您可以使用 IBM Cloud® API 自定义 IBM Cloud® 节点。在创建 IBM Cloud® 节点时，您必须考虑以下要求：

每个集群使用一个数据中心

OpenShift Container Platform 集群中的所有节点都必须在同一 IBM Cloud® 数据中心中运行。

创建公共和私有 VLAN

创建具有单个公共 VLAN 和单个私有 VLAN 的所有节点。

确定子网有足够的 IP 地址

IBM Cloud® 公共 VLAN 子网默认使用一个 /28 前缀，该前缀提供 16 个 IP 地址。这足以一个包含三个 control plane 节点、四个 worker 节点以及两个用于 baremetal 网络上的 API VIP 和 Ingress VIP 的集群。对于较大的集群，可能需要一个较小的前缀。

IBM Cloud® 私有 VLAN 子网默认使用 /26 前缀，该前缀提供 64 个 IP 地址。IBM Cloud® Bare Metal (Classic) 使用专用网络 IP 地址访问每个节点的 Baseboard Management Controller (BMC)。OpenShift Container Platform 为 provisioning 网络创建一个额外的子网。用于 provisioning 网络子网路由的网络流量，通过专用 VLAN 进行路由。对于较大的集群，可能需要一个较小的前缀。

表 17.1. 每个前缀的 IP 地址

IP 地址	prefix
32	**/27**
64	**/26**
128	**/25**
256	**/24**

配置 NIC

OpenShift Container Platform 使用两个网络部署：

-

provisioning ： provisioning 网络是一个不可路由的网络，用于在作为 OpenShift Container Platform 集群一部分的每个节点上置备底层操作系统。

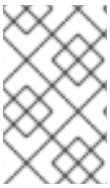
- **baremetal ： baremetal 网络是一个可路由的网络。您可以使用任何 NIC 顺序与 baremetal 网络进行接口，只要它不是 provisioningNetworkInterface 配置设置中指定的 NIC，或者 NIC 关联到用于 provisioning 网络的节点的 bootMACAddress 配置设置。**

虽然集群节点可以包含多于 2 个 NIC，但安装过程只关注于前两个 NIC:例如：

NIC	网络	VLAN
NIC1	**provisioning**	<provisioning_vlan>
NIC2	**baremetal**	<baremetal_vlan>

在上例中，所有 control plane 和 worker 节点上的 NIC1 连接到仅用于安装 OpenShift Container Platform 集群的不可路由网络（provisioning）。所有 control plane 和 worker 节点上的 NIC2 连接到 routable baremetal 网络。

PXE	引导顺序
NIC1 PXE-enabled **provisioning** 网络	1
NIC2 **baremetal** 网络.	2



注意

确保用于 provisioning 网络的 NIC 上启用了 PXE，并在所有其他 NIC 上禁用。

配置规范名称

客户端通过 baremetal 网络访问 OpenShift Container Platform 集群节点。配置 IBM Cloud® 子域或子区，其中规范名称扩展是集群名称。

```
<cluster_name>.<domain>
```

例如：

```
test-cluster.example.com
```

创建 DNS 条目

您必须为以下内容创建 DNS A 记录条目，解析为公共子网中未使用的 IP 地址：

用法	主机名	IP
API	api.<cluster_name>.<domain>	<ip>
Ingress LB(apps)	*.apps.<cluster_name>.<domain>	<ip>

在置备后，control plane 和 worker 节点已经有 DNS 条目。

下表提供了完全限定域名的示例。API 和 Nameserver 地址以规范名称扩展开头。control plane 和 worker 节点的主机名是示例，您可以使用您喜欢的任何主机命名约定。

用法	主机名	IP
API	api.<cluster_name>.<domain>	<ip>
Ingress LB(apps)	*.apps.<cluster_name>.<domain>	<ip>
provisioner 节点	provisioner.<cluster_name>.<domain>	<ip>
Master-0	openshift-master-0.<cluster_name>.<domain>	<ip>
Master-1	openshift-master-1.<cluster_name>.<domain>	<ip>
Master-2	openshift-master-2.<cluster_name>.<domain>	<ip>
Worker-0	openshift-worker-0.<cluster_name>.<domain>	<ip>
Worker-1	openshift-worker-1.<cluster_name>.<domain>	<ip>
Worker-n	openshift-worker-n.<cluster_name>.<domain>	<ip>

OpenShift Container Platform 包含使用集群成员资格信息来生成 A 记录的功能。这会将节点名称解析为其 IP 地址。使用 API 注册节点后，集群可以在不使用 CoreDNS-mDNS 的情况下分散节点信息。这可消除与多播 DNS 关联的网络流量。

重要

置备 IBM Cloud® 节点后，您必须为外部 DNS 上的 api.<cluster_name>.<domain> 域名创建一个 DNS 条目，因为删除 CoreDNS 会导致本地条目消失。未能在外部 DNS 服务器中为 api.<cluster_name>.<domain> 域名创建 DNS 记录会阻止 worker 节点加入集群。

网络时间协议(NTP)

集群中的每个 OpenShift Container Platform 节点都必须有权访问 NTP 服务器。OpenShift Container Platform 节点使用 NTP 来同步其时钟。例如，集群节点使用需要验证的 SSL 证书，如果节点之间的日期和时间未同步，则可能会失败。

重要

在每个群集节点的 BIOS 设置中定义一致的时钟日期和时间格式，或者安装可能会失败。

配置 DHCP 服务器

IBM Cloud® Bare Metal (Classic) 不会在公共或私有 VLAN 上运行 DHCP。在置备 IBM Cloud® 节点后，您必须为公共 VLAN 设置 DHCP 服务器，这与 OpenShift Container Platform 的 baremetal 网络对应。

注意

分配给每个节点的 IP 地址不需要与 IBM Cloud® Bare Metal (Classic) 置备系统分配的 IP 地址匹配。

详情请参阅"配置公共子网"部分。

确保 BMC 访问权限

仪表板上每个节点的"远程管理"页面包含节点的智能平台管理接口(IPMI)凭证。默认 IPMI 特权阻止用户进行某些引导目标更改。您必须将特权级别更改为 OPERATOR，以便 Ironic 能够进行这些更改。

在 install-config.yaml 文件中，将 privilegelevel 参数添加到用于配置每个 BMC 的 URL 中。如需了解更多详细信息，请参阅"配置 install-config.yaml 文件"部分。例如：

```
ipmi://<IP>:<port>?privilegelevel=OPERATOR
```

或者，联系 IBM Cloud® 支持并请求它们将 IPMI 权限增加到 ADMINISTRATOR。

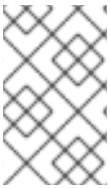
创建裸机服务器

在 IBM Cloud® dashboard 中，进入到 Create resource → Bare Metal Servers for Classic 来创建裸机服务器。

或者，您可以使用 ibmcloud CLI 实用程序创建裸机服务器。例如：

```
$ ibmcloud sl hardware create --hostname <SERVERNAME> \
                      --domain <DOMAIN> \
                      --size <SIZE> \
                      --os <OS-TYPE> \
                      --datacenter <DC-NAME> \
                      --port-speed <SPEED> \
                      --billing <BILLING>
```

有关装 IBM Cloud® CLI 的详情，请参阅安装独立 IBM Cloud® CLI。



注意

IBM Cloud® 服务器可能需要 3-5 小时才能使用。

17.2. 为 OPENSHIFT CONTAINER PLATFORM 安装设置环境

17.2.1. 在 IBM Cloud (R) Bare Metal (Classic)基础架构上准备 provisioner 节点

执行以下步骤准备 provisioner 节点。

流程

1. 通过 ssh 登录到 provisioner 节点。

2. 创建非 root 用户(kni)并为该用户提供 sudo 权限：

```
# useradd kni
```

```
# passwd kni
```

```
# echo "kni ALL=(root) NOPASSWD:ALL" | tee -a /etc/sudoers.d/kni
```

```
# chmod 0440 /etc/sudoers.d/kni
```

3. 为新用户创建 ssh 密钥：

```
# su - kni -c "ssh-keygen -f /home/kni/.ssh/id_rsa -N ""
```

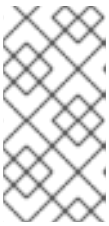
4. 以新用户身份登录到 provisioner 节点：

```
# su - kni
```

5. 使用 Red Hat Subscription Manager 注册 provisioner 节点：

```
$ sudo subscription-manager register --username=<user> --password=<pass> --auto-attach
```

```
$ sudo subscription-manager repos --enable=rhel-8-for-x86_64-appstream-rpms \
                          --enable=rhel-8-for-x86_64-baseos-rpms
```



注意

有关 Red Hat Subscription Manager 的详情，请参考 使用和配置 Red Hat Subscription Manager。

6. 安装以下软件包：

```
$ sudo dnf install -y libvirt qemu-kvm mkisofs python3-devel jq ipmitool
```

7. 修改用户，将 libvirt 组添加到新创建的用户：

```
$ sudo usermod --append --groups libvirt kni
```

8. 启动 **firewalld** ：

```
$ sudo systemctl start firewalld
```

9. 启用 **firewalld** ：

```
$ sudo systemctl enable firewalld
```

10. 启动 **http** 服务：

```
$ sudo firewall-cmd --zone=public --add-service=http --permanent
```

```
$ sudo firewall-cmd --reload
```

11. 启动并启用 **libvirtd** 服务：

```
$ sudo systemctl enable libvirtd --now
```

12. 设置 **provisioner** 节点的 **ID**：

```
$ PRVN_HOST_ID=<ID>
```

您可以使用以下 **ibmcloud** 命令查看 **ID**：

```
$ ibmcloud sl hardware list
```

13. 设置公共子网的 **ID**：

```
$ PUBLICSUBNETID=<ID>
```

您可以使用以下 **ibmcloud** 命令查看 **ID**：

```
$ ibmcloud sl subnet list
```

14. 设置专用子网的 ID：

```
$ PRIVSUBNETID=<ID>
```

您可以使用以下 ibmcloud 命令查看 ID：

```
$ ibmcloud sl subnet list
```

15. 设置 provisioner 节点公共 IP 地址：

```
$ PRVN_PUB_IP=$(ibmcloud sl hardware detail $PRVN_HOST_ID --output JSON | jq
.primaryIpAddress -r)
```

16. 为公共网络设置 CIDR:

```
$ PUBLICCIDR=$(ibmcloud sl subnet detail $PUBLICSUBNETID --output JSON | jq
.cidr)
```

17. 为公共网络设置 IP 地址和 CIDR：

```
$ PUB_IP_CIDR=$PRVN_PUB_IP/$PUBLICCIDR
```

18. 为公共网络设置网关：

```
$ PUB_GATEWAY=$(ibmcloud sl subnet detail $PUBLICSUBNETID --output JSON | jq
.gateway -r)
```

19. 设置 provisioner 节点的专用 IP 地址：

```
$ PRVN_PRIV_IP=$(ibmcloud sl hardware detail $PRVN_HOST_ID --output JSON | \
          jq .primaryBackendIpAddress -r)
```

20. 为专用网络设置 CIDR：

```
$ PRIVCIDR=$(ibmcloud sl subnet detail $PRIVSUBNETID --output JSON | jq .cidr)
```

21. 设置专用网络的 IP 地址和 CIDR：

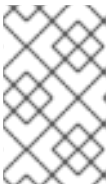
```
$ PRIV_IP_CIDR=$PRVN_PRIV_IP/$PRIVCIDR
```

22. 为专用网络设置网关：

```
$ PRIV_GATEWAY=$(ibmcloud sl subnet detail $PRIVSUBNETID --output JSON | jq
.gateway -r)
```

23. 为 baremetal 设置网桥并 置备 网络：

```
$ sudo nohup bash -c "
    nmcli --get-values UUID con show | xargs -n 1 nmcli con delete
    nmcli connection add ifname provisioning type bridge con-name provisioning
    nmcli con add type bridge-slave ifname eth1 master provisioning
    nmcli connection add ifname baremetal type bridge con-name baremetal
    nmcli con add type bridge-slave ifname eth2 master baremetal
    nmcli connection modify baremetal ipv4.addresses $PUB_IP_CIDR ipv4.method
manual ipv4.gateway $PUB_GATEWAY
    nmcli connection modify provisioning ipv4.addresses 172.22.0.1/24,$PRIV_IP_CIDR
ipv4.method manual
    nmcli connection modify provisioning +ipv4.routes \"10.0.0.0/8 $PRIV_GATEWAY\"
    nmcli con down baremetal
    nmcli con up baremetal
    nmcli con down provisioning
    nmcli con up provisioning
    init 6
"
```



注意

对于 eth1 和 eth2，根据需要替换相应的接口名称。

24. 如果需要，通过 SSH 重新连接到 provisioner 节点：

```
# ssh kni@provisioner.<cluster-name>.<domain>
```

25. 验证连接网桥是否已正确创建：

```
$ sudo nmcli con show
```

输出示例

```
NAME            UUID                                  TYPE      DEVICE
baremetal       4d5133a5-8351-4bb9-bfd4-3af264801530  bridge    baremetal
provisioning    43942805-017f-4d7d-a2c2-7cb3324482ed  bridge    provisioning
virbr0          d9bca40f-eee1-410b-8879-a2d4bb0465e7  bridge    virbr0
bridge-slave-eth1  76a8ed50-c7e5-4999-b4f6-6d9014dd0812  ethernet  eth1
bridge-slave-eth2  f31c3353-54b7-48de-893a-02d2b34c4736  ethernet  eth2
```

26. 创建 pull-secret.txt 文件：

```
$ vim pull-secret.txt
```

在 Web 浏览器中，导航到 Install on Bare Metal with user-provisioned infrastructure。在第 1 步中，点击 Download pull secret。将内容粘贴到 pull-secret.txt 文件中，并将内容保存到 kni 用户的主目录中。

17.2.2. 配置公共子网

所有 OpenShift Container Platform 集群节点都必须位于公共子网中。IBM Cloud® Bare Metal (Classic) 不会在子网上提供 DHCP 服务器。在 provisioner 节点上单独设置它。

您必须重置准备 provisioner 节点时定义的 BASH 变量。在准备完置备程序节点后，重新引导置备程序节点将删除之前设置的 BASH 变量。

流程

1. 安装 dnsmasq:

```
$ sudo dnf install dnsmasq
```

2. 打开 dnsmasq 配置文件：

```
$ sudo vi /etc/dnsmasq.conf
```

3. 在 dnsmasq 配置文件中添加以下配置：

```
interface=baremetal
except-interface=lo
bind-dynamic
log-dhcp

dhcp-range=<ip_addr>,<ip_addr>,<pub_cidr> 1
dhcp-option=baremetal,121,0.0.0.0/0,<pub_gateway>,<prvn_priv_ip>,<prvn_pub_ip>
2

dhcp-hostsfile=/var/lib/dnsmasq/dnsmasq.hostsfile
```

1 设置 DHCP 范围。将 <ip_addr> 实例替换为一个来自公共子网的未使用的 IP 地址，以便 baremetal 网络的 dhcp-range 并以相同的 IP 地址开始和结尾。将 <pub_cidr> 替换为公共子网的 CIDR。

2 设置 DHCP 选项。将 <pub_gateway> 替换为 baremetal 网络的网关 IP 地址。将 <prvn_priv_ip> 替换为 provisioning 网络上 provisioner 节点的专用 IP 地址的 IP 地址。将 <prvn_pub_ip> 替换为 baremetal 网络中 provisioner 节点的公共 IP 地址的 IP 地址。

要检索 <pub_cidr> 的值，请执行：

```
$ ibmcloud sl subnet detail <publicsubnetid> --output JSON | jq .cidr
```

将 <publicsubnetid> 替换为公共子网的 ID。

要检索 <pub_gateway> 的值，请执行：

```
$ ibmcloud sl subnet detail <publicsubnetid> --output JSON | jq .gateway -r
```

将 <publicsubnetid> 替换为公共子网的 ID。

要检索 <prvn_priv_ip> 的值，请执行：

```
$ ibmcloud  sl hardware detail <id> --output JSON | \
        jq .primaryBackendIpAddress -r
```

将 <id> 替换为 provisioner 节点的 ID。

要检索 <prvn_pub_ip> 的值，请执行：

```
$ ibmcloud sl hardware detail <id> --output JSON | jq .primaryIpAddress -r
```

将 <id> 替换为 provisioner 节点的 ID。

4. 获取集群的硬件列表：

```
$ ibmcloud sl hardware list
```

5. 获取每个节点的 MAC 地址和 IP 地址：

```
$ ibmcloud sl hardware detail <id> --output JSON | \
  jq '.networkComponents[] | \
  "\(.primaryIpAddress) \(.macAddress)"' | grep -v null
```

将 <id> 替换为节点的 ID。

输出示例

```
"10.196.130.144 00:e0:ed:6a:ca:b4"
"141.125.65.215 00:e0:ed:6a:ca:b5"
```

记录公共网络的 MAC 地址和 IP 地址。单独记录专用网络的 MAC 地址，稍后您将在 install-config.yaml 文件中使用它。对每个节点重复这个过程，直到您拥有公共 baremetal 网络的所有公共 MAC 和 IP 地址，以及私有 provisioning 网络的 MAC 地址。

6. 将每个节点的 public baremetal 网络的 MAC 和 IP 地址对添加到 dnsmasq.hostsfile 文件

中：

```
$ sudo vim /var/lib/dnsmasq/dnsmasq.hostsfile
```

输入示例

```
00:e0:ed:6a:ca:b5,141.125.65.215,master-0
<mac>,<ip>,master-1
<mac>,<ip>,master-2
<mac>,<ip>,worker-0
<mac>,<ip>,worker-1
...
```

将 <mac>,<ip> 替换为相应节点名称的公共 MAC 地址和公共 IP 地址。

7. 启动 dnsmasq:

```
$ sudo systemctl start dnsmasq
```

8. 启用 dnsmasq，以便在引导节点时启动：

```
$ sudo systemctl enable dnsmasq
```

9. 验证 dnsmasq 是否正在运行：

```
$ sudo systemctl status dnsmasq
```

输出示例

```
● dnsmasq.service - DNS caching server.
Loaded: loaded (/usr/lib/systemd/system/dnsmasq.service; enabled; vendor preset:
disabled)
Active: active (running) since Tue 2021-10-05 05:04:14 CDT; 49s ago
Main PID: 3101 (dnsmasq)
Tasks: 1 (limit: 204038)
```

```
Memory: 732.0K
CGroup: /system.slice/dnsmasq.service
└─3101 /usr/sbin/dnsmasq -k
```

10. 使用 UDP 协议打开端口 53 和 67 ：

```
$ sudo firewall-cmd --add-port 53/udp --permanent
```

```
$ sudo firewall-cmd --add-port 67/udp --permanent
```

11. 使用 masquerade 为外部区添加 置备 ：

```
$ sudo firewall-cmd --change-zone=provisioning --zone=external --permanent
```

这一步可确保将 IPMI 调用的网络地址转换为管理子网。

12. 重新载入 firewalld 配置：

```
$ sudo firewall-cmd --reload
```

17.2.3. 检索 OpenShift Container Platform 安装程序

使用安装程序的 stable-4.x 版本和您选择的架构来部署 OpenShift Container Platform 的一般稳定版本：

```
$ export VERSION=stable-4.16
```

```
$ export RELEASE_ARCH=<architecture>
```

```
$ export RELEASE_IMAGE=$(curl -s https://mirror.openshift.com/pub/openshift-v4/$RELEASE_ARCH/clients/ocp/$VERSION/release.txt | grep 'Pull From: quay.io' | awk -F ' ' '{print $3}')
```

17.2.4. 提取 OpenShift Container Platform 安装程序

在获取安装程序后，下一步是提取它。

流程

1. 设置环境变量：

```
$ export cmd=openshift-baremetal-install
```

```
$ export pullsecret_file=~/pull-secret.txt
```

```
$ export extract_dir=$(pwd)
```

2. 获取 oc 二进制文件：

```
$ curl -s https://mirror.openshift.com/pub/openshift-
v4/clients/ocp/$VERSION/openshift-client-linux.tar.gz | tar zxvf - oc
```

3. 解压安装程序：

```
$ sudo cp oc /usr/local/bin
```

```
$ oc adm release extract --registry-config "${pullsecret_file}" --command=$cmd --to
"${extract_dir}" ${RELEASE_IMAGE}
```

```
$ sudo cp openshift-baremetal-install /usr/local/bin
```

17.2.5. 配置 install-config.yaml 文件

install-config.yaml 文件需要一些额外的详情。大多数信息用于指导安装程序，从而让集群足够了解可用的 IBM Cloud® Bare Metal (Classic) 硬件，以便它可以完全管理它。在裸机上安装和在 IBM Cloud® Bare Metal (Classic)上安装之间的材料区别在于，您必须在 install-config.yaml 文件的 BMC 部分中明确设置 IPMI 的权限级别。

流程

1. 配置 install-config.yaml。更改适当的变量以匹配环境，包括 pullSecret 和 sshKey。

```
apiVersion: v1
baseDomain: <domain>
metadata:
  name: <cluster_name>
networking:
  machineNetwork:
  - cidr: <public-cidr>
```

```
  networkType: OVNKubernetes
compute:
- name: worker
  replicas: 2
controlPlane:
  name: master
  replicas: 3
  platform:
    baremetal: {}
platform:
  baremetal:
    apiVIP: <api_ip>
    ingressVIP: <wildcard_ip>
    provisioningNetworkInterface: <NIC1>
    provisioningNetworkCIDR: <CIDR>
    hosts:
      - name: openshift-master-0
        role: master
        bmc:
          address: ipmi://10.196.130.145?privilegelevel=OPERATOR 1
          username: root
          password: <password>
        bootMACAddress: 00:e0:ed:6a:ca:b4 2
        rootDeviceHints:
          deviceName: "/dev/sda"
      - name: openshift-worker-0
        role: worker
        bmc:
          address: ipmi://<out-of-band-ip>?privilegelevel=OPERATOR 3
          username: <user>
          password: <password>
        bootMACAddress: <NIC1_mac_address> 4
        rootDeviceHints:
          deviceName: "/dev/sda"
pullSecret: '<pull_secret>'
sshKey: '<ssh_pub_key>'
```

1 3

bmc.address 提供了 privilegelevel 配置设置，值设为 OPERATOR。这是 IBM Cloud® Bare Metal (Classic)基础架构所必需的。

2 4

为对应节点添加私有 provisioning 网络 NIC 的 MAC 地址。

注意

您可以使用 ibmcloud 命令行实用程序来检索密码。

```
$ ibmcloud sl hardware detail <id> --output JSON | \
  jq '"(.networkManagementIpAddress)
(.remoteManagementAccounts[0].password)"'
```

将 <id> 替换为节点的 ID。

2. 创建用于存储集群配置的目录：

```
$ mkdir ~/clusterconfigs
```

3. 将 install-config.yaml 文件复制到目录中：

```
$ cp install-config.yaml ~/clusterconfig
```

4. 在安装 OpenShift Container Platform 集群前，请确保关闭所有裸机节点：

```
$ ipmitool -I lanplus -U <user> -P <password> -H <management_server_ip> power off
```

5. 如果以前的部署尝试中保留了旧的 bootstrap 资源，请删除旧的 bootstrap 资源：

```
for i in $(sudo virsh list | tail -n +3 | grep bootstrap | awk {'print $2'});
do
  sudo virsh destroy $i;
  sudo virsh undefine $i;
  sudo virsh vol-delete $i --pool $i;
  sudo virsh vol-delete $i.ign --pool $i;
  sudo virsh pool-destroy $i;
  sudo virsh pool-undefine $i;
done
```

17.2.6. 其他 install-config 参数

下表列出了 install-config.yaml 文件所需的 参数、 hosts 参数和 bmc 参数。

表 17.2. 所需的参数

参数	default	描述
baseDomain		集群的域名。例如： **example.com**。
bootMode	**UEFI**	节点的引导模式。选项为 **legacy**、**U** EFI 和 **UEFISecureBoot**。如果没有设置 **bootMode**，Ironic 在检查节点时设置它。
bootstrapExternalStaticDNS		bootstrap 节点的静态网络 DNS。当裸机网络上没有动态主机配置协议 (DHCP) 服务器时，您必须使用静态 IP 地址部署集群时设置这个值。如果没有设置这个值，安装程序将使用 **bootstrapExternalStaticGateway** 的值，这会导致在网关和 DNS 的 IP 地址值不同时出现问题。
bootstrapExternalStaticIP		bootstrap 虚拟机的静态 IP 地址。当 bare-metal 网络中没有 DHCP 服务器时部署使用静态 IP 地址的集群时，必须设置这个值。
bootstrapExternalStaticGateway		bootstrap 虚拟机网关的静态 IP 地址。当 bare-metal 网络中没有 DHCP 服务器时部署使用静态 IP 地址的集群时，必须设置这个值。
sshKey		**sshKey** 配置设置包含 **~/.ssh/id_rsa.pub** 文件中访问 control plane 节点和计算节点所需的密钥。通常，这个密钥来自 **provisioner** 节点。
pullSecret		**pullSecret** 配置设置包含准备 provisioner 节点时从Install OpenShift on Bare Metal 页面下载的 pull secret 的副本。
metadata: name:		提供给 OpenShift Container Platform 集群的名称。例如： **openshift**。
networking: machineNetwork: - cidr:		外部网络的公共 CIDR(Classless Inter-Domain Routing)。例如： **10.0.0.0/24**。
compute: - name: worker		OpenShift Container Platform 集群需要为计算节点提供一个名称，即使没有节点也是如此。
compute: replicas: 2		replicas 设置 OpenShift Container Platform 集群中的计算节点数量。
controlPlane: name: master		OpenShift Container Platform 集群需要一个 control plane 节点的名称。

参数	default	描述
controlPlane: replicas: 3		replicas 设置作为 OpenShift Container Platform 集群一部分的 control plane 节点数量。
provisioningNetworkInterface		连接到 provisioning 网络的节点上的网络接口名称。对于 OpenShift Container Platform 4.9 及更新的版本，使用 **bootMACAddress** 配置设置来启用 Ironic 标识 NIC 的 IP 地址，**而不使用 provisioningNetworkInterface** 配置设置来标识 NIC 的名称。
defaultMachinePlatform		用于没有平台配置的机器池的默认配置。
apiVIPs		（可选）用于 Kubernetes API 通信的虚拟 IP 地址。 此设置必须在 **install-config.yaml** 文件中作为保留的 IP，或从 DNS 中预先配置到 DNS 中，以便正确解析默认名称。在 **install-config.yaml** 文件中的 **apiVIPs** 配置设置中添加值时，请使用虚拟 IP 地址而不是 FQDN。在使用双栈网络时，主 IP 地址必须来自 IPv4 网络。如果没有设置，安装程序将使用 **api.<cluster_name>.<base_domain>** 从 DNS 派生 IP 地址。  **注意** 在 OpenShift Container Platform 4.12 之前，集群安装程序只接受 **apiVIP** 配置设置的 IPv4 地址或 IPv6 地址。在 OpenShift Container Platform 4.12 或更高版本中，**apiVIP** 配置设置已弃用。反之，使用 **apiVIPs** 配置设置的列表格式来指定 IPv4 地址、IPv6 地址或两个 IP 地址格式。
disableCertificateVerification	**False**	**RedFish** 和 **redfish-virtualmedia** 需要这个参数来管理 BMC 地址。当 BMC 地址使用自签名证书时，这个值应该是 **True**。

参数	default	描述
ingressVIPs		（可选）入口流量的虚拟 IP 地址。 此设置必须在 **install-config.yaml** 文件中作为保留的 IP，或从 DNS 中预先配置到 DNS 中，以便正确解析默认名称。在 **install-config.yaml** 文件中的 **ingressVIPs** 配置设置中添加值时，请使用虚拟 IP 地址而不是 FQDN。在使用双栈网络时，主 IP 地址必须来自 IPv4 网络。如果没有设置，安装程序将使用 **test.apps.<cluster_name>.<base_domain>** 从 DNS 派生 IP 地址。  **注意** 在 OpenShift Container Platform 4.12 之前，集群安装程序只接受 **ingressVIP** 配置设置的 IPv4 地址或 IPv6 地址。在 OpenShift Container Platform 4.12 及更新的版本中，**ingressVIP** 配置设置已弃用。反之，使用 **ingressVIPs** 配置设置的列表格式来指定 IPv4 地址、IPv6 地址或两个 IP 地址格式。

表 17.3. 可选参数

参数	default	描述
provisioningDHCPRange	**172.22.0.10,172.22.0.100**	定义 provisioning 网络上节点的 IP 范围。
provisioningNetworkCIDR	**172.22.0.0/24**	用于置备的网络的 CIDR。在 provisioning 网络中不使用默认地址范围时需要这个选项。
clusterProvisioningIP	**provisioningNetworkCIDR** 的第三个 IP 地址。	运行置备服务的集群中的 IP 地址。默认为 provisioning 子网的第三个 IP 地址。例如： **172.22.0.3**。
bootstrapProvisioningIP	**provisioningNetworkCIDR** 的第二个 IP 地址。	在安装程序部署 control plane(master)节点时运行置备服务的 bootstrap 虚拟机上的 IP 地址。默认为 provisioning 子网的第二个 IP 地址。例如： **172.22.0.2 或 2620:52:0:1307::2**。
externalBridge	**baremetal**	附加到裸机网络的虚拟机监控程序的裸机网桥名称。
provisioningBridge	**provisioning**	附加到 provisioning 网络的 **provisioner** 主机上的 provisioning 网桥的名称。
架构		定义集群的主机架构。有效值为 **amd64** 或 **arm64**。
defaultMachinePlatform		用于没有平台配置的机器池的默认配置。

参数	default	描述
bootstrapOSImage		用于覆盖 bootstrap 节点的默认操作系统镜像的 URL。URL 必须包含镜像的 SHA-256 哈希。例如：**https://mirror.openshift.com/rhcos-<version>-qemu.qcow2.gz?sha256=<uncompressed_sha256>**。
provisioningNetwork		**provisioningNetwork** 配置设置决定了集群是否使用 provisioning 网络。如果存在，配置设置也会决定集群是否管理网络。 **disabled** ：将此参数设置为 **Disabled**，以禁用对 provisioning 网络的要求。当设置为 **Disabled** 时，您必须只使用基于虚拟介质的置备，或使用支持的安装程序启动集群。如果 **Disabled** 并使用电源管理，则必须可以从 bare-metal 网络访问 BMC。如果 **Disabled**，则必须在 bare-metal 网络上提供两个用于置备服务的 IP 地址。 **Managed** ：将此参数设置为 **Managed**（默认设置），以全面管理调配网络，包括 DHCP、TFTP 等。 **Unmanaged** ：将此参数 **设置为 Unmanaged** 以启用调配网络，但需要手动配置 DHCP。建议进行虚拟介质调配，但在需要时仍可使用 PXE。
httpProxy		将此参数设置为环境中使用的适当 HTTP 代理。
httpsProxy		将此参数设置为环境中使用的适当 HTTPS 代理。
noProxy		将此参数设置为适合环境中代理使用的排除项。

主机

hosts 参数是用于构建集群的独立裸机资产列表。

表 17.4. 主机

名称	default	描述
名称		与详细信息 **关联的 BareMetalHost** 资源的名称。例如： **openshift-master-0**。
role		裸机节点的角色。**master** (control plane 节点) 或 **worker** （计算节点）。
bmc		基板管理控制器的连接详情。如需了解更多详细信息，请参阅 BMC 寻址部分。

名称	default	描述
bootMACAddress		主机用于 provisioning 网络的 NIC 的 MAC 地址。Ironic 使用 **bootMACAddress** 配置设置检索 IP 地址。然后，它将绑定到 主机。  **注意** 如果您禁用了 provisioning 网络，则必须从主机提供有效的 MAC 地址。
networkConfig		设置此可选参数来配置主机的网络接口。如需了解更多详细信息，请参阅"（可选）配置主机网络接口"。

17.2.7. Root 设备提示

rootDeviceHints 参数可让安装程序将 Red Hat Enterprise Linux CoreOS(RHCOS)镜像置备到特定的设备。安装程序会按照发现设备的顺序检查设备，并将发现的值与 hint 值进行比较。安装程序使用第一个与 hint 值匹配的发现设备。配置可以组合多个 hint，但设备必须与所有提示匹配，以便安装程序进行选择。

表 17.5. 子字段

子字段	描述
deviceName	包含 Linux 设备名称的字符串（如 **/dev/vda** 或 **/dev/disk/by-path/** ）。建议您使用 **/dev/disk/by-path/<device_path>** 链接到存储位置。hint 必须与实际值完全匹配。
hctl	包含类似 **0:0:0:0:0 的 SCSI 总线地址的字符串**。hint 必须与实际值完全匹配。
model	包含特定厂商的设备标识符的字符串。hint 可以是实际值的子字符串。
vendor	包含该设备厂商或制造商名称的字符串。hint 可以是实际值的子字符串。
serialNumber	包含设备序列号的字符串。hint 必须与实际值完全匹配。
minSizeGigabytes	以 GB 为单位代表设备的最小大小的整数。
wwn	包含唯一存储标识符的字符串。hint 必须与实际值完全匹配。

子字段	描述
wwnWithExtension	包含唯一存储标识符的字符串，并附加厂商扩展。hint 必须与实际值完全匹配。
wwnVendorExtension	包含唯一厂商存储标识符的字符串。hint 必须与实际值完全匹配。
rotational	指明该设备为旋转磁盘(true)还是非旋转磁盘(false)的布尔值。

示例用法

```
- name: master-0
  role: master
  bmc:
    address: ipmi://10.10.0.3:6203
    username: admin
    password: redhat
  bootMACAddress: de:ad:be:ef:00:40
  rootDeviceHints:
    deviceName: "/dev/sda"
```

17.2.8. 创建 OpenShift Container Platform 清单

1. 创建 OpenShift Container Platform 清单。

```
$ ./openshift-baremetal-install --dir ~/clusterconfigs create manifests
```

```
INFO Consuming Install Config from target directory
WARNING Making control-plane schedulable by setting MastersSchedulable to true for Scheduler cluster settings
WARNING Discarding the OpenShift Manifest that was provided in the target directory because its dependencies are dirty and it needs to be regenerated
```

17.2.9. 通过 OpenShift Container Platform 安装程序部署集群

运行 OpenShift Container Platform 安装程序：

```
$ ./openshift-baremetal-install --dir ~/clusterconfigs --log-level debug create cluster
```

-

17.2.10. 安装后

在部署过程中，您可以通过向安装目录文件夹中的 .openshift_install.log 日志文件发出 tail 命令来检查安装的整体状态：

```
$ tail -f /path/to/install-dir/.openshift_install.log
```

第 18 章 在 IBM Z 和 IBM LINUXONE 上安装

18.1. 准备在 IBM Z 和 IBM LINUXONE 上安装

18.1.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读有关 选择集群安装方法的文档，并为用户准备它。
- 在开始安装过程前，您必须清理安装目录。这样可确保在安装过程中创建和更新所需的安装文件。
- 已为集群置备了 使用 OpenShift Data Foundation 或其他支持的存储协议的持久性存储。要部署私有镜像 registry，您必须使用 ReadWriteMany 访问设置持久性存储。
- 如果使用防火墙，将其配置为允许集群需要访问的站点。

注意

虽然本文档只涉及 IBM Z®，但它的所有信息也适用于 IBM® LinuxONE。

18.1.2. 选择在 IBM Z 或 IBM LinuxONE 上安装 OpenShift Container Platform 的方法

OpenShift Container Platform 安装程序提供在 IBM Z® 上部署集群的方法：

- 交互式 ：您可以使用基于 Web 的 辅助安装程序（Assisted Installer）部署集群。这个方法不需要安装程序的设置，对于 IBM Z® 等连接的环境来说是理想的选择。
- 本地基于代理的 ：您可以使用基于代理的安装程序在本地部署集群。它提供了 Assisted Installer 的许多优点，但您必须首先下载并配置基于代理的安装程序。使用命令行界面(CLI)进行配置。这种方法非常适合断开连接的网络。
- 完全控制：您可以在自己准备和维护的基础架构上部署集群，这种方法提供了最大的定制

性。您可以在有连接或断开连接的环境中部署集群。

表 18.1. IBM Z (R)安装选项

	支持的安装程序	基于代理的安装程序	用户置备的安装	安装程序置备的安装
带有 z/VM 的 IBM Z®	✓	✓	✓	
使用 z/VM 的受限网络 IBM Z®		✓	✓	
使用 RHEL KVM 的 IBM Z®	✓	✓	✓	
使用 RHEL KVM 的受限网络 IBM Z®		✓	✓	
LPAR 中的 IBM Z®			✓	
LPAR 中的受限网络 IBM Z®			✓	

有关安装过程的更多信息，请参阅 安装过程。

18.1.2.1. 在 IBM Z 上安装 OpenShift Container Platform 用户置备的基础架构

用户置备的基础架构要求用户置备 OpenShift Container Platform 所需的所有资源。

重要

执行用户置备的基础架构安装的步骤仅作为示例。使用您提供的基础架构安装集群需要了解 IBM Z® 平台和 OpenShift Container Platform 的安装过程。使用用户置备的基础架构安装说明作为指南；您可以通过其他方法创建所需的资源。

- 在 IBM Z® 和 IBM® LinuxONE 中使用 z/VM 安装集群 ：您可以在您置备的 IBM Z® 或 IBM® LinuxONE 基础架构中使用 z/VM 安装 OpenShift Container Platform。
- 在受限网络中的 IBM Z® 和 IBM® LinuxONE 上使用 z/VM 安装集群 ：您可以使用安装发行内容的内部镜像在 IBM Z® 或 IBM® LinuxONE 基础架构中使用 z/VM 安装 OpenShift Container Platform。您可以使用此方法安装不需要活跃互联网连接的集群来获取软件组件。您还可以使用此安装方法来确保集群只使用满足您组织对外部内容控制的容器镜像。
-

在 IBM Z® 和 IBM® LinuxONE 上使用 RHEL KVM 安装集群：您可以在您置备的 IBM Z® 或 IBM® LinuxONE 基础架构中使用 KVM 安装 OpenShift Container Platform。

- 在受限网络中的 IBM Z® 和 IBM® LinuxONE 上使用 RHEL KVM 安装集群：您可以使用 RHEL KVM 在受限或断开连接的网络中置备的 IBM Z® 或 IBM® LinuxONE 基础架构上安装 OpenShift Container Platform。您可以使用此方法安装不需要活跃互联网连接的集群来获取软件组件。您还可以使用此安装方法来确保集群只使用满足您组织对外部内容控制的容器镜像。

- 在 IBM Z® 和 IBM® LinuxONE 上的 LPAR 上安装集群 ：您可以在您置备的 IBM Z® 或 IBM® LinuxONE 基础架构中的逻辑分区(LPAR)中安装 OpenShift Container Platform。

- 在受限网络中的 IBM Z® 和 IBM® LinuxONE 上的 LPAR 安装集群：您可以使用 LPAR 在受限或断开连接的网络中置备的 IBM Z® 或 IBM® LinuxONE 基础架构上安装 OpenShift Container Platform。您可以使用此方法安装不需要活跃互联网连接的集群来获取软件组件。您还可以使用此安装方法来确保集群只使用满足您组织对外部内容控制的容器镜像。

18.2. 在 IBM Z 和 IBM LINUXONE 中使用 Z/VM 安装集群

在 OpenShift Container Platform 版本 4.16 中，您可以在您置备的 IBM Z® 或 IBM® LinuxONE 系统上安装集群。

注意

虽然本文档只涉及 IBM Z®，但它的所有信息也适用于 IBM® LinuxONE。

重要

非裸机平台还有其他注意事项。在安装 OpenShift Container Platform 集群前，请参阅有关在未经测试的平台上部署 OpenShift Container Platform 的指南 中的信息。

18.2.1. 先决条件

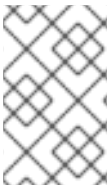
- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。

- 您可以阅读有关 选择集群安装方法的文档，并为用户准备它。

-

在开始安装过程前，您必须清理安装目录。这样可确保在安装过程中创建和更新所需的安装文件。

- 已为集群置备了 使用 OpenShift Data Foundation 或其他支持的存储协议的持久性存储。要部署私有镜像 registry，您必须使用 ReadWriteMany 访问设置持久性存储。
- 如果使用防火墙，则会 将其配置为允许集群需要访问的站点。



注意

如果要配置代理，请务必查看此站点列表。

18.2.2. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类型的基础架构上执行受限网络安装。在此过程中，您可以下载所需的内容，并使用它为镜像 registry 填充安装软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群前，您要更新镜像 registry 的内容。

18.2.3. 具有用户置备基础架构的集群的要求

对于包含用户置备的基础架构的集群，您必须部署所有所需的机器。

本节论述了在用户置备的基础架构上部署 OpenShift Container Platform 的要求。

18.2.3.1. 集群安装所需的机器

最小的 OpenShift Container Platform 集群需要以下主机：

表 18.2. 最低所需的主机

主机	描述
一个临时 bootstrap 机器	集群需要 bootstrap 机器在三台 control plane 机器上部署 OpenShift Container Platform 集群。您可在安装集群后删除 bootstrap 机器。
三台 control plane 机器	control plane 机器运行组成 control plane 的 Kubernetes 和 OpenShift Container Platform 服务。
至少两台计算机器，也称为 worker 机器。	OpenShift Container Platform 用户请求的工作负载在计算机器上运行。

重要

要提高集群的高可用性，请在至少两台物理机器的不同 z/VM 实例中分发 control plane 机器。

bootstrap、control plane 和计算机器必须使用 Red Hat Enterprise Linux CoreOS(RHCOS)作为操作系统。

请注意，RHCOS 基于 Red Hat Enterprise Linux(RHEL) 9.2，并继承其所有硬件认证和要求。查看红帽企业 Linux 技术功能和限制。

18.2.3.2. 集群安装的最低资源要求

每台集群机器都必须满足以下最低要求：

表 18.3. 最低资源要求

机器	操作系统	vCPU [1]	虚拟内存	Storage	每秒输入/输出(IOPS)
bootstrap	RHCOS	4	16 GB	100 GB	N/A
Control plane（控制平面）	RHCOS	4	16 GB	100 GB	N/A
Compute	RHCOS	2	8 GB	100 GB	N/A

1. 当启用 SMT-2 时，一个物理核心(IFL)提供两个逻辑核心（线程）。管理程序可以提供两个或多个 vCPU。

注意

从 OpenShift Container Platform 版本 4.13 开始，RHCOS 基于 RHEL 版本 9.2，它更新了微架构要求。以下列表包含每个架构需要的最小指令集架构 (ISA)：

- x86-64 体系结构需要 x86-64-v2 ISA
- ARM64 架构需要 ARMv8.0-A ISA
- IBM Power 架构需要 Power 9 ISA
- s390x 架构需要 z14 ISA

如需更多信息，请参阅 RHEL 架构。

如果平台的实例类型满足集群机器的最低要求，则 OpenShift Container Platform 支持使用它。

其他资源

- 优化存储

18.2.3.3. 最低 IBM Z 系统环境

您可以在以下 IBM® 硬件上安装 OpenShift Container Platform 版本 4.16：

- IBM® z16 （所有型号）、IBM® z15 （所有型号）、IBM® z14 （所有型号）
- IBM® LinuxONE 4 （所有型号）、IBM® LinuxONE （所有型号）、IBM® LinuxONE Emperor II、IBM® LinuxONE Rockhopper II

硬件要求

- Linux(IFL)等效的、启用了 SMT2 的 Linux 集成设施，每个群集都启用了 SMT2。
- 至少一个网络连接连接到 LoadBalancer 服务，并为集群外的流量提供数据。

注意

您可以使用专用或共享的 IFL 来分配足够的计算资源。资源共享是 IBM Z® 的关键优势之一。但是，您必须正确调整每个虚拟机监控程序层上的容量，并确保每个 OpenShift Container Platform 集群都有足够资源。

重要

由于集群的整体性能会受到影响，用于设置 OpenShift Container Platform 集群的 LPAR 必须提供足够的计算容量。就此而言，管理程序级别上的 LPAR 权重管理、授权和 CPU 共享扮演着重要角色。

操作系统要求

- 一个 z/VM 7.2 或更高版本实例

在 z/VM 实例中设置：

- 用于 OpenShift Container Platform control plane 机器的三台客户机虚拟机

- 用于 OpenShift Container Platform 计算机器的两个客户机虚拟机

- 一个客户虚拟机作为临时 OpenShift Container Platform bootstrap 机器

IBM Z 网络连接要求

要在 z/VM 中安装 IBM Z®，您需要使用第 2 层模式的单一 z/VM 虚拟 NIC。您还需要：

- 直接附加的 OSA 或 RoCE 网络适配器

- z/VM VSwitch 设置。对于首选设置，请使用 OSA 链接聚合。

磁盘存储

- 附加了 FICON 的磁盘存储(DASD)。可以是 z/VM Minidisks、fullpack Minidisks 或专用 DASD，它们都必须格式化为 CDL（默认 CDL）。要达到 Red Hat Enterprise Linux CoreOS(RHCOS)安装所需的最小 DASD 大小，您需要扩展地址卷(EAV)。如果可用，请使用 HyperPAV 来确保最佳性能。

- FCP 连接的磁盘存储

存储/主内存

- OpenShift Container Platform control plane 机器需要 16 GB

- OpenShift Container Platform 计算机器需要 8 GB

- 临时 OpenShift Container Platform bootstrap 机器需要 16 GB

18.2.3.4. 首选 IBM Z 系统环境

硬件要求

- 三个 LPARS，每个都相当于 6 个 IFL（每个集群启用了 SMT2）。

- 两个网络连接连接到 LoadBalancer 服务，并为集群外的流量提供数据。

- HiperSockets 作为设备直接附加到节点，或者通过使用一个 z/VM VSWITCH 进行桥接来对 z/VM 客户机进行透明连接。要将 HiperSockets 直接连接到节点，您必须通过 RHEL 8 客户机设置到外部网络的网关来桥接到 HiperSockets 网络。

操作系统要求

- 两个或多个 z/VM 7.2 或更高版本实例以实现高可用性

在 z/VM 实例中设置：

- 用于 OpenShift Container Platform control plane 机器的三台客户虚拟机，每个 z/VM 实例一个。

- 至少 6 个用于 OpenShift Container Platform 计算机器的虚拟机，分布在 z/VM 实例中。

- 一个客户虚拟机作为临时 OpenShift Container Platform bootstrap 机器。

- 要确保在过量使用的环境中集成组件可用，使用 CP 命令 SET SHARE 增加 control plane 的优先级。如果存在基础架构节点，则对它们执行相同的操作。请参阅 IBM® 文档中的 SET SHARE。

IBM Z 网络连接要求

要在 z/VM 中安装 IBM Z®，您需要使用第 2 层模式的单一 z/VM 虚拟 NIC。您还需要：

- 直接附加的 OSA 或 RoCE 网络适配器

- z/VM VSwitch 设置。对于首选设置，请使用 OSA 链接聚合。

磁盘存储

- 附加了 FICON 的磁盘存储(DASD)。可以是 z/VM Minidisks、fullpack Minidisks 或专用 DASD，它们都必须格式化为 CDL（默认 CDL）。要达到 Red Hat Enterprise Linux

CoreOS(RHCOS)安装所需的最小 DASD 大小，您需要扩展地址卷(EAV)。如果可用，请使用 HyperPAV 来确保最佳性能。

- FCP 连接的磁盘存储

存储/主内存

- OpenShift Container Platform control plane 机器需要 16 GB
- OpenShift Container Platform 计算机器需要 8 GB
- 临时 OpenShift Container Platform bootstrap 机器需要 16 GB

18.2.3.5. 证书签名请求管理

在使用您置备的基础架构时，集群只能有限地访问自动机器管理，因此您必须提供一种在安装后批准集群证书签名请求 (CSR) 的机制。kube-controller-manager 只能批准 kubelet 客户端 CSR。machine-approver 无法保证使用 kubelet 凭证请求的提供证书的有效性，因为它不能确认是正确的机器发出了该请求。您必须决定并实施一种方法，以验证 kubelet 提供证书请求的有效性并进行批准。

其他资源

- 请参阅 IBM® 文档中的使用 z/VM 虚拟交换机桥接 HiperSockets LAN。
- 请参阅在 z/VM 上的 Linux 客户端上扩展 HyperPAV 别名设备 以获得性能优化。
- 有关 LPAR 权重管理和权利的信息，请参阅 LPAR 性能的主题。
- IBM Z® 和 IBM® LinuxONE 环境的推荐主机实践

18.2.3.6. 用户置备的基础架构对网络的要求

所有 Red Hat Enterprise Linux CoreOS(RHCOS)机器都需要在启动时在 initramfs 中配置联网，以获取它们的 Ignition 配置文件。

在初次启动过程中，机器需要 HTTP 或 HTTPS 服务器建立网络连接，以下载其 Ignition 配置文件。

机器配置有静态 IP 地址。不需要 DHCP 服务器。确保机器具有持久的 IP 地址和主机名。

Kubernetes API 服务器必须能够解析集群机器的节点名称。如果 API 服务器和 worker 节点位于不同的区域中，您可以配置默认 DNS 搜索区域，以允许 API 服务器解析节点名称。另一种支持的方法是始终通过节点对象和所有 DNS 请求中的完全限定域名引用主机。

18.2.3.6.1. 网络连接要求

您必须配置机器之间的网络连接，以允许 OpenShift Container Platform 集群组件进行通信。每台机器都必须能够解析集群中所有其他机器的主机名。

本节详细介绍了所需的端口。

重要

在连接的 OpenShift Container Platform 环境中，所有节点都需要访问互联网才能为平台容器拉取镜像，并向红帽提供遥测数据。

表 18.4. 用于全机器到所有机器通信的端口

协议	port	描述
ICMP	N/A	网络可访问性测试
TCP	**1936**	指标
	9000-9999	主机级别的服务，包括端口 9**100-9101** 上的节点导出器，以及端口 **9099** 上的 Cluster Version Operator。
	10250-10259	Kubernetes 保留的默认端口
UDP	**4789**	VXLAN
	6081	Geneve
	9000-9999	主机级别的服务，包括端口 **9100**到**9101** 上的节点导出器。
	500	IPsec IKE 数据包

协议	port	描述
	4500	IPsec NAT-T 数据包
	123	UDP 端口 **123** 上的网络时间协议 (NTP) 如果配置了外部 NTP 时间服务器，需要打开 UDP 端口 **123**。
TCP/UDP	**30000-32767**	Kubernetes 节点端口
ESP	N/A	IPsec Encapsulating Security Payload(ESP)

表 18.5. 用于所有机器控制平面通信的端口

协议	port	描述
TCP	**6443**	Kubernetes API

表 18.6. control plane 机器用于 control plane 机器通信的端口

协议	port	描述
TCP	**2379-2380**	etcd 服务器和对等端口

用户置备的基础架构的 NTP 配置

OpenShift Container Platform 集群被配置为默认使用公共网络时间协议(NTP)服务器。如果要使用本地企业 NTP 服务器，或者集群部署在断开连接的网络中，您可以将集群配置为使用特定的时间服务器。如需更多信息，*请参阅配置 chrony 时间服务* 的文档。

其他资源

- **配置 chrony 时间服务**

18.2.3.7. 用户置备的 DNS 要求

在 OpenShift Container Platform 部署中，以下组件需要 DNS 名称解析：

- **The Kubernetes API**

- OpenShift Container Platform 应用程序通配符
- bootstrap、control plane 和计算机器

Kubernetes API、bootstrap 机器、control plane 机器和计算机器也需要反向 DNS 解析。

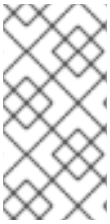
DNS A/AAAA 或 CNAME 记录用于名称解析，PTR 记录用于反向名称解析。反向记录很重要，因为 Red Hat Enterprise Linux CoreOS(RHCOS)使用反向记录为所有节点设置主机名，除非 DHCP 提供主机名。另外，反向记录用于生成 OpenShift Container Platform 需要操作的证书签名请求(CSR)。

用户置备的 OpenShift Container Platform 集群需要以下 DNS 记录，这些记录必须在安装前就位。在每个记录中，<cluster_name> 是集群名称，<base_domain> 是您在 install-config.yaml 文件中指定的基域。完整的 DNS 记录采用以下形式： <component>.<cluster_name>.<base_domain>.。

表 18.7. 所需的 DNS 记录

组件	记录	描述
Kubernetes API	**api.<cluster_name>.<base_domain>.**	DNS A/AAAA 或 CNAME 记录，以及用于标识 API 负载均衡器的 DNS PTR 记录。这些记录必须由集群外的客户端和集群中的所有节点解析。
	api-int.<cluster_name>.<base_domain>.	DNS A/AAAA 或 CNAME 记录，以及用于内部标识 API 负载均衡器的 DNS PTR 记录。这些记录必须可以从集群中的所有节点解析。  **重要** API 服务器必须能够根据 Kubernetes 中记录的主机名解析 worker 节点。如果 API 服务器无法解析节点名称，则代理的 API 调用会失败，且您无法从 pod 检索日志。
Routes	***.apps.<cluster_name>.<base_domain>.**	通配符 DNS A/AAAA 或 CNAME 记录，指向应用程序入口负载均衡器。应用程序入口负载均衡器以运行 Ingress Controller Pod 的机器为目标。默认情况下，Ingress Controller Pod 在计算机器上运行。这些记录必须由集群外的客户端和集群中的所有节点解析。 例如，console **-openshift-console.apps.<cluster_name>.<base_domain>** 用作到 OpenShift Container Platform 控制台的通配符路由。

组件	记录	描述
bootstrap 机器	**bootstrap.<cluster_name>.<base_domain>.**	DNS A/AAAA 或 CNAME 记录，以及用于标识 bootstrap 机器的 DNS PTR 记录。这些记录必须由集群中的节点解析。
control plane 机器	**<control_plane><n>.<cluster_name>.<base_domain>.**	DNS A/AAAA 或 CNAME 记录，以识别 control plane 节点的每台机器。这些记录必须由集群中的节点解析。
计算机器	**<compute><n>.<cluster_name>.<base_domain>.**	DNS A/AAAA 或 CNAME 记录，用于识别 worker 节点的每台机器。这些记录必须由集群中的节点解析。



注意

在 OpenShift Container Platform 4.4 及更新的版本中，您不需要在 DNS 配置中指定 etcd 主机和 SRV 记录。

提示

您可以使用 dig 命令验证名称和反向名称解析。如需了解详细的 *验证步骤，请参阅为用户置备的基础架构验证 DNS 解析* 一节。

18.2.3.7.1. 用户置备的集群的 DNS 配置示例

本节提供 A 和 PTR 记录配置示例，它们满足了在用户置备的基础架构上部署 OpenShift Container Platform 的 DNS 要求。样本不是为选择一个 DNS 解决方案提供建议。

在这个示例中，集群名称为 ocp4，基域是 example.com。

用户置备的集群的 DNS A 记录配置示例

以下示例是 BIND 区域文件，其中显示了用户置备的集群中名称解析的 A 记录示例。

例 18.1. DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
  2019070700 ; serial
  3H  ; refresh (3 hours)
  30M  ; retry (30 minutes)
```

```
  2W  ; expiry (2 weeks)
  1W ) ; minimum (1 week)
 IN NS ns1.example.com.
 IN MX 10 smtp.example.com.
;
;
ns1.example.com.  IN A 192.168.1.5
smtp.example.com.  IN A 192.168.1.5
;
helper.example.com.  IN A 192.168.1.5
helper.ocp4.example.com. IN A 192.168.1.5
;
api.ocp4.example.com.  IN A 192.168.1.5 1
api-int.ocp4.example.com. IN A 192.168.1.5 2
;
*.apps.ocp4.example.com. IN A 192.168.1.5 3
;
bootstrap.ocp4.example.com. IN A 192.168.1.96 4
;
control-plane0.ocp4.example.com. IN A 192.168.1.97 5
control-plane1.ocp4.example.com. IN A 192.168.1.98 6
control-plane2.ocp4.example.com. IN A 192.168.1.99 7
;
compute0.ocp4.example.com. IN A 192.168.1.11 8
compute1.ocp4.example.com. IN A 192.168.1.7 9
;
;EOF
```

1

为 Kubernetes API 提供名称解析。记录引用 API 负载均衡器的 IP 地址。

2

为 Kubernetes API 提供名称解析。记录引用 API 负载均衡器的 IP 地址，用于内部集群通信。

3

为通配符路由提供名称解析。记录引用应用程序入口负载均衡器的 IP 地址。应用程序入口负载均衡器以运行 Ingress Controller Pod 的机器为目标。默认情况下，Ingress Controller Pod 在计算机器上运行。

注意

在这个示例中，将相同的负载均衡器用于 Kubernetes API 和应用入口流量。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。

4

为 bootstrap 机器提供名称解析。

5 6 7

为 control plane 机器提供名称解析。

8 9

为计算机器提供名称解析。

用户置备的集群的 DNS PTR 记录配置示例

以下示例 BIND 区域文件显示了用户置备的集群中反向名称解析的 PTR 记录示例。

例 18.2. 反向记录的 DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
  2019070700 ; serial
  3H  ; refresh (3 hours)
  30M  ; retry (30 minutes)
  2W  ; expiry (2 weeks)
  1W )  ; minimum (1 week)
 IN NS ns1.example.com.
;
5.1.168.192.in-addr.arpa. IN PTR api.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. IN PTR api-int.ocp4.example.com. 2
;
96.1.168.192.in-addr.arpa. IN PTR bootstrap.ocp4.example.com. 3
;
97.1.168.192.in-addr.arpa. IN PTR control-plane0.ocp4.example.com. 4
98.1.168.192.in-addr.arpa. IN PTR control-plane1.ocp4.example.com. 5
99.1.168.192.in-addr.arpa. IN PTR control-plane2.ocp4.example.com. 6
;
11.1.168.192.in-addr.arpa. IN PTR compute0.ocp4.example.com. 7
7.1.168.192.in-addr.arpa. IN PTR compute1.ocp4.example.com. 8
;
;EOF
```

1

为 Kubernetes API 提供反向 DNS 解析。PTR 记录引用 API 负载均衡器的记录名称。

2

3

为 bootstrap 机器提供反向 DNS 解析。

4 5 6

为 control plane 机器提供反向 DNS 解析。

7 8

为计算机器提供反向 DNS 解析。

注意

OpenShift Container Platform 应用程序通配符不需要 PTR 记录。

18.2.3.8. 用户置备的基础架构的负载均衡要求

在安装 OpenShift Container Platform 前，您必须置备 API 和应用程序入口负载均衡基础架构。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。

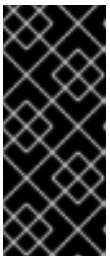
注意

如果要使用 Red Hat Enterprise Linux (RHEL) 实例部署 API 和应用程序入口负载均衡器，您必须单独购买 RHEL 订阅。

负载平衡基础架构必须满足以下要求：

1. API 负载均衡器 ：提供一个通用端点，供用户（包括人工和机器）与平台交互和配置。配置以下条件：

   - 仅第 4 层负载均衡.这可被称为 Raw TCP 或 SSL Passthrough 模式。
   - 无状态负载平衡算法。这些选项根据负载均衡器的实施而有所不同。



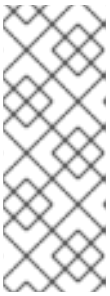
重要

不要为 **API** 负载均衡器配置会话持久性。为 **Kubernetes API** 服务器配置会话持久性可能会导致出现过量 **OpenShift Container Platform** 集群应用程序流量，以及过量的在集群中运行的 **Kubernetes API**。

在负载均衡器的前端和后端配置以下端口：

表 18.8. API 负载均衡器

port	后端机器（池成员）	internal	外部	描述
6443	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。您必须为 API 服务器健康检查探测配置 **/readyz** 端点。	X	X	Kubernetes API 服务器
22623	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。	X		机器配置服务器



注意

负载均衡器必须配置为，从 **API** 服务器关闭 **/readyz** 端点到从池中移除 **API** 服务器实例时最多需要 **30** 秒。在 **/readyz** 返回错误或健康后的时间范围内，端点必须被删除或添加。每 **5** 秒或 **10** 秒探测一次，有两个成功请求处于健康状态，三个成为不健康的请求是经过良好测试的值。

2. 应用程序入口负载均衡器 ：为应用程序流量从集群外部流提供入口点。**OpenShift Container Platform** 集群需要正确配置入口路由器。

配置以下条件：

- 仅第 **4** 层负载均衡.这可被称为 **Raw TCP** 或 **SSL Passthrough** 模式。
- 建议根据可用选项以及平台上托管的应用程序类型，使用基于连接的或基于会话的持久性。

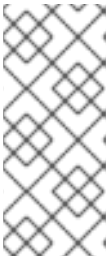
提示

如果应用程序入口负载均衡器可以看到客户端的真实 IP 地址，启用基于 IP 的会话持久性可以提高使用端到端 TLS 加密的应用程序的性能。

在负载均衡器的前端和后端配置以下端口：

表 18.9. 应用程序入口负载均衡器

port	后端机器（池成员）	internal	外部	描述
443	默认情况下，运行 Ingress Controller Pod、计算或 worker 的机器。	X	X	HTTPS 流量
80	默认情况下，运行 Ingress Controller Pod、计算或 worker 的机器。	X	X	HTTP 流量



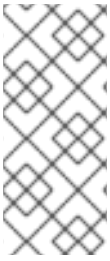
注意

如果要部署一个带有零计算节点的三节点集群，Ingress Controller Pod 在 control plane 节点上运行。在三节点集群部署中，您必须配置应用程序入口负载均衡器，将 HTTP 和 HTTPS 流量路由到 control plane 节点。

18.2.3.8.1. 用户置备的集群的负载均衡器配置示例

本节提供了一个满足用户置备集群的负载均衡要求的 API 和应用程序入口负载均衡器配置示例。示例是 HAProxy 负载均衡器的 /etc/haproxy/haproxy.cfg 配置。这个示例不是为选择一个负载平衡解决方案提供建议。

在这个示例中，将相同的负载均衡器用于 Kubernetes API 和应用入口流量。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。



注意

如果您使用 HAProxy 作为负载均衡器，并且 SELinux 设置为 enforcing，您必须通过运行 setsebool -P haproxy_connect_any=1 来确保 HAProxy 服务可以绑定到配置的 TCP 端口。

例 18.3. API 和应用程序入口负载均衡器配置示例

```
global
  log         127.0.0.1 local2
  pidfile     /var/run/haproxy.pid
  maxconn     4000
  daemon
defaults
  mode                    http
  log                     global
  option                  dontlognull
  option http-server-close
  option                  redispatch
  retries                 3
  timeout http-request    10s
  timeout queue           1m
  timeout connect         10s
  timeout client          1m
  timeout server          1m
  timeout http-keep-alive 10s
  timeout check           10s
  maxconn                 3000
listen api-server-6443 (1)
  bind *:6443
  mode tcp
  option  httpchk GET /readyz HTTP/1.0
  option  log-health-checks
  balance roundrobin
  server bootstrap bootstrap.ocp4.example.com:6443 verify none check check-ssl inter 10s
fall 2 rise 3 backup (2)
  server master0 master0.ocp4.example.com:6443 weight 1 verify none check check-ssl
inter 10s fall 2 rise 3
  server master1 master1.ocp4.example.com:6443 weight 1 verify none check check-ssl
inter 10s fall 2 rise 3
  server master2 master2.ocp4.example.com:6443 weight 1 verify none check check-ssl
inter 10s fall 2 rise 3
listen machine-config-server-22623 (3)
  bind *:22623
  mode tcp
  server bootstrap bootstrap.ocp4.example.com:22623 check inter 1s backup (4)
  server master0 master0.ocp4.example.com:22623 check inter 1s
  server master1 master1.ocp4.example.com:22623 check inter 1s
  server master2 master2.ocp4.example.com:22623 check inter 1s
listen ingress-router-443 (5)
  bind *:443
  mode tcp
  balance source
  server compute0 compute0.ocp4.example.com:443 check inter 1s
  server compute1 compute1.ocp4.example.com:443 check inter 1s
listen ingress-router-80 (6)
  bind *:80
  mode tcp
  balance source
  server compute0 compute0.ocp4.example.com:80 check inter 1s
  server compute1 compute1.ocp4.example.com:80 check inter 1s
```

(1)

2 4 bootstrap 条目必须在 OpenShift Container Platform 集群安装前就位，且必须在 bootstrap 过程完成后删除它们。

3 端口 22623 处理机器配置服务器流量并指向 control plane 机器。

5 端口 443 处理 HTTPS 流量，并指向运行 Ingress Controller pod 的机器。默认情况下，Ingress Controller Pod 在计算机器上运行。

6 端口 80 处理 HTTP 流量，并指向运行 Ingress Controller pod 的机器。默认情况下，Ingress Controller Pod 在计算机器上运行。

注意

如果要部署一个带有零计算节点的三节点集群，Ingress Controller Pod 在 control plane 节点上运行。在三节点集群部署中，您必须配置应用程序入口负载均衡器，将 HTTP 和 HTTPS 流量路由到 control plane 节点。

提示

如果您使用 HAProxy 作为负载均衡器，您可以通过在 HAProxy 节点上运行 netstat -nltupe 来检查 haproxy 进程是否在侦听端口 6443、22623、443 和 80。

18.2.4. 准备用户置备的基础架构

在用户置备的基础架构上安装 OpenShift Container Platform 之前，您必须准备底层基础架构。

本节详细介绍了设置集群基础架构以准备 OpenShift Container Platform 安装所需的高级别步骤。这包括为集群节点配置 IP 网络和网络连接，为 Ignition 文件准备 Web 服务器，通过防火墙启用所需的端口，以及设置所需的 DNS 和负载均衡基础架构。

准备后，集群基础架构必须满足 *带有用户置备的基础架构部分的集群要求*。

先决条件

- 您已参阅 OpenShift Container Platform 4.x Tested Integrations 页面。
- 您已查看了 *具有用户置备基础架构的集群要求部分中详述的基础架构* 要求。

流程

1. 设置静态 IP 地址。
2. 设置 HTTP 或 HTTPS 服务器，为集群节点提供 Ignition 文件。
3. 确保您的网络基础架构提供集群组件之间所需的网络连接。有关 *要求的详情，请参阅用户置备的基础架构* 的网络要求部分。
4. 将防火墙配置为启用 OpenShift Container Platform 集群组件进行通信所需的端口。*如需有关所需端口的详细信息，请参阅用户置备的基础架构* 部分的网络要求。

重要

默认情况下，OpenShift Container Platform 集群可以访问端口 1936，因为每个 control plane 节点都需要访问此端口。

避免使用 Ingress 负载均衡器公开此端口，因为这样做可能会导致公开敏感信息，如统计信息和指标（与 Ingress Controller 相关的统计信息和指标）。

5. 为集群设置所需的 DNS 基础架构。
   a. 为 Kubernetes API、应用程序通配符、bootstrap 机器、control plane 机器和计算机器配置 DNS 名称解析。
   b.

为 Kubernetes API、bootstrap 机器、control plane 机器和计算机器配置反向 DNS 解析。

如需有关 *OpenShift Container Platform DNS 要求的更多信息，请参阅用户置备* DNS 要求部分。

6. 验证您的 DNS 配置。

   a. 从安装节点，针对 Kubernetes API 的记录名称、通配符路由和集群节点运行 DNS 查找。验证响应中的 IP 地址是否与正确的组件对应。

   b. 从安装节点，针对负载均衡器和集群节点的 IP 地址运行反向 DNS 查找。验证响应中的记录名称是否与正确的组件对应。

   有关详细的 *DNS 验证步骤，请参阅用户置备的基础架构* 验证 DNS 解析部分。

7. 置备所需的 API 和应用程序入口负载平衡基础架构。有关 *要求的更多信息，请参阅用户置备的基础架构的负载平衡* 要求部分。

注意

某些负载平衡解决方案要求在初始化负载平衡之前，对群集节点进行 DNS 名称解析。

18.2.5. 验证用户置备的基础架构的 DNS 解析

您可以在在用户置备的基础架构上安装 OpenShift Container Platform 前验证 DNS 配置。

重要

本节中详述的验证步骤必须在安装集群前成功。

先决条件

- 已为您的用户置备的基础架构配置了所需的 DNS 记录。

流程

1. 从安装节点，针对 Kubernetes API 的记录名称、通配符路由和集群节点运行 DNS 查找。验证响应中包含的 IP 地址是否与正确的组件对应。

   a. 对 Kubernetes API 记录名称执行查询。检查结果是否指向 API 负载均衡器的 IP 地址：

   ```
   $ dig +noall +answer @<nameserver_ip> api.<cluster_name>.<base_domain> 1
   ```

   1 将 <nameserver_ip> 替换为 nameserver 的 IP 地址，<cluster_name> 替换为您的集群名称，<base_domain> 替换为您的基本域名。

   输出示例

   ```
   api.ocp4.example.com.  604800 IN A 192.168.1.5
   ```

   b. 对 Kubernetes 内部 API 记录名称执行查询。检查结果是否指向 API 负载均衡器的 IP 地址：

   ```
   $ dig +noall +answer @<nameserver_ip> api-int.<cluster_name>.<base_domain>
   ```

   输出示例

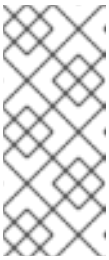
   ```
   api-int.ocp4.example.com.  604800 IN A 192.168.1.5
   ```

   c. 测试 *.apps.<cluster_name>.<base_domain> DNS 通配符查找示例。所有应用程序通配符查询都必须解析为应用程序入口负载均衡器的 IP 地址：

```
$ dig +noall +answer @<nameserver_ip> random.apps.<cluster_name>.
<base_domain>
```

输出示例

```
random.apps.ocp4.example.com.  604800 IN A 192.168.1.5
```



注意

在示例中，将相同的负载均衡器用于 Kubernetes API 和应用程序入口流量。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。

您可以使用另一个通配符值替换 random。例如，您可以查询到 OpenShift Container Platform 控制台的路由：

```
$ dig +noall +answer @<nameserver_ip> console-openshift-console.apps.
<cluster_name>.<base_domain>
```

输出示例

```
console-openshift-console.apps.ocp4.example.com. 604800 IN A 192.168.1.5
```

d. 针对 bootstrap DNS 记录名称运行查询。检查结果是否指向 bootstrap 节点的 IP 地址：

```
$ dig +noall +answer @<nameserver_ip> bootstrap.<cluster_name>.
<base_domain>
```

输出示例

```
bootstrap.ocp4.example.com.  604800 IN A 192.168.1.96
```

e. 使用此方法对 control plane 和计算节点的 DNS 记录名称执行查找。检查结果是否与每个节点的 IP 地址对应。

2. 从安装节点，针对负载均衡器和集群节点的 IP 地址运行反向 DNS 查找。验证响应中包含的记录名称是否与正确的组件对应。

a. 对 API 负载均衡器的 IP 地址执行反向查找。检查响应是否包含 Kubernetes API 和 Kubernetes 内部 API 的记录名称：

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.5
```

输出示例

```
5.1.168.192.in-addr.arpa. 604800 IN PTR api-int.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. 604800 IN PTR api.ocp4.example.com. 2
```

1 为 Kubernetes 内部 API 提供记录名称。

2 为 Kubernetes API 提供记录名称。

注意

OpenShift Container Platform 应用程序通配符不需要 PTR 记录。针对应用程序入口负载均衡器的 IP 地址解析反向 DNS 解析不需要验证步骤。

b. 对 bootstrap 节点的 IP 地址执行反向查找。检查结果是否指向 bootstrap 节点的 DNS

记录名称：

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.96
```

输出示例

```
96.1.168.192.in-addr.arpa. 604800 IN PTR bootstrap.ocp4.example.com.
```

c. 使用此方法对 control plane 和计算节点的 IP 地址执行反向查找。检查结果是否与每个节点的 DNS 记录名称对应。

18.2.6. 为集群节点 SSH 访问生成密钥对

在 OpenShift Container Platform 安装过程中，您可以为安装程序提供 SSH 公钥。密钥通过它们的 Ignition 配置文件传递给 Red Hat Enterprise Linux CoreOS(RHCOS)节点，用于验证对节点的 SSH 访问。密钥添加到每个节点上 core 用户的 ~/.ssh/authorized_keys 列表中，这将启用免密码身份验证。

将密钥传递给节点后，您可以使用密钥对作为用户 核心 通过 SSH 连接到 RHCOS 节点。若要通过 SSH 访问节点，必须由 SSH 为您的本地用户管理私钥身份。

如果要通过 SSH 连接到集群节点来执行安装调试或灾难恢复，则必须在安装过程中提供 SSH 公钥。./openshift-install gather 命令还需要在集群节点上设置 SSH 公钥。

重要

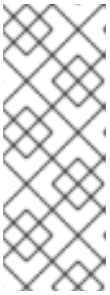
不要在生产环境中跳过这个过程，在生产环境中需要灾难恢复和调试。

流程

1. 如果您在本地计算机上没有可用于在集群节点上进行身份验证的现有 SSH 密钥对，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 ~/.ssh/id_ed25519。如果您已有密钥对，请确保您的公钥位于 ~/.ssh 目录中。



注意

如果您计划在 x86_64、ppc64le 和 s390x 架构上安装使用 RHEL 加密库（这些加密库已提交给 NIST 用于 FIPS 140-2/140-3 验证）的 OpenShift Container Platform 集群，则不要创建使用 ed25519 算法的密钥。相反，创建一个使用 rsa 或 ecdsa 算法的密钥。

2. 查看公共 SSH 密钥：

```
$ cat <path>/<file_name>.pub
```

例如，运行以下命令来查看 ~/.ssh/id_ed25519.pub 公钥：

```
$ cat ~/.ssh/id_ed25519.pub
```

3. 将 SSH 私钥身份添加到本地用户的 SSH 代理（如果尚未添加）。在集群节点上，或者要使用 ./openshift-install gather 命令，需要对该密钥进行 SSH 代理管理，才能在集群节点上进行免密码 SSH 身份验证。

注意

在某些发行版中，自动管理默认 SSH 私钥身份，如 ~/.ssh/id_rsa 和 ~/.ssh/id_dsa。

a. 如果 ssh-agent 进程尚未为您的本地用户运行，请将其作为后台任务启动：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果集群处于 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

4. 将 SSH 私钥添加到 ssh-agent ：

```
$ ssh-add <path>/<file_name> 1
```

1 指定 SSH 私钥的路径和文件名，如 ~/.ssh/id_ed25519.pub

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

后续步骤

- 安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

18.2.7. 获取安装程序

在安装 OpenShift Container Platform 之前，将安装文件下载到您置备的机器上。

先决条件

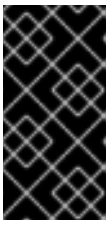
- 您有一个运行 Linux 的机器，如 Red Hat Enterprise Linux 8，本地磁盘空间为 500 MB。

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐户，请使用您的凭证登录。如果没有，请创建一个帐户。

2. 选择您的基础架构供应商。

3. 进入到安装类型的页面，下载与您的主机操作系统和架构对应的安装程序，并将该文件放在您要存储安装配置文件的目录中。



重要

安装程序会在用来安装集群的计算机上创建几个文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。

重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，请为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret。此 pull secret 允许您与所含授权机构提供的服务进行身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

18.2.8. 安装 OpenShift CLI

您可以安装 OpenShift CLI(oc)来使用命令行界面与 OpenShift Container Platform 进行交互。您可以在 Linux、Windows 或 macOS 上安装 oc。

重要

如果安装了旧版本的 oc，则可能无法使用 OpenShift Container Platform 4.16 中的所有命令。下载并安装新版本的 oc。

在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI(oc)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 产品变体 下拉列表中选择架构。
3. 从 版本 下拉列表中选择适当的版本。
4. 点 OpenShift v4.16 Linux Client 条目旁的 Download Now 来保存文件。
5. 解包存档：

```
$ tar xvf <file>
```

6. 将 oc 二进制文件放到 PATH 中的目录中。

   要查看您的 PATH，请执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 oc 命令：

```
$ oc <command>
```

在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI(oc)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 版本 下拉列表中选择适当的版本。
3. 点 OpenShift v4.16 Windows Client 条目旁的 Download Now 来保存文件。
4. 使用 ZIP 程序解压存档。
5. 将 oc 二进制文件移到 PATH 中的目录中。

   要查看您的 PATH，请打开命令提示并执行以下命令：

   ```
   C:\> path
   ```

验证

- 安装 OpenShift CLI 后，可以使用 oc 命令：

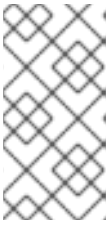
  ```
  C:\> oc <command>
  ```

在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI(oc)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 版本 下拉列表中选择适当的版本。
3. 点 OpenShift v4.16 macOS Client 条目旁的 Download Now 来保存文件。



注意

对于 **macOS arm64**，请选择 **OpenShift v4.16 macOS arm64 Client** 条目。

4. 解包和解压存档。

5. 将 **oc** 二进制文件移到 **PATH** 的目录中。

要查看您的 **PATH**，请打开终端并执行以下命令：

```
$ echo $PATH
```

验证

- 安装 **OpenShift CLI** 后，可以使用 **oc** 命令：

```
$ oc <command>
```

18.2.9. 手动创建安装配置文件

安装集群要求您手动创建安装配置文件。

先决条件

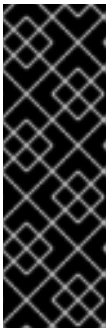
- 您在本地机器上有一个 **SSH** 公钥来提供给安装程序。该密钥将用于在集群节点上进行 **SSH** 身份验证，以进行调试和灾难恢复。

- 已获取 **OpenShift Container Platform** 安装程序和集群的 **pull secret**。

流程

1. 创建一个安装目录来存储所需的安装资产：

```
$ mkdir <installation_directory>
```



重要

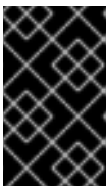
您必须创建一个目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

2. 自定义提供的 install-config.yaml 文件模板示例，并将其保存在 <installation_directory> 中。

注意

此配置文件必须命名为 install-config.yaml。

3. 备份 install-config.yaml 文件，以便您可以使用它安装多个集群。



重要

install-config.yaml 文件会在安装过程的下一步中使用。现在必须备份它。

其他资源

- IBM Z® 的安装配置参数

18.2.9.1. IBM Z 的 install-config.yaml 文件示例

您可以自定义 install-config.yaml 文件，以指定有关 OpenShift Container Platform 集群平台的更多详情，或修改所需参数的值。

```
apiVersion: v1
baseDomain: example.com (1)
compute: (2)
- hyperthreading: Enabled (3)
  name: worker
  replicas: 0 (4)
  architecture: s390x
controlPlane: (5)
  hyperthreading: Enabled (6)
```

```
  name: master
  replicas: 3 7
  architecture: s390x
metadata:
  name: test 8
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14 9
    hostPrefix: 23 10
  networkType: OVNKubernetes 11
  serviceNetwork: 12
  - 172.30.0.0/16
platform:
  none: {} 13
fips: false 14
pullSecret: '{"auths": ...}' 15
sshKey: 'ssh-ed25519 AAAA...' 16
```

1

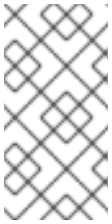
集群的基域。所有 DNS 记录都必须是这个基域的子域，并包含集群名称。

2 5

controlPlane 部分是一个单个映射，但 compute 部分是一系列映射。为满足不同数据结构的要求，compute 部分的第一行必须以连字符 - 开头，controlPlane 部分 的第一行则不以连字符开头。仅使用一个 control plane 池。

3 6

指定要启用或禁用并发多线程(SMT)还是超线程。默认情况下，启用 SMT 可提高机器中内核的性能。您可以通过将 参数值设置为 Disabled 来禁用它。如果禁用 SMT，则必须在所有集群机器中禁用它；这包括 control plane 和计算机器。



注意

默认启用并发多线程(SMT)。如果 OpenShift Container Platform 节点上没有 SMT，超线程 参数无效。

重要

如果您禁用 超线程，无论是在 OpenShift Container Platform 节点上，还是在 install-config.yaml 文件中，请确保您的容量规划考虑机器性能显著降低的情况。

4

注意

如果要安装一个三节点集群，在安装 Red Hat Enterprise Linux CoreOS(RHCOS)机器时不要部署任何计算机器。

7

您添加到集群的 control plane 机器数量。由于集群使用这些值作为集群中的 etcd 端点数量，所以该值必须与您部署的 control plane 机器数量匹配。

8

您在 DNS 记录中指定的集群名称。

9

从中分配 Pod IP 地址的 IP 地址块。此块不得与现有物理网络重叠。这些 IP 地址用于 pod 网络。如果需要从外部网络访问 pod，您必须配置负载均衡器和路由器来管理流量。

注意

类 E CIDR 范围被保留以供以后使用。要使用 Class E CIDR 范围，您必须确保您的网络环境接受 Class E CIDR 范围内的 IP 地址。

10

分配给每个节点的子网前缀长度。例如，如果 hostPrefix 设为 23，则每个节点从 given cidr 中分配 a /23 子网，这样就能有 510（2^(32 - 23)- 2）个 pod IP 地址。如果需要从外部网络访问节点，请配置负载均衡器和路由器来管理流量。

11

要安装的集群网络插件。默认值 OVNKubernetes 是唯一支持的值。

12

用于服务 IP 地址的 IP 地址池。您只能输入一个 IP 地址池。此块不得与现有物理网络重叠。如果您需要从外部网络访问服务，请配置负载均衡器和路由器来管理流量。

13

您必须将平台设置为 none。您无法为 IBM Z® 基础架构提供额外的平台配置变量。

重要

使用平台类型 none 安装的集群无法使用一些功能，如使用 Machine API 管理计算机器。即使附加到集群的计算机器安装在通常支持该功能的平台上，也会应用这个限制。在安装后无法更改此参数。

14

是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 Red Hat Enterprise Linux (RHEL) 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅在 FIPS 模式中安装该系统。

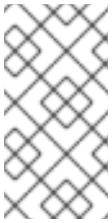
当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS)时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。

15

Red Hat OpenShift Cluster Manager 的 pull secret。此 pull secret 允许您与所含授权机构提供的服务进行身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

16

Red Hat Enterprise Linux CoreOS(RHCOS)中 core 用户的 SSH 公钥。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 ssh-agent 进程使用的 SSH 密钥。

18.2.9.2. 在安装过程中配置集群范围的代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 install-

config.yaml 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 install-config.yaml 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 Proxy 对象的 spec.noProxy 字段中添加站点来绕过代理。

注意

Proxy 对象 status.noProxy 字段使用安装配置中的 networking.machineNetwork[].cidr、networking.clusterNetwork[].cidr 和 networking.serviceNetwork[] 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，Proxy 对象 status.noProxy 字段也会使用实例元数据端点填充(169.254.169.254)。

流程

1. 编辑 install-config.yaml 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 http。

2.

3

要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 . 以仅匹配子域。例如，.y.com 匹配 x.y.com，但不匹配 y.com。使用 * 绕过所有目的地的代理。

4

如果提供，安装程序会在 openshift-config 命名空间中生成名为 user-ca-bundle 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 trusted-ca-bundle 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， Proxy 对象的 trustedCA 字段中也会引用此配置映射。additionalTrustBundle 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

5

可选：决定 Proxy 对象的配置以引用 trustedCA 字段中 user-ca-bundle 配置映射的策略。允许的值是 Proxyonly 和 Always。仅在配置了 http/https 代理时，使用 Proxyonly 引用 user-ca-bundle 配置映射。使用 Always 始终引用 user-ca-bundle 配置映射。默认值为 Proxyonly。

注意

安装程序不支持代理的 readinessEndpoints 字段。

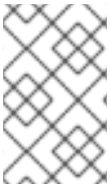
注意

如果安装程序超时，重启并使用安装程序的 wait-for 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. 保存该文件并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 cluster 的集群范围代理，该代理 使用 提供的 install-config.yaml 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 cluster Proxy 对象，但它会有一个空 spec。



注意

只支持名为 cluster 的 Proxy 对象，且无法创建额外的代理。

18.2.9.3. 配置三节点集群

另外，您可以在由三台 control plane 机器组成的最少三个节点集群中部署零台计算机器。这为集群管理员和开发人员提供了更小、效率更高的集群，用于测试、开发和生产。

在三节点 OpenShift Container Platform 环境中，三台 control plane 机器可以调度，这意味着应用程序工作负载被调度到它们上运行。

先决条件

- 您有一个现有的 install-config.yaml 文件。

流程

- 确保 install-config.yaml 文件中的计算副本数量设置为 0，如以下 计算 小节所示：

```
compute:
- name: worker
  platform: {}
  replicas: 0
```

注意

在用户置备的基础架构上安装 OpenShift Container Platform 时，无论您要部署的计算机器数量有多少，您必须将计算机器的 replicas 参数值设置为 0。在安装程序置备的安装中，参数控制集群为您创建和管理的计算机器数量。这不适用于手动部署计算机器的用户置备安装。

注意

control plane 节点的首选资源是 6 个 vCPU 和 21 GB。对于三个 control plane 节点，这是相当于至少五节点集群的内存 + vCPU。您应该为三个节点提供支持，每个节点安装在一个 120 GB 的磁盘上，并且启用了三个 SMT2 的 IFL。测试最小的设置是每个 control plane 节点在 120 GB 磁盘上的三个 vCPU 和 10 GB。

对于三节点集群安装，请按照以下步骤执行：

- 如果要部署一个带有零计算节点的三节点集群，Ingress Controller Pod 在 control plane 节点上运行。在三节点集群部署中，您必须配置应用程序入口负载均衡器，将 HTTP 和 HTTPS 流量路由到 control plane 节点。如需更多信息，*请参阅用户置备的基础架构的负载平衡要求* 部分。

- 在以下步骤中创建 Kubernetes 清单文件时，请确保 <installation_directory>/manifests/cluster-scheduler-02-config.yml 文件中的 mastersSchedulable 参数被设置为 true。这可让应用程序工作负载在 control plane 节点上运行。

- 在创建 Red Hat Enterprise Linux CoreOS(RHCOS)机器时，不要部署任何计算节点。

18.2.10. Cluster Network Operator 配置

集群网络的配置作为 Cluster Network Operator(CNO)配置的一部分指定，并存储在名为 cluster 的自定义资源(CR)对象中。CR 指定 operator.openshift.io API 组中的 Network API 的字段。

CNO 配置在集群安装过程中从 Network.config.openshift.io API 组中的 Network API 继承以下字段：

clusterNetwork

从中分配 Pod IP 地址的 IP 地址池。

serviceNetwork

服务的 IP 地址池.

defaultNetwork.type

集群网络插件。OVNKubernetes 是安装期间唯一支持的插件。

您可以通过在名为 cluster 的 CNO 对象中设置 defaultNetwork 对象的字段来为集群指定集群网络插件配置。

18.2.10.1. Cluster Network Operator 配置对象

下表中描述了 Cluster Network Operator(CNO)的字段：

表 18.10. Cluster Network Operator 配置对象

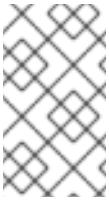
字段	类型	描述
metadata.name	**字符串**	CNO 对象的名称。这个名称始终是 **集群**。
spec.clusterNetwork	**array**	用于指定从哪些 IP 地址块分配 Pod IP 地址以及集群中每个节点的子网前缀长度的列表。例如： spec: clusterNetwork: - cidr: 10.128.0.0/19 hostPrefix: 23 - cidr: 10.128.32.0/19 hostPrefix: 23
spec.serviceNetwork	**array**	服务的 IP 地址块。OpenShift SDN 和 OVN-Kubernetes 网络插件只支持服务网络的一个 IP 地址块。例如： spec: serviceNetwork: - 172.30.0.0/14 您只能在创建清单前在 **install-config.yaml** 文件中自定义此字段。该值在清单文件中是只读的。
spec.defaultNetwork	**object**	为集群网络配置网络插件。
spec.kubeProxyConfig	**object**	此对象的字段指定 kube-proxy 配置。如果使用 OVN-Kubernetes 集群网络供应商，则 kube-proxy 配置不会起作用。

defaultNetwork 对象配置

下表列出了 **defaultNetwork** 对象的值：

表 18.11. defaultNetwork 对象

字段	类型	描述

字段	类型	描述
type	**字符串**	**OVNKubernetes**。Red Hat OpenShift Networking 网络插件在安装过程中被选择。此值在集群安装后无法更改。  **注意** OpenShift Container Platform 默认使用 OVN-Kubernetes 网络插件。OpenShift SDN 不再作为新集群的安装选择提供。
ovnKubernetesConfig	**object**	此对象仅对 OVN-Kubernetes 网络插件有效。

配置 OVN-Kubernetes 网络插件

下表描述了 OVN-Kubernetes 网络插件的配置字段：

表 18.12. ovnKubernetesConfig object

字段	类型	描述
mtu	**integer**	Geneve（通用网络虚拟化封装）覆盖网络的最大传输单元 (MTU)。这根据主网络接口的 MTU 自动探测。您通常不需要覆盖检测到的 MTU。 如果自动探测的值不是您期望的值，请确认节点上主网络接口上的 MTU 是否正确。您不能使用这个选项更改节点上主网络接口的 MTU 值。 如果集群中不同节点需要不同的 MTU 值，则必须将此值设置为 **比** 集群中的最低 MTU 值小 100。例如，如果集群中的某些节点的 MTU 为 **9001**，而某些节点的 MTU 为 **1500**，则必须将此值设置为 **1400**。
genevePort	**integer**	用于所有 Geneve 数据包的端口。默认值为 **6081**。此值在集群安装后无法更改。
ipsecConfig	**object**	指定用于自定义 IPsec 配置的配置对象。
ipv4	**object**	为 IPv4 设置指定配置对象。
ipv6	**object**	为 IPv6 设置指定配置对象。
policyAuditConfig	**object**	指定用于自定义网络策略审计日志的配置对象。如果未设置，则使用默认的审计日志设置。

字段	类型	描述
gatewayConfig	**object**	可选：指定一个配置对象来自定义如何将出口流量发送到节点网关。  **注意** 在迁移出口流量时，工作负载和服务流量会受到一定影响，直到 Cluster Network Operator (CNO) 成功推出更改。

表 18.13. ovnKubernetesConfig.ipv4 object

字段	类型	描述
internalTransitSwitchSubnet	字符串	如果您的现有网络基础架构与 **100.88.0.0/16** IPv4 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。启用东西流量的分布式传输交换机的子网。此子网不能与 OVN-Kubernetes 或主机本身使用的任何其他子网重叠。必须足够大，以适应集群中的每个节点一个 IP 地址。 默认值为 **100.88.0.0/16**。
internalJoinSubnet	字符串	如果您的现有网络基础架构与 **100.64.0.0/16** IPv4 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。您必须确保 IP 地址范围没有与 OpenShift Container Platform 安装使用的任何其他子网重叠。IP 地址范围必须大于可添加到集群的最大节点数。例如，如果 **clusterNetwork.cidr** 值为 **10.128.0.0/14**，并且 **clusterNetwork.hostPrefix** 值为 **/23**，则最大节点数量为 **2^(23-14)=512**。 默认值为 **100.64.0.0/16**。

表 18.14. ovnKubernetesConfig.ipv6 object

字段	类型	描述
internalTransitSwitchSubnet	字符串	如果您的现有网络基础架构与 **fd97::/64** IPv6 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。启用东西流量的分布式传输交换机的子网。此子网不能与 OVN-Kubernetes 或主机本身使用的任何其他子网重叠。必须足够大，以适应集群中的每个节点一个 IP 地址。 默认值为 **fd97::/64**。

字段	类型	描述
internalJoinSubnet	字符串	如果您的现有网络基础架构与 **fd98::/64** IPv6 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。您必须确保 IP 地址范围没有与 OpenShift Container Platform 安装使用的任何其他子网重叠。IP 地址范围必须大于可添加到集群的最大节点数。 默认值为 **fd98::/64**。

表 18.15. policyAuditConfig object

字段	类型	描述
rateLimit	整数	每个节点每秒生成一次的消息数量上限。默认值为每秒 **20** 条消息。
maxFileSize	整数	审计日志的最大大小，以字节为单位。默认值为 **50000000** 或 50 MB。
maxLogFiles	整数	保留的日志文件的最大数量。
目的地	字符串	以下附加审计日志目标之一： **libc** 主机上的 journald 进程的 libc **syslog（）** 函数。 **UDP:<host>:<port>** 一个 syslog 服务器。将 **<host>:<port>** 替换为 **syslog 服务器的主机** 和端口。 **Unix:<file>** 由 **<file>** 指定的 Unix 域套接字文件。 **null** 不要将审计日志发送到任何其他目标。
syslogFacility	字符串	syslog 工具，如 as **kern**，如 RFC5424 定义。默认值为 **local0**。

表 18.16. gatewayConfig object

字段	类型	描述

字段	类型	描述
routingViaHost	**布尔值**	将此字段设置为 **true**，将来自 pod 的出口流量发送到主机网络堆栈。对于依赖于在内核路由表中手动配置路由的高级别安装和应用程序，您可能需要将出口流量路由到主机网络堆栈。默认情况下，出口流量在 OVN 中进行处理以退出集群，不受内核路由表中的特殊路由的影响。默认值为 **false**。 此字段与 Open vSwitch 硬件卸载功能有交互。如果将此字段设置为 **true**，则不会获得卸载的性能优势，因为主机网络堆栈会处理出口流量。
ipForwarding	**object**	您可以使用 **Network** 资源中的 **ipForwarding** 规格来控制 OVN-Kubernetes 管理接口上所有流量的 IP 转发。指定 **Restricted** 只允许 Kubernetes 相关流量的 IP 转发。指定 **Global** 以允许转发所有 IP 流量。对于新安装，默认值为 **Restricted**。对于到 OpenShift Container Platform 4.14 或更高版本的更新，默认值为 **Global**。
ipv4	**object**	可选：指定一个对象来为主机配置内部 OVN-Kubernetes 伪装地址，以服务 IPv4 地址的流量。
ipv6	**object**	可选：指定一个对象来为主机配置内部 OVN-Kubernetes 伪装地址，以服务 IPv6 地址的流量。

表 18.17. gatewayConfig.ipv4 对象

字段	类型	描述
internalMasqueradeSubnet	**字符串**	内部使用的伪装 IPv4 地址，以启用主机服务流量。主机配置了这些 IP 地址和共享网关网桥接口。默认值为 **169.254.169.0/29**。

表 18.18. gatewayConfig.ipv6 对象

字段	类型	描述
internalMasqueradeSubnet	**字符串**	内部使用的伪装 IPv6 地址，以启用主机服务流量。主机配置了这些 IP 地址和共享网关网桥接口。默认值为 **fd69::/125**。

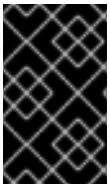
表 18.19. ipsecConfig 对象

字段	类型	描述

字段	类型	描述
模式	**字符串**	指定 IPsec 实现的行为。必须是以下值之一： • **Disabled**: 在集群节点上不启用 IPsec。 • **External** ：对于带有外部主机的网络流量，启用 IPsec。 • **Full**: IPsec 为带有外部主机的 pod 流量和网络流量启用 IPsec。

启用 IPSec 的 OVN-Kubernetes 配置示例

```
defaultNetwork:
  type: OVNKubernetes
  ovnKubernetesConfig:
    mtu: 1400
    genevePort: 6081
      ipsecConfig:
        mode: Full
```



重要

使用 OVNKubernetes 可能会导致 IBM Power® 上的堆栈耗尽问题。

kubeProxyConfig 对象配置（仅限 OpenShiftSDN 容器网络接口）

kubeProxyConfig 对象的值在下表中定义：

表 18.20. kubeProxyConfig object

字段	类型	描述

字段	类型	描述
iptablesSyncPeriod	**字符串**	**iptables** 规则的刷新周期。默认值为 **30s**。有效的后缀包括 **s**、**m** 和 **h**，具体参见 Go 时间 包 文档。 **注意** 由于 OpenShift Container Platform 4.3 及更高版本中引进了性能改进，不再需要调整 **iptablesSyncPeriod** 参数。
proxyArguments.iptables-min-sync-period	**array**	刷新 **iptables** 规则前的最短持续时间。此字段确保刷新的频率不会过于频繁。有效的后缀包括 **s**、**m** 和 **h**，具体参见 Go **time** 软件包。默认值为： kubeProxyConfig: proxyArguments: iptables-min-sync-period: - 0s

18.2.11. 创建 Kubernetes 清单和 Ignition 配置文件

由于您必须修改一些集群定义文件并手动启动集群机器，因此您必须生成 Kubernetes 清单和 Ignition 配置文件来配置机器。

安装配置文件转换为 Kubernetes 清单。清单嵌套到 Ignition 配置文件中，稍后用于配置集群机器。

重要

- OpenShift Container Platform 安装程序生成的 Ignition 配置文件包含 24 小时后过期的证书，然后在该时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 node-bootstrapper 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

注意

生成清单和 Ignition 文件的安装程序是特定的架构，可以从 客户端镜像镜像获取。安装程序的 Linux 版本仅在 s390x 上运行。此安装程序也可用作 Mac OS 版本。

先决条件

- 已获得 OpenShift Container Platform 安装程序。
- 已创建 install-config.yaml 安装配置文件。

流程

1. 进入包含 OpenShift Container Platform 安装程序的目录，并为集群生成 Kubernetes 清单：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 对于 <installation_directory>，请指定包含您创建的 install-config.yaml 文件的安装目录。

警告

如果您要安装一个三节点集群，请跳过以下步骤，以便可以调度 control plane 节点。

重要

当您将 control plane 节点从默认的不可调度配置为可以调度时，需要额外的订阅。这是因为 control plane 节点变为计算节点。

2. 检查 <installation_directory>/manifests/cluster-scheduler-02-config.yml Kubernetes 清单文件中的 mastersSchedulable 参数是否已设置为 false。此设置可防止在 control plane 机

器上调度 pod：

a. 打开 <installation_directory>/manifests/cluster-scheduler-02-config.yml 文件。

b. 找到 mastersSchedulable 参数，并确保它被设置为 false。

c. 保存并退出 文件。

3. 要创建 Ignition 配置文件，请从包含安装程序的目录运行以下命令：

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

1 对于 <installation_directory>，请指定相同的安装目录。

为安装目录中的 bootstrap、control plane 和计算节点创建 Ignition 配置文件。kubeadmin-password 和 kubeconfig 文件在 ./<installation_directory>/auth 目录中创建：

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

18.2.12. 在 IBM Z 或 IBM LinuxONE 环境中使用静态 IP 配置 NBDE

在 IBM Z® 或 IBM® LinuxONE 环境中启用 NBDE 磁盘加密需要额外的步骤，本节中详细介绍。

先决条件

- 您已设置了外部 Tang 服务器。具体步骤请查看 网络绑定磁盘加密。

- 您已安装了 with ane 实用程序。
- 您已查看如何使用 Butane 创建机器配置的说明。

流程

1. 为 control plane 和计算节点创建 Butane 配置文件。

   以下 control plane 节点的 Butane 配置示例为磁盘加密创建一个名为 master-storage.bu 的文件：

```
variant: openshift
version: 4.16.0
metadata:
  name: master-storage
  labels:
    machineconfiguration.openshift.io/role: master
storage:
  luks:
    - clevis:
        tang:
          - thumbprint: QcPr_NHFJammnRCA3fFMVdNBwjs
            url: http://clevis.example.com:7500
        options: (1)
          - --cipher
          - aes-cbc-essiv:sha256
      device: /dev/disk/by-partlabel/root (2)
      label: luks-root
      name: root
      wipe_volume: true
  filesystems:
    - device: /dev/mapper/root
      format: xfs
      label: root
      wipe_filesystem: true
openshift:
  fips: true (3)
```

(1) 只有在启用了 FIPS 模式时才需要 cipher 选项。如果禁用了 FIPS，则省略该条目。

(2) 对于在 DASD 类型磁盘中安装，使用 device: /dev/disk/by-label/root 替换。

3

是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

2. 运行以下命令，创建自定义 initramfs 文件来引导机器：

```
$ coreos-installer pxe customize \
  /root/rhcos-bootfiles/rhcos-<release>-live-initramfs.s390x.img \
  --dest-device /dev/disk/by-id/scsi-<serial_number> --dest-karg-append \
  ip=<ip_address>::<gateway_ip>:<subnet_mask>::<network_device>:none \
  --dest-karg-append nameserver=<nameserver_ip> \
  --dest-karg-append rd.neednet=1 -o \
  /root/rhcos-bootfiles/<node_name>-initramfs.s390x.img
```

注意

首次引导前，您必须为集群中的每个节点自定义 initramfs，并添加 PXE 内核参数。

3. 创建包含 ignition.platform.id=metal 和 ignition.firstboot 的参数文件。

control plane 机器的内核参数文件示例：

```
rd.neednet=1 \
console=ttysclp0 \
coreos.inst.install_dev=/dev/dasda \1
ignition.firstboot ignition.platform.id=metal \
coreos.live.rootfs_url=http://<http_server>/rhcos-<version>-live-rootfs.
<architecture>.img \2
coreos.inst.ignition_url=http://<http_server>/master.ign \3
ip=10.19.17.2::10.19.17.1:255.255.255.0::enbdd0:none nameserver=10.19.17.1 \
zfcp.allow_lun_scan=0 \4
rd.znet=qeth,0.0.bdd0,0.0.bdd1,0.0.bdd2,layer2=1 \
rd.zfcp=0.0.5677,0x600606680g7f0056,0x034F000000000000 \5
```

1

2

指定您要引导的 kernel 和 initramfs 的 rootfs 工件位置。仅支持 HTTP 和 HTTPS 协议。

3

指定 Ignition 配置文件的位置。使用 master.ign 或 worker.ign。仅支持 HTTP 和 HTTPS 协议。

4

对于在 FCP 类型磁盘中安装，请添加 zfcp.allow_lun_scan=0。为 DASD 类型磁盘省略这个值。

5

对于在 DASD 类型磁盘中安装，使用 rd.dasd=0.0.3490 替换来指定 DASD 设备。

注意

将参数文件中的所有选项写为一行，并确保您没有换行字符。

其他资源

- 使用 Butane 创建机器配置

18.2.13. 安装 RHCOS 并启动 OpenShift Container Platform bootstrap 过程

要在您置备的 IBM Z® 基础架构上安装 OpenShift Container Platform，您必须在 z/VM 客户机虚拟机上安装 Red Hat Enterprise Linux CoreOS (RHCOS)。安装 RHCOS 时，您必须为您要安装的机器类型提供 OpenShift Container Platform 安装程序生成的 Ignition 配置文件。如果您配置了适当的网络、DNS 和负载均衡基础架构，OpenShift Container Platform bootstrap 过程会在 RHCOS z/VM 客户机虚拟机重启后自动启动。

完成以下步骤以创建机器。

先决条件

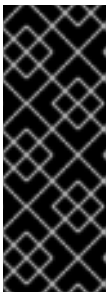
- 在置备机器上运行的 HTTP 或 HTTPS 服务器，可供您创建的机器访问。

- 如果要启用安全引导，需要已获取了适当的红帽产品签名密钥，并在 IBM 文档中的 IBM Z 和 IBM LinuxONE 上读取安全引导。

流程

1. 在您的置备机器上登录到 Linux。

2. 从 RHCOS 镜像镜像获取 Red Hat Enterprise Linux CoreOS(RHCOS)内核、initramfs 和 rootfs 文件。

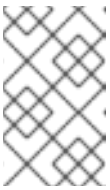


重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每个发行版本而改变。您必须下载最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。仅使用以下步骤中描述的适当 kernel、initramfs 和 rootfs 工件。

文件名包含 OpenShift Container Platform 版本号。它们类似以下示例：

- kernel: rhcos-<version>-live-kernel-<architecture>
- initramfs: rhcos-<version>-live-initramfs.<architecture>.img
- rootfs: rhcos-<version>-live-rootfs.<architecture>.img



注意

FCP 和 DASD 的 rootfs 镜像相同。

3. 创建参数文件。以下参数特定于特定虚拟机：

- 对于 ip=，请指定以下七项：

  i. 计算机的 IP 地址。

  ii. 个空字符串。

  iii. 网关

  iv. 子网掩码.

  v. hostname.domainname 格式的机器主机和域名。省略这个值可让 RHCOS 决定。

  vi. 网络接口名称。省略这个值可让 RHCOS 决定。

  vii. 如果使用静态 IP 地址，请指定 none。

- 对于 coreos.inst.ignition_url=，请为机器角色指定 Ignition 文件。使用 bootstrap.ign、master.ign 或 worker.ign。仅支持 HTTP 和 HTTPS 协议。

- 对于 coreos.live.rootfs_url=，请为您引导的内核和 initramfs 指定匹配的 rootfs 构件。仅支持 HTTP 和 HTTPS 协议。

- 可选：要启用安全引导，请添加 coreos.inst.secure_ipl

- 对于在 DASD 类型磁盘中安装，请完成以下任务：

  i. 对于 coreos.inst.install_dev=，请指定 /dev/dasda。

ii. Userd .dasd= 指定安装 RHCOS 的 DASD。

iii. 所有其他参数保持不变。

bootstrap 机器的参数文件示例：bootstrap-0.parm ：

```
rd.neednet=1 \
console=ttysclp0 \
coreos.inst.install_dev=/dev/dasda \
coreos.live.rootfs_url=http://<http_server>/rhcos-<version>-live-rootfs.
<architecture>.img \ 1
coreos.inst.ignition_url=http://<http_server>/bootstrap.ign \ 2
coreos.inst.secure_ipl \ 3
ip=172.18.78.2::172.18.78.1:255.255.255.0:::none nameserver=172.18.78.1 \
rd.znet=qeth,0.0.bdf0,0.0.bdf1,0.0.bdf2,layer2=1,portno=0 \
zfcp.allow_lun_scan=0 \
rd.dasd=0.0.3490
```

1 指定您要引导的 kernel 和 initramfs 的 rootfs 工件位置。仅支持 HTTP 和 HTTPS 协议。

2 指定 Ignition 配置文件的位置。使用 bootstrap.ign、master.ign 或 worker.ign。仅支持 HTTP 和 HTTPS 协议。

3 可选： 要启用安全引导，请添加 coreos.inst.secure_ipl。

将参数文件中的所有选项写为一行，并确保您没有换行字符。

- 对于在 FCP 类型磁盘中安装，请完成以下任务：

i. User d.zfcp=<adapter>,<wwpn>,<lun> 以指定要安装 RHCOS 的 FCP 磁盘。对于多路径，为每个额外路径重复此步骤。

注意

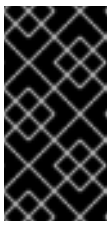
当使用多个路径安装时，您必须在安装后直接启用多路径，而不是在以后启用多路径，因为这可能导致问题。

ii. **将安装设备设置为：coreos.inst.install_dev=/dev/disk/by-id/scsi-<serial_number>。**

注意

如果使用 NPIV 配置额外的 LUN，FCP 需要 zfcp.allow_lun_scan=0。如果必须启用 zfcp.allow_lun_scan=1，因为您使用 CSI 驱动程序，则必须配置 NPIV，以便每个节点无法访问另一个节点的引导分区。

iii. **所有其他参数保持不变。**



重要

需要额外的安装后步骤才能完全启用多路径。如需更多信息，请参阅 *安装后机器配置任务* 中的"使用 RHCOS 上内核参数启用多路径"。

以下是使用多路径的计算节点的 worker-1.parm 示例参数文件：

```
rd.neednet=1 \
console=ttysclp0 \
coreos.inst.install_dev=/dev/disk/by-id/scsi-<serial_number> \
coreos.live.rootfs_url=http://<http_server>/rhcos-<version>-live-rootfs.
<architecture>.img \
coreos.inst.ignition_url=http://<http_server>/worker.ign \
ip=172.18.78.2::172.18.78.1:255.255.255.0:::none nameserver=172.18.78.1 \
rd.znet=qeth,0.0.bdf0,0.0.bdf1,0.0.bdf2,layer2=1,portno=0 \
zfcp.allow_lun_scan=0 \
rd.zfcp=0.0.1987,0x50050763070bc5e3,0x4008400B00000000 \
rd.zfcp=0.0.19C7,0x50050763070bc5e3,0x4008400B00000000 \
rd.zfcp=0.0.1987,0x50050763071bc5e3,0x4008400B00000000 \
rd.zfcp=0.0.19C7,0x50050763071bc5e3,0x4008400B00000000
```

将参数文件中的所有选项写为一行，并确保您没有换行字符。

4. 将 initramfs、内核、参数文件和 RHCOS 镜像传送到 z/VM 中，例如使用 FTP。有关如何使用 FTP 传输文件并从虚拟读取器引导的详情，请参阅 在 Z/VM 中安装。

5. 将文件 punch 到 z/VM 客户机虚拟机的虚拟读取器，即成为 bootstrap 节点。

   请参阅 IBM 文档中的 PUNCH。

   提示

   您可以使用 CP PUNCH 命令，或者，如果使用 Linux，则使用 vmur 命令在两个 z/VM 虚拟机之间传输文件。

6. 在 bootstrap 机器上登录到 CMS。

7. 从 reader IPL bootstrap 机器：

   ```
   $ ipl c
   ```

   请参阅 IBM 文档中的 IPL。

8. 对集群中的其他机器重复此步骤。

18.2.13.1. 高级 RHCOS 安装参考

本节演示了网络配置和其他高级选项，允许您修改 Red Hat Enterprise Linux CoreOS(RHCOS)手动安装过程。下表描述了您可以用于 RHCOS live 安装程序和 coreos-installer 命令的内核参数和命令行选项。

18.2.13.1.1. ISO 安装的网络和绑定选项

如果从 ISO 镜像安装 RHCOS，您可以在引导镜像时手动添加内核参数，以便为节点配置网络。如果没有指定网络参数，当 RHCOS 检测到需要网络来获取 Ignition 配置文件时，在 initramfs 中激活 DHCP。

重要

在手动添加网络参数时，还必须添加 rd.neednet=1 内核参数，以便在 initramfs 中启动网络。

以下信息提供了在 RHCOS 节点上为 ISO 安装配置网络和绑定的示例。示例描述了如何使用 ip=、name server = 和 bond= 内核参数。

注意

添加内核参数时顺序非常重要： ip=、 name server=，然后 bond=。

网络选项在系统引导过程中传递给 dracut 工具。有关 dracut 支持的网络选项的更多信息，请参阅 dracut.cmdline 手册页。

以下示例是 ISO 安装的网络选项。

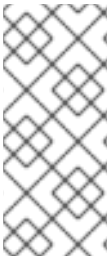
配置 DHCP 或静态 IP 地址

要配置 IP 地址，可使用 DHCP(ip=dhcp)或设置单独的静态 IP 地址(ip=<host_ip>)。如果设置静态 IP，则必须在每个节点上识别 DNS 服务器 IP 地址（名称服务器=<dns_ip>）。以下示例集：

- 节点的 IP 地址为 10.10.10.2
- 网关地址为 10.10.10.254
- 子网掩码为 255.255.255.0
- 到 core0.example.com 的主机名
- DNS 服务器地址为 4.4.4.41

- 自动配置值为 none。当以静态方式配置 IP 网络时，不需要自动配置。

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
nameserver=4.4.4.41
```



注意

当您使用 DHCP 为 RHCOS 机器配置 IP 寻址时，机器还通过 DHCP 获取 DNS 服务器信息。对于基于 DHCP 的部署，您可以通过 DHCP 服务器配置定义 RHCOS 节点使用的 DNS 服务器地址。

配置没有静态主机名的 IP 地址

您可以在不分配静态主机名的情况下配置 IP 地址。如果用户没有设置静态主机名，则会提取并通过反向 DNS 查找自动设置。要在没有静态主机名的情况下配置 IP 地址，请参考以下示例：

- 节点的 IP 地址为 10.10.10.2
- 网关地址为 10.10.10.254
- 子网掩码为 255.255.255.0
- DNS 服务器地址为 4.4.4.41
- 自动配置值为 none。当以静态方式配置 IP 网络时，不需要自动配置。

```
ip=10.10.10.2::10.10.10.254:255.255.255.0::enp1s0:none
nameserver=4.4.4.41
```

指定多个网络接口

您可以通过设置多个 ip= 条目来指定多个网络接口。

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
ip=10.10.10.3::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none
```

配置默认网关和路由

可选：您可以通过设置 a rd.route= 值来配置到额外网络的路由。

注意

当您配置一个或多个网络时，需要一个默认网关。如果额外网络网关与主要网络网关不同，则默认网关必须是主要网络网关。

- 运行以下命令来配置默认网关：

```
ip=::10.10.10.254::::
```

- 输入以下命令为额外网络配置路由：

```
rd.route=20.20.20.0/24:20.20.20.254:enp2s0
```

在单个接口中禁用 DHCP

您可以在单一接口中禁用 DHCP，例如当有两个或者多个网络接口时，且只有一个接口被使用。在示例中，enp1s0 接口具有一个静态网络配置，而 enp2s0 禁用了 DHCP，不使用它：

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
ip=::::core0.example.com:enp2s0:none
```

合并 DHCP 和静态 IP 配置

您可以将系统上的 DHCP 和静态 IP 配置与多个网络接口合并，例如：

```
ip=enp1s0:dhcp
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none
```

在独立接口上配置 VLAN

可选： 您可以使用 vlan= 参数在单个接口上配置 VLAN。

- 要在网络接口中配置 VLAN 并使用静态 IP 地址，请运行以下命令：

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0.100:none
vlan=enp2s0.100:enp2s0
```

- 要在网络接口中配置 VLAN 并使用 DHCP，请运行以下命令：

```
ip=enp2s0.100:dhcp
vlan=enp2s0.100:enp2s0
```

提供多个 DNS 服务器

您可以通过为每个服务器添加一个 nameserver= 条目来提供多个 DNS 服务器，例如

```
nameserver=1.1.1.1
nameserver=8.8.8.8
```

将多个网络接口绑定到一个接口

可选：您可以使用 bond= 选项将多个网络接口绑定到一个接口。请参见以下示例：

- 配置绑定接口的语法为：bond=<name>[:<network_interfaces>][:options]

  <name> 是绑定设备名称 (bond0)、<network_interfaces> 代表以逗号分隔的物理（以太网）接口列表(em1,em2)，*options* 是用逗号分开的绑定选项列表。输入 modinfo bonding 查看可用选项。

- 当使用 bond= 创建绑定接口时，您必须指定如何分配 IP 地址以及绑定接口的其他信息。

  - 要将绑定接口配置为使用 DHCP，请将绑定的 IP 地址设置为 dhcp。例如：

    ```
    bond=bond0:em1,em2:mode=active-backup
    ip=bond0:dhcp
    ```

  - 要将绑定接口配置为使用静态 IP 地址，请输入您需要的特定 IP 地址和相关信息。例如：

```
bond=bond0:em1,em2:mode=active-backup,fail_over_mac=1
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0:none
```

在 active-backup 模式中始终设置 fail_over_mac=1 选项，以避免使用共享 OSA/RoCE 卡时出现问题。

将多个网络接口绑定到一个接口

可选：您可以使用 vlan= 参数并在绑定接口上配置 VLAN，并使用 DHCP，例如：

```
ip=bond0.100:dhcp
bond=bond0:em1,em2:mode=active-backup
vlan=bond0.100:bond0
```

使用以下示例配置带有 VLAN 的绑定接口并使用静态 IP 地址：

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0.100:none
bond=bond0:em1,em2:mode=active-backup
vlan=bond0.100:bond0
```

使用网络团队

可选：您可以使用 team= 参数来将网络团队用作绑定的替代选择：

- 配置组接口的语法为：team=name[:network_interfaces]

  name 是组设备名称(team0)，*network_interfaces* 代表以逗号分隔的物理（以太网）接口（em1、em2）列表。

注意

当 RHCOS 切换到即将推出的 RHEL 版本时，团队(team)功能被计划弃用。如需更多信息，请参阅红帽知识库文章。

使用以下示例配置网络团队：

```
team=team0:em1,em2
ip=team0:dhcp
```

18.2.14. 等待 bootstrap 过程完成

OpenShift Container Platform bootstrap 过程在集群节点首次引导到安装到磁盘的持久 RHCOS 环境后开始。通过 Ignition 配置文件提供的配置信息用于初始化 bootstrap 过程并在机器上安装 OpenShift Container Platform。您必须等待 bootstrap 过程完成。

先决条件

- 已为集群创建 Ignition 配置文件。
- 您已配置了适当的网络、DNS 和负载平衡基础架构。
- 已获得安装程序，并为集群生成 Ignition 配置文件。
- 已在集群机器上安装 RHCOS，并提供 OpenShift Container Platform 安装程序生成的 Ignition 配置文件。
- 您的机器可以直接访问互联网，或者有 HTTP 或 HTTPS 代理可用。

流程

1. 监控 bootstrap 过程：

```
$ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \ 1
  --log-level=info 2
```

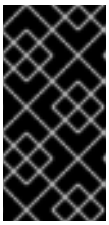
1. 对于 <installation_directory>，请指定安装文件保存到的目录的路径。
2. 要查看不同的安装详情，请指定 warn、debug 或 error，而不是 info。

输出示例

```
INFO Waiting up to 30m0s for the Kubernetes API at
https://api.test.example.com:6443...
INFO API v1.29.4 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
```

当 Kubernetes API 服务器提示已在 control plane 机器上引导它时，该命令会成功。

2. bootstrap 过程完成后，从负载均衡器中删除 bootstrap 机器。



重要

此时您必须从负载均衡器中删除 bootstrap 机器。您还可以删除或重新格式化 bootstrap 机器本身。

18.2.15. 使用 CLI 登录集群

您可以通过导出集群 kubeconfig 文件，以默认系统用户身份登录集群。kubeconfig 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。
- 已安装 oc CLI。

流程

1. 导出 kubeadmin 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 <installation_directory>，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 oc 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

18.2.16. 批准机器的证书签名请求

当您将机器添加到集群时，会为您添加的每台机器生成两个待处理证书签名请求(CSR)。您必须确认这些 CSR 已获得批准，或根据需要自行批准。必须首先批准客户端请求，然后批准服务器请求。

先决条件

- 您已将机器添加到集群中。

流程

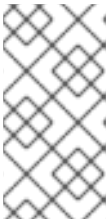
1. 确认集群可以识别这些机器：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0  Ready    master  63m  v1.29.4
master-1  Ready    master  63m  v1.29.4
master-2  Ready    master  64m  v1.29.4
```

输出中列出了您创建的所有机器。



注意

在有些 CSR 被批准前，前面的输出可能不包括计算节点（也称为 worker 节点）。

2. 检查待处理的 CSR，并确保添加到集群中的每台机器都有 Pending 或 Approved 状态的客户端请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE   REQUESTOR                                 CONDITION
csr-mddf5   20m   system:node:master-01.example.com   Approved,Issued
csr-z5rln   16m   system:node:worker-21.example.com   Approved,Issued
```

3. 如果 CSR 没有获得批准，在您添加的机器的所有待处理 CSR 都处于 Pending 状态后，请批准集群机器的 CSR：

注意

由于 CSR 会自动轮转，因此请在将机器添加到集群后一小时内批准您的 CSR。如果没有在一小时内批准它们，证书将会轮转，每个节点会存在多个证书。您必须批准所有这些证书。批准客户端 CSR 后，Kubelet 为服务证书创建一个二级 CSR，这需要手动批准。然后，如果 Kubelet 请求具有相同参数的新证书，则后续提供证书续订请求由 machine-approver 自动批准。

注意

对于在未启用机器 API 的平台上运行的集群，如裸机和其他用户置备的基础架构，您必须实施一种方法来自动批准 kubelet 提供证书请求(CSR)。如果没有批准请求，则 oc exec、ocrsh 和 oc logs 命令将无法成功，因为 API 服务器连接到 kubelet 时需要服务证书。与 Kubelet 端点联系的任何操作都需要此证书批准。该方法必须监视新的 CSR，确认 CSR 由 system: node 或 system:admin 组中的 node-bootstrapper 服务帐户提交，并确认节点的身份。

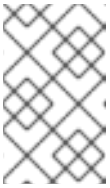
- 要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}
{{"\n"}}{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```



注意

在有些 CSR 被批准前，一些 Operator 可能无法使用。

4. 现在，您的客户端请求已被批准，您必须查看添加到集群中的每台机器的服务器请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE     REQUESTOR                                                   CONDITION
csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal       Pending
csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal       Pending
...
```

5. 如果剩余的 CSR 没有被批准，且处于 Pending 状态，请批准集群机器的 CSR：

- 要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1

<csr_name> 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

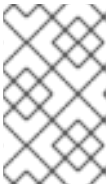
```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}
{{"\n"}}{{end}}{{end}}' | xargs oc adm certificate approve
```

6. 批准所有客户端和服务器 CSR 后，机器将 处于 Ready 状态。运行以下命令验证：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0  Ready    master  73m  v1.29.4
master-1  Ready    master  73m  v1.29.4
master-2  Ready    master  74m  v1.29.4
worker-0  Ready    worker  11m  v1.29.4
worker-1  Ready    worker  11m  v1.29.4
```



注意

批准服务器 CSR 后可能需要几分钟时间让机器过渡到 Ready 状态。

其他信息

- 如需有关 CSR 的更多信息，请参阅 证书签名请求。

18.2.17. 初始 Operator 配置

在 control plane 初始化后，您必须立即配置一些 Operator，以便它们都可用。

先决条件

- 您的 control plane 已初始化。

流程

1. 观察集群组件上线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION   AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.16.0    True        False         False      19m
baremetal                                  4.16.0    True        False         False      37m
cloud-credential                           4.16.0    True        False         False      40m
cluster-autoscaler                         4.16.0    True        False         False      37m
config-operator                            4.16.0    True        False         False      38m
console                                    4.16.0    True        False         False      26m
csi-snapshot-controller                    4.16.0    True        False         False      37m
dns                                        4.16.0    True        False         False      37m
etcd                                       4.16.0    True        False         False      36m
image-registry                             4.16.0    True        False         False      31m
ingress                                    4.16.0    True        False         False      30m
insights                                   4.16.0    True        False         False      31m
kube-apiserver                             4.16.0    True        False         False      26m
kube-controller-manager                    4.16.0    True        False         False      36m
kube-scheduler                             4.16.0    True        False         False      36m
kube-storage-version-migrator              4.16.0    True        False         False      37m
machine-api                                4.16.0    True        False         False      29m
machine-approver                           4.16.0    True        False         False      37m
machine-config                             4.16.0    True        False         False      36m
marketplace                                4.16.0    True        False         False      37m
monitoring                                 4.16.0    True        False         False      29m
network                                    4.16.0    True        False         False      38m
node-tuning                                4.16.0    True        False         False      37m
openshift-apiserver                        4.16.0    True        False         False      32m
openshift-controller-manager               4.16.0    True        False         False      30m
openshift-samples                          4.16.0    True        False         False      32m
operator-lifecycle-manager                 4.16.0    True        False         False      37m
operator-lifecycle-manager-catalog         4.16.0    True        False         False      37m
operator-lifecycle-manager-packageserver   4.16.0    True        False         False      32m
service-ca                                 4.16.0    True        False         False      38m
storage                                    4.16.0    True        False         False      37m
```

2. 配置不可用的 Operator。

18.2.17.1. 镜像 registry 存储配置

对于不提供默认存储的平台，Image Registry Operator 最初不可用。安装后，您必须将 registry 配置为使用存储，以便 Registry Operator 可用。

显示配置生产集群所需的持久性卷的说明。如果适用，显示有关将空目录配置为存储位置的说明，这仅适用于非生产集群。

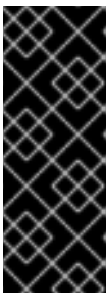
提供了在升级过程中使用 Recreate rollout 策略来允许镜像 registry 使用块存储类型的说明。

18.2.17.1.1. 为 IBM Z 配置 registry 存储

作为集群管理员，在安装后需要配置 registry 来使用存储。

先决条件

- 您可以使用具有 cluster-admin 角色的用户访问集群。
- 在 IBM Z® 上有一个集群。
- 您已为集群置备持久性存储，如 Red Hat OpenShift Data Foundation。



重要

当您只有一个副本时，OpenShift Container Platform 支持对镜像 registry 存储的 ReadWriteOnce 访问。ReadWriteOnce 访问还要求 registry 使用 Recreate rollout 策略。要部署支持高可用性的镜像 registry，需要两个或多个副本，ReadWriteMany 访问。

- 必须具有 100Gi 容量。

流程

1. 要将 registry 配置为使用存储，修改 configs.imageregistry/cluster 资源中的 spec.storage.pvc。

注意

使用共享存储时，请查看您的安全设置以防止外部访问。

2. 验证您没有 **registry pod:**

```
$ oc get pod -n openshift-image-registry -l docker-registry=default
```

输出示例

```
No resources found in openshift-image-registry namespace
```

注意

如果您的输出中有一个 registry pod，则不需要继续这个过程。

3. 检查 **registry** 配置：

```
$ oc edit configs.imageregistry.operator.openshift.io
```

输出示例

```
storage:
  pvc:
    claim:
```

将 claim 字段留空以允许自动创建 image-registry-storage PVC。

4. 检查 **clusteroperator** 状态：

```
$ oc get clusteroperator image-registry
```

输出示例

```
NAME             VERSION   AVAILABLE   PROGRESSING   DEGRADED   SINCE   MESSAGE
image-registry   4.16      True        False         False      6h50m
```

5. **确保 registry** 设置为 **managed，以启用镜像的构建和推送。**

- 运行：

```
$ oc edit configs.imageregistry/cluster
```

然后，更改行

```
managementState: Removed
```

至

```
managementState: Managed
```

18.2.17.1.2. 在非生产集群中为镜像 registry 配置存储

您必须为 Image Registry Operator 配置存储。对于非生产集群，您可以将镜像 registry 设置为空目录。如果您这样做，重启 registry 时会丢失所有镜像。

流程

- **将镜像 registry 存储设置为空目录：**

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch
'{"spec":{"storage":{"emptyDir":{}}}}'
```

警告

仅为非生产集群配置这个选项。

如果在 Image Registry Operator 初始化其组件前运行这个命令，oc patch 命令会失败并显示以下错误：

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster"
not found
```

等待几分钟，然后再次运行 命令。

18.2.18. 在用户置备的基础架构上完成安装

完成 Operator 配置后，可以在您提供的基础架构上完成集群安装。

先决条件

- 您的 control plane 已初始化。
- 已完成初始 Operator 配置。

流程

1. 使用以下命令确认所有集群组件都在线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION   AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.16.0    True        False         False      19m
baremetal                                  4.16.0    True        False         False      37m
cloud-credential                           4.16.0    True        False         False      40m
cluster-autoscaler                         4.16.0    True        False         False      37m
config-operator                            4.16.0    True        False         False      38m
console                                    4.16.0    True        False         False      26m
csi-snapshot-controller                    4.16.0    True        False         False      37m
dns                                        4.16.0    True        False         False      37m
etcd                                       4.16.0    True        False         False      36m
image-registry                             4.16.0    True        False         False      31m
ingress                                    4.16.0    True        False         False      30m
insights                                   4.16.0    True        False         False      31m
kube-apiserver                             4.16.0    True        False         False      26m
kube-controller-manager                    4.16.0    True        False         False      36m
kube-scheduler                             4.16.0    True        False         False      36m
kube-storage-version-migrator              4.16.0    True        False         False      37m
machine-api                                4.16.0    True        False         False      29m
machine-approver                           4.16.0    True        False         False      37m
machine-config                             4.16.0    True        False         False      36m
marketplace                                4.16.0    True        False         False      37m
monitoring                                 4.16.0    True        False         False      29m
network                                    4.16.0    True        False         False      38m
node-tuning                                4.16.0    True        False         False      37m
openshift-apiserver                        4.16.0    True        False         False      32m
openshift-controller-manager               4.16.0    True        False         False      30m
openshift-samples                          4.16.0    True        False         False      32m
operator-lifecycle-manager                 4.16.0    True        False         False      37m
operator-lifecycle-manager-catalog         4.16.0    True        False         False      37m
operator-lifecycle-manager-packageserver   4.16.0    True        False         False      32m
service-ca                                 4.16.0    True        False         False      38m
storage                                    4.16.0    True        False         False      37m
```

另外，当所有集群都可用时，以下命令会通知您。它还检索并显示凭证：

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

1

对于 <installation_directory>，请指定安装文件保存到的目录的路径。

输出示例

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

Cluster Version Operator 完成从 Kubernetes API 服务器部署 OpenShift Container Platform 集群时，该命令会成功。

重要

- 安装程序生成的 Ignition 配置文件包含 24 小时后过期的证书，然后在该时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 node-bootstrapper 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。

- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

2. 确认 Kubernetes API 服务器正在与 pod 通信。

   a. 要查看所有 pod 的列表，请使用以下命令：

```
$ oc get pods --all-namespaces
```

输出示例

```
NAMESPACE                  NAME                                        READY   STATUS
RESTARTS   AGE
openshift-apiserver-operator    openshift-apiserver-operator-85cb746d55-zqhs8
1/1   Running   1          9m
openshift-apiserver        apiserver-67b9g                             1/1     Running     0
3m
openshift-apiserver        apiserver-ljcmx                             1/1     Running     0
1m
openshift-apiserver        apiserver-z25h4                             1/1     Running     0
2m
```

```
openshift-authentication-operator authentication-operator-69d5d8bf84-vh2n8
1/1     Running     0          5m
...
```

b. 使用以下命令，查看上一命令的输出中所列 pod 的日志：

```
$ oc logs <pod_name> -n <namespace>
```

1 指定 pod 名称和命名空间，如上一命令的输出中所示。

如果 pod 日志显示，Kubernetes API 服务器可以与集群机器通信。

3. 对于使用光纤通道协议(FCP)的安装，还需要额外的步骤才能启用多路径。不要在安装过程中启用多路径。

如需更多信息，请参阅 *安装后机器配置任务* 文档中的"使用 RHCOS 上使用内核参数启用多路径"。

验证

如果您在 OpenShift Container Platform bootstrap 过程中启用了安全引导，则需要以下验证步骤：

1. 运行以下命令来调试节点：

```
$ oc debug node/<node_name>
chroot /host
```

2. 运行以下命令确认启用了安全引导：

```
$ cat /sys/firmware/ipl/secure
```

输出示例

1 1

1

如果启用了安全引导，则值为 1，如果未启用安全引导，则值为 0。

18.2.19. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于 远程健康监控
- 如何在没有 SSH 的情况下在 OpenShift4 节点中生成 SOSREPORT。

18.2.20. 后续步骤

- 在 RHCOS 上启用带有内核参数的多路径。
- 自定义集群。
- 如果需要，您可以选择 不使用远程健康报告。

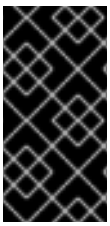
18.3. 在受限网络中的 IBM Z 和 IBM LINUXONE 中使用 Z/VM 安装集群

在 OpenShift Container Platform 版本 4.16 中，您可以在受限网络中置备的 IBM Z® 或 IBM®

LinuxONE 基础架构上安装集群。

注意

虽然本文档只涉及 IBM Z®，但它的所有信息也适用于 IBM® LinuxONE。

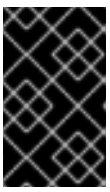


重要

非裸机平台还有其他注意事项。在安装 OpenShift Container Platform 集群前，请参阅有关在未经测试的平台上部署 OpenShift Container Platform 的指南 中的信息。

18.3.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读有关 选择集群安装方法的文档，并为用户准备它。
- 您 创建了用于在受限网络中安装的镜像 registry，并获取您的 OpenShift Container Platform 版本的 imageContentSources 数据。
- 在开始安装过程前，您必须移动或删除任何现有的安装文件。这样可确保在安装过程中创建和更新所需的安装文件。

  重要

  确保从可访问安装介质的机器中执行安装步骤。

- 已为集群置备了 使用 OpenShift Data Foundation 或其他支持的存储协议的持久性存储。要部署私有镜像 registry，您必须使用 ReadWriteMany 访问设置持久性存储。
- 如果您使用防火墙并计划使用 Telemetry 服务，则将防火墙配置为允许集群需要访问的站点。

注意

如果要配置代理，请务必查看此站点列表。

18.3.2. 关于在受限网络中安装

在 OpenShift Container Platform 4.16 中，可以执行不需要有效的互联网连接来获取软件组件的安装。受限网络安装可以使用安装程序置备的基础架构或用户置备的基础架构完成，具体取决于您要安装集群的云平台。

如果您选择在云平台中执行受限网络安装，您仍需要访问其云 API。有些云功能，比如 Amazon Web Service 的 Route 53 DNS 和 IAM 服务，需要访问互联网。根据您的网络，在裸机硬件、Nutanix 或 VMware vSphere 上安装可能需要较少的互联网访问。

要完成受限网络安装，您必须创建一个 registry，以镜像 OpenShift 镜像 registry 的内容并包含安装介质。您可以在镜像主机上创建此 registry，该主机可同时访问互联网和您的封闭网络，也可以使用满足您的限制条件的其他方法。

重要

由于用户置备安装配置的复杂性，在尝试使用用户置备的基础架构受限网络安装前，请考虑完成标准用户置备的基础架构安装。完成此测试安装后，您可以更轻松地隔离和排除在受限网络中安装过程中可能出现的任何问题。

18.3.2.1. 其他限制

受限网络中的集群有以下额外限制和限制：

- ClusterVersion 状态包含一个 Unable to retrieve available updates 错误。
- 默认情况下，您无法使用 Developer Catalog 的内容，因为您无法访问所需的镜像流标签。

18.3.3. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来获得用来安装集群的镜像。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

18.3.4. 具有用户置备基础架构的集群的要求

对于包含用户置备的基础架构的集群，您必须部署所有所需的机器。

本节论述了在用户置备的基础架构上部署 OpenShift Container Platform 的要求。

18.3.4.1. 集群安装所需的机器

最小的 OpenShift Container Platform 集群需要以下主机：

表 18.21. 最低所需的主机

主机	描述
一个临时 bootstrap 机器	集群需要 bootstrap 机器在三台 control plane 机器上部署 OpenShift Container Platform 集群。您可在安装集群后删除 bootstrap 机器。
三台 control plane 机器	control plane 机器运行组成 control plane 的 Kubernetes 和 OpenShift Container Platform 服务。
至少两台计算机器，也称为 worker 机器。	OpenShift Container Platform 用户请求的工作负载在计算机器上运行。

重要

要提高集群的高可用性，请在至少两台物理机器的不同 z/VM 实例中分发 control plane 机器。

bootstrap、control plane 和计算机器必须使用 Red Hat Enterprise Linux CoreOS(RHCOS)作为操作系统。

请注意，RHCOS 基于 Red Hat Enterprise Linux(RHEL) 9.2，并继承其所有硬件认证和要求。查看红帽企业 Linux 技术功能和限制。

18.3.4.2. 集群安装的最低资源要求

每台集群机器都必须满足以下最低要求：

表 18.22. 最低资源要求

机器	操作系统	vCPU [1]	虚拟内存	Storage	每秒输入/输出(IOPS)
bootstrap	RHCOS	4	16 GB	100 GB	N/A
Control plane（控制平面）	RHCOS	4	16 GB	100 GB	N/A
Compute	RHCOS	2	8 GB	100 GB	N/A

1. 当启用 SMT-2 时，一个物理核心(IFL)提供两个逻辑核心（线程）。管理程序可以提供两个或多个 vCPU。

注意

从 OpenShift Container Platform 版本 4.13 开始，RHCOS 基于 RHEL 版本 9.2，它更新了微架构要求。以下列表包含每个架构需要的最小指令集架构 (ISA)：

- x86-64 体系结构需要 x86-64-v2 ISA
- ARM64 架构需要 ARMv8.0-A ISA
- IBM Power 架构需要 Power 9 ISA
- s390x 架构需要 z14 ISA

如需更多信息，请参阅 RHEL 架构。

如果平台的实例类型满足集群机器的最低要求，则 OpenShift Container Platform 支持使用它。

其他资源

- 优化存储

18.3.4.3. 最低 IBM Z 系统环境

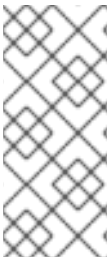
您可以在以下 IBM® 硬件上安装 OpenShift Container Platform 版本 4.16：

- IBM® z16 （所有型号）、IBM® z15 （所有型号）、IBM® z14 （所有型号）
- IBM® LinuxONE 4 （所有型号）、IBM® LinuxONE （所有型号）、IBM® LinuxONE Emperor II、IBM® LinuxONE Rockhopper II

硬件要求

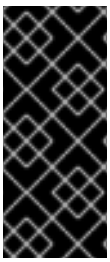
- Linux(IFL)等效的、启用了 SMT2 的 Linux 集成设施，每个群集都启用了 SMT2。

- 至少一个网络连接连接到 LoadBalancer 服务，并为集群外的流量提供数据。



注意

您可以使用专用或共享的 IFL 来分配足够的计算资源。资源共享是 IBM Z® 的关键优势之一。但是，您必须正确调整每个虚拟机监控程序层上的容量，并确保每个 OpenShift Container Platform 集群都有足够资源。



重要

由于集群的整体性能会受到影响，用于设置 OpenShift Container Platform 集群的 LPAR 必须提供足够的计算容量。就此而言，管理程序级别上的 LPAR 权重管理、授权和 CPU 共享扮演着重要角色。

操作系统要求

- 一个 z/VM 7.2 或更高版本实例

在 z/VM 实例中设置：

- 用于 OpenShift Container Platform control plane 机器的三台客户机虚拟机
- 用于 OpenShift Container Platform 计算机器的两个客户机虚拟机
- 一个客户虚拟机作为临时 OpenShift Container Platform bootstrap 机器

IBM Z 网络连接要求

要在 z/VM 中安装 IBM Z®，您需要使用第 2 层模式的单一 z/VM 虚拟 NIC。您还需要：

- 直接附加的 OSA 或 RoCE 网络适配器
- z/VM VSwitch 设置。对于首选设置，请使用 OSA 链接聚合。

磁盘存储

- 附加了 FICON 的磁盘存储(DASD)。可以是 z/VM Minidisks、fullpack Minidisks 或专用 DASD，它们都必须格式化为 CDL（默认 CDL）。要达到 Red Hat Enterprise Linux CoreOS(RHCOS)安装所需的最小 DASD 大小，您需要扩展地址卷(EAV)。如果可用，请使用 HyperPAV 来确保最佳性能。

- FCP 连接的磁盘存储

存储/主内存

- OpenShift Container Platform control plane 机器需要 16 GB

- OpenShift Container Platform 计算机器需要 8 GB

- 临时 OpenShift Container Platform bootstrap 机器需要 16 GB

18.3.4.4. 首选 IBM Z 系统环境

硬件要求

- 三个 LPARS，每个都相当于 6 个 IFL（每个集群启用了 SMT2）。

- 两个网络连接连接到 LoadBalancer 服务，并为集群外的流量提供数据。

- HiperSockets 作为设备直接附加到节点，或者通过使用一个 z/VM VSWITCH 进行桥接来对 z/VM 客户机进行透明连接。要将 HiperSockets 直接连接到节点，您必须通过 RHEL 8 客户机设置到外部网络的网关来桥接到 HiperSockets 网络。

操作系统要求

- 两个或多个 z/VM 7.2 或更高版本实例以实现高可用性

在 z/VM 实例中设置：

- 用于 OpenShift Container Platform control plane 机器的三台客户虚拟机，每个 z/VM 实

例一个。

- 至少 6 个用于 OpenShift Container Platform 计算机器的虚拟机，分布在 z/VM 实例中。
- 一个客户虚拟机作为临时 OpenShift Container Platform bootstrap 机器。
- 要确保在过量使用的环境中集成组件可用，使用 CP 命令 SET SHARE 增加 control plane 的优先级。如果存在基础架构节点，则对它们执行相同的操作。请参阅 IBM® 文档中的 SET SHARE。

IBM Z 网络连接要求

要在 z/VM 中安装 IBM Z®，您需要使用第 2 层模式的单一 z/VM 虚拟 NIC。您还需要：

- 直接附加的 OSA 或 RoCE 网络适配器
- z/VM VSwitch 设置。对于首选设置，请使用 OSA 链接聚合。

磁盘存储

- 附加了 FICON 的磁盘存储(DASD)。可以是 z/VM Minidisks、fullpack Minidisks 或专用 DASD，它们都必须格式化为 CDL（默认 CDL）。要达到 Red Hat Enterprise Linux CoreOS(RHCOS)安装所需的最小 DASD 大小，您需要扩展地址卷(EAV)。如果可用，请使用 HyperPAV 来确保最佳性能。
- FCP 连接的磁盘存储

存储/主内存

- OpenShift Container Platform control plane 机器需要 16 GB
- OpenShift Container Platform 计算机器需要 8 GB
- 临时 OpenShift Container Platform bootstrap 机器需要 16 GB

18.3.4.5. 证书签名请求管理

在使用您置备的基础架构时，集群只能有限地访问自动机器管理，因此您必须提供一种在安装后批准集群证书签名请求 (CSR) 的机制。kube-controller-manager 只能批准 kubelet 客户端 CSR。machine-approver 无法保证使用 kubelet 凭证请求的提供证书的有效性，因为它不能确认是正确的机器发出了该请求。您必须决定并实施一种方法，以验证 kubelet 提供证书请求的有效性并进行批准。

其他资源

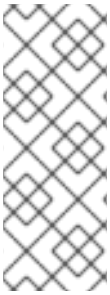
- 请参阅 IBM® 文档中的使用 z/VM 虚拟交换机桥接 HiperSockets LAN。
- 请参阅在 z/VM 上的 Linux 客户端上扩展 HyperPAV 别名设备 以获得性能优化。
- 有关 LPAR 权重管理和权利的信息，请参阅 LPAR 性能的主题。
- IBM Z® 和 IBM® LinuxONE 环境的推荐主机实践

18.3.4.6. 用户置备的基础架构对网络的要求

所有 Red Hat Enterprise Linux CoreOS(RHCOS)机器都需要在启动时在 initramfs 中配置联网，以获取它们的 Ignition 配置文件。

在初次启动过程中，机器需要 IP 地址配置，该配置通过 DHCP 服务器或静态设置，提供所需的引导选项。建立网络连接后，机器会从 HTTP 或 HTTPS 服务器下载 Ignition 配置文件。然后，Ignition 配置文件用于设置每台机器的确切状态。Machine Config Operator 在安装后完成对机器的更多更改，如应用新证书或密钥。

建议使用 DHCP 服务器对集群机器进行长期管理。确保 DHCP 服务器已配置为向集群机器提供持久的 IP 地址、DNS 服务器信息和主机名。



注意

如果用户置备的基础架构没有 DHCP 服务，您可以在 RHCOS 安装时向节点提供 IP 网络配置和 DNS 服务器地址。如果要从 ISO 镜像安装，这些参数可作为引导参数传递。如需有关静态 IP 置备和高级网络选项的更多信息，请参阅 *安装 RHCOS 并启动 OpenShift Container Platform bootstrap 过程* 部分。

Kubernetes API 服务器必须能够解析集群机器的节点名称。如果 API 服务器和 worker 节点位于不同的区域中，您可以配置默认 DNS 搜索区域，以允许 API 服务器解析节点名称。另一种支持的方法是始终通过节点对象和所有 DNS 请求中的完全限定域名引用主机。

18.3.4.6.1. 通过 DHCP 设置集群节点主机名

在 Red Hat Enterprise Linux CoreOS(RHCOS)机器上，主机名是通过 NetworkManager 设置的。默认情况下，机器通过 DHCP 获取其主机名。如果主机名不是由 DHCP 提供，请通过内核参数或者其它方法进行静态设置，请通过反向 DNS 查找获取。反向 DNS 查找在网络初始化后进行，可能需要一些时间来解决。其他系统服务可以在此之前启动，并将主机名检测为 localhost 或类似的内容。您可以使用 DHCP 为每个集群节点提供主机名来避免这种情况。

另外，通过 DHCP 设置主机名可以绕过实施 DNS split-horizon 的环境中的手动 DNS 记录名称配置错误。

18.3.4.6.2. 网络连接要求

您必须配置机器之间的网络连接，以允许 OpenShift Container Platform 集群组件进行通信。每台机器都必须能够解析集群中所有其他机器的主机名。

本节详细介绍了所需的端口。

表 18.23. 用于全机器到所有机器通信的端口

协议	port	描述
ICMP	N/A	网络可访问性测试
TCP	**1936**	指标
	9000-**9999**	主机级别的服务，包括端口 9**100**-**9101** 上的节点导出器，以及端口 **9099** 上的 Cluster Version Operator。
	10250-**10259**	Kubernetes 保留的默认端口
UDP	**4789**	VXLAN
	6081	Geneve
	9000-**9999**	主机级别的服务，包括端口 **9100**到**9101** 上的节点导出器。
	500	IPsec IKE 数据包

协议	port	描述
	4500	IPsec NAT-T 数据包
	123	UDP 端口 **123** 上的网络时间协议 (NTP) 如果配置了外部 NTP 时间服务器，需要打开 UDP 端口 **123**。
TCP/UDP	**30000-32767**	Kubernetes 节点端口
ESP	N/A	IPsec Encapsulating Security Payload(ESP)

表 18.24. 用于所有机器控制平面通信的端口

协议	port	描述
TCP	**6443**	Kubernetes API

表 18.25. control plane 机器用于 control plane 机器通信的端口

协议	port	描述
TCP	**2379-2380**	etcd 服务器和对等端口

用户置备的基础架构的 NTP 配置

OpenShift Container Platform 集群被配置为默认使用公共网络时间协议(NTP)服务器。如果要使用本地企业 NTP 服务器，或者集群部署在断开连接的网络中，您可以将集群配置为使用特定的时间服务器。如需更多信息，*请参阅配置 chrony 时间服务* 的文档。

其他资源

- **配置 chrony 时间服务**

18.3.4.7. 用户置备的 DNS 要求

在 OpenShift Container Platform 部署中，以下组件需要 DNS 名称解析：

- **The Kubernetes API**

- **OpenShift Container Platform 应用程序通配符**
- **bootstrap、control plane 和计算机器**

Kubernetes API、bootstrap 机器、control plane 机器和计算机器也需要反向 DNS 解析。

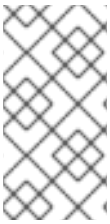
DNS A/AAAA 或 CNAME 记录用于名称解析，PTR 记录用于反向名称解析。反向记录很重要，因为 Red Hat Enterprise Linux CoreOS(RHCOS)使用反向记录为所有节点设置主机名，除非 DHCP 提供主机名。另外，反向记录用于生成 OpenShift Container Platform 需要操作的证书签名请求(CSR)。

用户置备的 OpenShift Container Platform 集群需要以下 DNS 记录，这些记录必须在安装前就位。在每个记录中，<cluster_name> 是集群名称，<base_domain> 是您在 install-config.yaml 文件中指定的基域。完整的 DNS 记录采用以下形式： <component>.<cluster_name>.<base_domain>.。

表 18.26. 所需的 DNS 记录

组件	记录	描述
Kubernetes API	**api.<cluster_name>.<base_domain>.**	DNS A/AAAA 或 CNAME 记录，以及用于标识 API 负载均衡器的 DNS PTR 记录。这些记录必须由集群外的客户端和集群中的所有节点解析。
	api-int.<cluster_name>.<base_domain>.	DNS A/AAAA 或 CNAME 记录，以及用于内部标识 API 负载均衡器的 DNS PTR 记录。这些记录必须可以从集群中的所有节点解析。  **重要** API 服务器必须能够根据 Kubernetes 中记录的主机名解析 worker 节点。如果 API 服务器无法解析节点名称，则代理的 API 调用会失败，且您无法从 pod 检索日志。
Routes	***.apps.<cluster_name>.<base_domain>.**	通配符 DNS A/AAAA 或 CNAME 记录，指向应用程序入口负载均衡器。应用程序入口负载均衡器以运行 Ingress Controller Pod 的机器为目标。默认情况下，Ingress Controller Pod 在计算机器上运行。这些记录必须由集群外的客户端和集群中的所有节点解析。 例如，console **-openshift-console.apps.<cluster_name>.<base_domain>** 用作到 OpenShift Container Platform 控制台的通配符路由。

组件	记录	描述
bootstrap 机器	**bootstrap.<cluster_name>.<base_domain>.**	DNS A/AAAA 或 CNAME 记录，以及用于标识 bootstrap 机器的 DNS PTR 记录。这些记录必须由集群中的节点解析。
control plane 机器	**<control_plane><n>.<cluster_name>.<base_domain>.**	DNS A/AAAA 或 CNAME 记录，以识别 control plane 节点的每台机器。这些记录必须由集群中的节点解析。
计算机器	**<compute><n>.<cluster_name>.<base_domain>.**	DNS A/AAAA 或 CNAME 记录，用于识别 worker 节点的每台机器。这些记录必须由集群中的节点解析。



注意

在 OpenShift Container Platform 4.4 及更新的版本中，您不需要在 DNS 配置中指定 etcd 主机和 SRV 记录。

提示

您可以使用 dig 命令验证名称和反向名称解析。如需了解详细的 *验证步骤，请参阅为用户置备的基础架构验证 DNS 解析* 一节。

18.3.4.7.1. 用户置备的集群的 DNS 配置示例

本节提供 A 和 PTR 记录配置示例，它们满足了在用户置备的基础架构上部署 OpenShift Container Platform 的 DNS 要求。样本不是为选择一个 DNS 解决方案提供建议。

在这个示例中，集群名称为 ocp4，基域是 example.com。

用户置备的集群的 DNS A 记录配置示例

以下示例是 BIND 区域文件，其中显示了用户置备的集群中名称解析的 A 记录示例。

例 18.4. DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
  2019070700 ; serial
  3H  ; refresh (3 hours)
  30M  ; retry (30 minutes)
```

```
    2W  ; expiry (2 weeks)
    1W ) ; minimum (1 week)
  IN NS ns1.example.com.
  IN MX 10 smtp.example.com.
;
;
ns1.example.com.  IN A 192.168.1.5
smtp.example.com.  IN A 192.168.1.5
;
helper.example.com.  IN A 192.168.1.5
helper.ocp4.example.com. IN A 192.168.1.5
;
api.ocp4.example.com.  IN A 192.168.1.5 1
api-int.ocp4.example.com. IN A 192.168.1.5 2
;
*.apps.ocp4.example.com. IN A 192.168.1.5 3
;
bootstrap.ocp4.example.com. IN A 192.168.1.96 4
;
control-plane0.ocp4.example.com. IN A 192.168.1.97 5
control-plane1.ocp4.example.com. IN A 192.168.1.98 6
control-plane2.ocp4.example.com. IN A 192.168.1.99 7
;
compute0.ocp4.example.com. IN A 192.168.1.11 8
compute1.ocp4.example.com. IN A 192.168.1.7 9
;
;EOF
```

1

为 Kubernetes API 提供名称解析。记录引用 API 负载均衡器的 IP 地址。

2

为 Kubernetes API 提供名称解析。记录引用 API 负载均衡器的 IP 地址，用于内部集群通信。

3

为通配符路由提供名称解析。记录引用应用程序入口负载均衡器的 IP 地址。应用程序入口负载均衡器以运行 Ingress Controller Pod 的机器为目标。默认情况下，Ingress Controller Pod 在计算机器上运行。

注意

在这个示例中，将相同的负载均衡器用于 Kubernetes API 和应用入口流量。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。

4

为 bootstrap 机器提供名称解析。

5 6 7

为 control plane 机器提供名称解析。

8 9

为计算机器提供名称解析。

用户置备的集群的 DNS PTR 记录配置示例

以下示例 BIND 区域文件显示了用户置备的集群中反向名称解析的 PTR 记录示例。

例 18.5. 反向记录的 DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
  2019070700 ; serial
  3H  ; refresh (3 hours)
  30M  ; retry (30 minutes)
  2W  ; expiry (2 weeks)
  1W )  ; minimum (1 week)
 IN NS ns1.example.com.
;
5.1.168.192.in-addr.arpa. IN PTR api.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. IN PTR api-int.ocp4.example.com. 2
;
96.1.168.192.in-addr.arpa. IN PTR bootstrap.ocp4.example.com. 3
;
97.1.168.192.in-addr.arpa. IN PTR control-plane0.ocp4.example.com. 4
98.1.168.192.in-addr.arpa. IN PTR control-plane1.ocp4.example.com. 5
99.1.168.192.in-addr.arpa. IN PTR control-plane2.ocp4.example.com. 6
;
11.1.168.192.in-addr.arpa. IN PTR compute0.ocp4.example.com. 7
7.1.168.192.in-addr.arpa. IN PTR compute1.ocp4.example.com. 8
;
;EOF
```

1

为 Kubernetes API 提供反向 DNS 解析。PTR 记录引用 API 负载均衡器的记录名称。

2

3

为 bootstrap 机器提供反向 DNS 解析。

4 5 6

为 control plane 机器提供反向 DNS 解析。

7 8

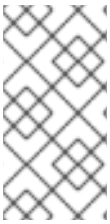
为计算机器提供反向 DNS 解析。

注意

OpenShift Container Platform 应用程序通配符不需要 PTR 记录。

18.3.4.8. 用户置备的基础架构的负载均衡要求

在安装 OpenShift Container Platform 前，您必须置备 API 和应用程序入口负载均衡基础架构。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。



注意

如果要使用 Red Hat Enterprise Linux (RHEL) 实例部署 API 和应用程序入口负载均衡器，您必须单独购买 RHEL 订阅。

负载平衡基础架构必须满足以下要求：

1. API 负载均衡器 ：提供一个通用端点，供用户（包括人工和机器）与平台交互和配置。配置以下条件：

   - 仅第 4 层负载均衡.这可被称为 Raw TCP 或 SSL Passthrough 模式。

- 无状态负载平衡算法。这些选项根据负载均衡器的实施而有所不同。

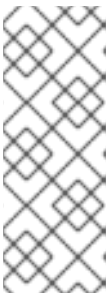
重要

不要为 API 负载均衡器配置会话持久性。为 Kubernetes API 服务器配置会话持久性可能会导致出现过量 OpenShift Container Platform 集群应用程序流量，以及过量的在集群中运行的 Kubernetes API。

在负载均衡器的前端和后端配置以下端口：

表 18.27. API 负载均衡器

port	后端机器（池成员）	internal	外部	描述
6443	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。您必须为 API 服务器健康检查探测配置 **/readyz** 端点。	X	X	Kubernetes API 服务器
22623	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。	X		机器配置服务器



注意

负载均衡器必须配置为，从 API 服务器关闭 /readyz 端点到从池中移除 API 服务器实例时最多需要 30 秒。在 /readyz 返回错误或健康后的时间范围内，端点必须被删除或添加。每 5 秒或 10 秒探测一次，有两个成功请求处于健康状态，三个成为不健康的请求是经过良好测试的值。

2. 应用程序入口负载均衡器 ：为应用程序流量从集群外部流提供入口点。OpenShift Container Platform 集群需要正确配置入口路由器。

配置以下条件：

- 仅第 4 层负载均衡.这可被称为 Raw TCP 或 SSL Passthrough 模式。
-

建议根据可用选项以及平台上托管的应用程序类型，使用基于连接的或基于会话的持久性。

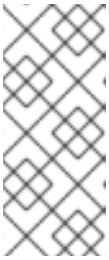
提示

如果应用程序入口负载均衡器可以看到客户端的真实 IP 地址，启用基于 IP 的会话持久性可以提高使用端到端 TLS 加密的应用程序的性能。

在负载均衡器的前端和后端配置以下端口：

表 18.28. 应用程序入口负载均衡器

port	后端机器（池成员）	internal	外部	描述
443	默认情况下，运行 Ingress Controller Pod、计算或 worker 的机器。	X	X	HTTPS 流量
80	默认情况下，运行 Ingress Controller Pod、计算或 worker 的机器。	X	X	HTTP 流量



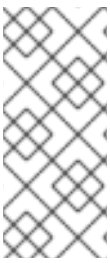
注意

如果要部署一个带有零计算节点的三节点集群，Ingress Controller Pod 在 control plane 节点上运行。在三节点集群部署中，您必须配置应用程序入口负载均衡器，将 HTTP 和 HTTPS 流量路由到 control plane 节点。

18.3.4.8.1. 用户置备的集群的负载均衡器配置示例

本节提供了一个满足用户置备集群的负载均衡要求的 API 和应用程序入口负载均衡器配置示例。示例是 HAProxy 负载均衡器的 /etc/haproxy/haproxy.cfg 配置。这个示例不是为选择一个负载平衡解决方案提供建议。

在这个示例中，将相同的负载均衡器用于 Kubernetes API 和应用入口流量。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。



注意

如果您使用 HAProxy 作为负载均衡器，并且 SELinux 设置为 enforcing，您必须通过运行 setsebool -P haproxy_connect_any=1 来确保 HAProxy 服务可以绑定到配置的 TCP 端口。

例 18.6. API 和应用程序入口负载均衡器配置示例

```
global
  log         127.0.0.1 local2
  pidfile     /var/run/haproxy.pid
  maxconn     4000
  daemon
defaults
  mode                    http
  log                     global
  option                  dontlognull
  option http-server-close
  option                  redispatch
  retries                 3
  timeout http-request    10s
  timeout queue           1m
  timeout connect         10s
  timeout client          1m
  timeout server          1m
  timeout http-keep-alive 10s
  timeout check           10s
  maxconn                 3000
listen api-server-6443 (1)
  bind *:6443
  mode tcp
  option  httpchk GET /readyz HTTP/1.0
  option  log-health-checks
  balance roundrobin
  server bootstrap bootstrap.ocp4.example.com:6443 verify none check check-ssl inter 10s
fall 2 rise 3 backup (2)
  server master0 master0.ocp4.example.com:6443 weight 1 verify none check check-ssl
inter 10s fall 2 rise 3
  server master1 master1.ocp4.example.com:6443 weight 1 verify none check check-ssl
inter 10s fall 2 rise 3
  server master2 master2.ocp4.example.com:6443 weight 1 verify none check check-ssl
inter 10s fall 2 rise 3
listen machine-config-server-22623 (3)
  bind *:22623
  mode tcp
  server bootstrap bootstrap.ocp4.example.com:22623 check inter 1s backup (4)
  server master0 master0.ocp4.example.com:22623 check inter 1s
  server master1 master1.ocp4.example.com:22623 check inter 1s
  server master2 master2.ocp4.example.com:22623 check inter 1s
listen ingress-router-443 (5)
  bind *:443
  mode tcp
  balance source
  server compute0 compute0.ocp4.example.com:443 check inter 1s
  server compute1 compute1.ocp4.example.com:443 check inter 1s
listen ingress-router-80 (6)
  bind *:80
  mode tcp
  balance source
  server compute0 compute0.ocp4.example.com:80 check inter 1s
  server compute1 compute1.ocp4.example.com:80 check inter 1s
```

1

端口 6443 处理 Kubernetes API 流量并指向 control plane 机器。

2 4

bootstrap 条目必须在 OpenShift Container Platform 集群安装前就位，且必须在 bootstrap 过程完成后删除它们。

3

端口 22623 处理机器配置服务器流量并指向 control plane 机器。

5

端口 443 处理 HTTPS 流量，并指向运行 Ingress Controller pod 的机器。默认情况下，Ingress Controller Pod 在计算机器上运行。

6

端口 80 处理 HTTP 流量，并指向运行 Ingress Controller pod 的机器。默认情况下，Ingress Controller Pod 在计算机器上运行。

注意

如果要部署一个带有零计算节点的三节点集群，Ingress Controller Pod 在 control plane 节点上运行。在三节点集群部署中，您必须配置应用程序入口负载均衡器，将 HTTP 和 HTTPS 流量路由到 control plane 节点。

提示

如果您使用 HAProxy 作为负载均衡器，您可以通过在 HAProxy 节点上运行 netstat -nltupe 来检查 haproxy 进程是否在侦听端口 6443、22623、443 和 80。

18.3.5. 准备用户置备的基础架构

在用户置备的基础架构上安装 OpenShift Container Platform 之前，您必须准备底层基础架构。

本节详细介绍了设置集群基础架构以准备 OpenShift Container Platform 安装所需的高级别步骤。这包括为集群节点配置 IP 网络和网络连接，为 Ignition 文件准备 Web 服务器，通过防火墙启用所需的端

口，以及设置所需的 DNS 和负载均衡基础架构。

准备后，集群基础架构必须满足 *带有用户置备的基础架构部分的集群要求*。

先决条件

- 您已参阅 OpenShift Container Platform 4.x Tested Integrations 页面。
- 您已查看了 *具有用户置备基础架构的集群要求部分中详述的基础架构* 要求。

流程

1. 设置静态 IP 地址。
2. 设置 HTTP 或 HTTPS 服务器，为集群节点提供 Ignition 文件。
3. 确保您的网络基础架构提供集群组件之间所需的网络连接。有关 *要求的详情，请参阅用户置备的基础架构* 的网络要求部分。
4. 将防火墙配置为启用 OpenShift Container Platform 集群组件进行通信所需的端口。*如需有关所需端口的详细信息，请参阅用户置备的基础架构* 部分的网络要求。

重要

默认情况下，OpenShift Container Platform 集群可以访问端口 1936，因为每个 control plane 节点都需要访问此端口。

避免使用 Ingress 负载均衡器公开此端口，因为这样做可能会导致公开敏感信息，如统计信息和指标（与 Ingress Controller 相关的统计信息和指标）。

5. 为集群设置所需的 DNS 基础架构。
   a. 为 Kubernetes API、应用程序通配符、bootstrap 机器、control plane 机器和计算机

器配置 DNS 名称解析。

b. 为 Kubernetes API、bootstrap 机器、control plane 机器和计算机器配置反向 DNS 解析。

如需有关 *OpenShift Container Platform DNS 要求的更多信息，请参阅用户置备* DNS 要求部分。

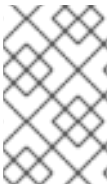
6. 验证您的 DNS 配置。

a. 从安装节点，针对 Kubernetes API 的记录名称、通配符路由和集群节点运行 DNS 查找。验证响应中的 IP 地址是否与正确的组件对应。

b. 从安装节点，针对负载均衡器和集群节点的 IP 地址运行反向 DNS 查找。验证响应中的记录名称是否与正确的组件对应。

有关详细的 *DNS 验证步骤，请参阅用户置备的基础架构* 验证 DNS 解析部分。

7. 置备所需的 API 和应用程序入口负载平衡基础架构。有关 *要求的更多信息，请参阅用户置备的基础架构的负载平衡* 要求部分。



注意

某些负载平衡解决方案要求在初始化负载平衡之前，对群集节点进行 DNS 名称解析。

18.3.6. 验证用户置备的基础架构的 DNS 解析

您可以在在用户置备的基础架构上安装 OpenShift Container Platform 前验证 DNS 配置。

重要

本节中详述的验证步骤必须在安装集群前成功。

先决条件

- 已为您的用户置备的基础架构配置了所需的 DNS 记录。

流程

1. 从安装节点，针对 Kubernetes API 的记录名称、通配符路由和集群节点运行 DNS 查找。验证响应中包含的 IP 地址是否与正确的组件对应。

   a. 对 Kubernetes API 记录名称执行查询。检查结果是否指向 API 负载均衡器的 IP 地址：

   ```
   $ dig +noall +answer @<nameserver_ip> api.<cluster_name>.<base_domain> 1
   ```

   1 将 <nameserver_ip> 替换为 nameserver 的 IP 地址，<cluster_name> 替换为您的集群名称，<base_domain> 替换为您的基本域名。

   输出示例

   ```
   api.ocp4.example.com.  604800 IN A 192.168.1.5
   ```

   b. 对 Kubernetes 内部 API 记录名称执行查询。检查结果是否指向 API 负载均衡器的 IP 地址：

   ```
   $ dig +noall +answer @<nameserver_ip> api-int.<cluster_name>.<base_domain>
   ```

   输出示例

   ```
   api-int.ocp4.example.com.  604800 IN A 192.168.1.5
   ```

   c.

测试 *.apps.<cluster_name>.<base_domain> DNS 通配符查找示例。所有应用程序通配符查询都必须解析为应用程序入口负载均衡器的 IP 地址：

```
$ dig +noall +answer @<nameserver_ip> random.apps.<cluster_name>.
<base_domain>
```

输出示例

```
random.apps.ocp4.example.com.  604800 IN A 192.168.1.5
```

注意

在示例中，将相同的负载均衡器用于 Kubernetes API 和应用程序入口流量。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。

您可以使用另一个通配符值替换 random。例如，您可以查询到 OpenShift Container Platform 控制台的路由：

```
$ dig +noall +answer @<nameserver_ip> console-openshift-console.apps.
<cluster_name>.<base_domain>
```

输出示例

```
console-openshift-console.apps.ocp4.example.com. 604800 IN A 192.168.1.5
```

d. 针对 bootstrap DNS 记录名称运行查询。检查结果是否指向 bootstrap 节点的 IP 地址：

```
$ dig +noall +answer @<nameserver_ip> bootstrap.<cluster_name>.
<base_domain>
```

输出示例

```
bootstrap.ocp4.example.com.  604800 IN A 192.168.1.96
```

e. 使用此方法对 control plane 和计算节点的 DNS 记录名称执行查找。检查结果是否与每个节点的 IP 地址对应。

2. 从安装节点，针对负载均衡器和集群节点的 IP 地址运行反向 DNS 查找。验证响应中包含的记录名称是否与正确的组件对应。

a. 对 API 负载均衡器的 IP 地址执行反向查找。检查响应是否包含 Kubernetes API 和 Kubernetes 内部 API 的记录名称：

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.5
```

输出示例

```
5.1.168.192.in-addr.arpa. 604800 IN PTR api-int.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. 604800 IN PTR api.ocp4.example.com. 2
```

1 为 Kubernetes 内部 API 提供记录名称。

2 为 Kubernetes API 提供记录名称。

注意

OpenShift Container Platform 应用程序通配符不需要 PTR 记录。针对应用程序入口负载均衡器的 IP 地址解析反向 DNS 解析不需要验证步骤。

b. 对 bootstrap 节点的 IP 地址执行反向查找。检查结果是否指向 bootstrap 节点的 DNS 记录名称：

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.96
```

输出示例

```
96.1.168.192.in-addr.arpa. 604800 IN PTR bootstrap.ocp4.example.com.
```

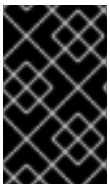
c. 使用此方法对 control plane 和计算节点的 IP 地址执行反向查找。检查结果是否与每个节点的 DNS 记录名称对应。

18.3.7. 为集群节点 SSH 访问生成密钥对

在 OpenShift Container Platform 安装过程中，您可以为安装程序提供 SSH 公钥。密钥通过它们的 Ignition 配置文件传递给 Red Hat Enterprise Linux CoreOS(RHCOS)节点，用于验证对节点的 SSH 访问。密钥添加到每个节点上 core 用户的 ~/.ssh/authorized_keys 列表中，这将启用免密码身份验证。

将密钥传递给节点后，您可以使用密钥对作为用户 核心 通过 SSH 连接到 RHCOS 节点。若要通过 SSH 访问节点，必须由 SSH 为您的本地用户管理私钥身份。

如果要通过 SSH 连接到集群节点来执行安装调试或灾难恢复，则必须在安装过程中提供 SSH 公钥。./openshift-install gather 命令还需要在集群节点上设置 SSH 公钥。



重要

不要在生产环境中跳过这个过程，在生产环境中需要灾难恢复和调试。

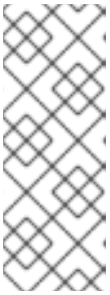
流程

1. 如果您在本地计算机上没有可用于在集群节点上进行身份验证的现有 SSH 密钥对，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1

指定新 SSH 密钥的路径和文件名，如 ~/.ssh/id_ed25519。如果您已有密钥对，请确保您的公钥位于 ~/.ssh 目录中。



注意

如果您计划在 x86_64、ppc64le 和 s390x 架构上安装使用 RHEL 加密库（这些加密库已提交给 NIST 用于 FIPS 140-2/140-3 验证）的 OpenShift Container Platform 集群，则不要创建使用 ed25519 算法的密钥。相反，创建一个使用 rsa 或 ecdsa 算法的密钥。

2. 查看公共 SSH 密钥：

```
$ cat <path>/<file_name>.pub
```

例如，运行以下命令来查看 ~/.ssh/id_ed25519.pub 公钥：

```
$ cat ~/.ssh/id_ed25519.pub
```

3. 将 SSH 私钥身份添加到本地用户的 SSH 代理（如果尚未添加）。在集群节点上，或者要使用 ./openshift-install gather 命令，需要对该密钥进行 SSH 代理管理，才能在集群节点上进行免密码 SSH 身份验证。

注意

在某些发行版中，自动管理默认 SSH 私钥身份，如 ~/.ssh/id_rsa 和 ~/.ssh/id_dsa。

a. 如果 ssh-agent 进程尚未为您的本地用户运行，请将其作为后台任务启动：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果集群处于 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

4. 将 SSH 私钥添加到 ssh-agent ：

```
$ ssh-add <path>/<file_name> 1
```

1 指定 SSH 私钥的路径和文件名，如 ~/.ssh/id_ed25519.pub

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

后续步骤

- 安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

18.3.8. 手动创建安装配置文件

安装集群要求您手动创建安装配置文件。

先决条件

- 您在本地机器上有一个 SSH 公钥来提供给安装程序。该密钥将用于在集群节点上进行 SSH 身份验证，以进行调试和灾难恢复。

- 已获取 OpenShift Container Platform 安装程序和集群的 pull secret。

流程

1. 创建一个安装目录来存储所需的安装资产：

```
$ mkdir <installation_directory>
```

重要

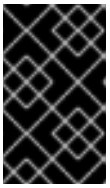
您必须创建一个目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

2. 自定义提供的 install-config.yaml 文件模板示例，并将其保存在 <installation_directory> 中。

注意

此配置文件必须命名为 install-config.yaml。

3. 备份 install-config.yaml 文件，以便您可以使用它安装多个集群。



重要

install-config.yaml 文件会在安装过程的下一步中使用。现在必须备份它。

其他资源

- IBM Z® 的安装配置参数

18.3.8.1. IBM Z 的 install-config.yaml 文件示例

您可以自定义 install-config.yaml 文件，以指定有关 OpenShift Container Platform 集群平台的更多详情，或修改所需参数的值。

```
apiVersion: v1
baseDomain: example.com (1)
compute: (2)
- hyperthreading: Enabled (3)
  name: worker
  replicas: 0 (4)
  architecture: s390x
controlPlane: (5)
  hyperthreading: Enabled (6)
  name: master
  replicas: 3 (7)
  architecture: s390x
metadata:
  name: test (8)
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14 (9)
    hostPrefix: 23 (10)
  networkType: OVNKubernetes (11)
  serviceNetwork: (12)
  - 172.30.0.0/16
platform:
  none: {} (13)
fips: false (14)
pullSecret: '{"auths":{"<local_registry>": {"auth": "<credentials>","email":
"you@example.com"}}}' (15)
sshKey: 'ssh-ed25519 AAAA...' (16)
additionalTrustBundle: | (17)
  -----BEGIN CERTIFICATE-----
  ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
  -----END CERTIFICATE-----
imageContentSources: (18)
- mirrors:
  - <local_repository>/ocp4/openshift4
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <local_repository>/ocp4/openshift4
  source: quay.io/openshift-release-dev/ocp-v4.0-art-dev
```

(1)

集群的基域。所有 DNS 记录都必须是这个基域的子域，并包含集群名称。

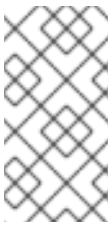
(2) (5)

controlPlane 部分是一个单个映射，但 compute 部分是一系列映射。为满足不同数据结构的要求，compute 部分的第一行必须以连字符 - 开头，controlPlane 部分 的第一行则不以连字符开

头。仅使用一个 control plane 池。

3 6

指定要启用或禁用并发多线程(SMT)还是超线程。默认情况下，启用 SMT 可提高机器中内核的性能。您可以通过将 参数值设置为 Disabled 来禁用它。如果禁用 SMT，则必须在所有集群机器中禁用它；这包括 control plane 和计算机器。



注意

默认启用并发多线程(SMT)。如果 OpenShift Container Platform 节点上没有 SMT，超线程 参数无效。

重要

如果您禁用 超线程，无论是在 OpenShift Container Platform 节点上，还是在 install-config.yaml 文件中，请确保您的容量规划考虑机器性能显著降低的情况。

4

在用户置备的基础架构上安装 OpenShift Container Platform 时，必须将这个值设置为 0。在安装程序置备的安装中，参数控制集群为您创建和管理的计算机器数量。在用户置备的安装中，您必须在完成集群安装前手动部署计算机器。

注意

如果要安装一个三节点集群，在安装 Red Hat Enterprise Linux CoreOS(RHCOS)机器时不要部署任何计算机器。

7

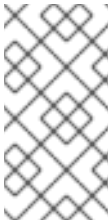
您添加到集群的 control plane 机器数量。由于集群使用这些值作为集群中的 etcd 端点数量，所以该值必须与您部署的 control plane 机器数量匹配。

8

您在 DNS 记录中指定的集群名称。

9

从中分配 Pod IP 地址的 IP 地址块。此块不得与现有物理网络重叠。这些 IP 地址用于 pod 网络。如果需要从外部网络访问 pod，您必须配置负载均衡器和路由器来管理流量。



注意

类 E CIDR 范围被保留以供以后使用。要使用 Class E CIDR 范围，您必须确保您的网络环境接受 Class E CIDR 范围内的 IP 地址。

10

分配给每个节点的子网前缀长度。例如，如果 hostPrefix 设为 23，则每个节点从 given cidr 中分配 a /23 子网，这样就能有 510（2^(32 - 23)- 2）个 pod IP 地址。如果需要从外部网络访问节点，请配置负载均衡器和路由器来管理流量。

11

要安装的集群网络插件。默认值 OVNKubernetes 是唯一支持的值。

12

用于服务 IP 地址的 IP 地址池。您只能输入一个 IP 地址池。此块不得与现有物理网络重叠。如果您需要从外部网络访问服务，请配置负载均衡器和路由器来管理流量。

13

您必须将平台设置为 none。您无法为 IBM Z® 基础架构提供额外的平台配置变量。

重要

使用平台类型 none 安装的集群无法使用一些功能，如使用 Machine API 管理计算机器。即使附加到集群的计算机器安装在通常支持该功能的平台上，也会应用这个限制。在安装后无法更改此参数。

14

是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 Red Hat Enterprise Linux (RHEL) 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅在 FIPS 模式中安装该系统。

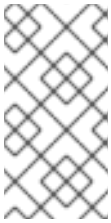
当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS)时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。

15

对于 <local_registry>，请指定 registry 域名，以及您的镜像 registry 用来提供内容的可选端口。例如： registry.example.com 或 registry.example.com:5000。对于 <credentials>，请为您的镜像 registry 指定 base64 编码的用户名和密码。

16

Red Hat Enterprise Linux CoreOS(RHCOS)中 core 用户的 SSH 公钥。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 ssh-agent 进程使用的 SSH 密钥。

17

添加 additionalTrustBundle 参数和值。该值必须是您用于镜像 registry 的证书文件内容。证书文件可以是现有的可信证书颁发机构，也可以是您为镜像 registry 生成的自签名证书。

18

根据您用来镜像存储库的命令输出提供 imageContentSources 部分。

重要

- 使用 oc adm release mirror 命令时，请使用 imageContentSources 部分中的输出。
- 使用 oc mirror 命令时，请使用运行该命令结果的 ImageContentSourcePolicy 文件的 repositoryDigestMirrors 部分。
- ImageContentSourcePolicy 已被弃用。如需更多信息，请参阅*配置镜像 registry 存储库镜像*。

18.3.8.2. 在安装过程中配置集群范围的代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 install-config.yaml 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 install-config.yaml 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 Proxy 对象的 spec.noProxy 字段中添加站点来绕过代理。

注意

Proxy 对象 status.noProxy 字段使用安装配置中的 networking.machineNetwork[].cidr、networking.clusterNetwork[].cidr 和 networking.serviceNetwork[] 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，Proxy 对象 status.noProxy 字段也会使用实例元数据端点填充(169.254.169.254)。

流程

1. 编辑 install-config.yaml 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 http。

2 用于创建集群外 HTTPS 连接的代理 URL。

3 要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 . 以仅匹配子域。例如，.y.com 匹配 x.y.com，但不匹配 y.com。使用 * 绕过所有目的地的代理。

4 如果提供，安装程序会在 openshift-config 命名空间中生成名为 user-ca-bundle 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 trusted-ca-bundle 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， Proxy 对象的 trustedCA 字段中也会引用此配置映射。additionalTrustBundle 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

5 可选：决定 Proxy 对象的配置以引用 trustedCA 字段中 user-ca-bundle 配置映射的策略。允许的值是 Proxyonly 和 Always。仅在配置了 http/https 代理时，使用 Proxyonly 引用 user-ca-bundle 配置映射。使用 Always 始终引用 user-ca-bundle 配置映射。默认值为 Proxyonly。

注意

安装程序不支持代理的 readinessEndpoints 字段。

注意

如果安装程序超时，重启并使用安装程序的 wait-for 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. **保存该文件并在安装 OpenShift Container Platform 时引用。**

安装程序会创建一个名为 cluster 的集群范围代理，该代理 使用 提供的 install-config.yaml 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 cluster Proxy 对象，但它会有一个空 spec。

注意

只支持名为 cluster 的 Proxy 对象，且无法创建额外的代理。

18.3.8.3. 配置三节点集群

另外，您可以在由三台 control plane 机器组成的最少三个节点集群中部署零台计算机器。这为集群管理员和开发人员提供了更小、效率更高的集群，用于测试、开发和生产。

在三节点 OpenShift Container Platform 环境中，三台 control plane 机器可以调度，这意味着应用程序工作负载被调度到它们上运行。

先决条件

- **您有一个现有的 install-config.yaml 文件。**

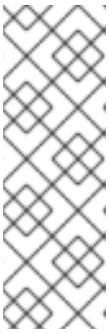
流程

- **确保 install-config.yaml 文件中的计算副本数量设置为 0，如以下 计算 小节所示：**

```
compute:
- name: worker
  platform: {}
  replicas: 0
```

注意

在用户置备的基础架构上安装 OpenShift Container Platform 时，无论您要部署的计算机器数量有多少，您必须将计算机器的 replicas 参数值设置为 0。在安装程序置备的安装中，参数控制集群为您创建和管理的计算机器数量。这不适用于手动部署计算机器的用户置备安装。



注意

control plane 节点的首选资源是 6 个 vCPU 和 21 GB。对于三个 control plane 节点，这是相当于至少五节点集群的内存 + vCPU。您应该为三个节点提供支持，每个节点安装在一个 120 GB 的磁盘上，并且启用了三个 SMT2 的 IFL。测试最小的设置是每个 control plane 节点在 120 GB 磁盘上的三个 vCPU 和 10 GB。

对于三节点集群安装，请按照以下步骤执行：

- 如果要部署一个带有零计算节点的三节点集群，Ingress Controller Pod 在 control plane 节点上运行。在三节点集群部署中，您必须配置应用程序入口负载均衡器，将 HTTP 和 HTTPS 流量路由到 control plane 节点。如需更多信息，*请参阅用户置备的基础架构的负载平衡要求* 部分。

- 在以下步骤中创建 Kubernetes 清单文件时，请确保 <installation_directory>/manifests/cluster-scheduler-02-config.yml 文件中的 mastersSchedulable 参数被设置为 true。这可让应用程序工作负载在 control plane 节点上运行。

- 在创建 Red Hat Enterprise Linux CoreOS(RHCOS)机器时，不要部署任何计算节点。

18.3.9. Cluster Network Operator 配置

集群网络的配置作为 Cluster Network Operator(CNO)配置的一部分指定，并存储在名为 cluster 的自定义资源(CR)对象中。CR 指定 operator.openshift.io API 组中的 Network API 的字段。

CNO 配置在集群安装过程中从 Network.config.openshift.io API 组中的 Network API 继承以下字段：

clusterNetwork

从中分配 Pod IP 地址的 IP 地址池。

serviceNetwork

服务的 IP 地址池.

defaultNetwork.type

集群网络插件。OVNKubernetes 是安装期间唯一支持的插件。

您可以通过在名为 cluster 的 CNO 对象中设置 defaultNetwork 对象的字段来为集群指定集群网络插件配置。

18.3.9.1. Cluster Network Operator 配置对象

下表中描述了 Cluster Network Operator(CNO)的字段：

表 18.29. Cluster Network Operator 配置对象

字段	类型	描述
metadata.name	**字符串**	CNO 对象的名称。这个名称始终是 **集群**。
spec.clusterNetwork	**array**	用于指定从哪些 IP 地址块分配 Pod IP 地址以及集群中每个节点的子网前缀长度的列表。例如： spec: clusterNetwork: - cidr: 10.128.0.0/19 hostPrefix: 23 - cidr: 10.128.32.0/19 hostPrefix: 23

字段	类型	描述
spec.serviceNetwork	**array**	服务的 IP 地址块。OpenShift SDN 和 OVN-Kubernetes 网络插件只支持服务网络的一个 IP 地址块。例如： spec: serviceNetwork: - 172.30.0.0/14 您只能在创建清单前在 **install-config.yaml** 文件中自定义此字段。该值在清单文件中是只读的。
spec.defaultNetwork	**object**	为集群网络配置网络插件。
spec.kubeProxyConfig	**object**	此对象的字段指定 kube-proxy 配置。如果使用 OVN-Kubernetes 集群网络供应商，则 kube-proxy 配置不会起作用。

defaultNetwork 对象配置

下表列出了 defaultNetwork 对象的值：

表 18.30. defaultNetwork 对象

字段	类型	描述
type	**字符串**	**OVNKubernetes**。Red Hat OpenShift Networking 网络插件在安装过程中被选择。此值在集群安装后无法更改。 **注意** OpenShift Container Platform 默认使用 OVN-Kubernetes 网络插件。OpenShift SDN 不再作为新集群的安装选择提供。
ovnKubernetesConfig	**object**	此对象仅对 OVN-Kubernetes 网络插件有效。

配置 OVN-Kubernetes 网络插件

下表描述了 OVN-Kubernetes 网络插件的配置字段：

表 18.31. ovnKubernetesConfig object

字段	类型	描述
mtu	**integer**	Geneve（通用网络虚拟化封装）覆盖网络的最大传输单元 (MTU)。这根据主网络接口的 MTU 自动探测。您通常不需要覆盖检测到的 MTU。 如果自动探测的值不是您期望的值，请确认节点上主网络接口上的 MTU 是否正确。您不能使用这个选项更改节点上主网络接口的 MTU 值。 如果集群中不同节点需要不同的 MTU 值，则必须将此值设置为**比** 集群中的最低 MTU 值小 100。例如，如果集群中的某些节点的 MTU 为 **9001**，而某些节点的 MTU 为 **1500**，则必须将此值设置为 **1400**。
genevePort	**integer**	用于所有 Geneve 数据包的端口。默认值为 **6081**。此值在集群安装后无法更改。
ipsecConfig	**object**	指定用于自定义 IPsec 配置的配置对象。
ipv4	**object**	为 IPv4 设置指定配置对象。
ipv6	**object**	为 IPv6 设置指定配置对象。
policyAuditConfig	**object**	指定用于自定义网络策略审计日志的配置对象。如果未设置，则使用默认的审计日志设置。
gatewayConfig	**object**	可选：指定一个配置对象来自定义如何将出口流量发送到节点网关。 **注意** 在迁移出口流量时，工作负载和服务流量会受到一定影响，直到 Cluster Network Operator (CNO) 成功推出更改。

表 18.32. ovnKubernetesConfig.ipv4 object

字段	类型	描述
internalTransitSwitchSubnet	字符串	如果您的现有网络基础架构与 **100.88.0.0/16** IPv4 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。启用东西流量的分布式传输交换机的子网。此子网不能与 OVN-Kubernetes 或主机本身使用的任何其他子网重叠。必须足够大，以适应集群中的每个节点一个 IP 地址。 默认值为 **100.88.0.0/16**。

字段	类型	描述
internalJoinSubnet	字符串	如果您的现有网络基础架构与 **100.64.0.0/16** IPv4 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。您必须确保 IP 地址范围没有与 OpenShift Container Platform 安装使用的任何其他子网重叠。IP 地址范围必须大于可添加到集群的最大节点数。例如，如果 **clusterNetwork.cidr** 值为 **10.128.0.0/14**，并且 **clusterNetwork.hostPrefix** 值为 **/23**，则最大节点数量为 **2^(23-14)=512**。 默认值为 **100.64.0.0/16**。

表 18.33. ovnKubernetesConfig.ipv6 object

字段	类型	描述
internalTransitSwitchSubnet	字符串	如果您的现有网络基础架构与 **fd97::/64** IPv6 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。启用东西流量的分布式传输交换机的子网。此子网不能与 OVN-Kubernetes 或主机本身使用的任何其他子网重叠。必须足够大，以适应集群中的每个节点一个 IP 地址。 默认值为 **fd97::/64**。
internalJoinSubnet	字符串	如果您的现有网络基础架构与 **fd98::/64** IPv6 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。您必须确保 IP 地址范围没有与 OpenShift Container Platform 安装使用的任何其他子网重叠。IP 地址范围必须大于可添加到集群的最大节点数。 默认值为 **fd98::/64**。

表 18.34. policyAuditConfig object

字段	类型	描述
rateLimit	整数	每个节点每秒生成一次的消息数量上限。默认值为每秒 **20** 条消息。
maxFileSize	整数	审计日志的最大大小，以字节为单位。默认值为 **50000000** 或 50 MB。
maxLogFiles	整数	保留的日志文件的最大数量。

字段	类型	描述
目的地	字符串	以下附加审计日志目标之一： **libc** 主机上的 journald 进程的 libc **syslog（）** 函数。 **UDP:<host>:<port>** 一个 syslog 服务器。将 **<host>:<port>** 替换为 **syslog 服务器的主机** 和端口。 **Unix:<file>** 由 **<file>** 指定的 Unix 域套接字文件。 **null** 不要将审计日志发送到任何其他目标。
syslogFacility	字符串	syslog 工具，如 as **kern**，如 RFC5424 定义。默认值为 **local0。**

表 18.35. gatewayConfig object

字段	类型	描述
routingViaHost	**布尔值**	将此字段设置为 **true**，将来自 pod 的出口流量发送到主机网络堆栈。对于依赖于在内核路由表中手动配置路由的高级别安装和应用程序，您可能需要将出口流量路由到主机网络堆栈。默认情况下，出口流量在 OVN 中进行处理以退出集群，不受内核路由表中的特殊路由的影响。默认值为 **false**。 此字段与 Open vSwitch 硬件卸载功能有交互。如果将此字段设置为 **true**，则不会获得卸载的性能优势，因为主机网络堆栈会处理出口流量。
ipForwarding	**object**	您可以使用 **Network** 资源中的 **ipForwarding** 规格来控制 OVN-Kubernetes 管理接口上所有流量的 IP 转发。指定 **Restricted** 只允许 Kubernetes 相关流量的 IP 转发。指定 **Global** 以允许转发所有 IP 流量。对于新安装，默认值为 **Restricted**。对于到 OpenShift Container Platform 4.14 或更高版本的更新，默认值为 **Global**。
ipv4	**object**	可选：指定一个对象来为主机配置内部 OVN-Kubernetes 伪装地址，以服务 IPv4 地址的流量。
ipv6	**object**	可选：指定一个对象来为主机配置内部 OVN-Kubernetes 伪装地址，以服务 IPv6 地址的流量。

表 18.36. gatewayConfig.ipv4 对象

字段	类型	描述
internalMasqueradeSubnet	**字符串**	内部使用的伪装 IPv4 地址，以启用主机服务流量。主机配置了这些 IP 地址和共享网关网桥接口。默认值为 **169.254.169.0/29**。

表 18.37. gatewayConfig.ipv6 对象

字段	类型	描述
internalMasqueradeSubnet	**字符串**	内部使用的伪装 IPv6 地址，以启用主机服务流量。主机配置了这些 IP 地址和共享网关网桥接口。默认值为 **fd69::/125**。

表 18.38. ipsecConfig 对象

字段	类型	描述
模式	**字符串**	指定 IPsec 实现的行为。必须是以下值之一： • **Disabled**: 在集群节点上不启用 IPsec。 • **External** ：对于带有外部主机的网络流量，启用 IPsec。 • **Full**: IPsec 为带有外部主机的 pod 流量和网络流量启用 IPsec。

启用 IPSec 的 OVN-Kubernetes 配置示例

```
defaultNetwork:
  type: OVNKubernetes
  ovnKubernetesConfig:
    mtu: 1400
    genevePort: 6081
      ipsecConfig:
        mode: Full
```

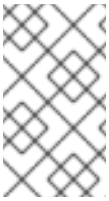
重要

使用 OVNKubernetes 可能会导致 IBM Power® 上的堆栈耗尽问题。

kubeProxyConfig 对象配置（仅限 OpenShiftSDN 容器网络接口）

kubeProxyConfig 对象的值在下表中定义：

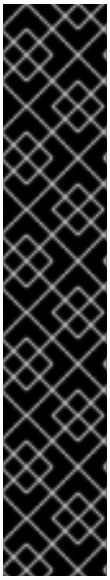
表 18.39. kubeProxyConfig object

字段	类型	描述
iptablesSyncPeriod	**字符串**	**iptables** 规则的刷新周期。默认值为 **30s**。有效的后缀包括 **s**、**m** 和 **h**，具体参见 Go 时间 包 文档。  **注意** 由于 OpenShift Container Platform 4.3 及更高版本中引进了性能改进，不再需要调整 **iptablesSyncPeriod** 参数。
proxyArguments.iptables-min-sync-period	**array**	刷新 **iptables** 规则前的最短持续时间。此字段确保刷新的频率不会过于频繁。有效的后缀包括 **s**、**m** 和 **h**，具体参见 Go **time** 软件包。默认值为： kubeProxyConfig: proxyArguments: iptables-min-sync-period: - 0s

18.3.10. 创建 Kubernetes 清单和 Ignition 配置文件

由于您必须修改一些集群定义文件并手动启动集群机器，因此您必须生成 Kubernetes 清单和 Ignition 配置文件来配置机器。

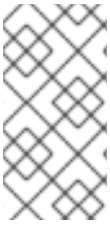
安装配置文件转换为 Kubernetes 清单。清单嵌套到 Ignition 配置文件中，稍后用于配置集群机器。



重要

- OpenShift Container Platform 安装程序生成的 Ignition 配置文件包含 24 小时后过期的证书，然后在该时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 node-bootstrapper 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。

- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。



注意

生成清单和 Ignition 文件的安装程序是特定的架构，可以从 客户端镜像镜像获取。安装程序的 Linux 版本仅在 s390x 上运行。此安装程序也可用作 Mac OS 版本。

先决条件

- 已获得 OpenShift Container Platform 安装程序。对于受限网络安装，这些文件位于您的镜像主机上。

- 已创建 install-config.yaml 安装配置文件。

流程

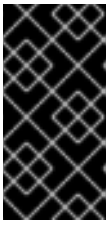
1. 进入包含 OpenShift Container Platform 安装程序的目录，并为集群生成 Kubernetes 清单：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 对于 <installation_directory>，请指定包含您创建的 install-config.yaml 文件的安装目录。

警告

如果您要安装一个三节点集群，请跳过以下步骤，以便可以调度 control plane 节点。



重要

当您将 control plane 节点从默认的不可调度配置为可以调度时，需要额外的订阅。这是因为 control plane 节点变为计算节点。

2. 检查 <installation_directory>/manifests/cluster-scheduler-02-config.yml Kubernetes 清单文件中的 mastersSchedulable 参数是否已设置为 false。此设置可防止在 control plane 机器上调度 pod：

   a. 打开 <installation_directory>/manifests/cluster-scheduler-02-config.yml 文件。

   b. 找到 mastersSchedulable 参数，并确保它被设置为 false。

   c. 保存并退出 文件。

3. 要创建 Ignition 配置文件，请从包含安装程序的目录运行以下命令：

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

1 对于 <installation_directory>，请指定相同的安装目录。

为安装目录中的 bootstrap、control plane 和计算节点创建 Ignition 配置文件。kubeadmin-password 和 kubeconfig 文件在 ./<installation_directory>/auth 目录中创建：

```
.
```

```
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

18.3.11. 在 IBM Z 或 IBM LinuxONE 环境中使用静态 IP 配置 NBDE

在 IBM Z® 或 IBM® LinuxONE 环境中启用 NBDE 磁盘加密需要额外的步骤，本节中详细介绍。

先决条件

- 您已设置了外部 Tang 服务器。具体步骤请查看 网络绑定磁盘加密。
- 您已安装了 with ane 实用程序。
- 您已查看如何使用 Butane 创建机器配置的说明。

流程

1. 为 control plane 和计算节点创建 Butane 配置文件。

   以下 control plane 节点的 Butane 配置示例为磁盘加密创建一个名为 master-storage.bu 的文件：

```
variant: openshift
version: 4.16.0
metadata:
  name: master-storage
  labels:
    machineconfiguration.openshift.io/role: master
storage:
  luks:
    - clevis:
        tang:
          - thumbprint: QcPr_NHFJammnRCA3fFMVdNBwjs
            url: http://clevis.example.com:7500
        options: 1
          - --cipher
          - aes-cbc-essiv:sha256
      device: /dev/disk/by-partlabel/root 2
```

```
    label: luks-root
    name: root
    wipe_volume: true
  filesystems:
   - device: /dev/mapper/root
    format: xfs
    label: root
    wipe_filesystem: true
openshift:
  fips: true 3
```

1

只有在启用了 FIPS 模式时才需要 cipher 选项。如果禁用了 FIPS，则省略该条目。

2

对于在 DASD 类型磁盘中安装，使用 device: /dev/disk/by-label/root 替换。

3

是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

2. 运行以下命令，创建自定义 initramfs 文件来引导机器：

```
$ coreos-installer pxe customize \
   /root/rhcos-bootfiles/rhcos-<release>-live-initramfs.s390x.img \
   --dest-device /dev/disk/by-id/scsi-<serial_number> --dest-karg-append \
   ip=<ip_address>::<gateway_ip>:<subnet_mask>::<network_device>:none \
   --dest-karg-append nameserver=<nameserver_ip> \
   --dest-karg-append rd.neednet=1 -o \
   /root/rhcos-bootfiles/<node_name>-initramfs.s390x.img
```

注意

首次引导前，您必须为集群中的每个节点自定义 initramfs，并添加 PXE 内核参数。

3. 创建包含 ignition.platform.id=metal 和 ignition.firstboot 的参数文件。

control plane 机器的内核参数文件示例：

```
rd.neednet=1 \
console=ttysclp0 \
coreos.inst.install_dev=/dev/dasda \(1)
ignition.firstboot ignition.platform.id=metal \
coreos.live.rootfs_url=http://<http_server>/rhcos-<version>-live-rootfs.
<architecture>.img \(2)
coreos.inst.ignition_url=http://<http_server>/master.ign \(3)
ip=10.19.17.2::10.19.17.1:255.255.255.0::enbdd0:none nameserver=10.19.17.1 \
zfcp.allow_lun_scan=0 \(4)
rd.znet=qeth,0.0.bdd0,0.0.bdd1,0.0.bdd2,layer2=1 \
rd.zfcp=0.0.5677,0x600606680g7f0056,0x034F000000000000 \(5)
```

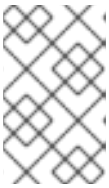
(1) 对于在 DASD 类型磁盘中安装，请添加 coreos.inst.install_dev=/dev/dasda。为 FCP 类型磁盘省略这个值。

(2) 指定您要引导的 kernel 和 initramfs 的 rootfs 工件位置。仅支持 HTTP 和 HTTPS 协议。

(3) 指定 Ignition 配置文件的位置。使用 master.ign 或 worker.ign。仅支持 HTTP 和 HTTPS 协议。

(4) 对于在 FCP 类型磁盘中安装，请添加 zfcp.allow_lun_scan=0。为 DASD 类型磁盘省略这个值。

(5) 对于在 DASD 类型磁盘中安装，使用 rd.dasd=0.0.3490 替换来指定 DASD 设备。



注意

将参数文件中的所有选项写为一行，并确保您没有换行字符。

其他资源

- 使用 Butane 创建机器配置

18.3.12. 安装 RHCOS 并启动 OpenShift Container Platform bootstrap 过程

要在您置备的 IBM Z® 基础架构上安装 OpenShift Container Platform，您必须在 z/VM 客户机虚拟机上安装 Red Hat Enterprise Linux CoreOS (RHCOS)。安装 RHCOS 时，您必须为您要安装的机器类型提供 OpenShift Container Platform 安装程序生成的 Ignition 配置文件。如果您配置了适当的网络、DNS 和负载均衡基础架构，OpenShift Container Platform bootstrap 过程会在 RHCOS z/VM 客户机虚拟机重启后自动启动。

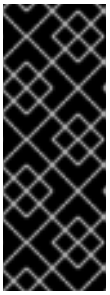
完成以下步骤以创建机器。

先决条件

- 在置备机器上运行的 HTTP 或 HTTPS 服务器，可供您创建的机器访问。
- 如果要启用安全引导，需要已获取了适当的红帽产品签名密钥，并在 IBM 文档中的 IBM Z 和 IBM LinuxONE 上读取安全引导。

流程

1. 在您的置备机器上登录到 Linux。
2. 从 RHCOS 镜像镜像获取 Red Hat Enterprise Linux CoreOS(RHCOS)内核、initramfs 和 rootfs 文件。

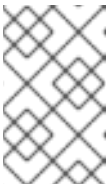


重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每个发行版本而改变。您必须下载最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。仅使用以下步骤中描述的适当 kernel、initramfs 和 rootfs 工件。

文件名包含 OpenShift Container Platform 版本号。它们类似以下示例：

- kernel: rhcos-<version>-live-kernel-<architecture>
- initramfs: rhcos-<version>-live-initramfs.<architecture>.img
- rootfs: rhcos-<version>-live-rootfs.<architecture>.img



注意

FCP 和 DASD 的 rootfs 镜像相同。

3. 创建参数文件。以下参数特定于特定虚拟机：

- 对于 ip=，请指定以下七项：
  i. 计算机的 IP 地址。
  ii. 个空字符串。
  iii. 网关
  iv. 子网掩码.
  v. hostname.domainname 格式的机器主机和域名。省略这个值可让 RHCOS 决定。
  vi. 网络接口名称。省略这个值可让 RHCOS 决定。
  vii. 如果使用静态 IP 地址，请指定 none。
-

对于 coreos.inst.ignition_url=，请为机器角色指定 Ignition 文件。使用 bootstrap.ign、master.ign 或 worker.ign。仅支持 HTTP 和 HTTPS 协议。

- 对于 coreos.live.rootfs_url=，请为您引导的内核和 initramfs 指定匹配的 rootfs 构件。仅支持 HTTP 和 HTTPS 协议。

- 可选：要启用安全引导，请添加 coreos.inst.secure_ipl

- 对于在 DASD 类型磁盘中安装，请完成以下任务：

  i. 对于 coreos.inst.install_dev=，请指定 /dev/dasda。

  ii. Userd .dasd= 指定安装 RHCOS 的 DASD。

  iii. 所有其他参数保持不变。

  bootstrap 机器的参数文件示例：bootstrap-0.parm ：

```
rd.neednet=1 \
console=ttysclp0 \
coreos.inst.install_dev=/dev/dasda \
coreos.live.rootfs_url=http://<http_server>/rhcos-<version>-live-rootfs.
<architecture>.img \ 1
coreos.inst.ignition_url=http://<http_server>/bootstrap.ign \ 2
coreos.inst.secure_ipl \ 3
ip=172.18.78.2::172.18.78.1:255.255.255.0:::none nameserver=172.18.78.1 \
rd.znet=qeth,0.0.bdf0,0.0.bdf1,0.0.bdf2,layer2=1,portno=0 \
zfcp.allow_lun_scan=0 \
rd.dasd=0.0.3490
```

1 指定您要引导的 kernel 和 initramfs 的 rootfs 工件位置。仅支持 HTTP 和 HTTPS 协议。

2 指定 Ignition 配置文件的位置。使用 bootstrap.ign、master.ign 或 worker.ign。仅支持 HTTP 和 HTTPS 协议。

3

可选：要启用安全引导，请添加 coreos.inst.secure_ipl。

将参数文件中的所有选项写为一行，并确保您没有换行字符。

- 对于在 FCP 类型磁盘中安装，请完成以下任务：

i. User d.zfcp=<adapter>,<wwpn>,<lun> 以指定要安装 RHCOS 的 FCP 磁盘。对于多路径，为每个额外路径重复此步骤。

注意

当使用多个路径安装时，您必须在安装后直接启用多路径，而不是在以后启用多路径，因为这可能导致问题。

ii. 将安装设备设置为：coreos.inst.install_dev=/dev/disk/by-id/scsi-<serial_number>。

注意

如果使用 NPIV 配置额外的 LUN，FCP 需要 zfcp.allow_lun_scan=0。如果必须启用 zfcp.allow_lun_scan=1，因为您使用 CSI 驱动程序，则必须配置 NPIV，以便每个节点无法访问另一个节点的引导分区。

iii. 所有其他参数保持不变。

重要

需要额外的安装后步骤才能完全启用多路径。如需更多信息，请参阅 *安装后机器配置任务* 中的"使用 RHCOS 上内核参数启用多路径"。

以下是使用多路径的计算节点的 worker-1.parm 示例参数文件：

```
rd.neednet=1 \
console=ttysclp0 \
coreos.inst.install_dev=/dev/disk/by-id/scsi-<serial_number> \
coreos.live.rootfs_url=http://<http_server>/rhcos-<version>-live-rootfs.
<architecture>.img \
coreos.inst.ignition_url=http://<http_server>/worker.ign \
ip=172.18.78.2::172.18.78.1:255.255.255.0:::none nameserver=172.18.78.1 \
rd.znet=qeth,0.0.bdf0,0.0.bdf1,0.0.bdf2,layer2=1,portno=0 \
zfcp.allow_lun_scan=0 \
rd.zfcp=0.0.1987,0x50050763070bc5e3,0x4008400B00000000 \
rd.zfcp=0.0.19C7,0x50050763070bc5e3,0x4008400B00000000 \
rd.zfcp=0.0.1987,0x50050763071bc5e3,0x4008400B00000000 \
rd.zfcp=0.0.19C7,0x50050763071bc5e3,0x4008400B00000000
```

将参数文件中的所有选项写为一行，并确保您没有换行字符。

4. 将 initramfs、内核、参数文件和 RHCOS 镜像传送到 z/VM 中，例如使用 FTP。有关如何使用 FTP 传输文件并从虚拟读取器引导的详情，请参阅 在 Z/VM 中安装。

5. 将文件 punch 到 z/VM 客户机虚拟机的虚拟读取器，即成为 bootstrap 节点。

   请参阅 IBM 文档中的 PUNCH。

   提示

   您可以使用 CP PUNCH 命令，或者，如果使用 Linux，则使用 vmur 命令在两个 z/VM 虚拟机之间传输文件。

6. 在 bootstrap 机器上登录到 CMS。

7. 从 reader IPL bootstrap 机器：

   ```
   $ ipl c
   ```

   请参阅 IBM 文档中的 IPL。

8. 对集群中的其他机器重复此步骤。

18.3.12.1. 高级 RHCOS 安装参考

本节演示了网络配置和其他高级选项，允许您修改 Red Hat Enterprise Linux CoreOS(RHCOS)手动安装过程。下表描述了您可以用于 RHCOS live 安装程序和 coreos-installer 命令的内核参数和命令行选项。

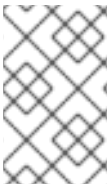
18.3.12.1.1. ISO 安装的网络和绑定选项

如果从 ISO 镜像安装 RHCOS，您可以在引导镜像时手动添加内核参数，以便为节点配置网络。如果没有指定网络参数，当 RHCOS 检测到需要网络来获取 Ignition 配置文件时，在 initramfs 中激活 DHCP。

重要

在手动添加网络参数时，还必须添加 rd.neednet=1 内核参数，以便在 initramfs 中启动网络。

以下信息提供了在 RHCOS 节点上为 ISO 安装配置网络和绑定的示例。示例描述了如何使用 ip=、name server = 和 bond= 内核参数。



注意

添加内核参数时顺序非常重要： ip=、 name server=，然后 bond=。

网络选项在系统引导过程中传递给 dracut 工具。有关 dracut 支持的网络选项的更多信息，请参阅 dracut.cmdline 手册页。

以下示例是 ISO 安装的网络选项。

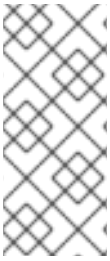
配置 DHCP 或静态 IP 地址

要配置 IP 地址，可使用 DHCP(ip=dhcp)或设置单独的静态 IP 地址(ip=<host_ip>)。如果设置静态 IP，则必须在每个节点上识别 DNS 服务器 IP 地址（名称服务器=<dns_ip>）。以下示例集：

- 节点的 IP 地址为 10.10.10.2

- 网关地址为 10.10.10.254
- 子网掩码为 255.255.255.0
- 到 core0.example.com 的主机名
- DNS 服务器地址为 4.4.4.41
- 自动配置值为 none。当以静态方式配置 IP 网络时，不需要自动配置。

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
nameserver=4.4.4.41
```



注意

当您使用 DHCP 为 RHCOS 机器配置 IP 寻址时，机器还通过 DHCP 获取 DNS 服务器信息。对于基于 DHCP 的部署，您可以通过 DHCP 服务器配置定义 RHCOS 节点使用的 DNS 服务器地址。

配置没有静态主机名的 IP 地址

您可以在不分配静态主机名的情况下配置 IP 地址。如果用户没有设置静态主机名，则会提取并通过反向 DNS 查找自动设置。要在没有静态主机名的情况下配置 IP 地址，请参考以下示例：

- 节点的 IP 地址为 10.10.10.2
- 网关地址为 10.10.10.254
- 子网掩码为 255.255.255.0
- DNS 服务器地址为 4.4.4.41

- 自动配置值为 none。当以静态方式配置 IP 网络时，不需要自动配置。

```
ip=10.10.10.2::10.10.10.254:255.255.255.0::enp1s0:none
nameserver=4.4.4.41
```

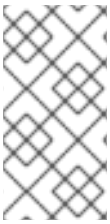
指定多个网络接口

您可以通过设置多个 ip= 条目来指定多个网络接口。

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
ip=10.10.10.3::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none
```

配置默认网关和路由

可选：您可以通过设置 a rd.route= 值来配置到额外网络的路由。



注意

当您配置一个或多个网络时，需要一个默认网关。如果额外网络网关与主要网络网关不同，则默认网关必须是主要网络网关。

- 运行以下命令来配置默认网关：

```
ip=::10.10.10.254::::
```

- 输入以下命令为额外网络配置路由：

```
rd.route=20.20.20.0/24:20.20.20.254:enp2s0
```

在单个接口中禁用 DHCP

您可以在单一接口中禁用 DHCP，例如当有两个或者多个网络接口时，且只有一个接口被使用。在示例中，enp1s0 接口具有一个静态网络配置，而 enp2s0 禁用了 DHCP，不使用它：

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
ip=::::core0.example.com:enp2s0:none
```

合并 DHCP 和静态 IP 配置

您可以将系统上的 DHCP 和静态 IP 配置与多个网络接口合并，例如：

```
ip=enp1s0:dhcp
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none
```

在独立接口上配置 VLAN

可选：您可以使用 vlan= 参数在单个接口上配置 VLAN。

- 要在网络接口中配置 VLAN 并使用静态 IP 地址，请运行以下命令：

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0.100:none
vlan=enp2s0.100:enp2s0
```

- 要在网络接口中配置 VLAN 并使用 DHCP，请运行以下命令：

```
ip=enp2s0.100:dhcp
vlan=enp2s0.100:enp2s0
```

提供多个 DNS 服务器

您可以通过为每个服务器添加一个 nameserver= 条目来提供多个 DNS 服务器，例如

```
nameserver=1.1.1.1
nameserver=8.8.8.8
```

将多个网络接口绑定到一个接口

可选：您可以使用 bond= 选项将多个网络接口绑定到一个接口。请参见以下示例：

- 配置绑定接口的语法为：bond=<name>[:<network_interfaces>][:options]

  <name> 是绑定设备名称 (bond0)、<network_interfaces> 代表以逗号分隔的物理（以太网）接口列表(em1,em2)，*options* 是用逗号分开的绑定选项列表。输入 modinfo bonding 查看可用选项。

- 当使用 bond= 创建绑定接口时，您必须指定如何分配 IP 地址以及绑定接口的其他信息。

  - 要将绑定接口配置为使用 DHCP，请将绑定的 IP 地址设置为 dhcp。例如：

```
bond=bond0:em1,em2:mode=active-backup
ip=bond0:dhcp
```

- 要将绑定接口配置为使用静态 IP 地址，请输入您需要的特定 IP 地址和相关信息。例如：

```
bond=bond0:em1,em2:mode=active-backup,fail_over_mac=1
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0:none
```

在 active-backup 模式中始终设置 fail_over_mac=1 选项，以避免使用共享 OSA/RoCE 卡时出现问题。

将多个网络接口绑定到一个接口

可选：您可以使用 vlan= 参数并在绑定接口上配置 VLAN，并使用 DHCP，例如：

```
ip=bond0.100:dhcp
bond=bond0:em1,em2:mode=active-backup
vlan=bond0.100:bond0
```

使用以下示例配置带有 VLAN 的绑定接口并使用静态 IP 地址：

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0.100:none
bond=bond0:em1,em2:mode=active-backup
vlan=bond0.100:bond0
```

使用网络团队

可选：您可以使用 team= 参数来将网络团队用作绑定的替代选择：

- 配置组接口的语法为：team=name[:network_interfaces]

  name 是组设备名称(team0)，*network_interfaces* 代表以逗号分隔的物理（以太网）接口（em1、em2）列表。

注意

当 RHCOS 切换到即将推出的 RHEL 版本时，团队(team)功能被计划弃用。如需更多信息，请参阅红帽知识库文章。

使用以下示例配置网络团队：

```
team=team0:em1,em2
ip=team0:dhcp
```

18.3.13. 等待 bootstrap 过程完成

OpenShift Container Platform bootstrap 过程在集群节点首次引导到安装到磁盘的持久 RHCOS 环境后开始。通过 Ignition 配置文件提供的配置信息用于初始化 bootstrap 过程并在机器上安装 OpenShift Container Platform。您必须等待 bootstrap 过程完成。

先决条件

- 已为集群创建 Ignition 配置文件。
- 您已配置了适当的网络、DNS 和负载平衡基础架构。
- 已获得安装程序，并为集群生成 Ignition 配置文件。
- 已在集群机器上安装 RHCOS，并提供 OpenShift Container Platform 安装程序生成的 Ignition 配置文件。

流程

1. 监控 bootstrap 过程：

```
$ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \ 1
  --log-level=info 2
```

1. 对于 <installation_directory>，请指定安装文件保存到的目录的路径。
2. 要查看不同的安装详情，请指定 warn、debug 或 error，而不是 info。

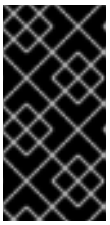
输出示例

-

```
INFO Waiting up to 30m0s for the Kubernetes API at
https://api.test.example.com:6443...
INFO API v1.29.4 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
```

当 Kubernetes API 服务器提示已在 control plane 机器上引导它时，该命令会成功。

2. bootstrap 过程完成后，从负载均衡器中删除 bootstrap 机器。



重要

此时您必须从负载均衡器中删除 bootstrap 机器。您还可以删除或重新格式化 bootstrap 机器本身。

18.3.14. 使用 CLI 登录集群

您可以通过导出集群 kubeconfig 文件，以默认系统用户身份登录集群。kubeconfig 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。
- 已安装 oc CLI。

流程

1. 导出 kubeadmin 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1

对于 <installation_directory>，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 oc 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

18.3.15. 批准机器的证书签名请求

当您将机器添加到集群时，会为您添加的每台机器生成两个待处理证书签名请求(CSR)。您必须确认这些 CSR 已获得批准，或根据需要自行批准。必须首先批准客户端请求，然后批准服务器请求。

先决条件

- 您已将机器添加到集群中。

流程

1. 确认集群可以识别这些机器：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0  Ready    master  63m  v1.29.4
master-1  Ready    master  63m  v1.29.4
master-2  Ready    master  64m  v1.29.4
```

输出中列出了您创建的所有机器。

注意

在有些 CSR 被批准前，前面的输出可能不包括计算节点（也称为 worker 节点）。

2. 检查待处理的 CSR，并确保添加到集群中的每台机器都有 Pending 或 Approved 状态的客户端请求：

```
$ oc get csr
```

输出示例

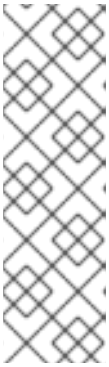
```
NAME        AGE    REQUESTOR                                                   CONDITION
csr-8b2br   15m    system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
csr-8vnps   15m    system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
...
```

在本例中，两台机器加入集群。您可能会在列表中看到更多已批准的 CSR。

3. 如果 CSR 没有获得批准，在您添加的机器的所有待处理 CSR 都处于 Pending 状态后，请批准集群机器的 CSR：

注意

由于 CSR 会自动轮转，因此请在将机器添加到集群后一小时内批准您的 CSR。如果没有在一小时内批准它们，证书将会轮转，每个节点会存在多个证书。您必须批准所有这些证书。批准客户端 CSR 后，Kubelet 为服务证书创建一个二级 CSR，这需要手动批准。然后，如果 Kubelet 请求具有相同参数的新证书，则后续提供证书续订请求由 machine-approver 自动批准。



注意

对于在未启用机器 API 的平台上运行的集群，如裸机和其他用户置备的基础架构，您必须实施一种方法来自动批准 kubelet 提供证书请求(CSR)。如果没有批准请求，则 oc exec、ocrsh 和 oc logs 命令将无法成功，因为 API 服务器连接到 kubelet 时需要服务证书。与 Kubelet 端点联系的任何操作都需要此证书批准。该方法必须监视新的 CSR，确认 CSR 由 system: node 或 system:admin 组中的 node-bootstrapper 服务帐户提交，并确认节点的身份。

- 要单独批准，请对每个有效的 CSR 运行以下命令：

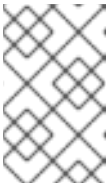
```
$ oc adm certificate approve <csr_name> 1
```

1

<csr_name> 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}
{{"\n"}}{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```



注意

在有些 CSR 被批准前，一些 Operator 可能无法使用。

4. 现在，您的客户端请求已被批准，您必须查看添加到集群中的每台机器的服务器请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE     REQUESTOR                                                   CONDITION
csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal       Pending
csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal       Pending
...
```

5. 如果剩余的 CSR 没有被批准，且处于 Pending 状态，请批准集群机器的 CSR：

- 要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 <csr_name> 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}
{{"\n"}}{{end}}{{end}}' | xargs oc adm certificate approve
```

6. 批准所有客户端和服务器 CSR 后，机器将 处于 Ready 状态。运行以下命令验证：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0  Ready    master  73m  v1.29.4
master-1  Ready    master  73m  v1.29.4
master-2  Ready    master  74m  v1.29.4
worker-0  Ready    worker  11m  v1.29.4
worker-1  Ready    worker  11m  v1.29.4
```

注意

批准服务器 CSR 后可能需要几分钟时间让机器过渡到 Ready 状态。

其他信息

- 如需有关 CSR 的更多信息，请参阅 证书签名请求。

18.3.16. 初始 Operator 配置

在 control plane 初始化后，您必须立即配置一些 Operator，以便它们都可用。

先决条件

- 您的 control plane 已初始化。

流程

1. 观察集群组件上线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION   AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.16.0    True        False         False      19m
baremetal                                  4.16.0    True        False         False      37m
cloud-credential                           4.16.0    True        False         False      40m
cluster-autoscaler                         4.16.0    True        False         False      37m
config-operator                            4.16.0    True        False         False      38m
console                                    4.16.0    True        False         False      26m
csi-snapshot-controller                    4.16.0    True        False         False      37m
dns                                        4.16.0    True        False         False      37m
etcd                                       4.16.0    True        False         False      36m
image-registry                             4.16.0    True        False         False      31m
ingress                                    4.16.0    True        False         False      30m
```

```
insights                                   4.16.0    True        False         False      31m
kube-apiserver                             4.16.0    True        False         False      26m
kube-controller-manager                    4.16.0    True        False         False      36m
kube-scheduler                             4.16.0    True        False         False      36m
kube-storage-version-migrator              4.16.0    True        False         False      37m
machine-api                                4.16.0    True        False         False      29m
machine-approver                           4.16.0    True        False         False      37m
machine-config                             4.16.0    True        False         False      36m
marketplace                                4.16.0    True        False         False      37m
monitoring                                 4.16.0    True        False         False      29m
network                                    4.16.0    True        False         False      38m
node-tuning                                4.16.0    True        False         False      37m
openshift-apiserver                        4.16.0    True        False         False      32m
openshift-controller-manager               4.16.0    True        False         False      30m
openshift-samples                          4.16.0    True        False         False      32m
operator-lifecycle-manager                 4.16.0    True        False         False      37m
operator-lifecycle-manager-catalog         4.16.0    True        False         False      37m
operator-lifecycle-manager-packageserver   4.16.0    True        False         False      32m
service-ca                                 4.16.0    True        False         False      38m
storage                                    4.16.0    True        False         False      37m
```

2. 配置不可用的 Operator。

18.3.16.1. 禁用默认的 OperatorHub 目录源

在 OpenShift Container Platform 安装过程中，默认为 OperatorHub 配置由红帽和社区项目提供的源内容的 operator 目录。在受限网络环境中，必须以集群管理员身份禁用默认目录。

流程

- 通过在 OperatorHub 对象中添加 disableAllDefaultSources: true 来 禁用默认目录的源：

```
$ oc patch OperatorHub cluster --type json \
    -p '[{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]'
```

提示

或者，您可以使用 Web 控制台管理目录源。在 Administration → Cluster Settings → Configuration → OperatorHub 页面中，点 Sources 选项卡，您可以在其中创建、更新、删除、禁用和启用单独的源。

18.3.16.2. 镜像 registry 存储配置

对于不提供默认存储的平台，Image Registry Operator 最初不可用。安装后，您必须将 registry 配置为使用存储，以便 Registry Operator 可用。

显示配置生产集群所需的持久性卷的说明。如果适用，显示有关将空目录配置为存储位置的说明，这仅适用于非生产集群。

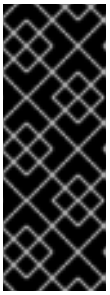
提供了在升级过程中使用 Recreate rollout 策略来允许镜像 registry 使用块存储类型的说明。

18.3.16.2.1. 为 IBM Z 配置 registry 存储

作为集群管理员，在安装后需要配置 registry 来使用存储。

先决条件

- 您可以使用具有 cluster-admin 角色的用户访问集群。
- 在 IBM Z® 上有一个集群。
- 您已为集群置备持久性存储，如 Red Hat OpenShift Data Foundation。



重要

当您只有一个副本时，OpenShift Container Platform 支持对镜像 registry 存储的 ReadWriteOnce 访问。ReadWriteOnce 访问还要求 registry 使用 Recreate rollout 策略。要部署支持高可用性的镜像 registry，需要两个或多个副本，ReadWriteMany 访问。

- 必须具有 100Gi 容量。

流程

1. 要将 registry 配置为使用存储，修改 configs.imageregistry/cluster 资源中的 spec.storage.pvc。

注意

使用共享存储时，请查看您的安全设置以防止外部访问。

2. 验证您没有 registry pod:

```
$ oc get pod -n openshift-image-registry -l docker-registry=default
```

输出示例

```
No resources found in openshift-image-registry namespace
```

注意

如果您的输出中有一个 registry pod，则不需要继续这个过程。

3. 检查 registry 配置：

```
$ oc edit configs.imageregistry.operator.openshift.io
```

输出示例

```
storage:
  pvc:
    claim:
```

将 claim 字段留空以允许自动创建 image-registry-storage PVC。

4. 检查 **clusteroperator** 状态：

```
$ oc get clusteroperator image-registry
```

输出示例

```
NAME             VERSION   AVAILABLE   PROGRESSING   DEGRADED   SINCE   MESSAGE
image-registry   4.16      True        False         False      6h50m
```

5. **确保 registry** 设置为 **managed，以启用镜像的构建和推送。**

- 运行：

```
$ oc edit configs.imageregistry/cluster
```

然后，更改行

```
managementState: Removed
```

至

```
managementState: Managed
```

18.3.16.2.2. 在非生产集群中为镜像 registry 配置存储

您必须为 Image Registry Operator 配置存储。对于非生产集群，您可以将镜像 registry 设置为空目录。如果您这样做，重启 registry 时会丢失所有镜像。

流程

- **将镜像 registry 存储设置为空目录：**

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch
'{"spec":{"storage":{"emptyDir":{}}}}'
```

警告

仅为非生产集群配置这个选项。

如果在 Image Registry Operator 初始化其组件前运行这个命令，oc patch 命令会失败并显示以下错误：

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster"
not found
```

等待几分钟，然后再次运行 命令。

18.3.17. 在用户置备的基础架构上完成安装

完成 Operator 配置后，可以在您提供的基础架构上完成集群安装。

先决条件

- 您的 control plane 已初始化。
- 已完成初始 Operator 配置。

流程

1. 使用以下命令确认所有集群组件都在线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION   AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.16.0    True        False         False      19m
baremetal                                  4.16.0    True        False         False      37m
cloud-credential                           4.16.0    True        False         False      40m
cluster-autoscaler                         4.16.0    True        False         False      37m
config-operator                            4.16.0    True        False         False      38m
console                                    4.16.0    True        False         False      26m
csi-snapshot-controller                    4.16.0    True        False         False      37m
dns                                        4.16.0    True        False         False      37m
etcd                                       4.16.0    True        False         False      36m
image-registry                             4.16.0    True        False         False      31m
ingress                                    4.16.0    True        False         False      30m
insights                                   4.16.0    True        False         False      31m
kube-apiserver                             4.16.0    True        False         False      26m
kube-controller-manager                    4.16.0    True        False         False      36m
kube-scheduler                             4.16.0    True        False         False      36m
kube-storage-version-migrator              4.16.0    True        False         False      37m
machine-api                                4.16.0    True        False         False      29m
machine-approver                           4.16.0    True        False         False      37m
machine-config                             4.16.0    True        False         False      36m
marketplace                                4.16.0    True        False         False      37m
monitoring                                 4.16.0    True        False         False      29m
network                                    4.16.0    True        False         False      38m
node-tuning                                4.16.0    True        False         False      37m
openshift-apiserver                        4.16.0    True        False         False      32m
openshift-controller-manager               4.16.0    True        False         False      30m
openshift-samples                          4.16.0    True        False         False      32m
operator-lifecycle-manager                 4.16.0    True        False         False      37m
operator-lifecycle-manager-catalog         4.16.0    True        False         False      37m
operator-lifecycle-manager-packageserver   4.16.0    True        False         False      32m
service-ca                                 4.16.0    True        False         False      38m
storage                                    4.16.0    True        False         False      37m
```

另外，当所有集群都可用时，以下命令会通知您。它还检索并显示凭证：

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

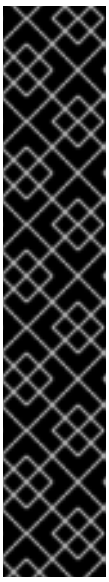
1

对于 <installation_directory>，请指定安装文件保存到的目录的路径。

输出示例

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

Cluster Version Operator 完成从 Kubernetes API 服务器部署 OpenShift Container Platform 集群时，该命令会成功。



重要

- 安装程序生成的 Ignition 配置文件包含 24 小时后过期的证书，然后在该时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 node-bootstrapper 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。

- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

2. 确认 Kubernetes API 服务器正在与 pod 通信。

   a. 要查看所有 pod 的列表，请使用以下命令：

```
$ oc get pods --all-namespaces
```

输出示例

```
NAMESPACE                  NAME                                        READY   STATUS
RESTARTS   AGE
openshift-apiserver-operator    openshift-apiserver-operator-85cb746d55-zqhs8
1/1    Running    1          9m
openshift-apiserver        apiserver-67b9g                             1/1     Running     0
3m
openshift-apiserver        apiserver-ljcmx                             1/1     Running     0
1m
openshift-apiserver        apiserver-z25h4                             1/1     Running     0
2m
```

```
openshift-authentication-operator authentication-operator-69d5d8bf84-vh2n8
1/1     Running     0          5m
...
```

b. 使用以下命令，查看上一命令的输出中所列 pod 的日志：

```
$ oc logs <pod_name> -n <namespace>
```

1 指定 pod 名称和命名空间，如上一命令的输出中所示。

如果 pod 日志显示，Kubernetes API 服务器可以与集群机器通信。

3. 对于使用光纤通道协议(FCP)的安装，还需要额外的步骤才能启用多路径。不要在安装过程中启用多路径。

如需更多信息，请参阅 *安装后机器配置任务* 文档中的"使用 RHCOS 上使用内核参数启用多路径"。

4. 在 Cluster registration 页面注册 您的集群。

验证

如果您在 OpenShift Container Platform bootstrap 过程中启用了安全引导，则需要以下验证步骤：

1. 运行以下命令来调试节点：

```
$ oc debug node/<node_name>
chroot /host
```

2. 运行以下命令确认启用了安全引导：

```
$ cat /sys/firmware/ipl/secure
```

输出示例

```
1 1
```

1

如果启用了安全引导，则值为 1，如果未启用安全引导，则值为 0。

其他资源

- 如何在没有 SSH 的 OpenShift Container Platform 版本 4 节点中生成 SOSREPORT。

18.3.18. 后续步骤

- 自定义集群。
- 如果您用来安装集群的镜像 registry 具有可信任的 CA，请通过 配置额外的信任存储将其添加到集群中。
- 如果需要，您可以选择 不使用远程健康报告。
- 如果需要，请参阅 注册断开连接的集群

18.4. 在 IBM Z 和 IBM LINUXONE 中使用 RHEL KVM 安装集群

在 OpenShift Container Platform 版本 4.16 中，您可以在您置备的 IBM Z® 或 IBM® LinuxONE 系统上安装集群。

注意

虽然本文档只涉及 IBM Z®，但它的所有信息也适用于 IBM® LinuxONE。

重要

非裸机平台还有其他注意事项。在安装 OpenShift Container Platform 集群前，请参阅有关在未经测试的平台上部署 OpenShift Container Platform 的指南 中的信息。

18.4.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读有关 选择集群安装方法的文档，并为用户准备它。
- 在开始安装过程前，您必须清理安装目录。这样可确保在安装过程中创建和更新所需的安装文件。
- 已为集群置备了 使用 OpenShift Data Foundation 或其他支持的存储协议的持久性存储。要部署私有镜像 registry，您必须使用 ReadWriteMany 访问设置持久性存储。
- 如果使用防火墙，则会 将其配置为允许集群需要访问的站点。

注意

如果要配置代理，请务必查看此站点列表。

- 置备基于 RHEL 8.6 或更高版本的、托管在逻辑分区(LPAR)上的 RHEL Kernel Virtual Machine(KVM)系统。请参阅 Red Hat Enterprise Linux 8 和 9 生命周期。

18.4.2. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。

- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类型的基础架构上执行受限网络安装。在此过程中，您可以下载所需的内容，并使用它为镜像 registry 填充安装软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群前，您要更新镜像 registry 的内容。

18.4.3. 具有用户置备基础架构的集群的机器要求

对于包含用户置备的基础架构的集群，您必须部署所有所需的机器。

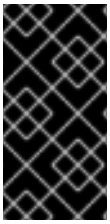
基于 RHEL 8.6 或更高版本的一个或多个 KVM 主机机器。每一台 RHEL KVM 主机机器都必须安装并运行 libvirt。虚拟机在每台 RHEL KVM 主机下调配。

18.4.3.1. 所需的机器

最小的 OpenShift Container Platform 集群需要以下主机：

表 18.40. 最低所需的主机

主机	描述
一个临时 bootstrap 机器	集群需要 bootstrap 机器在三台 control plane 机器上部署 OpenShift Container Platform 集群。您可在安装集群后删除 bootstrap 机器。
三台 control plane 机器	control plane 机器运行组成 control plane 的 Kubernetes 和 OpenShift Container Platform 服务。
至少两台计算机器，也称为 worker 机器。	OpenShift Container Platform 用户请求的工作负载在计算机器上运行。



重要

要提高集群的高可用性，请在至少两台物理机器的不同 RHEL 实例上分发 control plane 机器。

bootstrap、control plane 和计算机器必须使用 Red Hat Enterprise Linux CoreOS(RHCOS)作为操作系统。

查看 红帽企业 Linux 技术功能和限制。

18.4.3.2. 网络连接要求

OpenShift Container Platform 安装程序创建 Ignition 文件，这是所有 Red Hat Enterprise Linux CoreOS(RHCOS)虚拟机所必需的。OpenShift Container Platform 的自动安装由 bootstrap 机器执行。它在每个节点上启动 OpenShift Container Platform 安装，启动 Kubernetes 集群，然后完成。在这个 bootstrap 中，虚拟机必须通过 DHCP 服务器或静态 IP 地址建立网络连接。

18.4.3.3. IBM Z 网络连接要求

要在 RHEL KVM 中安装 IBM Z®，您需要：

- 使用 OSA 或 RoCE 网络适配器配置的 RHEL KVM 主机。
- 在 libvirt 中使用桥接网络的 RHEL KVM 主机或 MacVTap 将网络连接到客户机。

  请参阅 虚拟网络连接的类型。

18.4.3.4. 主机机器资源要求

环境中的 RHEL KVM 主机必须满足以下要求，才能托管您计划用于 OpenShift Container Platform 环境的虚拟机。请参阅 开始使用虚拟化。

您可以在以下 IBM® 硬件上安装 OpenShift Container Platform 版本 4.16：

- IBM® z16 （所有型号）、IBM® z15 （所有型号）、IBM® z14 （所有型号）

- IBM® LinuxONE 4 （所有型号）、IBM® LinuxONE （所有型号）、IBM® LinuxONE Emperor II、IBM® LinuxONE Rockhopper II

18.4.3.5. 最低 IBM Z 系统环境

硬件要求

- Linux(IFL)等效的、启用了 SMT2 的 Linux 集成设施，每个群集都启用了 SMT2。

- 至少一个网络连接连接到 LoadBalancer 服务，并为集群外的流量提供数据。

注意

您可以使用专用或共享的 IFL 来分配足够的计算资源。资源共享是 IBM Z® 的关键优势之一。但是，您必须正确调整每个虚拟机监控程序层上的容量，并确保每个 OpenShift Container Platform 集群都有足够资源。

重要

由于集群的整体性能会受到影响，用于设置 OpenShift Container Platform 集群的 LPAR 必须提供足够的计算容量。就此而言，管理程序级别上的 LPAR 权重管理、授权和 CPU 共享扮演着重要角色。

操作系统要求

- 使用 KVM 在 RHEL 8.6 或更高版本上运行的一个 LPAR，由 libvirt 管理

在 RHEL KVM 主机上设置：

- 用于 OpenShift Container Platform control plane 机器的三台客户机虚拟机

- 用于 OpenShift Container Platform 计算机器的两个客户机虚拟机

- 一个客户虚拟机作为临时 OpenShift Container Platform bootstrap 机器

18.4.3.6. 最低资源要求

每个集群虚拟机都必须满足以下最低要求：

虚拟机	操作系统	vCPU [1]	虚拟内存	Storage	IOPS
bootstrap	RHCOS	4	16 GB	100 GB	N/A
Control plane（控制平面）	RHCOS	4	16 GB	100 GB	N/A
Compute	RHCOS	2	8 GB	100 GB	N/A

1. 当启用 SMT-2 时，一个物理核心(IFL)提供两个逻辑核心（线程）。管理程序可以提供两个或多个 vCPU。

18.4.3.7. 首选 IBM Z 系统环境

硬件要求

- 三个 LPARS，每个都相当于 6 个 IFL（每个集群启用了 SMT2）。
- 两个网络连接连接到 LoadBalancer 服务，并为集群外的流量提供数据。

操作系统要求

- 为获得高可用性，在 RHEL 8.6 或更高版本上运行的两个或者三个 LPAR 使用 KVM（由 libvirt 管理）。

在 RHEL KVM 主机上设置：

- 3 个用于 OpenShift Container Platform control plane 机器的虚拟机，分布在 RHEL KVM 主机中。

- 至少 6 个用于 OpenShift Container Platform 计算机器的虚拟机，分布在 RHEL KVM 主机中。

- 一个客户虚拟机作为临时 OpenShift Container Platform bootstrap 机器。

- 要确保过量使用环境中组件的可用性，请使用 cpu_shares 增加 control plane 的优先级。如果存在基础架构节点，则对它们执行相同的操作。请参阅 IBM® 文档中的 schedinfo。

18.4.3.8. 首选资源要求

每个集群虚拟机的首选要求如下：

虚拟机	操作系统	vCPU	虚拟内存	Storage
bootstrap	RHCOS	4	16 GB	120 GB
Control plane（控制平面）	RHCOS	8	16 GB	120 GB
Compute	RHCOS	6	8 GB	120 GB

18.4.3.9. 证书签名请求管理

在使用您置备的基础架构时，集群只能有限地访问自动机器管理，因此您必须提供一种在安装后批准集群证书签名请求 (CSR) 的机制。kube-controller-manager 只能批准 kubelet 客户端 CSR。machine-approver 无法保证使用 kubelet 凭证请求的提供证书的有效性，因为它不能确认是正确的机器发出了该请求。您必须决定并实施一种方法，以验证 kubelet 提供证书请求的有效性并进行批准。

其他资源

- IBM Z® 和 IBM® LinuxONE 环境的推荐主机实践

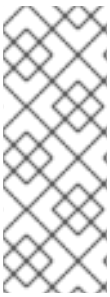
18.4.3.10. 用户置备的基础架构对网络的要求

所有 Red Hat Enterprise Linux CoreOS(RHCOS)机器都需要在启动时在 initramfs 中配置联网，以获取它们的 Ignition 配置文件。

在初次启动过程中，机器需要 IP 地址配置，该配置通过 DHCP 服务器或静态设置，提供所需的引导

选项。建立网络连接后，机器会从 HTTP 或 HTTPS 服务器下载 Ignition 配置文件。然后，Ignition 配置文件用于设置每台机器的确切状态。Machine Config Operator 在安装后完成对机器的更多更改，如应用新证书或密钥。

建议使用 DHCP 服务器对集群机器进行长期管理。确保 DHCP 服务器已配置为向集群机器提供持久的 IP 地址、DNS 服务器信息和主机名。



注意

如果用户置备的基础架构没有 DHCP 服务，您可以在 RHCOS 安装时向节点提供 IP 网络配置和 DNS 服务器地址。如果要从 ISO 镜像安装，这些参数可作为引导参数传递。如需有关静态 IP 置备和高级网络选项的更多信息，请参阅 *安装 RHCOS 并启动 OpenShift Container Platform bootstrap 过程* 部分。

Kubernetes API 服务器必须能够解析集群机器的节点名称。如果 API 服务器和 worker 节点位于不同的区域中，您可以配置默认 DNS 搜索区域，以允许 API 服务器解析节点名称。另一种支持的方法是始终通过节点对象和所有 DNS 请求中的完全限定域名引用主机。

18.4.3.10.1. 通过 DHCP 设置集群节点主机名

在 Red Hat Enterprise Linux CoreOS(RHCOS)机器上，主机名是通过 NetworkManager 设置的。默认情况下，机器通过 DHCP 获取其主机名。如果主机名不是由 DHCP 提供，请通过内核参数或者其它方法进行静态设置，请通过反向 DNS 查找获取。反向 DNS 查找在网络初始化后进行，可能需要一些时间来解决。其他系统服务可以在此之前启动，并将主机名检测为 localhost 或类似的内容。您可以使用 DHCP 为每个集群节点提供主机名来避免这种情况。

另外，通过 DHCP 设置主机名可以绕过实施 DNS split-horizon 的环境中的手动 DNS 记录名称配置错误。

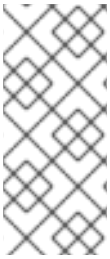
18.4.3.10.2. 网络连接要求

您必须配置机器之间的网络连接，以允许 OpenShift Container Platform 集群组件进行通信。每台机器都必须能够解析集群中所有其他机器的主机名。

本节详细介绍了所需的端口。

重要

在连接的 OpenShift Container Platform 环境中，所有节点都需要访问互联网才能为平台容器拉取镜像，并向红帽提供遥测数据。



注意

RHEL KVM 主机必须配置为使用 libvirt 或 MacVTap 中的桥接网络，才能将网络连接到虚拟机。虚拟机必须有权访问网络，该网络附加到 RHEL KVM 主机。KVM 中的虚拟网络（如网络地址转换(NAT)）并不是受支持的配置。

表 18.41. 用于全机器到所有机器通信的端口

协议	port	描述
ICMP	N/A	网络可访问性测试
TCP	**1936**	指标
	9000-**9999**	主机级别的服务，包括端口 9 **100**-**9101** 上的节点导出器，以及端口 **9099** 上的 Cluster Version Operator。
	10250-**10259**	Kubernetes 保留的默认端口
UDP	**4789**	VXLAN
	6081	Geneve
	9000-**9999**	主机级别的服务，包括端口 **9100**到**9101** 上的节点导出器。
	500	IPsec IKE 数据包
	4500	IPsec NAT-T 数据包
	123	UDP 端口 **123** 上的网络时间协议 (NTP) 如果配置了外部 NTP 时间服务器，需要打开 UDP 端口 **123**。
TCP/UDP	**30000**-**32767**	Kubernetes 节点端口
ESP	N/A	IPsec Encapsulating Security Payload(ESP)

表 18.42. 用于所有机器控制平面通信的端口

协议	port	描述
TCP	**6443**	Kubernetes API

表 18.43. control plane 机器用于 control plane 机器通信的端口

协议	port	描述
TCP	**2379-2380**	etcd 服务器和对等端口

用户置备的基础架构的 NTP 配置

OpenShift Container Platform 集群被配置为默认使用公共网络时间协议(NTP)服务器。如果要使用本地企业 NTP 服务器，或者集群部署在断开连接的网络中，您可以将集群配置为使用特定的时间服务器。如需更多信息，*请参阅配置 chrony 时间服务* 的文档。

如果 DHCP 服务器提供 NTP 服务器信息，Red Hat Enterprise Linux CoreOS(RHCOS)机器上的 chrony 时间服务会读取信息，并可以把时钟与 NTP 服务器同步。

其他资源

- **配置 chrony 时间服务**

18.4.3.11. 用户置备的 DNS 要求

在 OpenShift Container Platform 部署中，以下组件需要 DNS 名称解析：

- **The Kubernetes API**
- **OpenShift Container Platform 应用程序通配符**
- **bootstrap、control plane 和计算机器**

Kubernetes API、bootstrap 机器、control plane 机器和计算机器也需要反向 DNS 解析。

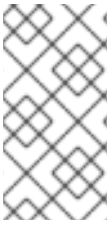
DNS A/AAAA 或 CNAME 记录用于名称解析，PTR 记录用于反向名称解析。反向记录很重要，因为 Red Hat Enterprise Linux CoreOS(RHCOS)使用反向记录为所有节点设置主机名，除非 DHCP 提供主

机名。另外，反向记录用于生成 OpenShift Container Platform 需要操作的证书签名请求(CSR)。

用户置备的 OpenShift Container Platform 集群需要以下 DNS 记录，这些记录必须在安装前就位。在每个记录中，<cluster_name> 是集群名称，<base_domain> 是您在 install-config.yaml 文件中指定的基域。完整的 DNS 记录采用以下形式： <component>.<cluster_name>.<base_domain>.。

表 18.44. 所需的 DNS 记录

组件	记录	描述
Kubernetes API	**api.<cluster_name>.<base_domain>.**	DNS A/AAAA 或 CNAME 记录，以及用于标识 API 负载均衡器的 DNS PTR 记录。这些记录必须由集群外的客户端和集群中的所有节点解析。
	api-int.<cluster_name>.<base_domain>.	DNS A/AAAA 或 CNAME 记录，以及用于内部标识 API 负载均衡器的 DNS PTR 记录。这些记录必须可以从集群中的所有节点解析。  **重要** API 服务器必须能够根据 Kubernetes 中记录的主机名解析 worker 节点。如果 API 服务器无法解析节点名称，则代理的 API 调用会失败，且您无法从 pod 检索日志。
Routes	***.apps.<cluster_name>.<base_domain>.**	通配符 DNS A/AAAA 或 CNAME 记录，指向应用程序入口负载均衡器。应用程序入口负载均衡器以运行 Ingress Controller Pod 的机器为目标。默认情况下，Ingress Controller Pod 在计算机器上运行。这些记录必须由集群外的客户端和集群中的所有节点解析。 例如，console **-openshift-console.apps.<cluster_name>.<base_domain>** 用作到 OpenShift Container Platform 控制台的通配符路由。
bootstrap 机器	**bootstrap.<cluster_name>.<base_domain>.**	DNS A/AAAA 或 CNAME 记录，以及用于标识 bootstrap 机器的 DNS PTR 记录。这些记录必须由集群中的节点解析。
control plane 机器	**<control_plane><n>.<cluster_name>.<base_domain>.**	DNS A/AAAA 或 CNAME 记录，以识别 control plane 节点的每台机器。这些记录必须由集群中的节点解析。
计算机器	**<compute><n>.<cluster_name>.<base_domain>.**	DNS A/AAAA 或 CNAME 记录，用于识别 worker 节点的每台机器。这些记录必须由集群中的节点解析。



注意

在 OpenShift Container Platform 4.4 及更新的版本中，您不需要在 DNS 配置中指定 etcd 主机和 SRV 记录。

提示

您可以使用 dig 命令验证名称和反向名称解析。如需了解详细的 *验证步骤，请参阅为用户置备的基础架构验证 DNS 解析* 一节。

18.4.3.11.1. 用户置备的集群的 DNS 配置示例

本节提供 A 和 PTR 记录配置示例，它们满足了在用户置备的基础架构上部署 OpenShift Container Platform 的 DNS 要求。样本不是为选择一个 DNS 解决方案提供建议。

在这个示例中，集群名称为 ocp4，基域是 example.com。

用户置备的集群的 DNS A 记录配置示例

以下示例是 BIND 区域文件，其中显示了用户置备的集群中名称解析的 A 记录示例。

例 18.7. DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
  2019070700 ; serial
  3H  ; refresh (3 hours)
  30M  ; retry (30 minutes)
  2W  ; expiry (2 weeks)
  1W )  ; minimum (1 week)
 IN NS ns1.example.com.
 IN MX 10 smtp.example.com.
;
;
ns1.example.com.  IN A 192.168.1.5
smtp.example.com.  IN A 192.168.1.5
;
helper.example.com.  IN A 192.168.1.5
helper.ocp4.example.com. IN A 192.168.1.5
;
api.ocp4.example.com.  IN A 192.168.1.5 1
api-int.ocp4.example.com. IN A 192.168.1.5 2
;
*.apps.ocp4.example.com. IN A 192.168.1.5 3
;
```

```
bootstrap.ocp4.example.com. IN A 192.168.1.96 4
;
control-plane0.ocp4.example.com. IN A 192.168.1.97 5
control-plane1.ocp4.example.com. IN A 192.168.1.98 6
control-plane2.ocp4.example.com. IN A 192.168.1.99 7
;
compute0.ocp4.example.com. IN A 192.168.1.11 8
compute1.ocp4.example.com. IN A 192.168.1.7 9
;
;EOF
```

1

为 Kubernetes API 提供名称解析。记录引用 API 负载均衡器的 IP 地址。

2

为 Kubernetes API 提供名称解析。记录引用 API 负载均衡器的 IP 地址，用于内部集群通信。

3

为通配符路由提供名称解析。记录引用应用程序入口负载均衡器的 IP 地址。应用程序入口负载均衡器以运行 Ingress Controller Pod 的机器为目标。默认情况下，Ingress Controller Pod 在计算机器上运行。

注意

在这个示例中，将相同的负载均衡器用于 Kubernetes API 和应用入口流量。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。

4

为 bootstrap 机器提供名称解析。

5 6 7

为 control plane 机器提供名称解析。

8 9

为计算机器提供名称解析。

用户置备的集群的 DNS PTR 记录配置示例

以下示例 BIND 区域文件显示了用户置备的集群中反向名称解析的 PTR 记录示例。

例 18.8. 反向记录的 DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
  2019070700 ; serial
  3H  ; refresh (3 hours)
  30M  ; retry (30 minutes)
  2W  ; expiry (2 weeks)
  1W )  ; minimum (1 week)
 IN NS ns1.example.com.
;
5.1.168.192.in-addr.arpa. IN PTR api.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. IN PTR api-int.ocp4.example.com. 2
;
96.1.168.192.in-addr.arpa. IN PTR bootstrap.ocp4.example.com. 3
;
97.1.168.192.in-addr.arpa. IN PTR control-plane0.ocp4.example.com. 4
98.1.168.192.in-addr.arpa. IN PTR control-plane1.ocp4.example.com. 5
99.1.168.192.in-addr.arpa. IN PTR control-plane2.ocp4.example.com. 6
;
11.1.168.192.in-addr.arpa. IN PTR compute0.ocp4.example.com. 7
7.1.168.192.in-addr.arpa. IN PTR compute1.ocp4.example.com. 8
;
;EOF
```

1 为 Kubernetes API 提供反向 DNS 解析。PTR 记录引用 API 负载均衡器的记录名称。

2 为 Kubernetes API 提供反向 DNS 解析。PTR 记录引用 API 负载均衡器的记录名称，用于内部集群通信。

3 为 bootstrap 机器提供反向 DNS 解析。

4 5 6 为 control plane 机器提供反向 DNS 解析。

7 8

为计算机器提供反向 DNS 解析。

注意

OpenShift Container Platform 应用程序通配符不需要 PTR 记录。

18.4.3.12. 用户置备的基础架构的负载均衡要求

在安装 OpenShift Container Platform 前，您必须置备 API 和应用程序入口负载均衡基础架构。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。

注意

如果要使用 Red Hat Enterprise Linux (RHEL) 实例部署 API 和应用程序入口负载均衡器，您必须单独购买 RHEL 订阅。

负载平衡基础架构必须满足以下要求：

1. API 负载均衡器 ：提供一个通用端点，供用户（包括人工和机器）与平台交互和配置。配置以下条件：

   - 仅第 4 层负载均衡.这可被称为 Raw TCP 或 SSL Passthrough 模式。
   - 无状态负载平衡算法。这些选项根据负载均衡器的实施而有所不同。

重要

不要为 API 负载均衡器配置会话持久性。为 Kubernetes API 服务器配置会话持久性可能会导致出现过量 OpenShift Container Platform 集群应用程序流量，以及过量的在集群中运行的 Kubernetes API。

在负载均衡器的前端和后端配置以下端口：

表 18.45. API 负载均衡器

port	后端机器（池成员）	internal	外部	描述
6443	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。您必须为 API 服务器健康检查探测配置 /readyz 端点。	X	X	Kubernetes API 服务器
22623	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。	X		机器配置服务器

注意

负载均衡器必须配置为，从 API 服务器关闭 /readyz 端点到从池中移除 API 服务器实例时最多需要 30 秒。在 /readyz 返回错误或健康后的时间范围内，端点必须被删除或添加。每 5 秒或 10 秒探测一次，有两个成功请求处于健康状态，三个成为不健康的请求是经过良好测试的值。

2. 应用程序入口负载均衡器 ：为应用程序流量从集群外部流提供入口点。OpenShift Container Platform 集群需要正确配置入口路由器。

配置以下条件：

- 仅第 4 层负载均衡.这可被称为 Raw TCP 或 SSL Passthrough 模式。
- 建议根据可用选项以及平台上托管的应用程序类型，使用基于连接的或基于会话的持久性。

提示

如果应用程序入口负载均衡器可以看到客户端的真实 IP 地址，启用基于 IP 的会话持久性可以提高使用端到端 TLS 加密的应用程序的性能。

在负载均衡器的前端和后端配置以下端口：

表 18.46. 应用程序入口负载均衡器

port	后端机器（池成员）	internal	外部	描述
443	默认情况下，运行 Ingress Controller Pod、计算或 worker 的机器。	X	X	HTTPS 流量
80	默认情况下，运行 Ingress Controller Pod、计算或 worker 的机器。	X	X	HTTP 流量

注意

如果要部署一个带有零计算节点的三节点集群，Ingress Controller Pod 在 control plane 节点上运行。在三节点集群部署中，您必须配置应用程序入口负载均衡器，将 HTTP 和 HTTPS 流量路由到 control plane 节点。

18.4.3.12.1. 用户置备的集群的负载均衡器配置示例

本节提供了一个满足用户置备集群的负载均衡要求的 API 和应用程序入口负载均衡器配置示例。示例是 HAProxy 负载均衡器的 /etc/haproxy/haproxy.cfg 配置。这个示例不是为选择一个负载平衡解决方案提供建议。

在这个示例中，将相同的负载均衡器用于 Kubernetes API 和应用入口流量。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。

注意

如果您使用 HAProxy 作为负载均衡器，并且 SELinux 设置为 enforcing，您必须通过运行 setsebool -P haproxy_connect_any=1 来确保 HAProxy 服务可以绑定到配置的 TCP 端口。

例 18.9. API 和应用程序入口负载均衡器配置示例

```
global
  log         127.0.0.1 local2
  pidfile     /var/run/haproxy.pid
  maxconn     4000
  daemon
defaults
  mode                    http
  log                     global
```

```
  option                  dontlognull
  option http-server-close
  option                  redispatch
  retries                 3
  timeout http-request    10s
  timeout queue           1m
  timeout connect         10s
  timeout client          1m
  timeout server          1m
  timeout http-keep-alive 10s
  timeout check           10s
  maxconn                 3000
listen api-server-6443 (1)
  bind *:6443
  mode tcp
  option  httpchk GET /readyz HTTP/1.0
  option  log-health-checks
  balance roundrobin
  server bootstrap bootstrap.ocp4.example.com:6443 verify none check check-ssl inter 10s
fall 2 rise 3 backup (2)
  server master0 master0.ocp4.example.com:6443 weight 1 verify none check check-ssl
inter 10s fall 2 rise 3
  server master1 master1.ocp4.example.com:6443 weight 1 verify none check check-ssl
inter 10s fall 2 rise 3
  server master2 master2.ocp4.example.com:6443 weight 1 verify none check check-ssl
inter 10s fall 2 rise 3
listen machine-config-server-22623 (3)
  bind *:22623
  mode tcp
  server bootstrap bootstrap.ocp4.example.com:22623 check inter 1s backup (4)
  server master0 master0.ocp4.example.com:22623 check inter 1s
  server master1 master1.ocp4.example.com:22623 check inter 1s
  server master2 master2.ocp4.example.com:22623 check inter 1s
listen ingress-router-443 (5)
  bind *:443
  mode tcp
  balance source
  server compute0 compute0.ocp4.example.com:443 check inter 1s
  server compute1 compute1.ocp4.example.com:443 check inter 1s
listen ingress-router-80 (6)
  bind *:80
  mode tcp
  balance source
  server compute0 compute0.ocp4.example.com:80 check inter 1s
  server compute1 compute1.ocp4.example.com:80 check inter 1s
```

(1)

端口 6443 处理 Kubernetes API 流量并指向 control plane 机器。

(2) (4)

bootstrap 条目必须在 OpenShift Container Platform 集群安装前就位，且必须在 bootstrap 过程完成后删除它们。

3

端口 22623 处理机器配置服务器流量并指向 control plane 机器。

5

端口 443 处理 HTTPS 流量，并指向运行 Ingress Controller pod 的机器。默认情况下，Ingress Controller Pod 在计算机器上运行。

6

端口 80 处理 HTTP 流量，并指向运行 Ingress Controller pod 的机器。默认情况下，Ingress Controller Pod 在计算机器上运行。

注意

如果要部署一个带有零计算节点的三节点集群，Ingress Controller Pod 在 control plane 节点上运行。在三节点集群部署中，您必须配置应用程序入口负载均衡器，将 HTTP 和 HTTPS 流量路由到 control plane 节点。

提示

如果您使用 HAProxy 作为负载均衡器，您可以通过在 HAProxy 节点上运行 **netstat -nltupe** 来检查 haproxy 进程是否在侦听端口 6443、22623、443 和 80。

18.4.4. 准备用户置备的基础架构

在用户置备的基础架构上安装 OpenShift Container Platform 之前，您必须准备底层基础架构。

本节详细介绍了设置集群基础架构以准备 OpenShift Container Platform 安装所需的高级别步骤。这包括为您的集群节点配置 IP 网络和网络连接，通过防火墙启用所需的端口，以及设置所需的 DNS 和负载均衡基础架构。

准备后，集群基础架构必须满足 *带有用户置备的基础架构部分的集群要求*。

先决条件

- 您已参阅 OpenShift Container Platform 4.x Tested Integrations 页面。
- 您已查看了 *具有用户置备基础架构的集群要求部分中详述的基础架构* 要求。

流程

1. 如果您使用 DHCP 向集群节点提供 IP 网络配置，请配置 DHCP 服务。

   a. 将节点的持久 IP 地址添加到您的 DHCP 服务器配置。在您的配置中，将相关网络接口的 MAC 地址与每个节点的预期 IP 地址匹配。

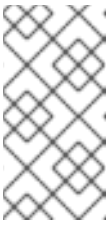
   b. 当您使用 DHCP 为集群机器配置 IP 寻址时，机器还通过 DHCP 获取 DNS 服务器信息。定义集群节点通过 DHCP 服务器配置使用的持久性 DNS 服务器地址。

   注意

   如果没有使用 DHCP 服务，则必须在 RHCOS 安装时为节点提供 IP 网络配置和 DNS 服务器地址。如果要从 ISO 镜像安装，这些参数可作为引导参数传递。如需有关静态 IP 置备和高级网络选项的更多信息，请参阅 *安装 RHCOS 并启动 OpenShift Container Platform bootstrap 过程* 部分。

   c. 在 DHCP 服务器配置中定义集群节点的主机名。有关 *主机名注意事项的详情，请参阅通过 DHCP 设置集群节点* 主机名部分。

   注意

   如果没有使用 DHCP 服务，集群节点可以通过反向 DNS 查找来获取其主机名。

2. 选择执行 Red Hat Enterprise Linux CoreOS(RHCOS)的快速跟踪安装或 Red Hat Enterprise Linux CoreOS(RHCOS)的完整安装。要进行完整安装，您必须设置 HTTP 或 HTTPS 服务器，以便提供 Ignition 文件，并将镜像安装到集群节点。对于快速跟踪安装，不需要 HTTP 或 HTTPS 服务器，但需要 DHCP 服务器。请参阅"Fast-track 安装：创建 Red Hat Enterprise Linux CoreOS(RHCOS)机器"和"Full installation: Creating Red Hat Enterprise Linux CoreOS(RHCOS)机器"部分。

3.

确保您的网络基础架构提供集群组件之间所需的网络连接。有关 *要求的详情，请参阅用户置备的基础架构* 的网络要求部分。

4. 将防火墙配置为启用 OpenShift Container Platform 集群组件进行通信所需的端口。*如需有关所需端口的详细信息，请参阅用户置备的基础架构* 部分的网络要求。

重要

默认情况下，OpenShift Container Platform 集群可以访问端口 1936，因为每个 control plane 节点都需要访问此端口。

避免使用 Ingress 负载均衡器公开此端口，因为这样做可能会导致公开敏感信息，如统计信息和指标（与 Ingress Controller 相关的统计信息和指标）。

5. 为集群设置所需的 DNS 基础架构。

   a. 为 Kubernetes API、应用程序通配符、bootstrap 机器、control plane 机器和计算机器配置 DNS 名称解析。

   b. 为 Kubernetes API、bootstrap 机器、control plane 机器和计算机器配置反向 DNS 解析。

   如需有关 *OpenShift Container Platform DNS 要求的更多信息，请参阅用户置备* DNS 要求部分。

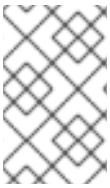
6. 验证您的 DNS 配置。

   a. 从安装节点，针对 Kubernetes API 的记录名称、通配符路由和集群节点运行 DNS 查找。验证响应中的 IP 地址是否与正确的组件对应。

   b. 从安装节点，针对负载均衡器和集群节点的 IP 地址运行反向 DNS 查找。验证响应中的记录名称是否与正确的组件对应。

   有关详细的 *DNS 验证步骤，请参阅用户置备的基础架构* 验证 DNS 解析部分。

7. 置备所需的 API 和应用程序入口负载平衡基础架构。有关 *要求的更多信息，请参阅用户置备的基础架构的负载平衡* 要求部分。

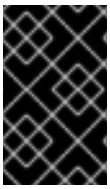


注意

某些负载平衡解决方案要求在初始化负载平衡之前，对群集节点进行 DNS 名称解析。

18.4.5. 验证用户置备的基础架构的 DNS 解析

您可以在在用户置备的基础架构上安装 OpenShift Container Platform 前验证 DNS 配置。



重要

本节中详述的验证步骤必须在安装集群前成功。

先决条件

- 已为您的用户置备的基础架构配置了所需的 DNS 记录。

流程

1. 从安装节点，针对 Kubernetes API 的记录名称、通配符路由和集群节点运行 DNS 查找。验证响应中包含的 IP 地址是否与正确的组件对应。

   a. 对 Kubernetes API 记录名称执行查询。检查结果是否指向 API 负载均衡器的 IP 地址：

   ```
   $ dig +noall +answer @<nameserver_ip> api.<cluster_name>.<base_domain> 1
   ```

   1 将 <nameserver_ip> 替换为 nameserver 的 IP 地址，<cluster_name> 替换为您的集群名称，<base_domain> 替换为您的基本域名。

   输出示例

   ```
   -
   ```

```
api.ocp4.example.com.  604800 IN A 192.168.1.5
```

b. 对 Kubernetes 内部 API 记录名称执行查询。检查结果是否指向 API 负载均衡器的 IP 地址：

```
$ dig +noall +answer @<nameserver_ip> api-int.<cluster_name>.<base_domain>
```

输出示例

```
api-int.ocp4.example.com.  604800 IN A 192.168.1.5
```

c. 测试 *.apps.<cluster_name>.<base_domain> DNS 通配符查找示例。所有应用程序通配符查询都必须解析为应用程序入口负载均衡器的 IP 地址：

```
$ dig +noall +answer @<nameserver_ip> random.apps.<cluster_name>.
<base_domain>
```

输出示例

```
random.apps.ocp4.example.com.  604800 IN A 192.168.1.5
```

注意

在示例中，将相同的负载均衡器用于 Kubernetes API 和应用程序入口流量。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。

您可以使用另一个通配符值替换 random。例如，您可以查询到 OpenShift Container Platform 控制台的路由：

```
$ dig +noall +answer @<nameserver_ip> console-openshift-console.apps.
<cluster_name>.<base_domain>
```

输出示例

```
console-openshift-console.apps.ocp4.example.com. 604800 IN A 192.168.1.5
```

d. 针对 bootstrap DNS 记录名称运行查询。检查结果是否指向 bootstrap 节点的 IP 地址：

```
$ dig +noall +answer @<nameserver_ip> bootstrap.<cluster_name>.
<base_domain>
```

输出示例

```
bootstrap.ocp4.example.com.  604800 IN A 192.168.1.96
```

e. 使用此方法对 control plane 和计算节点的 DNS 记录名称执行查找。检查结果是否与每个节点的 IP 地址对应。

2. 从安装节点，针对负载均衡器和集群节点的 IP 地址运行反向 DNS 查找。验证响应中包含的记录名称是否与正确的组件对应。

a. 对 API 负载均衡器的 IP 地址执行反向查找。检查响应是否包含 Kubernetes API 和 Kubernetes 内部 API 的记录名称：

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.5
```

输出示例

```
5.1.168.192.in-addr.arpa. 604800 IN PTR api-int.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. 604800 IN PTR api.ocp4.example.com. 2
```

1 为 Kubernetes 内部 API 提供记录名称。

2 为 Kubernetes API 提供记录名称。

注意

OpenShift Container Platform 应用程序通配符不需要 PTR 记录。针对应用程序入口负载均衡器的 IP 地址解析反向 DNS 解析不需要验证步骤。

b. 对 bootstrap 节点的 IP 地址执行反向查找。检查结果是否指向 bootstrap 节点的 DNS 记录名称：

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.96
```

输出示例

```
96.1.168.192.in-addr.arpa. 604800 IN PTR bootstrap.ocp4.example.com.
```

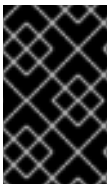
c. 使用此方法对 control plane 和计算节点的 IP 地址执行反向查找。检查结果是否与每个节点的 DNS 记录名称对应。

18.4.6. 为集群节点 SSH 访问生成密钥对

在 OpenShift Container Platform 安装过程中，您可以为安装程序提供 SSH 公钥。密钥通过它们的 Ignition 配置文件传递给 Red Hat Enterprise Linux CoreOS(RHCOS)节点，用于验证对节点的 SSH 访问。密钥添加到每个节点上 core 用户的 ~/.ssh/authorized_keys 列表中，这将启用免密码身份验证。

将密钥传递给节点后，您可以使用密钥对作为用户 核心 通过 SSH 连接到 RHCOS 节点。若要通过 SSH 访问节点，必须由 SSH 为您的本地用户管理私钥身份。

如果要通过 SSH 连接到集群节点来执行安装调试或灾难恢复，则必须在安装过程中提供 SSH 公钥。./openshift-install gather 命令还需要在集群节点上设置 SSH 公钥。



重要

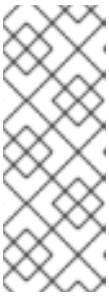
不要在生产环境中跳过这个过程，在生产环境中需要灾难恢复和调试。

流程

1. 如果您在本地计算机上没有可用于在集群节点上进行身份验证的现有 SSH 密钥对，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 ~/.ssh/id_ed25519。如果您已有密钥对，请确保您的公钥位于 ~/.ssh 目录中。



注意

如果您计划在 x86_64、ppc64le 和 s390x 架构上安装使用 RHEL 加密库（这些加密库已提交给 NIST 用于 FIPS 140-2/140-3 验证）的 OpenShift Container Platform 集群，则不要创建使用 ed25519 算法的密钥。相反，创建一个使用 rsa 或 ecdsa 算法的密钥。

2. 查看公共 SSH 密钥：

```
$ cat <path>/<file_name>.pub
```

例如，运行以下命令来查看 ~/.ssh/id_ed25519.pub 公钥：

```
$ cat ~/.ssh/id_ed25519.pub
```

3. 将 SSH 私钥身份添加到本地用户的 SSH 代理（如果尚未添加）。在集群节点上，或者要使用 ./openshift-install gather 命令，需要对该密钥进行 SSH 代理管理，才能在集群节点上进行免密码 SSH 身份验证。

注意

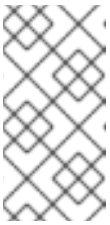
在某些发行版中，自动管理默认 SSH 私钥身份，如 ~/.ssh/id_rsa 和 ~/.ssh/id_dsa。

a. 如果 ssh-agent 进程尚未为您的本地用户运行，请将其作为后台任务启动：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```



注意

如果集群处于 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

4. 将 SSH 私钥添加到 ssh-agent ：

```
$ ssh-add <path>/<file_name> 1
```

1 指定 SSH 私钥的路径和文件名，如 ~/.ssh/id_ed25519.pub

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

后续步骤

- 安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

18.4.7. 获取安装程序

在安装 OpenShift Container Platform 之前，将安装文件下载到您置备的机器上。

先决条件

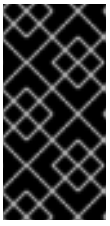
- 您有一个运行 Linux 的机器，如 Red Hat Enterprise Linux 8，本地磁盘空间为 500 MB。

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐户，请使用您的凭证登录。如果没有，请创建一个帐户。

2. 选择您的基础架构供应商。

3. 进入到安装类型的页面，下载与您的主机操作系统和架构对应的安装程序，并将该文件放在您要存储安装配置文件的目录中。



重要

安装程序会在用来安装集群的计算机上创建几个文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。

重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，请为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret。此 pull secret 允许您与所含授权机构提供的服务进行身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

18.4.8. 安装 OpenShift CLI

您可以安装 OpenShift CLI(oc)来使用命令行界面与 OpenShift Container Platform 进行交互。您可以在 Linux、Windows 或 macOS 上安装 oc。

重要

如果安装了旧版本的 oc，则可能无法使用 OpenShift Container Platform 4.16 中的所有命令。下载并安装新版本的 oc。

在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI(oc)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 产品变体 下拉列表中选择架构。
3. 从 版本 下拉列表中选择适当的版本。

4. 点 OpenShift v4.16 Linux Client 条目旁的 Download Now 来保存文件。

5. 解包存档：

```
$ tar xvf <file>
```

6. 将 oc 二进制文件放到 PATH 中的目录中。

   要查看您的 PATH，请执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 oc 命令：

```
$ oc <command>
```

在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI(oc)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。

2. 从 版本 下拉列表中选择适当的版本。

3. 点 OpenShift v4.16 Windows Client 条目旁的 Download Now 来保存文件。

4. 使用 ZIP 程序解压存档。

5. 将 oc 二进制文件移到 PATH 中的目录中。

要查看您的 PATH，请打开命令提示并执行以下命令：

```
C:\> path
```

验证

- 安装 OpenShift CLI 后，可以使用 oc 命令：

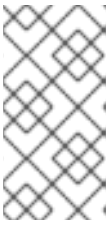
```
C:\> oc <command>
```

在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI(oc)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 版本 下拉列表中选择适当的版本。
3. 点 OpenShift v4.16 macOS Client 条目旁的 Download Now 来保存文件。



注意

对于 macOS arm64，请选择 OpenShift v4.16 macOS arm64 Client 条目。

4. 解包和解压存档。
5. 将 oc 二进制文件移到 PATH 的目录中。

要查看您的 PATH，请打开终端并执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 oc 命令：

```
$ oc <command>
```

18.4.9. 手动创建安装配置文件

安装集群要求您手动创建安装配置文件。

先决条件

- 您在本地机器上有一个 SSH 公钥来提供给安装程序。该密钥将用于在集群节点上进行 SSH 身份验证，以进行调试和灾难恢复。

- 已获取 OpenShift Container Platform 安装程序和集群的 pull secret。

流程

1. 创建一个安装目录来存储所需的安装资产：

```
$ mkdir <installation_directory>
```

重要

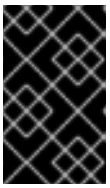
您必须创建一个目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

2. 自定义提供的 install-config.yaml 文件模板示例，并将其保存在 <installation_directory> 中。

注意

此配置文件必须命名为 install-config.yaml。

3. 备份 install-config.yaml 文件，以便您可以使用它安装多个集群。



重要

install-config.yaml 文件会在安装过程的下一步中使用。现在必须备份它。

其他资源

- IBM Z® 的安装配置参数

18.4.9.1. IBM Z 的 install-config.yaml 文件示例

您可以自定义 install-config.yaml 文件，以指定有关 OpenShift Container Platform 集群平台的更多详情，或修改所需参数的值。

```
apiVersion: v1
baseDomain: example.com (1)
compute: (2)
- hyperthreading: Enabled (3)
  name: worker
  replicas: 0 (4)
  architecture: s390x
controlPlane: (5)
  hyperthreading: Enabled (6)
  name: master
  replicas: 3 (7)
  architecture: s390x
metadata:
  name: test (8)
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14 (9)
    hostPrefix: 23 (10)
  networkType: OVNKubernetes (11)
  serviceNetwork: (12)
  - 172.30.0.0/16
platform:
  none: {} (13)
fips: false (14)
pullSecret: '{"auths": ...}' (15)
sshKey: 'ssh-ed25519 AAAA...' (16)
```

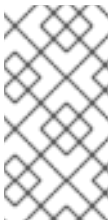
(1)

2 5

controlPlane 部分是一个单个映射，但 compute 部分是一系列映射。为满足不同数据结构的要求，compute 部分的第一行必须以连字符 - 开头，controlPlane 部分 的第一行则不以连字符开头。仅使用一个 control plane 池。

3 6

指定要启用或禁用并发多线程(SMT)还是超线程。默认情况下，启用 SMT 可提高机器中内核的性能。您可以通过将 参数值设置为 Disabled 来禁用它。如果禁用 SMT，则必须在所有集群机器中禁用它；这包括 control plane 和计算机器。



注意

默认启用并发多线程(SMT)。如果 OpenShift Container Platform 节点上没有 SMT，超线程 参数无效。

重要

如果您禁用 超线程，无论是在 OpenShift Container Platform 节点上，还是在 install-config.yaml 文件中，请确保您的容量规划考虑机器性能显著降低的情况。

4

在用户置备的基础架构上安装 OpenShift Container Platform 时，必须将这个值设置为 0。在安装程序置备的安装中，参数控制集群为您创建和管理的计算机器数量。在用户置备的安装中，您必须在完成集群安装前手动部署计算机器。

注意

如果要安装一个三节点集群，在安装 Red Hat Enterprise Linux CoreOS(RHCOS)机器时不要部署任何计算机器。

7

您添加到集群的 control plane 机器数量。由于集群使用这些值作为集群中的 etcd 端点数量，所以该值必须与您部署的 control plane 机器数量匹配。

8

您在 DNS 记录中指定的集群名称。

9

注意

类 E CIDR 范围被保留以供以后使用。要使用 Class E CIDR 范围，您必须确保您的网络环境接受 Class E CIDR 范围内的 IP 地址。

10

分配给每个节点的子网前缀长度。例如，如果 hostPrefix 设为 23，则每个节点从 given cidr 中分配 a /23 子网，这样就能有 510（2^(32 - 23)- 2）个 pod IP 地址。如果需要从外部网络访问节点，请配置负载均衡器和路由器来管理流量。

11

要安装的集群网络插件。默认值 OVNKubernetes 是唯一支持的值。

12

用于服务 IP 地址的 IP 地址池。您只能输入一个 IP 地址池。此块不得与现有物理网络重叠。如果您需要从外部网络访问服务，请配置负载均衡器和路由器来管理流量。

13

您必须将平台设置为 none。您无法为 IBM Z® 基础架构提供额外的平台配置变量。

重要

使用平台类型 none 安装的集群无法使用一些功能，如使用 Machine API 管理计算机器。即使附加到集群的计算机器安装在通常支持该功能的平台上，也会应用这个限制。在安装后无法更改此参数。

14

是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 Red Hat Enterprise Linux (RHEL) 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅在 FIPS 模式中安装该系统。

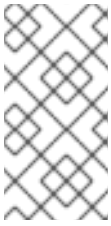
当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS)时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。

15

Red Hat OpenShift Cluster Manager 的 pull secret。此 pull secret 允许您与所含授权机构提供的服务进行身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

16

Red Hat Enterprise Linux CoreOS(RHCOS)中 core 用户的 SSH 公钥。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 ssh-agent 进程使用的 SSH 密钥。

18.4.9.2. 在安装过程中配置集群范围的代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 install-config.yaml 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 install-config.yaml 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 Proxy 对象的 spec.noProxy 字段中添加站点来绕过代理。

注意

Proxy 对象 status.noProxy 字段使用安装配置中的 networking.machineNetwork[].cidr、networking.clusterNetwork[].cidr 和 networking.serviceNetwork[] 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，Proxy 对象 status.noProxy 字段也会使用实例元数据端点填充(169.254.169.254)。

流程

1. 编辑 install-config.yaml 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 http。

2. 用于创建集群外 HTTPS 连接的代理 URL。

3. 要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 . 以仅匹配子域。例如，.y.com 匹配 x.y.com，但不匹配 y.com。使用 * 绕过所有目的地的代理。

4. 如果提供，安装程序会在 openshift-config 命名空间中生成名为 user-ca-bundle 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 trusted-ca-bundle 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， Proxy 对象的 trustedCA 字段中

也会引用此配置映射。additionalTrustBundle 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

5 可选：决定 Proxy 对象的配置以引用 trustedCA 字段中 user-ca-bundle 配置映射的策略。允许的值是 Proxyonly 和 Always。仅在配置了 http/https 代理时，使用 Proxyonly 引用 user-ca-bundle 配置映射。使用 Always 始终引用 user-ca-bundle 配置映射。默认值为 Proxyonly。

注意

安装程序不支持代理的 readinessEndpoints 字段。

注意

如果安装程序超时，重启并使用安装程序的 wait-for 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. 保存该文件并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 cluster 的集群范围代理，该代理 使用 提供的 install-config.yaml 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 cluster Proxy 对象，但它会有一个空 spec。

注意

只支持名为 cluster 的 Proxy 对象，且无法创建额外的代理。

18.4.9.3. 配置三节点集群

另外，您可以在由三台 control plane 机器组成的最少三个节点集群中部署零台计算机器。这为集群管理员和开发人员提供了更小、效率更高的集群，用于测试、开发和生产。

在三节点 OpenShift Container Platform 环境中，三台 control plane 机器可以调度，这意味着应用程序工作负载被调度到它们上运行。

先决条件

- 您有一个现有的 install-config.yaml 文件。

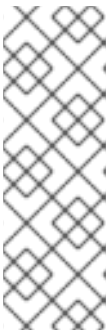
流程

- 确保 install-config.yaml 文件中的计算副本数量设置为 0，如以下 计算 小节所示：

```
compute:
- name: worker
  platform: {}
  replicas: 0
```

注意

在用户置备的基础架构上安装 OpenShift Container Platform 时，无论您要部署的计算机器数量有多少，您必须将计算机器的 replicas 参数值设置为 0。在安装程序置备的安装中，参数控制集群为您创建和管理的计算机器数量。这不适用于手动部署计算机器的用户置备安装。



注意

control plane 节点的首选资源是 6 个 vCPU 和 21 GB。对于三个 control plane 节点，这是相当于至少五节点集群的内存 + vCPU。您应该为三个节点提供支持，每个节点安装在一个 120 GB 的磁盘上，并且启用了三个 SMT2 的 IFL。测试最小的设置是每个 control plane 节点在 120 GB 磁盘上的三个 vCPU 和 10 GB。

对于三节点集群安装，请按照以下步骤执行：

- 如果要部署一个带有零计算节点的三节点集群，Ingress Controller Pod 在 control plane 节点上运行。在三节点集群部署中，您必须配置应用程序入口负载均衡器，将 HTTP 和 HTTPS 流量路由到 control plane 节点。如需更多信息，*请参阅用户置备的基础架构的负载平衡要求* 部分。

- 在以下步骤中创建 Kubernetes 清单文件时，请确保 <installation_directory>/manifests/cluster-scheduler-02-config.yml 文件中的 mastersSchedulable 参数被设置为 true。这可让应用程序工作负载在 control plane 节点上运行。

- 在创建 Red Hat Enterprise Linux CoreOS(RHCOS)机器时，不要部署任何计算节点。

18.4.10. Cluster Network Operator 配置

集群网络的配置作为 Cluster Network Operator(CNO)配置的一部分指定，并存储在名为 cluster 的自定义资源(CR)对象中。CR 指定 operator.openshift.io API 组中的 Network API 的字段。

CNO 配置在集群安装过程中从 Network.config.openshift.io API 组中的 Network API 继承以下字段：

clusterNetwork

从中分配 Pod IP 地址的 IP 地址池。

serviceNetwork

服务的 IP 地址池.

defaultNetwork.type

集群网络插件。OVNKubernetes 是安装期间唯一支持的插件。

您可以通过在名为 cluster 的 CNO 对象中设置 defaultNetwork 对象的字段来为集群指定集群网络插件配置。

18.4.10.1. Cluster Network Operator 配置对象

下表中描述了 Cluster Network Operator(CNO)的字段：

表 18.47. Cluster Network Operator 配置对象

字段	类型	描述
metadata.name	字符串	CNO 对象的名称。这个名称始终是 集群。

字段	类型	描述
spec.clusterNetwork	**array**	用于指定从哪些 IP 地址块分配 Pod IP 地址以及集群中每个节点的子网前缀长度的列表。例如： spec: clusterNetwork: - cidr: 10.128.0.0/19 hostPrefix: 23 - cidr: 10.128.32.0/19 hostPrefix: 23
spec.serviceNetwork	**array**	服务的 IP 地址块。OpenShift SDN 和 OVN-Kubernetes 网络插件只支持服务网络的一个 IP 地址块。例如： spec: serviceNetwork: - 172.30.0.0/14 您只能在创建清单前在 **install-config.yaml** 文件中自定义此字段。该值在清单文件中是只读的。
spec.defaultNetwork	**object**	为集群网络配置网络插件。
spec.kubeProxyConfig	**object**	此对象的字段指定 kube-proxy 配置。如果使用 OVN-Kubernetes 集群网络供应商，则 kube-proxy 配置不会起作用。

defaultNetwork 对象配置

下表列出了 defaultNetwork 对象的值：

表 18.48. defaultNetwork 对象

字段	类型	描述
type	**字符串**	**OVNKubernetes**。Red Hat OpenShift Networking 网络插件在安装过程中被选择。此值在集群安装后无法更改。 **注意** OpenShift Container Platform 默认使用 OVN-Kubernetes 网络插件。OpenShift SDN 不再作为新集群的安装选择提供。
ovnKubernetesConfig	**object**	此对象仅对 OVN-Kubernetes 网络插件有效。

配置 OVN-Kubernetes 网络插件

下表描述了 OVN-Kubernetes 网络插件的配置字段：

表 18.49. ovnKubernetesConfig object

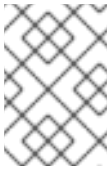
字段	类型	描述
mtu	**integer**	Geneve（通用网络虚拟化封装）覆盖网络的最大传输单元 (MTU)。这根据主网络接口的 MTU 自动探测。您通常不需要覆盖检测到的 MTU。 如果自动探测的值不是您期望的值，请确认节点上主网络接口上的 MTU 是否正确。您不能使用这个选项更改节点上主网络接口的 MTU 值。 如果集群中不同节点需要不同的 MTU 值，则必须将此值设置为 **比** 集群中的最低 MTU 值小 100。例如，如果集群中的某些节点的 MTU 为 **9001**，而某些节点的 MTU 为 **1500**，则必须将此值设置为 **1400**。
genevePort	**integer**	用于所有 Geneve 数据包的端口。默认值为 **6081**。此值在集群安装后无法更改。
ipsecConfig	**object**	指定用于自定义 IPsec 配置的配置对象。
ipv4	**object**	为 IPv4 设置指定配置对象。
ipv6	**object**	为 IPv6 设置指定配置对象。
policyAuditConfig	**object**	指定用于自定义网络策略审计日志的配置对象。如果未设置，则使用默认的审计日志设置。
gatewayConfig	**object**	可选：指定一个配置对象来自定义如何将出口流量发送到节点网关。  **注意** 在迁移出口流量时，工作负载和服务流量会受到一定影响，直到 Cluster Network Operator (CNO) 成功推出更改。

表 18.50. ovnKubernetesConfig.ipv4 object

字段	类型	描述

字段	类型	描述
internalTransitSwitchSubnet	字符串	如果您的现有网络基础架构与 **100.88.0.0/16** IPv4 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。启用东西流量的分布式传输交换机的子网。此子网不能与 OVN-Kubernetes 或主机本身使用的任何其他子网重叠。必须足够大，以适应集群中的每个节点一个 IP 地址。 默认值为 **100.88.0.0/16**。
internalJoinSubnet	字符串	如果您的现有网络基础架构与 **100.64.0.0/16** IPv4 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。您必须确保 IP 地址范围没有与 OpenShift Container Platform 安装使用的任何其他子网重叠。IP 地址范围必须大于可添加到集群的最大节点数。例如，如果 **clusterNetwork.cidr** 值为 **10.128.0.0/14**，并且 **clusterNetwork.hostPrefix** 值为 **/23**，则最大节点数量为 **2^(23-14)=512**。 默认值为 **100.64.0.0/16**。

表 18.51. ovnKubernetesConfig.ipv6 object

字段	类型	描述
internalTransitSwitchSubnet	字符串	如果您的现有网络基础架构与 **fd97::/64** IPv6 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。启用东西流量的分布式传输交换机的子网。此子网不能与 OVN-Kubernetes 或主机本身使用的任何其他子网重叠。必须足够大，以适应集群中的每个节点一个 IP 地址。 默认值为 **fd97::/64**。
internalJoinSubnet	字符串	如果您的现有网络基础架构与 **fd98::/64** IPv6 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。您必须确保 IP 地址范围没有与 OpenShift Container Platform 安装使用的任何其他子网重叠。IP 地址范围必须大于可添加到集群的最大节点数。 默认值为 **fd98::/64**。

表 18.52. policyAuditConfig object

字段	类型	描述
rateLimit	整数	每个节点每秒生成一次的消息数量上限。默认值为每秒 **20** 条消息。

字段	类型	描述
maxFileSize	整数	审计日志的最大大小，以字节为单位。默认值为 **50000000** 或 50 MB。
maxLogFiles	整数	保留的日志文件的最大数量。
目的地	字符串	以下附加审计日志目标之一： **libc** 主机上的 journald 进程的 libc **syslog（）** 函数。 **UDP:<host>:<port>** 一个 syslog 服务器。将 **<host>:<port>** 替换为 **syslog 服务器的主机** 和端口。 **Unix:<file>** 由 **<file>** 指定的 Unix 域套接字文件。 **null** 不要将审计日志发送到任何其他目标。
syslogFacility	字符串	syslog 工具，如 as **kern**，如 RFC5424 定义。默认值为 **local0。**

表 18.53. gatewayConfig object

字段	类型	描述
routingViaHost	**布尔值**	将此字段设置为 **true**，将来自 pod 的出口流量发送到主机网络堆栈。对于依赖于在内核路由表中手动配置路由的高级别安装和应用程序，您可能需要将出口流量路由到主机网络堆栈。默认情况下，出口流量在 OVN 中进行处理以退出集群，不受内核路由表中的特殊路由的影响。默认值为 **false**。 此字段与 Open vSwitch 硬件卸载功能有交互。如果将此字段设置为 **true**，则不会获得卸载的性能优势，因为主机网络堆栈会处理出口流量。
ipForwarding	**object**	您可以使用 **Network** 资源中的 **ipForwarding** 规格来控制 OVN-Kubernetes 管理接口上所有流量的 IP 转发。指定 **Restricted** 只允许 Kubernetes 相关流量的 IP 转发。指定 **Global** 以允许转发所有 IP 流量。对于新安装，默认值为 **Restricted**。对于到 OpenShift Container Platform 4.14 或更高版本的更新，默认值为 **Global**。
ipv4	**object**	可选：指定一个对象来为主机配置内部 OVN-Kubernetes 伪装地址，以服务 IPv4 地址的流量。
ipv6	**object**	可选：指定一个对象来为主机配置内部 OVN-Kubernetes 伪装地址，以服务 IPv6 地址的流量。

表 18.54. gatewayConfig.ipv4 对象

字段	类型	描述
internalMasqueradeSubnet	**字符串**	内部使用的伪装 IPv4 地址，以启用主机服务流量。主机配置了这些 IP 地址和共享网关网桥接口。默认值为 **169.254.169.0/29**。

表 18.55. gatewayConfig.ipv6 对象

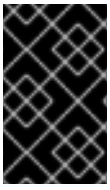
字段	类型	描述
internalMasqueradeSubnet	**字符串**	内部使用的伪装 IPv6 地址，以启用主机服务流量。主机配置了这些 IP 地址和共享网关网桥接口。默认值为 **fd69::/125**。

表 18.56. ipsecConfig 对象

字段	类型	描述
模式	**字符串**	指定 IPsec 实现的行为。必须是以下值之一： • **Disabled**: 在集群节点上不启用 IPsec。 • **External** ：对于带有外部主机的网络流量，启用 IPsec。 • **Full**: IPsec 为带有外部主机的 pod 流量和网络流量启用 IPsec。

启用 IPSec 的 OVN-Kubernetes 配置示例

```
defaultNetwork:
  type: OVNKubernetes
  ovnKubernetesConfig:
    mtu: 1400
    genevePort: 6081
      ipsecConfig:
        mode: Full
```



重要

使用 OVNKubernetes 可能会导致 IBM Power® 上的堆栈耗尽问题。

kubeProxyConfig 对象配置（仅限 OpenShiftSDN 容器网络接口）

kubeProxyConfig 对象的值在下表中定义：

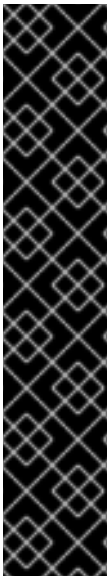
表 18.57. kubeProxyConfig object

字段	类型	描述
iptablesSyncPeriod	**字符串**	**iptables** 规则的刷新周期。默认值为 **30s**。有效的后缀包括 **s**、**m** 和 **h**，具体参见 Go 时间 包 文档。  **注意** 由于 OpenShift Container Platform 4.3 及更高版本中引进了性能改进，不再需要调整 **iptablesSyncPeriod** 参数。
proxyArguments.iptables-min-sync-period	**array**	刷新 **iptables** 规则前的最短持续时间。此字段确保刷新的频率不会过于频繁。有效的后缀包括 **s**、**m** 和 **h**，具体参见 Go **time** 软件包。默认值为： kubeProxyConfig: proxyArguments: iptables-min-sync-period: - 0s

18.4.11. 创建 Kubernetes 清单和 Ignition 配置文件

由于您必须修改一些集群定义文件并手动启动集群机器，因此您必须生成 Kubernetes 清单和 Ignition 配置文件来配置机器。

安装配置文件转换为 Kubernetes 清单。清单嵌套到 Ignition 配置文件中，稍后用于配置集群机器。



重要

- OpenShift Container Platform 安装程序生成的 Ignition 配置文件包含 24 小时后过期的证书，然后在该时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 node-bootstrapper 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。

- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

注意

生成清单和 Ignition 文件的安装程序是特定的架构，可以从 客户端镜像镜像获取。安装程序的 Linux 版本仅在 s390x 上运行。此安装程序也可用作 Mac OS 版本。

先决条件

- 已获得 OpenShift Container Platform 安装程序。

- 已创建 install-config.yaml 安装配置文件。

流程

1. 进入包含 OpenShift Container Platform 安装程序的目录，并为集群生成 Kubernetes 清单：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 对于 <installation_directory>，请指定包含您创建的 install-config.yaml 文件的安装目录。

警告

如果您要安装一个三节点集群，请跳过以下步骤，以便可以调度 control plane 节点。

重要

当您将 control plane 节点从默认的不可调度配置为可以调度时，需要额外的订阅。这是因为 control plane 节点变为计算节点。

2. 检查 <installation_directory>/manifests/cluster-scheduler-02-config.yml Kubernetes 清单文件中的 mastersSchedulable 参数是否已设置为 false。此设置可防止在 control plane 机器上调度 pod：

   a. 打开 <installation_directory>/manifests/cluster-scheduler-02-config.yml 文件。

   b. 找到 mastersSchedulable 参数，并确保它被设置为 false。

   c. 保存并退出 文件。

3. 要创建 Ignition 配置文件，请从包含安装程序的目录运行以下命令：

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

1 对于 <installation_directory>，请指定相同的安装目录。

为安装目录中的 bootstrap、control plane 和计算节点创建 Ignition 配置文件。kubeadmin-password 和 kubeconfig 文件在 ./<installation_directory>/auth 目录中创建：

```
.
```

```
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

18.4.12. 安装 RHCOS 并启动 OpenShift Container Platform bootstrap 过程

要在您置备的 IBM Z® 基础架构上安装 OpenShift Container Platform，您必须将 Red Hat Enterprise Linux CoreOS(RHCOS)安装为 Red Hat Enterprise Linux(RHEL)客户机虚拟机。安装 RHCOS 时，您必须为您要安装的机器类型提供 OpenShift Container Platform 安装程序生成的 Ignition 配置文件。如果您配置了适当的网络、DNS 和负载均衡基础架构，OpenShift Container Platform bootstrap 过程会在 RHCOS 机器重启后自动启动。

您可以使用预打包的 QEMU 写时复制(QCOW2)磁盘镜像对 RHCOS 执行快速跟踪安装。或者，您可以在新 QCOW2 磁盘镜像上执行完整安装。

要在系统中添加更多安全性，您可以选择使用 IBM® Secure Execution 安装 RHCOS，然后才能继续快速跟踪安装。

18.4.12.1. 使用 IBM 安全执行安装 RHCOS

在使用 IBM® Secure Execution 安装 RHCOS 前，您必须准备底层基础架构。

先决条件

- IBM® z15 或更高版本，或 IBM® LinuxONE114 或更高版本。
- Red Hat Enterprise Linux (RHEL) 8 或更高版本。
- 您有一个 bootstrap Ignition 文件。该文件不受保护，其他人能够查看和编辑它。
- 您已确认引导镜像在安装后没有更改。
- 您必须以 IBM® Secure Execution 客户机身份运行所有节点。

流程

1. **准备 RHEL KVM 主机以支持 IBM® 安全执行。**

- **默认情况下，KVM 主机不支持 IBM® 安全执行模式中的客户机。要在 IBM® Secure Execution 模式中支持客户机，KVM 主机必须使用内核参数规格 prot_virt=1 在 LPAR 模式中引导。要在 RHEL 8 上启用 prot_virt=1，请按照以下步骤执行：**

a. **进入到 /boot/loader/entries/，以修改您的引导装载程序配置文件 slirpconf。**

b. **添加内核命令行参数 prot_virt=1。**

c. **运行 zipl 命令并重启系统。**

从支持 IBM® Secure Execution 开始的 KVM 主机发出以下内核信息：

```
prot_virt: Reserving <amount>MB as ultravisor base storage.
```

d. **要验证 KVM 主机现在支持 IBM® Secure Execution，请运行以下命令：**

```
# cat /sys/firmware/uv/prot_virt_host
```

输出示例

```
1
```

对于 Linux 实例，此属性的值是 1，它检测其环境与安全主机的一致性。对于其他实例，值为 0。

2. **通过 Ignition 将您的主机密钥添加到 KVM 客户机。**

第一次引导过程中，RHCOS 会查找主机密钥以使用它们重新加密自身。RHCOS 在

/etc/se-hostkeys 目录中搜索以 ibm-z-hostkey- 开头的文件。集群运行的每台机器的所有主机密钥都必须由管理员加载到目录中。首次启动后，您无法在任何其他机器上运行虚拟机。

注意

您需要在安全系统上准备 Ignition 文件。例如，另一个 IBM® Secure Execution 客户机。

例如：

```
{
 "ignition": { "version": "3.0.0" },
 "storage": {
  "files": [
   {
    "path": "/etc/se-hostkeys/ibm-z-hostkey-<your-hostkey>.crt",
    "contents": {
     "source": "data:;base64,<base64 encoded hostkey document>"
    },
    "mode": 420
   },
   {
    "path": "/etc/se-hostkeys/ibm-z-hostkey-<your-hostkey>.crt",
    "contents": {
     "source": "data:;base64,<base64 encoded hostkey document>"
    },
    "mode": 420
   }
  ]
 }
}
```

注意

如果您希望节点能够在多个 IBM Z® 机器上运行，您可以根据需要添加任意数量的主机密钥。

3. 要生成 Base64 编码字符串，请运行以下命令：

```
base64 <your-hostkey>.crt
```

与不运行 IBM® Secure Execution 的客户机相比，机器的第一次引导将较长，因为整个镜

像在 Ignition 阶段之前使用随机生成的 LUKS 密码短语进行加密。

4. 添加 Ignition 保护

为了保护存储在 Ignition 配置文件中的 secret 被读取或修改，您必须加密 Ignition 配置文件。

注意

为了实现所需的安全性，在运行 IBM® Secure Execution 时默认禁用 Ignition 日志记录和本地登录。

a. 运行以下命令，获取 secex-qemu.qcow2 镜像的公共 GPG 密钥，并使用密钥加密 Ignition 配置：

```
gpg --recipient-file /path/to/ignition.gpg.pub --yes --output /path/to/config.ign.gpg --verbose --armor --encrypt /path/to/config.ign
```

5. 按照 RHCOS 的 fast-track 安装步骤，使用 IBM® Secure Execution QCOW 镜像安装节点。

注意

在启动虚拟机前，请将 serial=ignition 替换为 serial=ignition_crypted，并添加 launchSecurity 参数。

验证

当您完成 RHCOS 的快速跟踪安装，以及 Ignition 在第一次引导时运行后，验证解密是否成功。

- 如果解密成功，您可以预期类似以下示例的输出：

输出示例

```
[    2.801433] systemd[1]: Starting coreos-ignition-setup-user.service - CoreOS Ignition
```

```
User Config Setup...

[    2.803959] coreos-secex-ignition-decrypt[731]: gpg: key <key_name>: public key
"Secure Execution (secex) 38.20230323.dev.0" imported
[    2.808874] coreos-secex-ignition-decrypt[740]: gpg: encrypted with rsa4096 key, ID
<key_name>, created <yyyy-mm-dd>
[  OK  ] Finished coreos-secex-igni…S Secex Ignition Config Decryptor.
```

- 如果解密失败，您可以预期类似以下示例的输出：

输出示例

```
Starting coreos-ignition-s…reOS Ignition User Config Setup...
[    2.863675] coreos-secex-ignition-decrypt[729]: gpg: key <key_name>: public key
"Secure Execution (secex) 38.20230323.dev.0" imported
[    2.869178] coreos-secex-ignition-decrypt[738]: gpg: encrypted with RSA key, ID
<key_name>
[    2.870347] coreos-secex-ignition-decrypt[738]: gpg: public key decryption failed: No
secret key
[    2.870371] coreos-secex-ignition-decrypt[738]: gpg: decryption failed: No secret key
```

其他资源

- Linux 的 IBM® 安全执行简介
- Linux 作为 IBM® Secure Execution 主机或客户机
- 在 IBM Z 上设置 IBM® Secure Execution

18.4.12.2. 在 IBM Z 或 IBM LinuxONE 环境中使用静态 IP 配置 NBDE

在 IBM Z® 或 IBM® LinuxONE 环境中启用 NBDE 磁盘加密需要额外的步骤，本节中详细介绍。

先决条件

- 您已设置了外部 Tang 服务器。具体步骤请查看 网络绑定磁盘加密。
- 您已安装了 with ane 实用程序。
- 您已查看如何使用 Butane 创建机器配置的说明。

流程

1. 为 control plane 和计算节点创建 Butane 配置文件。

   以下 control plane 节点的 Butane 配置示例为磁盘加密创建一个名为 master-storage.bu 的文件：

```
variant: openshift
version: 4.16.0
metadata:
  name: master-storage
  labels:
    machineconfiguration.openshift.io/role: master
storage:
  luks:
    - clevis:
        tang:
          - thumbprint: QcPr_NHFJammnRCA3fFMVdNBwjs
            url: http://clevis.example.com:7500
        options: 1
          - --cipher
          - aes-cbc-essiv:sha256
      device: /dev/disk/by-partlabel/root
      label: luks-root
      name: root
      wipe_volume: true
  filesystems:
    - device: /dev/mapper/root
      format: xfs
      label: root
      wipe_filesystem: true
openshift:
  fips: true 2
```

1. 只有在启用了 FIPS 模式时才需要 cipher 选项。如果禁用了 FIPS，则省略该条目。

2.

2. 运行以下命令，创建自定义 initramfs 文件来引导机器：

```
$ coreos-installer pxe customize \
    /root/rhcos-bootfiles/rhcos-<release>-live-initramfs.s390x.img \
    --dest-device /dev/disk/by-id/scsi-<serial_number> --dest-karg-append \
    ip=<ip_address>::<gateway_ip>:<subnet_mask>::<network_device>:none \
    --dest-karg-append nameserver=<nameserver_ip> \
    --dest-karg-append rd.neednet=1 -o \
    /root/rhcos-bootfiles/<node_name>-initramfs.s390x.img
```

注意

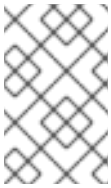
首次引导前，您必须为集群中的每个节点自定义 initramfs，并添加 PXE 内核参数。

3. 创建包含 ignition.platform.id=metal 和 ignition.firstboot 的参数文件。

```
rd.neednet=1 \
console=ttysclp0 \
ignition.firstboot ignition.platform.id=metal \
coreos.live.rootfs_url=http://<http_server>/rhcos-<version>-live-rootfs.<architecture>.img \ 1
coreos.inst.ignition_url=http://<http_server>/master.ign \ 2
ip=10.19.17.2::10.19.17.1:255.255.255.0::enbdd0:none nameserver=10.19.17.1 \
zfcp.allow_lun_scan=0 \
rd.znet=qeth,0.0.bdd0,0.0.bdd1,0.0.bdd2,layer2=1 \
rd.zfcp=0.0.5677,0x600606680g7f0056,0x034F000000000000
```

1 指定您要引导的 kernel 和 initramfs 的 rootfs 工件位置。仅支持 HTTP 和 HTTPS 协议。

2 指定 Ignition 配置文件的位置。使用 master.ign 或 worker.ign。仅支持 HTTP 和 HTTPS 协议。



注意

将参数文件中的所有选项写为一行，并确保您没有换行字符。

其他资源

- 使用 Butane 创建机器配置

18.4.12.3. 使用预打包的 QCOW2 磁盘镜像快速跟踪安装

完成以下步骤，在 Red Hat Enterprise Linux CoreOS(RHCOS)快速跟踪安装中创建机器，导入预打包的 Red Hat Enterprise Linux CoreOS(RHCOS)QEMU 写时复制(QCOW2)磁盘镜像。

先决条件

- 至少有一个 LPAR 在 KVM 的 RHEL 8.6 或更高版本中运行，在此过程中称为 RHEL KVM 主机。
- KVM/QEMU 管理程序安装在 RHEL KVM 主机上。
- 一个域名服务器(DNS)，它可以对节点执行主机名和反向查找。
- 提供 IP 地址的 DHCP 服务器。

流程

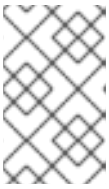
1. 从红帽客户门户网站的产品下载页或 RHCOS image mirror 页获得 RHEL QEMU copy-on-write (QCOW2) 磁盘镜像。

重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每个发行版本而改变。您必须下载最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。仅使用以下流程中描述的适当 RHCOS QCOW2 镜像。

2. 将 QCOW2 磁盘镜像和 Ignition 文件下载到 RHEL KVM 主机上的通用目录中。

例如： /var/lib/libvirt/images



注意

Ignition 文件由 OpenShift Container Platform 安装程序生成。

3. 为每个 KVM 客户机节点使用 QCOW2 磁盘镜像文件创建新磁盘镜像。

```
$ qemu-img create -f qcow2 -F qcow2 -b /var/lib/libvirt/images/{source_rhcos_qemu}
/var/lib/libvirt/images/{vmname}.qcow2 {size}
```

4. 使用 Ignition 文件和新磁盘镜像创建新的 KVM 客户机节点。

```
$ virt-install --noautoconsole \
  --connect qemu:///system \
  --name {vm_name} \
  --memory {memory} \
  --vcpus {vcpus} \
  --disk {disk} \
  --launchSecurity type="s390-pv" \ 1
  --import \
  --network network={network},mac={mac} \
  --disk path=
{ign_file},format=raw,readonly=on,serial=ignition,startup_policy=optional 2
```

1. 如果启用了 IBM® Secure Execution，请添加 launchSecurity type="s390-pv" 参数。

2. 如果启用了 IBM® Secure Execution，请将 serial=ignition 替换为 serial=ignition_crypted。

18.4.12.4. 在新 QCOW2 磁盘镜像上完全安装

完成以下步骤，在新的 QEMU 写时复制(QCOW2)磁盘镜像上在完整安装中创建机器。

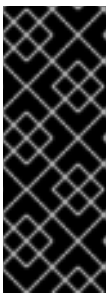
先决条件

- 至少有一个 LPAR 在 KVM 的 RHEL 8.6 或更高版本中运行，在此过程中称为 RHEL KVM 主机。

- KVM/QEMU 管理程序安装在 RHEL KVM 主机上。
- 一个域名服务器(DNS)，它可以对节点执行主机名和反向查找。
- 设置了 HTTP 或 HTTPS 服务器。

流程

1. 从红帽客户门户网站的产品下载页或 RHCOS image mirror 页获取 RHEL kernel, initramfs, 和 rootfs 文件。



重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每个发行版本而改变。您必须下载最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。仅使用以下流程中描述的适当 RHCOS QCOW2 镜像。

文件名包含 OpenShift Container Platform 版本号。它们类似以下示例：

- kernel: rhcos-<version>-live-kernel-<architecture>
- initramfs: rhcos-<version>-live-initramfs.<architecture>.img
- rootfs: rhcos-<version>-live-rootfs.<architecture>.img

2. 在启动 virt-install 前，将下载的 RHEL live kernel、initramfs 和 rootfs 以及 Ignition 文件移到 HTTP 或 HTTPS 服务器中。

注意

Ignition 文件由 OpenShift Container Platform 安装程序生成。

3. 使用 RHEL 内核、initramfs 和 Ignition 文件、新磁盘镜像并调整 parm 行参数，创建新的 KVM 客户机节点。

- 对于 --location，指定 kernel/initrd 在 HTTP 或 HTTPS 服务器上的位置。
- 对于 coreos.inst.ignition_url=，请为机器角色指定 Ignition 文件。使用 bootstrap.ign、master.ign 或 worker.ign。仅支持 HTTP 和 HTTPS 协议。
- 对于 coreos.live.rootfs_url=，请为您引导的内核和 initramfs 指定匹配的 rootfs 构件。仅支持 HTTP 和 HTTPS 协议。

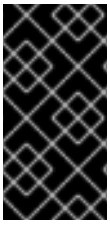
```
$ virt-install \
  --connect qemu:///system \
  --name {vm_name} \
  --vcpus {vcpus} \
  --memory {memory_mb} \
  --disk {vm_name}.qcow2,size={image_size| default(10,true)} \
  --network network={virt_network_parm} \
  --boot hd \
  --location {media_location},kernel={rhcos_kernel},initrd={rhcos_initrd} \
  --extra-args "rd.neednet=1 coreos.inst.install_dev=/dev/vda
coreos.live.rootfs_url={rhcos_liveos} ip={ip}::{default_gateway}:
{subnet_mask_length}:{vm_name}:enc1:none:{MTU} nameserver={dns}
coreos.inst.ignition_url={rhcos_ign}" \
  --noautoconsole \
  --wait
```

18.4.12.5. 高级 RHCOS 安装参考

本节演示了网络配置和其他高级选项，允许您修改 Red Hat Enterprise Linux CoreOS(RHCOS)手动安装过程。下表描述了您可以用于 RHCOS live 安装程序和 coreos-installer 命令的内核参数和命令行选项。

18.4.12.5.1. ISO 安装的网络选项

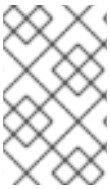
如果从 ISO 镜像安装 RHCOS，您可以在引导镜像时手动添加内核参数，以便为节点配置网络。如果没有指定网络参数，当 RHCOS 检测到需要网络来获取 Ignition 配置文件时，在 initramfs 中激活 DHCP。



重要

在手动添加网络参数时，还必须添加 rd.neednet=1 内核参数，以便在 initramfs 中启动网络。

以下信息提供了在 RHCOS 节点上为 ISO 安装配置网络的示例。示例描述了如何使用 ip= 和 nameserver= 内核参数。



注意

在添加内核参数时，顺序非常重要：ip= 和 nameserver=。

网络选项在系统引导过程中传递给 dracut 工具。有关由 dracut 支持的网络选项的更多信息，请参阅 dracut.cmdline 手册页。

以下示例是 ISO 安装的网络选项。

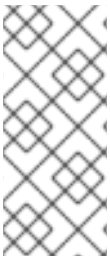
配置 DHCP 或静态 IP 地址

要配置 IP 地址，可使用 DHCP(ip=dhcp)或设置单独的静态 IP 地址(ip=<host_ip>)。如果设置静态 IP，则必须在每个节点上识别 DNS 服务器 IP 地址（名称服务器=<dns_ip>）。以下示例集：

- 节点的 IP 地址为 10.10.10.2
- 网关地址为 10.10.10.254
- 子网掩码为 255.255.255.0
- 到 core0.example.com 的主机名
- DNS 服务器地址为 4.4.4.41

- 自动配置值为 none。当以静态方式配置 IP 网络时，不需要自动配置。

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
nameserver=4.4.4.41
```



注意

当您使用 DHCP 为 RHCOS 机器配置 IP 寻址时，机器还通过 DHCP 获取 DNS 服务器信息。对于基于 DHCP 的部署，您可以通过 DHCP 服务器配置定义 RHCOS 节点使用的 DNS 服务器地址。

配置没有静态主机名的 IP 地址

您可以在不分配静态主机名的情况下配置 IP 地址。如果用户没有设置静态主机名，则会提取并通过反向 DNS 查找自动设置。要在没有静态主机名的情况下配置 IP 地址，请参考以下示例：

- 节点的 IP 地址为 10.10.10.2
- 网关地址为 10.10.10.254
- 子网掩码为 255.255.255.0
- DNS 服务器地址为 4.4.4.41
- 自动配置值为 none。当以静态方式配置 IP 网络时，不需要自动配置。

```
ip=10.10.10.2::10.10.10.254:255.255.255.0::enp1s0:none
nameserver=4.4.4.41
```

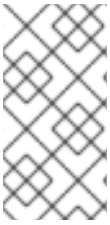
指定多个网络接口

您可以通过设置多个 ip= 条目来指定多个网络接口。

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
ip=10.10.10.3::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none
```

配置默认网关和路由

可选：您可以通过设置 a rd.route= 值来配置到额外网络的路由。



注意

当您配置一个或多个网络时，需要一个默认网关。如果额外网络网关与主要网络网关不同，则默认网关必须是主要网络网关。

- 运行以下命令来配置默认网关：

```
ip=::10.10.10.254::::
```

- 输入以下命令为额外网络配置路由：

```
rd.route=20.20.20.0/24:20.20.20.254:enp2s0
```

在单个接口中禁用 DHCP

您可以在单一接口中禁用 DHCP，例如当有两个或者多个网络接口时，且只有一个接口被使用。在示例中，enp1s0 接口具有一个静态网络配置，而 enp2s0 禁用了 DHCP，不使用它：

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
ip=::::core0.example.com:enp2s0:none
```

合并 DHCP 和静态 IP 配置

您可以将系统上的 DHCP 和静态 IP 配置与多个网络接口合并，例如：

```
ip=enp1s0:dhcp
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none
```

在独立接口上配置 VLAN

可选： 您可以使用 vlan= 参数在单个接口上配置 VLAN。

- 要在网络接口中配置 VLAN 并使用静态 IP 地址，请运行以下命令：

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0.100:none
vlan=enp2s0.100:enp2s0
```

- 要在网络接口中配置 VLAN 并使用 DHCP，请运行以下命令：

```
ip=enp2s0.100:dhcp
vlan=enp2s0.100:enp2s0
```

提供多个 DNS 服务器

您可以通过为每个服务器添加一个 nameserver= 条目来提供多个 DNS 服务器，例如

```
nameserver=1.1.1.1
nameserver=8.8.8.8
```

18.4.13. 等待 bootstrap 过程完成

OpenShift Container Platform bootstrap 过程在集群节点首次引导到安装到磁盘的持久 RHCOS 环境后开始。通过 Ignition 配置文件提供的配置信息用于初始化 bootstrap 过程并在机器上安装 OpenShift Container Platform。您必须等待 bootstrap 过程完成。

先决条件

- 已为集群创建 Ignition 配置文件。
- 您已配置了适当的网络、DNS 和负载平衡基础架构。
- 已获得安装程序，并为集群生成 Ignition 配置文件。
- 已在集群机器上安装 RHCOS，并提供 OpenShift Container Platform 安装程序生成的 Ignition 配置文件。
- 您的机器可以直接访问互联网，或者有 HTTP 或 HTTPS 代理可用。

流程

1. 监控 bootstrap 过程：

```
$ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \ 1
    --log-level=info 2
```

1 对于 <installation_directory>，请指定安装文件保存到的目录的路径。

2 要查看不同的安装详情，请指定 warn、debug 或 error，而不是 info。

输出示例

```
INFO Waiting up to 30m0s for the Kubernetes API at
https://api.test.example.com:6443...
INFO API v1.29.4 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
```

当 Kubernetes API 服务器提示已在 control plane 机器上引导它时，该命令会成功。

2. bootstrap 过程完成后，从负载均衡器中删除 bootstrap 机器。

重要

此时您必须从负载均衡器中删除 bootstrap 机器。您还可以删除或重新格式化 bootstrap 机器本身。

18.4.14. 使用 CLI 登录集群

您可以通过导出集群 kubeconfig 文件，以默认系统用户身份登录集群。kubeconfig 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。

- 已安装 oc CLI。

流程

1. 导出 kubeadmin 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 <installation_directory>，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 oc 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

18.4.15. 批准机器的证书签名请求

当您将机器添加到集群时，会为您添加的每台机器生成两个待处理证书签名请求(CSR)。您必须确认这些 CSR 已获得批准，或根据需要自行批准。必须首先批准客户端请求，然后批准服务器请求。

先决条件

- 您已将机器添加到集群中。

流程

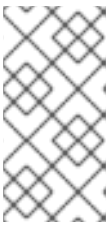
1. 确认集群可以识别这些机器：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0  Ready    master  63m  v1.29.4
master-1  Ready    master  63m  v1.29.4
master-2  Ready    master  64m  v1.29.4
```

输出中列出了您创建的所有机器。



注意

在有些 CSR 被批准前，前面的输出可能不包括计算节点（也称为 worker 节点）。

2. 检查待处理的 CSR，并确保添加到集群中的每台机器都有 Pending 或 Approved 状态的客户端请求：

```
$ oc get csr
```

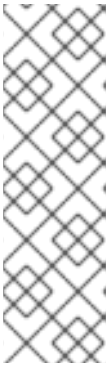
输出示例

```
NAME        AGE   REQUESTOR                             CONDITION
csr-mddf5   20m   system:node:master-01.example.com   Approved,Issued
csr-z5rln   16m   system:node:worker-21.example.com   Approved,Issued
```

3. 如果 CSR 没有获得批准，在您添加的机器的所有待处理 CSR 都处于 Pending 状态 后，请批准集群机器的 CSR：

注意

由于 CSR 会自动轮转，因此请在将机器添加到集群后一小时内批准您的 CSR。如果没有在一小时内批准它们，证书将会轮转，每个节点会存在多个证书。您必须批准所有这些证书。批准客户端 CSR 后，Kubelet 为服务证书创建一个二级 CSR，这需要手动批准。然后，如果 Kubelet 请求具有相同参数的新证书，则后续提供证书续订请求由 machine-approver 自动批准。



注意

对于在未启用机器 API 的平台上运行的集群，如裸机和其他用户置备的基础架构，您必须实施一种方法来自动批准 kubelet 提供证书请求(CSR)。如果没有批准请求，则 oc exec、ocrsh 和 oc logs 命令将无法成功，因为 API 服务器连接到 kubelet 时需要服务证书。与 Kubelet 端点联系的任何操作都需要此证书批准。该方法必须监视新的 CSR，确认 CSR 由 system: node 或 system:admin 组中的 node-bootstrapper 服务帐户提交，并确认节点的身份。

- 要单独批准，请对每个有效的 CSR 运行以下命令：

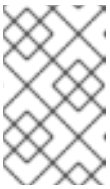
```
$ oc adm certificate approve <csr_name> 1
```

1

<csr_name> 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}
{{"\n"}}{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```



注意

在有些 CSR 被批准前，一些 Operator 可能无法使用。

4. 现在，您的客户端请求已被批准，您必须查看添加到集群中的每台机器的服务器请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE     REQUESTOR                                                   CONDITION
csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal       Pending
csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal       Pending
...
```

5. 如果剩余的 CSR 没有被批准，且处于 Pending 状态，请批准集群机器的 CSR：

- 要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 <csr_name> 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}
{{"\n"}}{{end}}{{end}}' | xargs oc adm certificate approve
```

6. 批准所有客户端和服务器 CSR 后，机器将 处于 Ready 状态。运行以下命令验证：

```
$ oc get nodes
```

输出示例

```
NAME       STATUS   ROLES    AGE  VERSION
master-0   Ready    master   73m  v1.29.4
master-1   Ready    master   73m  v1.29.4
master-2   Ready    master   74m  v1.29.4
worker-0   Ready    worker   11m  v1.29.4
worker-1   Ready    worker   11m  v1.29.4
```

注意

批准服务器 CSR 后可能需要几分钟时间让机器过渡到 Ready 状态。

其他信息

- 如需有关 CSR 的更多信息，请参阅 证书签名请求。

18.4.16. 初始 Operator 配置

在 control plane 初始化后，您必须立即配置一些 Operator，以便它们都可用。

先决条件

- 您的 control plane 已初始化。

流程

1. 观察集群组件上线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION   AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.16.0    True        False         False      19m
baremetal                                  4.16.0    True        False         False      37m
cloud-credential                           4.16.0    True        False         False      40m
cluster-autoscaler                         4.16.0    True        False         False      37m
config-operator                            4.16.0    True        False         False      38m
console                                    4.16.0    True        False         False      26m
csi-snapshot-controller                    4.16.0    True        False         False      37m
dns                                        4.16.0    True        False         False      37m
etcd                                       4.16.0    True        False         False      36m
image-registry                             4.16.0    True        False         False      31m
ingress                                    4.16.0    True        False         False      30m
```

```
insights                                   4.16.0    True        False         False      31m
kube-apiserver                             4.16.0    True        False         False      26m
kube-controller-manager                    4.16.0    True        False         False      36m
kube-scheduler                             4.16.0    True        False         False      36m
kube-storage-version-migrator              4.16.0    True        False         False      37m
machine-api                                4.16.0    True        False         False      29m
machine-approver                           4.16.0    True        False         False      37m
machine-config                             4.16.0    True        False         False      36m
marketplace                                4.16.0    True        False         False      37m
monitoring                                 4.16.0    True        False         False      29m
network                                    4.16.0    True        False         False      38m
node-tuning                                4.16.0    True        False         False      37m
openshift-apiserver                        4.16.0    True        False         False      32m
openshift-controller-manager               4.16.0    True        False         False      30m
openshift-samples                          4.16.0    True        False         False      32m
operator-lifecycle-manager                 4.16.0    True        False         False      37m
operator-lifecycle-manager-catalog         4.16.0    True        False         False      37m
operator-lifecycle-manager-packageserver   4.16.0    True        False         False      32m
service-ca                                 4.16.0    True        False         False      38m
storage                                    4.16.0    True        False         False      37m
```

2. 配置不可用的 Operator。

18.4.16.1. 镜像 registry 存储配置

对于不提供默认存储的平台，Image Registry Operator 最初不可用。安装后，您必须将 registry 配置为使用存储，以便 Registry Operator 可用。

显示配置生产集群所需的持久性卷的说明。如果适用，显示有关将空目录配置为存储位置的说明，这仅适用于非生产集群。

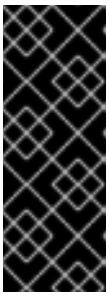
提供了在升级过程中使用 Recreate rollout 策略来允许镜像 registry 使用块存储类型的说明。

18.4.16.1.1. 为 IBM Z 配置 registry 存储

作为集群管理员，在安装后需要配置 registry 来使用存储。

先决条件

- 您可以使用具有 cluster-admin 角色的用户访问集群。
- 在 IBM Z® 上有一个集群。
- 您已为集群置备持久性存储，如 Red Hat OpenShift Data Foundation。



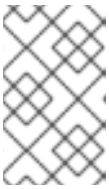
重要

当您只有一个副本时，OpenShift Container Platform 支持对镜像 registry 存储的 ReadWriteOnce 访问。ReadWriteOnce 访问还要求 registry 使用 Recreate rollout 策略。要部署支持高可用性的镜像 registry，需要两个或多个副本，ReadWriteMany 访问。

- 必须具有 100Gi 容量。

流程

1. 要将 registry 配置为使用存储，修改 configs.imageregistry/cluster 资源中的 spec.storage.pvc。



注意

使用共享存储时，请查看您的安全设置以防止外部访问。

2. 验证您没有 registry pod:

```
$ oc get pod -n openshift-image-registry -l docker-registry=default
```

输出示例

```
No resources found in openshift-image-registry namespace
```

注意

如果您的输出中有一个 registry pod，则不需要继续这个过程。

3. 检查 **registry** 配置：

```
$ oc edit configs.imageregistry.operator.openshift.io
```

输出示例

```
storage:
  pvc:
    claim:
```

将 **claim** 字段留空以允许自动创建 **image-registry-storage PVC**。

4. 检查 **clusteroperator** 状态：

```
$ oc get clusteroperator image-registry
```

输出示例

```
NAME            VERSION         AVAILABLE   PROGRESSING   DEGRADED   SINCE
MESSAGE
image-registry   4.16            True        False         False      6h50m
```

5. 确保 **registry** 设置为 **managed**，以启用镜像的构建和推送。

- 运行：

```
$ oc edit configs.imageregistry/cluster
```

然后，更改行

```
managementState: Removed
```

至

```
managementState: Managed
```

18.4.16.1.2. 在非生产集群中为镜像 registry 配置存储

您必须为 Image Registry Operator 配置存储。对于非生产集群，您可以将镜像 registry 设置为空目录。如果您这样做，重启 registry 时会丢失所有镜像。

流程

- 将镜像 registry 存储设置为空目录：

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch
'{"spec":{"storage":{"emptyDir":{}}}}'
```

警告

仅为非生产集群配置这个选项。

如果在 Image Registry Operator 初始化其组件前运行这个命令，oc patch 命令会失败并显示以下错误：

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster"
not found
```

等待几分钟，然后再次运行 命令。

18.4.17. 在用户置备的基础架构上完成安装

完成 Operator 配置后，可以在您提供的基础架构上完成集群安装。

先决条件

- 您的 control plane 已初始化。
- 已完成初始 Operator 配置。

流程

1. 使用以下命令确认所有集群组件都在线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION   AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.16.0    True        False         False      19m
baremetal                                  4.16.0    True        False         False      37m
cloud-credential                           4.16.0    True        False         False      40m
cluster-autoscaler                         4.16.0    True        False         False      37m
config-operator                            4.16.0    True        False         False      38m
console                                    4.16.0    True        False         False      26m
csi-snapshot-controller                    4.16.0    True        False         False      37m
dns                                        4.16.0    True        False         False      37m
etcd                                       4.16.0    True        False         False      36m
image-registry                             4.16.0    True        False         False      31m
ingress                                    4.16.0    True        False         False      30m
insights                                   4.16.0    True        False         False      31m
kube-apiserver                             4.16.0    True        False         False      26m
kube-controller-manager                    4.16.0    True        False         False      36m
kube-scheduler                             4.16.0    True        False         False      36m
kube-storage-version-migrator              4.16.0    True        False         False      37m
machine-api                                4.16.0    True        False         False      29m
machine-approver                           4.16.0    True        False         False      37m
machine-config                             4.16.0    True        False         False      36m
marketplace                                4.16.0    True        False         False      37m
monitoring                                 4.16.0    True        False         False      29m
network                                    4.16.0    True        False         False      38m
node-tuning                                4.16.0    True        False         False      37m
openshift-apiserver                        4.16.0    True        False         False      32m
```

```
openshift-controller-manager             4.16.0    True        False         False      30m
openshift-samples                        4.16.0    True        False         False      32m
operator-lifecycle-manager               4.16.0    True        False         False      37m
operator-lifecycle-manager-catalog       4.16.0    True        False         False      37m
operator-lifecycle-manager-packageserver 4.16.0    True        False         False      32m
service-ca                               4.16.0    True        False         False      38m
storage                                  4.16.0    True        False         False      37m
```

另外，当所有集群都可用时，以下命令会通知您。它还检索并显示凭证：

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

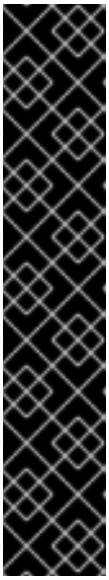
1

对于 <installation_directory>，请指定安装文件保存到的目录的路径。

输出示例

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

Cluster Version Operator 完成从 Kubernetes API 服务器部署 OpenShift Container Platform 集群时，该命令会成功。



重要

- 安装程序生成的 Ignition 配置文件包含 24 小时后过期的证书，然后在该时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 node-bootstrapper 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。

- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

2. 确认 Kubernetes API 服务器正在与 pod 通信。

   a. 要查看所有 pod 的列表，请使用以下命令：

```
$ oc get pods --all-namespaces
```

   输出示例

```
NAMESPACE                       NAME                                        READY   STATUS
RESTARTS   AGE
openshift-apiserver-operator    openshift-apiserver-operator-85cb746d55-zqhs8
1/1     Running   1          9m
openshift-apiserver             apiserver-67b9g                             1/1     Running   0
3m
openshift-apiserver             apiserver-ljcmx                             1/1     Running   0
1m
openshift-apiserver             apiserver-z25h4                             1/1     Running   0
2m
openshift-authentication-operator authentication-operator-69d5d8bf84-vh2n8
1/1     Running   0          5m
...
```

   b. 使用以下命令，查看上一命令的输出中所列 pod 的日志：

```
$ oc logs <pod_name> -n <namespace> 1
```

1

指定 pod 名称和命名空间，如上一命令的输出中所示。

如果 pod 日志显示，Kubernetes API 服务器可以与集群机器通信。

3. 对于使用光纤通道协议(FCP)的安装，还需要额外的步骤才能启用多路径。不要在安装过程中启用多路径。

如需更多信息，请参阅 *安装后机器配置任务* 文档中的"使用 RHCOS 上使用内核参数启用多路径"。

18.4.18. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

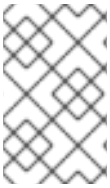
- 有关 Telemetry 服务的更多信息，请参阅关于 远程健康监控
- 如何在没有 SSH 的情况下在 OpenShift4 节点中生成 SOSREPORT。

18.4.19. 后续步骤

- 自定义集群。
- 如果需要，您可以选择 不使用远程健康报告。

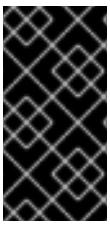
18.5. 在受限网络中的 IBM Z 和 IBM LINUXONE 上使用 RHEL KVM 安装集群

在 OpenShift Container Platform 版本 4.16 中，您可以在受限网络中置备的 IBM Z® 或 IBM® LinuxONE 基础架构上安装集群。



注意

虽然本文档只涉及 IBM Z®，但它的所有信息也适用于 IBM® LinuxONE。



重要

非裸机平台还有其他注意事项。在安装 OpenShift Container Platform 集群前，请参阅有关在未经测试的平台上部署 OpenShift Container Platform 的指南 中的信息。

18.5.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读有关 选择集群安装方法的文档，并为用户准备它。
- 您在镜像主机上创建 registry，并获取您的 OpenShift Container Platform 版本的 imageContentSources 数据。
- 在开始安装过程前，您必须移动或删除任何现有的安装文件。这样可确保在安装过程中创建和更新所需的安装文件。

  重要

  确保从可访问安装介质的机器中执行安装步骤。

- 已为集群置备了 使用 OpenShift Data Foundation 或其他支持的存储协议的持久性存储。要部署私有镜像 registry，您必须使用 ReadWriteMany 访问设置持久性存储。
- 如果使用防火墙，则会 将其配置为允许集群需要访问的站点。

注意

如果要配置代理，请务必查看此站点列表。

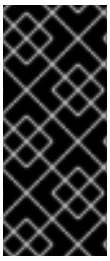
- 置备基于 RHEL 8.6 或更高版本的、托管在逻辑分区(LPAR)上的 RHEL Kernel Virtual Machine(KVM)系统。请参阅 Red Hat Enterprise Linux 8 和 9 生命周期。

18.5.2. 关于在受限网络中安装

在 OpenShift Container Platform 4.16 中，可以执行不需要有效的互联网连接来获取软件组件的安装。受限网络安装可以使用安装程序置备的基础架构或用户置备的基础架构完成，具体取决于您要安装集群的云平台。

如果您选择在云平台中执行受限网络安装，您仍需要访问其云 API。有些云功能，比如 Amazon Web Service 的 Route 53 DNS 和 IAM 服务，需要访问互联网。根据您的网络，在裸机硬件、Nutanix 或 VMware vSphere 上安装可能需要较少的互联网访问。

要完成受限网络安装，您必须创建一个 registry，以镜像 OpenShift 镜像 registry 的内容并包含安装介质。您可以在镜像主机上创建此 registry，该主机可同时访问互联网和您的封闭网络，也可以使用满足您的限制条件的其他方法。



重要

由于用户置备安装配置的复杂性，在尝试使用用户置备的基础架构受限网络安装前，请考虑完成标准用户置备的基础架构安装。完成此测试安装后，您可以更轻松地隔离和排除在受限网络中安装过程中可能出现的任何问题。

18.5.2.1. 其他限制

受限网络中的集群有以下额外限制和限制：

- ClusterVersion 状态包含一个 Unable to retrieve available updates 错误。
- 默认情况下，您无法使用 Developer Catalog 的内容，因为您无法访问所需的镜像流标签。

18.5.3. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来获得用来安装集群的镜像。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

18.5.4. 具有用户置备基础架构的集群的机器要求

对于包含用户置备的基础架构的集群，您必须部署所有所需的机器。

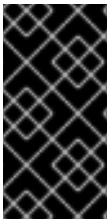
基于 RHEL 8.6 或更高版本的一个或多个 KVM 主机机器。每一台 RHEL KVM 主机机器都必须安装并运行 libvirt。虚拟机在每台 RHEL KVM 主机下调配。

18.5.4.1. 所需的机器

最小的 OpenShift Container Platform 集群需要以下主机：

表 18.58. 最低所需的主机

主机	描述
一个临时 bootstrap 机器	集群需要 bootstrap 机器在三台 control plane 机器上部署 OpenShift Container Platform 集群。您可在安装集群后删除 bootstrap 机器。
三台 control plane 机器	control plane 机器运行组成 control plane 的 Kubernetes 和 OpenShift Container Platform 服务。
至少两台计算机器，也称为 worker 机器。	OpenShift Container Platform 用户请求的工作负载在计算机器上运行。



重要

要提高集群的高可用性，请在至少两台物理机器的不同 RHEL 实例上分发 control plane 机器。

bootstrap、control plane 和计算机器必须使用 Red Hat Enterprise Linux CoreOS(RHCOS)作为操作系统。

查看 红帽企业 Linux 技术功能和限制。

18.5.4.2. 网络连接要求

OpenShift Container Platform 安装程序创建 Ignition 文件，这是所有 Red Hat Enterprise Linux CoreOS(RHCOS)虚拟机所必需的。OpenShift Container Platform 的自动安装由 bootstrap 机器执行。它在每个节点上启动 OpenShift Container Platform 安装，启动 Kubernetes 集群，然后完成。在这个 bootstrap 中，虚拟机必须通过 DHCP 服务器或静态 IP 地址建立网络连接。

18.5.4.3. IBM Z 网络连接要求

要在 RHEL KVM 中安装 IBM Z®，您需要：

- 使用 OSA 或 RoCE 网络适配器配置的 RHEL KVM 主机。
- 在 libvirt 中使用桥接网络的 RHEL KVM 主机或 MacVTap 将网络连接到客户机。

  请参阅 虚拟网络连接的类型。

18.5.4.4. 主机机器资源要求

环境中的 RHEL KVM 主机必须满足以下要求，才能托管您计划用于 OpenShift Container Platform 环境的虚拟机。请参阅 开始使用虚拟化。

您可以在以下 IBM® 硬件上安装 OpenShift Container Platform 版本 4.16：

- IBM® z16 （所有型号）、IBM® z15 （所有型号）、IBM® z14 （所有型号）

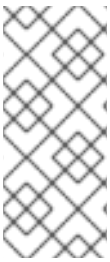
- IBM® LinuxONE 4 （所有型号）、IBM® LinuxONE （所有型号）、IBM® LinuxONE Emperor II、IBM® LinuxONE Rockhopper II

18.5.4.5. 最低 IBM Z 系统环境

硬件要求

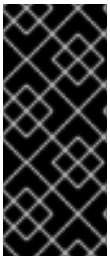
- Linux(IFL)等效的、启用了 SMT2 的 Linux 集成设施，每个群集都启用了 SMT2。

- 至少一个网络连接连接到 LoadBalancer 服务，并为集群外的流量提供数据。



注意

您可以使用专用或共享的 IFL 来分配足够的计算资源。资源共享是 IBM Z® 的关键优势之一。但是，您必须正确调整每个虚拟机监控程序层上的容量，并确保每个 OpenShift Container Platform 集群都有足够资源。



重要

由于集群的整体性能会受到影响，用于设置 OpenShift Container Platform 集群的 LPAR 必须提供足够的计算容量。就此而言，管理程序级别上的 LPAR 权重管理、授权和 CPU 共享扮演着重要角色。

操作系统要求

- 使用 KVM 在 RHEL 8.6 或更高版本上运行的一个 LPAR，由 libvirt 管理

在 RHEL KVM 主机上设置：

- 用于 OpenShift Container Platform control plane 机器的三台客户机虚拟机

- 用于 OpenShift Container Platform 计算机器的两个客户机虚拟机

- 一个客户虚拟机作为临时 OpenShift Container Platform bootstrap 机器

18.5.4.6. 最低资源要求

每个集群虚拟机都必须满足以下最低要求：

虚拟机	操作系统	vCPU [1]	虚拟内存	Storage	IOPS
bootstrap	RHCOS	4	16 GB	100 GB	N/A
Control plane（控制平面）	RHCOS	4	16 GB	100 GB	N/A
Compute	RHCOS	2	8 GB	100 GB	N/A

1. 当启用 SMT-2 时，一个物理核心(IFL)提供两个逻辑核心（线程）。管理程序可以提供两个或多个 vCPU。

18.5.4.7. 首选 IBM Z 系统环境

硬件要求

- 三个 LPARS，每个都相当于 6 个 IFL（每个集群启用了 SMT2）。
- 两个网络连接连接到 LoadBalancer 服务，并为集群外的流量提供数据。

操作系统要求

- 为获得高可用性，在 RHEL 8.6 或更高版本上运行的两个或者三个 LPAR 使用 KVM（由 libvirt 管理）。

在 RHEL KVM 主机上设置：

- 3 个用于 OpenShift Container Platform control plane 机器的虚拟机，分布在 RHEL KVM 主机中。
-

至少 6 个用于 OpenShift Container Platform 计算机器的虚拟机，分布在 RHEL KVM 主机中。

- 一个客户虚拟机作为临时 OpenShift Container Platform bootstrap 机器。

- 要确保过量使用环境中组件的可用性，请使用 cpu_shares 增加 control plane 的优先级。如果存在基础架构节点，则对它们执行相同的操作。请参阅 IBM® 文档中的 schedinfo。

18.5.4.8. 首选资源要求

每个集群虚拟机的首选要求如下：

虚拟机	操作系统	vCPU	虚拟内存	Storage
bootstrap	RHCOS	4	16 GB	120 GB
Control plane（控制平面）	RHCOS	8	16 GB	120 GB
Compute	RHCOS	6	8 GB	120 GB

18.5.4.9. 证书签名请求管理

在使用您置备的基础架构时，集群只能有限地访问自动机器管理，因此您必须提供一种在安装后批准集群证书签名请求 (CSR) 的机制。kube-controller-manager 只能批准 kubelet 客户端 CSR。machine-approver 无法保证使用 kubelet 凭证请求的提供证书的有效性，因为它不能确认是正确的机器发出了该请求。您必须决定并实施一种方法，以验证 kubelet 提供证书请求的有效性并进行批准。

其他资源

- IBM Z® 和 IBM® LinuxONE 环境的推荐主机实践

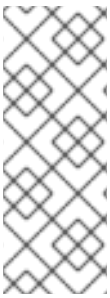
18.5.4.10. 用户置备的基础架构对网络的要求

所有 Red Hat Enterprise Linux CoreOS(RHCOS)机器都需要在启动时在 initramfs 中配置联网，以获取它们的 Ignition 配置文件。

在初次启动过程中，机器需要 IP 地址配置，该配置通过 DHCP 服务器或静态设置，提供所需的引导选项。建立网络连接后，机器会从 HTTP 或 HTTPS 服务器下载 Ignition 配置文件。然后，Ignition 配置

文件用于设置每台机器的确切状态。Machine Config Operator 在安装后完成对机器的更多更改，如应用新证书或密钥。

建议使用 DHCP 服务器对集群机器进行长期管理。确保 DHCP 服务器已配置为向集群机器提供持久的 IP 地址、DNS 服务器信息和主机名。



注意

如果用户置备的基础架构没有 DHCP 服务，您可以在 RHCOS 安装时向节点提供 IP 网络配置和 DNS 服务器地址。如果要从 ISO 镜像安装，这些参数可作为引导参数传递。如需有关静态 IP 置备和高级网络选项的更多信息，请参阅 *安装 RHCOS 并启动 OpenShift Container Platform bootstrap 过程* 部分。

Kubernetes API 服务器必须能够解析集群机器的节点名称。如果 API 服务器和 worker 节点位于不同的区域中，您可以配置默认 DNS 搜索区域，以允许 API 服务器解析节点名称。另一种支持的方法是始终通过节点对象和所有 DNS 请求中的完全限定域名引用主机。

18.5.4.10.1. 通过 DHCP 设置集群节点主机名

在 Red Hat Enterprise Linux CoreOS(RHCOS)机器上，主机名是通过 NetworkManager 设置的。默认情况下，机器通过 DHCP 获取其主机名。如果主机名不是由 DHCP 提供，请通过内核参数或者其它方法进行静态设置，请通过反向 DNS 查找获取。反向 DNS 查找在网络初始化后进行，可能需要一些时间来解决。其他系统服务可以在此之前启动，并将主机名检测为 localhost 或类似的内容。您可以使用 DHCP 为每个集群节点提供主机名来避免这种情况。

另外，通过 DHCP 设置主机名可以绕过实施 DNS split-horizon 的环境中的手动 DNS 记录名称配置错误。

18.5.4.10.2. 网络连接要求

您必须配置机器之间的网络连接，以允许 OpenShift Container Platform 集群组件进行通信。每台机器都必须能够解析集群中所有其他机器的主机名。

本节详细介绍了所需的端口。

表 18.59. 用于全机器到所有机器通信的端口

协议	port	描述
ICMP	N/A	网络可访问性测试
TCP	**1936**	指标
	9000-9999	主机级别的服务，包括端口 9 **100-9101 上的节点导出器**，以及端口 **9099** 上的 Cluster Version Operator。
	10250-10259	Kubernetes 保留的默认端口
UDP	**4789**	VXLAN
	6081	Geneve
	9000-9999	主机级别的服务，包括端口 **9100**到**9101** 上的节点导出器。
	500	IPsec IKE 数据包
	4500	IPsec NAT-T 数据包
	123	UDP 端口 **123** 上的网络时间协议 (NTP) 如果配置了外部 NTP 时间服务器，需要打开 UDP 端口 **123**。
TCP/UDP	**30000-32767**	Kubernetes 节点端口
ESP	N/A	IPsec Encapsulating Security Payload(ESP)

表 18.60. 用于所有机器控制平面通信的端口

协议	port	描述
TCP	**6443**	Kubernetes API

表 18.61. control plane 机器用于 control plane 机器通信的端口

协议	port	描述
TCP	**2379-2380**	etcd 服务器和对等端口

用户置备的基础架构的 NTP 配置

OpenShift Container Platform 集群被配置为默认使用公共网络时间协议(NTP)服务器。如果要使用本地企业 NTP 服务器，或者集群部署在断开连接的网络中，您可以将集群配置为使用特定的时间服务

器。如需更多信息，*请参阅配置 chrony 时间服务* 的文档。

如果 DHCP 服务器提供 NTP 服务器信息，Red Hat Enterprise Linux CoreOS(RHCOS)机器上的 chrony 时间服务会读取信息，并可以把时钟与 NTP 服务器同步。

其他资源

- 配置 chrony 时间服务

18.5.4.11. 用户置备的 DNS 要求

在 OpenShift Container Platform 部署中，以下组件需要 DNS 名称解析：

- The Kubernetes API
- OpenShift Container Platform 应用程序通配符
- bootstrap、control plane 和计算机器

Kubernetes API、bootstrap 机器、control plane 机器和计算机器也需要反向 DNS 解析。

DNS A/AAAA 或 CNAME 记录用于名称解析，PTR 记录用于反向名称解析。反向记录很重要，因为 Red Hat Enterprise Linux CoreOS(RHCOS)使用反向记录为所有节点设置主机名，除非 DHCP 提供主机名。另外，反向记录用于生成 OpenShift Container Platform 需要操作的证书签名请求(CSR)。

用户置备的 OpenShift Container Platform 集群需要以下 DNS 记录，这些记录必须在安装前就位。在每个记录中，<cluster_name> 是集群名称，<base_domain> 是您在 install-config.yaml 文件中指定的基域。完整的 DNS 记录采用以下形式：<component>.<cluster_name>.<base_domain>.。

表 18.62. 所需的 DNS 记录

组件	记录	描述
Kubernetes API	api.<cluster_name>.<base_domain>.	DNS A/AAAA 或 CNAME 记录，以及用于标识 API 负载均衡器的 DNS PTR 记录。这些记录必须由集群外的客户端和集群中的所有节点解析。

组件	记录	描述
	api-int.<cluster_name>.<base_domain>.	DNS A/AAAA 或 CNAME 记录，以及用于内部标识 API 负载均衡器的 DNS PTR 记录。这些记录必须可以从集群中的所有节点解析。  **重要** API 服务器必须能够根据 Kubernetes 中记录的主机名解析 worker 节点。如果 API 服务器无法解析节点名称，则代理的 API 调用会失败，且您无法从 pod 检索日志。
Routes	***.apps.<cluster_name>.<base_domain>.**	通配符 DNS A/AAAA 或 CNAME 记录，指向应用程序入口负载均衡器。应用程序入口负载均衡器以运行 Ingress Controller Pod 的机器为目标。默认情况下，Ingress Controller Pod 在计算机器上运行。这些记录必须由集群外的客户端和集群中的所有节点解析。 例如，console **-openshift-console.apps.<cluster_name>.<base_domain>** 用作到 OpenShift Container Platform 控制台的通配符路由。
bootstrap 机器	**bootstrap.<cluster_name>.<base_domain>.**	DNS A/AAAA 或 CNAME 记录，以及用于标识 bootstrap 机器的 DNS PTR 记录。这些记录必须由集群中的节点解析。
control plane 机器	**<control_plane><n>.<cluster_name>.<base_domain>.**	DNS A/AAAA 或 CNAME 记录，以识别 control plane 节点的每台机器。这些记录必须由集群中的节点解析。
计算机器	**<compute><n>.<cluster_name>.<base_domain>.**	DNS A/AAAA 或 CNAME 记录，用于识别 worker 节点的每台机器。这些记录必须由集群中的节点解析。

注意

在 OpenShift Container Platform 4.4 及更新的版本中，您不需要在 DNS 配置中指定 etcd 主机和 SRV 记录。

提示

您可以使用 dig 命令验证名称和反向名称解析。如需了解详细的 *验证步骤，请参阅为用户置备的基础架构验证 DNS 解析* 一节。

18.5.4.11.1. 用户置备的集群的 DNS 配置示例

本节提供 A 和 PTR 记录配置示例，它们满足了在用户置备的基础架构上部署 OpenShift Container Platform 的 DNS 要求。样本不是为选择一个 DNS 解决方案提供建议。

在这个示例中，集群名称为 ocp4，基域是 example.com。

用户置备的集群的 DNS A 记录配置示例

以下示例是 BIND 区域文件，其中显示了用户置备的集群中名称解析的 A 记录示例。

例 18.10. DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
  2019070700 ; serial
  3H  ; refresh (3 hours)
  30M  ; retry (30 minutes)
  2W  ; expiry (2 weeks)
  1W )  ; minimum (1 week)
 IN NS ns1.example.com.
 IN MX 10 smtp.example.com.
;
;
ns1.example.com.  IN A 192.168.1.5
smtp.example.com.  IN A 192.168.1.5
;
helper.example.com.  IN A 192.168.1.5
helper.ocp4.example.com. IN A 192.168.1.5
;
api.ocp4.example.com.  IN A 192.168.1.5 1
api-int.ocp4.example.com. IN A 192.168.1.5 2
;
*.apps.ocp4.example.com. IN A 192.168.1.5 3
;
bootstrap.ocp4.example.com. IN A 192.168.1.96 4
;
control-plane0.ocp4.example.com. IN A 192.168.1.97 5
control-plane1.ocp4.example.com. IN A 192.168.1.98 6
control-plane2.ocp4.example.com. IN A 192.168.1.99 7
;
compute0.ocp4.example.com. IN A 192.168.1.11 8
compute1.ocp4.example.com. IN A 192.168.1.7 9
;
;EOF
```

1

为 Kubernetes API 提供名称解析。记录引用 API 负载均衡器的 IP 地址。

2

3

为通配符路由提供名称解析。记录引用应用程序入口负载均衡器的 IP 地址。应用程序入口负载均衡器以运行 Ingress Controller Pod 的机器为目标。默认情况下，Ingress Controller Pod 在计算机器上运行。

注意

在这个示例中，将相同的负载均衡器用于 Kubernetes API 和应用入口流量。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。

4

为 bootstrap 机器提供名称解析。

5 6 7

为 control plane 机器提供名称解析。

8 9

为计算机器提供名称解析。

用户置备的集群的 DNS PTR 记录配置示例

以下示例 BIND 区域文件显示了用户置备的集群中反向名称解析的 PTR 记录示例。

例 18.11. 反向记录的 DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
  2019070700 ; serial
  3H  ; refresh (3 hours)
  30M  ; retry (30 minutes)
  2W  ; expiry (2 weeks)
  1W ) ; minimum (1 week)
 IN NS ns1.example.com.
;
5.1.168.192.in-addr.arpa. IN PTR api.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. IN PTR api-int.ocp4.example.com. 2
;
96.1.168.192.in-addr.arpa. IN PTR bootstrap.ocp4.example.com. 3
```

```
;
97.1.168.192.in-addr.arpa. IN PTR control-plane0.ocp4.example.com. 4
98.1.168.192.in-addr.arpa. IN PTR control-plane1.ocp4.example.com. 5
99.1.168.192.in-addr.arpa. IN PTR control-plane2.ocp4.example.com. 6
;
11.1.168.192.in-addr.arpa. IN PTR compute0.ocp4.example.com. 7
7.1.168.192.in-addr.arpa. IN PTR compute1.ocp4.example.com. 8
;
;EOF
```

1 为 Kubernetes API 提供反向 DNS 解析。PTR 记录引用 API 负载均衡器的记录名称。

2 为 Kubernetes API 提供反向 DNS 解析。PTR 记录引用 API 负载均衡器的记录名称，用于内部集群通信。

3 为 bootstrap 机器提供反向 DNS 解析。

4 5 6 为 control plane 机器提供反向 DNS 解析。

7 8 为计算机器提供反向 DNS 解析。

注意

OpenShift Container Platform 应用程序通配符不需要 PTR 记录。

18.5.4.12. 用户置备的基础架构的负载均衡要求

在安装 OpenShift Container Platform 前，您必须置备 API 和应用程序入口负载均衡基础架构。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。

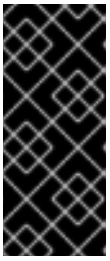
注意

如果要使用 Red Hat Enterprise Linux (RHEL) 实例部署 API 和应用程序入口负载均衡器，您必须单独购买 RHEL 订阅。

负载平衡基础架构必须满足以下要求：

1. API 负载均衡器 ：提供一个通用端点，供用户（包括人工和机器）与平台交互和配置。配置以下条件：

   - 仅第 4 层负载均衡.这可被称为 Raw TCP 或 SSL Passthrough 模式。
   - 无状态负载平衡算法。这些选项根据负载均衡器的实施而有所不同。



重要

不要为 API 负载均衡器配置会话持久性。为 Kubernetes API 服务器配置会话持久性可能会导致出现过量 OpenShift Container Platform 集群应用程序流量，以及过量的在集群中运行的 Kubernetes API。

在负载均衡器的前端和后端配置以下端口：

表 18.63. API 负载均衡器

port	后端机器（池成员）	internal	外部	描述
6443	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。您必须为 API 服务器健康检查探测配置 **/readyz** 端点。	X	X	Kubernetes API 服务器
22623	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。	X		机器配置服务器

注意

负载均衡器必须配置为，从 API 服务器关闭 /readyz 端点到从池中移除 API 服务器实例时最多需要 30 秒。在 /readyz 返回错误或健康后的时间范围内，端点必须被删除或添加。每 5 秒或 10 秒探测一次，有两个成功请求处于健康状态，三个成为不健康的请求是经过良好测试的值。

2.

应用程序入口负载均衡器 ：为应用程序流量从集群外部流提供入口点。OpenShift Container Platform 集群需要正确配置入口路由器。

配置以下条件：

- 仅第 4 层负载均衡.这可被称为 Raw TCP 或 SSL Passthrough 模式。
- 建议根据可用选项以及平台上托管的应用程序类型，使用基于连接的或基于会话的持久性。

提示

如果应用程序入口负载均衡器可以看到客户端的真实 IP 地址，启用基于 IP 的会话持久性可以提高使用端到端 TLS 加密的应用程序的性能。

在负载均衡器的前端和后端配置以下端口：

表 18.64. 应用程序入口负载均衡器

port	后端机器（池成员）	internal	外部	描述
443	默认情况下，运行 Ingress Controller Pod、计算或 worker 的机器。	X	X	HTTPS 流量
80	默认情况下，运行 Ingress Controller Pod、计算或 worker 的机器。	X	X	HTTP 流量

注意

如果要部署一个带有零计算节点的三节点集群，Ingress Controller Pod 在 control plane 节点上运行。在三节点集群部署中，您必须配置应用程序入口负载均衡器，将 HTTP 和 HTTPS 流量路由到 control plane 节点。

18.5.4.12.1. 用户置备的集群的负载均衡器配置示例

本节提供了一个满足用户置备集群的负载均衡要求的 API 和应用程序入口负载均衡器配置示例。示例是 HAProxy 负载均衡器的 /etc/haproxy/haproxy.cfg 配置。这个示例不是为选择一个负载平衡解决方案提供建议。

在这个示例中，将相同的负载均衡器用于 Kubernetes API 和应用入口流量。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。

注意

如果您使用 HAProxy 作为负载均衡器，并且 SELinux 设置为 enforcing，您必须通过运行 setsebool -P haproxy_connect_any=1 来确保 HAProxy 服务可以绑定到配置的 TCP 端口。

例 18.12. API 和应用程序入口负载均衡器配置示例

```
global
  log         127.0.0.1 local2
  pidfile     /var/run/haproxy.pid
  maxconn     4000
  daemon
defaults
  mode                    http
  log                     global
  option                  dontlognull
  option http-server-close
  option                  redispatch
  retries                 3
  timeout http-request    10s
  timeout queue           1m
  timeout connect         10s
  timeout client          1m
  timeout server          1m
  timeout http-keep-alive 10s
  timeout check           10s
  maxconn                 3000
listen api-server-6443 1
  bind *:6443
  mode tcp
  option  httpchk GET /readyz HTTP/1.0
```

```
  option  log-health-checks
  balance roundrobin
  server bootstrap bootstrap.ocp4.example.com:6443 verify none check check-ssl inter 10s
fall 2 rise 3 backup 2
  server master0 master0.ocp4.example.com:6443 weight 1 verify none check check-ssl
inter 10s fall 2 rise 3
  server master1 master1.ocp4.example.com:6443 weight 1 verify none check check-ssl
inter 10s fall 2 rise 3
  server master2 master2.ocp4.example.com:6443 weight 1 verify none check check-ssl
inter 10s fall 2 rise 3
listen machine-config-server-22623 3
  bind *:22623
  mode tcp
  server bootstrap bootstrap.ocp4.example.com:22623 check inter 1s backup 4
  server master0 master0.ocp4.example.com:22623 check inter 1s
  server master1 master1.ocp4.example.com:22623 check inter 1s
  server master2 master2.ocp4.example.com:22623 check inter 1s
listen ingress-router-443 5
  bind *:443
  mode tcp
  balance source
  server compute0 compute0.ocp4.example.com:443 check inter 1s
  server compute1 compute1.ocp4.example.com:443 check inter 1s
listen ingress-router-80 6
  bind *:80
  mode tcp
  balance source
  server compute0 compute0.ocp4.example.com:80 check inter 1s
  server compute1 compute1.ocp4.example.com:80 check inter 1s
```

1

端口 6443 处理 Kubernetes API 流量并指向 control plane 机器。

2 4

bootstrap 条目必须在 OpenShift Container Platform 集群安装前就位，且必须在 bootstrap 过程完成后删除它们。

3

端口 22623 处理机器配置服务器流量并指向 control plane 机器。

5

端口 443 处理 HTTPS 流量，并指向运行 Ingress Controller pod 的机器。默认情况下，Ingress Controller Pod 在计算机器上运行。

6

端口 80 处理 HTTP 流量，并指向运行 Ingress Controller pod 的机器。默认情况下，Ingress Controller Pod 在计算机器上运行。

注意

如果要部署一个带有零计算节点的三节点集群，Ingress Controller Pod 在 control plane 节点上运行。在三节点集群部署中，您必须配置应用程序入口负载均衡器，将 HTTP 和 HTTPS 流量路由到 control plane 节点。

提示

如果您使用 HAProxy 作为负载均衡器，您可以通过在 HAProxy 节点上运行 netstat -nltupe 来检查 haproxy 进程是否在侦听端口 6443、22623、443 和 80。

18.5.5. 准备用户置备的基础架构

在用户置备的基础架构上安装 OpenShift Container Platform 之前，您必须准备底层基础架构。

本节详细介绍了设置集群基础架构以准备 OpenShift Container Platform 安装所需的高级别步骤。这包括为您的集群节点配置 IP 网络和网络连接，通过防火墙启用所需的端口，以及设置所需的 DNS 和负载均衡基础架构。

准备后，集群基础架构必须满足 *带有用户置备的基础架构部分的集群要求*。

先决条件

- 您已参阅 OpenShift Container Platform 4.x Tested Integrations 页面。
- 您已查看了 *具有用户置备基础架构的集群要求部分中详述的基础架构* 要求。

流程

1. 如果您使用 DHCP 向集群节点提供 IP 网络配置，请配置 DHCP 服务。
   a. 将节点的持久 IP 地址添加到您的 DHCP 服务器配置。在您的配置中，将相关网络接口的 MAC 地址与每个节点的预期 IP 地址匹配。

b.

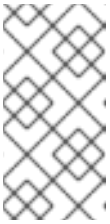
当您使用 DHCP 为集群机器配置 IP 寻址时，机器还通过 DHCP 获取 DNS 服务器信息。定义集群节点通过 DHCP 服务器配置使用的持久性 DNS 服务器地址。

注意

如果没有使用 DHCP 服务，则必须在 RHCOS 安装时为节点提供 IP 网络配置和 DNS 服务器地址。如果要从 ISO 镜像安装，这些参数可作为引导参数传递。如需有关静态 IP 置备和高级网络选项的更多信息，请参阅 *安装 RHCOS 并启动 OpenShift Container Platform bootstrap 过程* 部分。

c.

在 DHCP 服务器配置中定义集群节点的主机名。有关 *主机名注意事项的详情，请参阅通过 DHCP 设置集群节点* 主机名部分。



注意

如果没有使用 DHCP 服务，集群节点可以通过反向 DNS 查找来获取其主机名。

2.

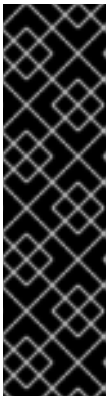
选择执行 Red Hat Enterprise Linux CoreOS(RHCOS)的快速跟踪安装或 Red Hat Enterprise Linux CoreOS(RHCOS)的完整安装。要进行完整安装，您必须设置 HTTP 或 HTTPS 服务器，以便提供 Ignition 文件，并将镜像安装到集群节点。对于快速跟踪安装，不需要 HTTP 或 HTTPS 服务器，但需要 DHCP 服务器。请参阅"Fast-track 安装：创建 Red Hat Enterprise Linux CoreOS(RHCOS)机器"和"Full installation: Creating Red Hat Enterprise Linux CoreOS(RHCOS)机器"部分。

3.

确保您的网络基础架构提供集群组件之间所需的网络连接。有关 *要求的详情，请参阅用户置备的基础架构* 的网络要求部分。

4.

将防火墙配置为启用 OpenShift Container Platform 集群组件进行通信所需的端口。*如需有关所需端口的详细信息，请参阅用户置备的基础架构* 部分的网络要求。



重要

默认情况下，OpenShift Container Platform 集群可以访问端口 1936，因为每个 control plane 节点都需要访问此端口。

避免使用 Ingress 负载均衡器公开此端口，因为这样做可能会导致公开敏感信息，如统计信息和指标（与 Ingress Controller 相关的统计信息和指标）。

5. 为集群设置所需的 DNS 基础架构。

   a. 为 Kubernetes API、应用程序通配符、bootstrap 机器、control plane 机器和计算机器配置 DNS 名称解析。

   b. 为 Kubernetes API、bootstrap 机器、control plane 机器和计算机器配置反向 DNS 解析。

   如需有关 *OpenShift Container Platform DNS 要求的更多信息，请参阅用户置备* DNS 要求部分。

6. 验证您的 DNS 配置。

   a. 从安装节点，针对 Kubernetes API 的记录名称、通配符路由和集群节点运行 DNS 查找。验证响应中的 IP 地址是否与正确的组件对应。

   b. 从安装节点，针对负载均衡器和集群节点的 IP 地址运行反向 DNS 查找。验证响应中的记录名称是否与正确的组件对应。

   有关详细的 *DNS 验证步骤，请参阅用户置备的基础架构*验证 DNS 解析部分。

7. 置备所需的 API 和应用程序入口负载平衡基础架构。有关 *要求的更多信息，请参阅用户置备的基础架构的负载平衡* 要求部分。

注意

某些负载平衡解决方案要求在初始化负载平衡之前，对群集节点进行 DNS 名称解析。

18.5.6. 验证用户置备的基础架构的 DNS 解析

您可以在在用户置备的基础架构上安装 OpenShift Container Platform 前验证 DNS 配置。

重要

本节中详述的验证步骤必须在安装集群前成功。

先决条件

- 已为您的用户置备的基础架构配置了所需的 DNS 记录。

流程

1. 从安装节点，针对 Kubernetes API 的记录名称、通配符路由和集群节点运行 DNS 查找。验证响应中包含的 IP 地址是否与正确的组件对应。

   a. 对 Kubernetes API 记录名称执行查询。检查结果是否指向 API 负载均衡器的 IP 地址：

   ```
   $ dig +noall +answer @<nameserver_ip> api.<cluster_name>.<base_domain> 1
   ```

   1 将 <nameserver_ip> 替换为 nameserver 的 IP 地址，<cluster_name> 替换为您的集群名称，<base_domain> 替换为您的基本域名。

   输出示例

   ```
   api.ocp4.example.com.  604800 IN A 192.168.1.5
   ```

b.

对 Kubernetes 内部 API 记录名称执行查询。检查结果是否指向 API 负载均衡器的 IP 地址：

```
$ dig +noall +answer @<nameserver_ip> api-int.<cluster_name>.<base_domain>
```

输出示例

```
api-int.ocp4.example.com.  604800 IN A 192.168.1.5
```

c.

测试 *.apps.<cluster_name>.<base_domain> DNS 通配符查找示例。所有应用程序通配符查询都必须解析为应用程序入口负载均衡器的 IP 地址：

```
$ dig +noall +answer @<nameserver_ip> random.apps.<cluster_name>.
<base_domain>
```

输出示例

```
random.apps.ocp4.example.com.  604800 IN A 192.168.1.5
```

注意

在示例中，将相同的负载均衡器用于 Kubernetes API 和应用程序入口流量。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。

您可以使用另一个通配符值替换 random。例如，您可以查询到 OpenShift Container Platform 控制台的路由：

```
$ dig +noall +answer @<nameserver_ip> console-openshift-console.apps.
<cluster_name>.<base_domain>
```

输出示例

```
console-openshift-console.apps.ocp4.example.com. 604800 IN A 192.168.1.5
```

d. 针对 bootstrap DNS 记录名称运行查询。检查结果是否指向 bootstrap 节点的 IP 地址：

```
$ dig +noall +answer @<nameserver_ip> bootstrap.<cluster_name>.
<base_domain>
```

输出示例

```
bootstrap.ocp4.example.com.  604800 IN A 192.168.1.96
```

e. 使用此方法对 control plane 和计算节点的 DNS 记录名称执行查找。检查结果是否与每个节点的 IP 地址对应。

2. 从安装节点，针对负载均衡器和集群节点的 IP 地址运行反向 DNS 查找。验证响应中包含的记录名称是否与正确的组件对应。

a. 对 API 负载均衡器的 IP 地址执行反向查找。检查响应是否包含 Kubernetes API 和 Kubernetes 内部 API 的记录名称：

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.5
```

输出示例

```
5.1.168.192.in-addr.arpa. 604800 IN PTR api-int.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. 604800 IN PTR api.ocp4.example.com. 2
```

1 为 Kubernetes 内部 API 提供记录名称。

2 为 Kubernetes API 提供记录名称。

注意

OpenShift Container Platform 应用程序通配符不需要 PTR 记录。针对应用程序入口负载均衡器的 IP 地址解析反向 DNS 解析不需要验证步骤。

b. 对 bootstrap 节点的 IP 地址执行反向查找。检查结果是否指向 bootstrap 节点的 DNS 记录名称：

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.96
```

输出示例

```
96.1.168.192.in-addr.arpa. 604800 IN PTR bootstrap.ocp4.example.com.
```

c. 使用此方法对 control plane 和计算节点的 IP 地址执行反向查找。检查结果是否与每个节点的 DNS 记录名称对应。

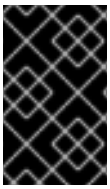
18.5.7. 为集群节点 SSH 访问生成密钥对

在 OpenShift Container Platform 安装过程中，您可以为安装程序提供 SSH 公钥。密钥通过它们的 Ignition 配置文件传递给 Red Hat Enterprise Linux CoreOS(RHCOS)节点，用于验证对节点的 SSH 访问。密钥添加到每个节点上 core 用户的 ~/.ssh/authorized_keys 列表中，这将启用免密码身份验证。

将密钥传递给节点后，您可以使用密钥对作为用户 核心 通过 SSH 连接到 RHCOS 节点。若要通过

SSH 访问节点，必须由 SSH 为您的本地用户管理私钥身份。

如果要通过 SSH 连接到集群节点来执行安装调试或灾难恢复，则必须在安装过程中提供 SSH 公钥。./openshift-install gather 命令还需要在集群节点上设置 SSH 公钥。



重要

不要在生产环境中跳过这个过程，在生产环境中需要灾难恢复和调试。

流程

1. 如果您在本地计算机上没有可用于在集群节点上进行身份验证的现有 SSH 密钥对，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 ~/.ssh/id_ed25519。如果您已有密钥对，请确保您的公钥位于 ~/.ssh 目录中。

注意

如果您计划在 x86_64、ppc64le 和 s390x 架构上安装使用 RHEL 加密库（这些加密库已提交给 NIST 用于 FIPS 140-2/140-3 验证）的 OpenShift Container Platform 集群，则不要创建使用 ed25519 算法的密钥。相反，创建一个使用 rsa 或 ecdsa 算法的密钥。

2. 查看公共 SSH 密钥：

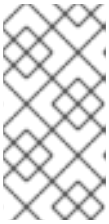
```
$ cat <path>/<file_name>.pub
```

例如，运行以下命令来查看 ~/.ssh/id_ed25519.pub 公钥：

```
$ cat ~/.ssh/id_ed25519.pub
```

3. 将 SSH 私钥身份添加到本地用户的 SSH 代理（如果尚未添加）。在集群节点上，或者要使

用 ./openshift-install gather 命令，需要对该密钥进行 SSH 代理管理，才能在集群节点上进行免密码 SSH 身份验证。



注意

在某些发行版中，自动管理默认 SSH 私钥身份，如 ~/.ssh/id_rsa 和 ~/.ssh/id_dsa。

a. 如果 ssh-agent 进程尚未为您的本地用户运行，请将其作为后台任务启动：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果集群处于 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

4. 将 SSH 私钥添加到 ssh-agent ：

```
$ ssh-add <path>/<file_name> 1
```

1 指定 SSH 私钥的路径和文件名，如 ~/.ssh/id_ed25519.pub

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

后续步骤

- 安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

18.5.8. 手动创建安装配置文件

安装集群要求您手动创建安装配置文件。

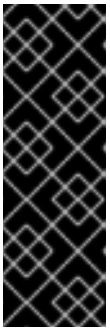
先决条件

- 您在本地机器上有一个 SSH 公钥来提供给安装程序。该密钥将用于在集群节点上进行 SSH 身份验证，以进行调试和灾难恢复。
- 已获取 OpenShift Container Platform 安装程序和集群的 pull secret。

流程

1. 创建一个安装目录来存储所需的安装资产：

```
$ mkdir <installation_directory>
```



重要

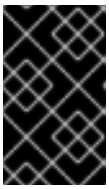
您必须创建一个目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

2. 自定义提供的 install-config.yaml 文件模板示例，并将其保存在 <installation_directory> 中。

注意

此配置文件必须命名为 **install-config.yaml**。

3. 备份 **install-config.yaml** 文件，以便您可以使用它安装多个集群。



重要

install-config.yaml 文件会在安装过程的下一步中使用。现在必须备份它。

其他资源

- IBM Z® 的安装配置参数

18.5.8.1. IBM Z 的 install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多详情，或修改所需参数的值。

```
apiVersion: v1
baseDomain: example.com (1)
compute: (2)
- hyperthreading: Enabled (3)
  name: worker
  replicas: 0 (4)
  architecture: s390x
controlPlane: (5)
  hyperthreading: Enabled (6)
  name: master
  replicas: 3 (7)
  architecture: s390x
metadata:
  name: test (8)
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14 (9)
    hostPrefix: 23 (10)
  networkType: OVNKubernetes (11)
  serviceNetwork: (12)
  - 172.30.0.0/16
platform:
  none: {} (13)
fips: false (14)
```

```
pullSecret: '{"auths":{"<local_registry>": {"auth": "<credentials>","email":
"you@example.com"}}}' 15
sshKey: 'ssh-ed25519 AAAA...' 16
additionalTrustBundle: | 17
  -----BEGIN CERTIFICATE-----
  ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
  -----END CERTIFICATE-----
imageContentSources: 18
- mirrors:
  - <local_repository>/ocp4/openshift4
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <local_repository>/ocp4/openshift4
  source: quay.io/openshift-release-dev/ocp-v4.0-art-dev
```

1

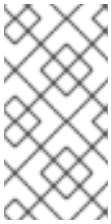
集群的基域。所有 DNS 记录都必须是这个基域的子域，并包含集群名称。

2 5

controlPlane 部分是一个单个映射，但 compute 部分是一系列映射。为满足不同数据结构的要求，compute 部分的第一行必须以连字符 - 开头，controlPlane 部分 的第一行则不以连字符开头。仅使用一个 control plane 池。

3 6

指定要启用或禁用并发多线程(SMT)还是超线程。默认情况下，启用 SMT 可提高机器中内核的性能。您可以通过将 参数值设置为 Disabled 来禁用它。如果禁用 SMT，则必须在所有集群机器中禁用它；这包括 control plane 和计算机器。



注意

默认启用并发多线程(SMT)。如果 OpenShift Container Platform 节点上没有 SMT，超线程 参数无效。

重要

如果您禁用 超线程，无论是在 OpenShift Container Platform 节点上，还是在 install-config.yaml 文件中，请确保您的容量规划考虑机器性能显著降低的情况。

4

在用户置备的基础架构上安装 OpenShift Container Platform 时，必须将这个值设置为 0。在安装程序置备的安装中，参数控制集群为您创建和管理的计算机器数量。在用户置备的安装中，您必须在完成集群安装前手动部署计算机器。

注意

如果要安装一个三节点集群，在安装 Red Hat Enterprise Linux CoreOS(RHCOS)机器时不要部署任何计算机器。

7

您添加到集群的 control plane 机器数量。由于集群使用这些值作为集群中的 etcd 端点数量，所以该值必须与您部署的 control plane 机器数量匹配。

8

您在 DNS 记录中指定的集群名称。

9

从中分配 Pod IP 地址的 IP 地址块。此块不得与现有物理网络重叠。这些 IP 地址用于 pod 网络。如果需要从外部网络访问 pod，您必须配置负载均衡器和路由器来管理流量。

注意

类 E CIDR 范围被保留以供以后使用。要使用 Class E CIDR 范围，您必须确保您的网络环境接受 Class E CIDR 范围内的 IP 地址。

10

分配给每个节点的子网前缀长度。例如，如果 hostPrefix 设为 23，则每个节点从 given cidr 中分配 a /23 子网，这样就能有 510（2^(32 - 23)- 2）个 pod IP 地址。如果需要从外部网络访问节点，请配置负载均衡器和路由器来管理流量。

11

要安装的集群网络插件。默认值 OVNKubernetes 是唯一支持的值。

12

用于服务 IP 地址的 IP 地址池。您只能输入一个 IP 地址池。此块不得与现有物理网络重叠。如果您需要从外部网络访问服务，请配置负载均衡器和路由器来管理流量。

13

您必须将平台设置为 none。您无法为 IBM Z® 基础架构提供额外的平台配置变量。

重要

使用平台类型 none 安装的集群无法使用一些功能，如使用 Machine API 管理计算机器。即使附加到集群的计算机器安装在通常支持该功能的平台上，也会应用这个限制。在安装后无法更改此参数。

14

是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 Red Hat Enterprise Linux (RHEL) 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅在 FIPS 模式中安装该系统。

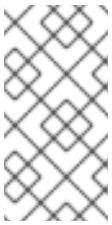
当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS)时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。

15

对于 <local_registry>，请指定 registry 域名，以及您的镜像 registry 用来提供内容的可选端口。例如： registry.example.com 或 registry.example.com:5000。对于 <credentials>，请为您的镜像 registry 指定 base64 编码的用户名和密码。

16

Red Hat Enterprise Linux CoreOS(RHCOS)中 core 用户的 SSH 公钥。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 ssh-agent 进程使用的 SSH 密钥。

17

添加 additionalTrustBundle 参数和值。该值必须是您用于镜像 registry 的证书文件内容。证书文件可以是现有的可信证书颁发机构，也可以是您为镜像 registry 生成的自签名证书。

18

根据您用来镜像存储库的命令输出提供 imageContentSources 部分。

重要

- 使用 oc adm release mirror 命令时，请使用 imageContentSources 部分中的输出。
- 使用 oc mirror 命令时，请使用运行该命令结果的 ImageContentSourcePolicy 文件的 repositoryDigestMirrors 部分。
- ImageContentSourcePolicy 已被弃用。如需更多信息，请参阅*配置镜像 registry 存储库镜像*。

18.5.8.2. 在安装过程中配置集群范围的代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 install-config.yaml 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 install-config.yaml 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 Proxy 对象的 spec.noProxy 字段中添加站点来绕过代理。

注意

Proxy 对象 status.noProxy 字段使用安装配置中的 networking.machineNetwork[].cidr、networking.clusterNetwork[].cidr 和 networking.serviceNetwork[] 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，Proxy 对象 status.noProxy 字段也会使用实例元数据端点填充(169.254.169.254)。

流程

1. 编辑 install-config.yaml 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 http。

2. 用于创建集群外 HTTPS 连接的代理 URL。

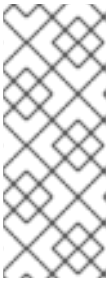
3. 要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 . 以仅匹配子域。例如，.y.com 匹配 x.y.com，但不匹配 y.com。使用 * 绕过所有目的地的代理。

4. 如果提供，安装程序会在 openshift-config 命名空间中生成名为 user-ca-bundle 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 trusted-ca-bundle 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， Proxy 对象的 trustedCA 字段中也会引用此配置映射。additionalTrustBundle 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

5. 可选：决定 Proxy 对象的配置以引用 trustedCA 字段中 user-ca-bundle 配置映射的策略。允许的值是 Proxyonly 和 Always。仅在配置了 http/https 代理时，使用 Proxyonly 引用 user-ca-bundle 配置映射。使用 Always 始终引用 user-ca-bundle 配置映射。默认值为 Proxyonly。

注意

安装程序不支持代理的 readinessEndpoints 字段。



注意

如果安装程序超时，重启并使用安装程序的 wait-for 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. **保存该文件并在安装 OpenShift Container Platform 时引用。**

安装程序会创建一个名为 cluster 的集群范围代理，该代理 使用 提供的 install-config.yaml 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 cluster Proxy 对象，但它会有一个空 spec。

注意

只支持名为 cluster 的 Proxy 对象，且无法创建额外的代理。

18.5.8.3. 配置三节点集群

另外，您可以在由三台 control plane 机器组成的最少三个节点集群中部署零台计算机器。这为集群管理员和开发人员提供了更小、效率更高的集群，用于测试、开发和生产。

在三节点 OpenShift Container Platform 环境中，三台 control plane 机器可以调度，这意味着应用程序工作负载被调度到它们上运行。

先决条件

- **您有一个现有的 install-config.yaml 文件。**

流程

- **确保 install-config.yaml 文件中的计算副本数量设置为 0，如以下 计算 小节所示：**

```
compute:
- name: worker
  platform: {}
  replicas: 0
```

注意

在用户置备的基础架构上安装 OpenShift Container Platform 时，无论您要部署的计算机器数量有多少，您必须将计算机器的 replicas 参数值设置为 0。在安装程序置备的安装中，参数控制集群为您创建和管理的计算机器数量。这不适用于手动部署计算机器的用户置备安装。

注意

control plane 节点的首选资源是 6 个 vCPU 和 21 GB。对于三个 control plane 节点，这是相当于至少五节点集群的内存 + vCPU。您应该为三个节点提供支持，每个节点安装在一个 120 GB 的磁盘上，并且启用了三个 SMT2 的 IFL。测试最小的设置是每个 control plane 节点在 120 GB 磁盘上的三个 vCPU 和 10 GB。

对于三节点集群安装，请按照以下步骤执行：

- 如果要部署一个带有零计算节点的三节点集群，Ingress Controller Pod 在 control plane 节点上运行。在三节点集群部署中，您必须配置应用程序入口负载均衡器，将 HTTP 和 HTTPS 流量路由到 control plane 节点。如需更多信息，*请参阅用户置备的基础架构的负载平衡要求* 部分。

- 在以下步骤中创建 Kubernetes 清单文件时，请确保 <installation_directory>/manifests/cluster-scheduler-02-config.yml 文件中的 mastersSchedulable 参数被设置为 true。这可让应用程序工作负载在 control plane 节点上运行。

- 在创建 Red Hat Enterprise Linux CoreOS(RHCOS)机器时，不要部署任何计算节点。

18.5.9. Cluster Network Operator 配置

集群网络的配置作为 Cluster Network Operator(CNO)配置的一部分指定，并存储在名为 cluster 的自定义资源(CR)对象中。CR 指定 operator.openshift.io API 组中的 Network API 的字段。

CNO 配置在集群安装过程中从 Network.config.openshift.io API 组中的 Network API 继承以下字段：

clusterNetwork

从中分配 Pod IP 地址的 IP 地址池。

serviceNetwork

服务的 IP 地址池.

defaultNetwork.type

集群网络插件。OVNKubernetes 是安装期间唯一支持的插件。

您可以通过在名为 cluster 的 CNO 对象中设置 defaultNetwork 对象的字段来为集群指定集群网络插件配置。

18.5.9.1. Cluster Network Operator 配置对象

下表中描述了 Cluster Network Operator(CNO)的字段：

表 18.65. Cluster Network Operator 配置对象

字段	类型	描述
metadata.name	**字符串**	CNO 对象的名称。这个名称始终是 **集群**。
spec.clusterNetwork	**array**	用于指定从哪些 IP 地址块分配 Pod IP 地址以及集群中每个节点的子网前缀长度的列表。例如： spec: clusterNetwork: - cidr: 10.128.0.0/19 hostPrefix: 23 - cidr: 10.128.32.0/19 hostPrefix: 23

字段	类型	描述
spec.serviceNetwork	**array**	服务的 IP 地址块。OpenShift SDN 和 OVN-Kubernetes 网络插件只支持服务网络的一个 IP 地址块。例如： spec: serviceNetwork: - 172.30.0.0/14 您只能在创建清单前在 **install-config.yaml** 文件中自定义此字段。该值在清单文件中是只读的。
spec.defaultNetwork	**object**	为集群网络配置网络插件。
spec.kubeProxyConfig	**object**	此对象的字段指定 kube-proxy 配置。如果使用 OVN-Kubernetes 集群网络供应商，则 kube-proxy 配置不会起作用。

defaultNetwork 对象配置

下表列出了 defaultNetwork 对象的值：

表 18.66. defaultNetwork 对象

字段	类型	描述
type	**字符串**	**OVNKubernetes**。Red Hat OpenShift Networking 网络插件在安装过程中被选择。此值在集群安装后无法更改。 **注意** OpenShift Container Platform 默认使用 OVN-Kubernetes 网络插件。OpenShift SDN 不再作为新集群的安装选择提供。
ovnKubernetesConfig	**object**	此对象仅对 OVN-Kubernetes 网络插件有效。

配置 OVN-Kubernetes 网络插件

下表描述了 OVN-Kubernetes 网络插件的配置字段：

表 18.67. ovnKubernetesConfig object

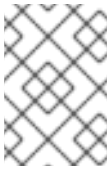
字段	类型	描述
mtu	**integer**	Geneve（通用网络虚拟化封装）覆盖网络的最大传输单元 (MTU)。这根据主网络接口的 MTU 自动探测。您通常不需要覆盖检测到的 MTU。 如果自动探测的值不是您期望的值，请确认节点上主网络接口上的 MTU 是否正确。您不能使用这个选项更改节点上主网络接口的 MTU 值。 如果集群中不同节点需要不同的 MTU 值，则必须将此值设置为 **比** 集群中的最低 MTU 值小 100。例如，如果集群中的某些节点的 MTU 为 **9001**，而某些节点的 MTU 为 **1500**，则必须将此值设置为 **1400**。
genevePort	**integer**	用于所有 Geneve 数据包的端口。默认值为 **6081**。此值在集群安装后无法更改。
ipsecConfig	**object**	指定用于自定义 IPsec 配置的配置对象。
ipv4	**object**	为 IPv4 设置指定配置对象。
ipv6	**object**	为 IPv6 设置指定配置对象。
policyAuditConfig	**object**	指定用于自定义网络策略审计日志的配置对象。如果未设置，则使用默认的审计日志设置。
gatewayConfig	**object**	可选：指定一个配置对象来自定义如何将出口流量发送到节点网关。  **注意** 在迁移出口流量时，工作负载和服务流量会受到一定影响，直到 Cluster Network Operator (CNO) 成功推出更改。

表 18.68. ovnKubernetesConfig.ipv4 object

字段	类型	描述
internalTransitSwitchSubnet	字符串	如果您的现有网络基础架构与 **100.88.0.0/16** IPv4 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。启用东西流量的分布式传输交换机的子网。此子网不能与 OVN-Kubernetes 或主机本身使用的任何其他子网重叠。必须足够大，以适应集群中的每个节点一个 IP 地址。 默认值为 **100.88.0.0/16**。

字段	类型	描述
internalJoinSubnet	字符串	如果您的现有网络基础架构与 **100.64.0.0/16** IPv4 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。您必须确保 IP 地址范围没有与 OpenShift Container Platform 安装使用的任何其他子网重叠。IP 地址范围必须大于可添加到集群的最大节点数。例如，如果 **clusterNetwork.cidr** 值为 **10.128.0.0/14**，并且 **clusterNetwork.hostPrefix** 值为 **/23**，则最大节点数量为 **2^(23-14)=512**。 默认值为 **100.64.0.0/16**。

表 18.69. ovnKubernetesConfig.ipv6 object

字段	类型	描述
internalTransitSwitchSubnet	字符串	如果您的现有网络基础架构与 **fd97::/64** IPv6 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。启用东西流量的分布式传输交换机的子网。此子网不能与 OVN-Kubernetes 或主机本身使用的任何其他子网重叠。必须足够大，以适应集群中的每个节点一个 IP 地址。 默认值为 **fd97::/64**。
internalJoinSubnet	字符串	如果您的现有网络基础架构与 **fd98::/64** IPv6 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。您必须确保 IP 地址范围没有与 OpenShift Container Platform 安装使用的任何其他子网重叠。IP 地址范围必须大于可添加到集群的最大节点数。 默认值为 **fd98::/64**。

表 18.70. policyAuditConfig object

字段	类型	描述
rateLimit	整数	每个节点每秒生成一次的消息数量上限。默认值为每秒 **20** 条消息。
maxFileSize	整数	审计日志的最大大小，以字节为单位。默认值为 **50000000** 或 50 MB。
maxLogFiles	整数	保留的日志文件的最大数量。

字段	类型	描述
目的地	字符串	以下附加审计日志目标之一： **libc** 主机上的 journald 进程的 libc **syslog（）** 函数。 **UDP:<host>:<port>** 一个 syslog 服务器。将 **<host>:<port>** 替换为 **syslog 服务器的主机** 和端口。 **Unix:<file>** 由 **<file>** 指定的 Unix 域套接字文件。 **null** 不要将审计日志发送到任何其他目标。
syslogFacility	字符串	syslog 工具，如 as **kern**，如 RFC5424 定义。默认值为 **local0。**

表 18.71. gatewayConfig object

字段	类型	描述
routingViaHost	**布尔值**	将此字段设置为 **true**，将来自 pod 的出口流量发送到主机网络堆栈。对于依赖于在内核路由表中手动配置路由的高级别安装和应用程序，您可能需要将出口流量路由到主机网络堆栈。默认情况下，出口流量在 OVN 中进行处理以退出集群，不受内核路由表中的特殊路由的影响。默认值为 **false**。 此字段与 Open vSwitch 硬件卸载功能有交互。如果将此字段设置为 **true**，则不会获得卸载的性能优势，因为主机网络堆栈会处理出口流量。
ipForwarding	**object**	您可以使用 **Network** 资源中的 **ipForwarding** 规格来控制 OVN-Kubernetes 管理接口上所有流量的 IP 转发。指定 **Restricted** 只允许 Kubernetes 相关流量的 IP 转发。指定 **Global** 以允许转发所有 IP 流量。对于新安装，默认值为 **Restricted**。对于到 OpenShift Container Platform 4.14 或更高版本的更新，默认值为 **Global**。
ipv4	**object**	可选：指定一个对象来为主机配置内部 OVN-Kubernetes 伪装地址，以服务 IPv4 地址的流量。
ipv6	**object**	可选：指定一个对象来为主机配置内部 OVN-Kubernetes 伪装地址，以服务 IPv6 地址的流量。

表 18.72. gatewayConfig.ipv4 对象

字段	类型	描述
internalMasqueradeSubnet	**字符串**	内部使用的伪装 IPv4 地址，以启用主机服务流量。主机配置了这些 IP 地址和共享网关网桥接口。默认值为 **169.254.169.0/29**。

表 18.73. gatewayConfig.ipv6 对象

字段	类型	描述
internalMasqueradeSubnet	**字符串**	内部使用的伪装 IPv6 地址，以启用主机服务流量。主机配置了这些 IP 地址和共享网关网桥接口。默认值为 **fd69::/125**。

表 18.74. ipsecConfig 对象

字段	类型	描述
模式	**字符串**	指定 IPsec 实现的行为。必须是以下值之一： • **Disabled**: 在集群节点上不启用 IPsec。 • **External** ：对于带有外部主机的网络流量，启用 IPsec。 • **Full**: IPsec 为带有外部主机的 pod 流量和网络流量启用 IPsec。

启用 IPSec 的 OVN-Kubernetes 配置示例

```
defaultNetwork:
  type: OVNKubernetes
  ovnKubernetesConfig:
    mtu: 1400
    genevePort: 6081
      ipsecConfig:
        mode: Full
```

重要

使用 OVNKubernetes 可能会导致 IBM Power® 上的堆栈耗尽问题。

kubeProxyConfig 对象配置（仅限 OpenShiftSDN 容器网络接口）

kubeProxyConfig 对象的值在下表中定义：

表 18.75. kubeProxyConfig object

字段	类型	描述
iptablesSyncPeriod	**字符串**	**iptables** 规则的刷新周期。默认值为 **30s**。有效的后缀包括 **s**、**m** 和 **h**，具体参见 Go 时间 包 文档。  **注意** 由于 OpenShift Container Platform 4.3 及更高版本中引进了性能改进，不再需要调整 **iptablesSyncPeriod** 参数。

proxyArguments.iptables-min-sync-period	**array**	刷新 **iptables** 规则前的最短持续时间。此字段确保刷新的频率不会过于频繁。有效的后缀包括 **s**、**m** 和 **h**，具体参见 Go **time** 软件包。默认值为：
		`kubeProxyConfig:`
		`  proxyArguments:`
		`  iptables-min-sync-period:`
		`  - 0s`

18.5.10. 创建 Kubernetes 清单和 Ignition 配置文件

由于您必须修改一些集群定义文件并手动启动集群机器，因此您必须生成 Kubernetes 清单和 Ignition 配置文件来配置机器。

安装配置文件转换为 Kubernetes 清单。清单嵌套到 Ignition 配置文件中，稍后用于配置集群机器。

重要

- OpenShift Container Platform 安装程序生成的 Ignition 配置文件包含 24 小时后过期的证书，然后在该时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 node-bootstrapper 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。

- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

注意

生成清单和 Ignition 文件的安装程序是特定的架构，可以从 客户端镜像镜像获取。安装程序的 Linux 版本仅在 s390x 上运行。此安装程序也可用作 Mac OS 版本。

先决条件

- 已获得 OpenShift Container Platform 安装程序。对于受限网络安装，这些文件位于您的镜像主机上。

- 已创建 install-config.yaml 安装配置文件。

流程

1. 进入包含 OpenShift Container Platform 安装程序的目录，并为集群生成 Kubernetes 清单：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 对于 <installation_directory>，请指定包含您创建的 install-config.yaml 文件的安装目录。

警告

如果您要安装一个三节点集群，请跳过以下步骤，以便可以调度 control plane 节点。

重要

当您将 control plane 节点从默认的不可调度配置为可以调度时，需要额外的订阅。这是因为 control plane 节点变为计算节点。

2. 检查 <installation_directory>/manifests/cluster-scheduler-02-config.yml Kubernetes 清单文件中的 mastersSchedulable 参数是否已设置为 false。此设置可防止在 control plane 机器上调度 pod：

   a. 打开 <installation_directory>/manifests/cluster-scheduler-02-config.yml 文件。

   b. 找到 mastersSchedulable 参数，并确保它被设置为 false。

   c. 保存并退出 文件。

3. 要创建 Ignition 配置文件，请从包含安装程序的目录运行以下命令：

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

1 对于 <installation_directory>，请指定相同的安装目录。

为安装目录中的 bootstrap、control plane 和计算节点创建 Ignition 配置文件。kubeadmin-password 和 kubeconfig 文件在 ./<installation_directory>/auth 目录中创建：

```
.
```

```
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

18.5.11. 安装 RHCOS 并启动 OpenShift Container Platform bootstrap 过程

要在您置备的 IBM Z® 基础架构上安装 OpenShift Container Platform，您必须将 Red Hat Enterprise Linux CoreOS(RHCOS)安装为 Red Hat Enterprise Linux(RHEL)客户机虚拟机。安装 RHCOS 时，您必须为您要安装的机器类型提供 OpenShift Container Platform 安装程序生成的 Ignition 配置文件。如果您配置了适当的网络、DNS 和负载均衡基础架构，OpenShift Container Platform bootstrap 过程会在 RHCOS 机器重启后自动启动。

您可以使用预打包的 QEMU 写时复制(QCOW2)磁盘镜像对 RHCOS 执行快速跟踪安装。或者，您可以在新 QCOW2 磁盘镜像上执行完整安装。

要在系统中添加更多安全性，您可以选择使用 IBM® Secure Execution 安装 RHCOS，然后才能继续快速跟踪安装。

18.5.11.1. 使用 IBM 安全执行安装 RHCOS

在使用 IBM® Secure Execution 安装 RHCOS 前，您必须准备底层基础架构。

先决条件

- IBM® z15 或更高版本，或 IBM® LinuxONE114 或更高版本。
- Red Hat Enterprise Linux (RHEL) 8 或更高版本。
- 您有一个 bootstrap Ignition 文件。该文件不受保护，其他人能够查看和编辑它。
- 您已确认引导镜像在安装后没有更改。
- 您必须以 IBM® Secure Execution 客户机身份运行所有节点。

流程

1. **准备 RHEL KVM 主机以支持 IBM® 安全执行。**

- **默认情况下，KVM 主机不支持 IBM® 安全执行模式中的客户机。要在 IBM® Secure Execution 模式中支持客户机，KVM 主机必须使用内核参数规格 prot_virt=1 在 LPAR 模式中引导。要在 RHEL 8 上启用 prot_virt=1，请按照以下步骤执行：**

a. **进入到 /boot/loader/entries/，以修改您的引导装载程序配置文件 slirpconf。**

b. **添加内核命令行参数 prot_virt=1。**

c. **运行 zipl 命令并重启系统。**

从支持 IBM® Secure Execution 开始的 KVM 主机发出以下内核信息：

```
prot_virt: Reserving <amount>MB as ultravisor base storage.
```

d. **要验证 KVM 主机现在支持 IBM® Secure Execution，请运行以下命令：**

```
# cat /sys/firmware/uv/prot_virt_host
```

输出示例

```
1
```

对于 Linux 实例，此属性的值是 1，它检测其环境与安全主机的一致性。对于其他实例，值为 0。

2. **通过 Ignition 将您的主机密钥添加到 KVM 客户机。**

第一次引导过程中，RHCOS 会查找主机密钥以使用它们重新加密自身。RHCOS 在

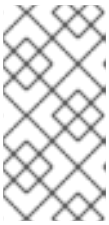
/etc/se-hostkeys 目录中搜索以 ibm-z-hostkey- 开头的文件。集群运行的每台机器的所有主机密钥都必须由管理员加载到目录中。首次启动后，您无法在任何其他机器上运行虚拟机。

注意

您需要在安全系统上准备 Ignition 文件。例如，另一个 IBM® Secure Execution 客户机。

例如：

```
{
 "ignition": { "version": "3.0.0" },
 "storage": {
  "files": [
   {
    "path": "/etc/se-hostkeys/ibm-z-hostkey-<your-hostkey>.crt",
    "contents": {
     "source": "data:;base64,<base64 encoded hostkey document>"
    },
    "mode": 420
   },
   {
    "path": "/etc/se-hostkeys/ibm-z-hostkey-<your-hostkey>.crt",
    "contents": {
     "source": "data:;base64,<base64 encoded hostkey document>"
    },
    "mode": 420
   }
  ]
 }
}
```



注意

如果您希望节点能够在多个 IBM Z® 机器上运行，您可以根据需要添加任意数量的主机密钥。

3. 要生成 Base64 编码字符串，请运行以下命令：

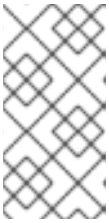
```
base64 <your-hostkey>.crt
```

与不运行 IBM® Secure Execution 的客户机相比，机器的第一次引导将较长，因为整个镜

像在 Ignition 阶段之前使用随机生成的 LUKS 密码短语进行加密。

4. 添加 Ignition 保护

为了保护存储在 Ignition 配置文件中的 secret 被读取或修改，您必须加密 Ignition 配置文件。



注意

为了实现所需的安全性，在运行 IBM® Secure Execution 时默认禁用 Ignition 日志记录和本地登录。

a. 运行以下命令，获取 secex-qemu.qcow2 镜像的公共 GPG 密钥，并使用密钥加密 Ignition 配置：

```
gpg --recipient-file /path/to/ignition.gpg.pub --yes --output /path/to/config.ign.gpg --verbose --armor --encrypt /path/to/config.ign
```

5. 按照 RHCOS 的 fast-track 安装步骤，使用 IBM® Secure Execution QCOW 镜像安装节点。

注意

在启动虚拟机前，请将 serial=ignition 替换为 serial=ignition_crypted，并添加 launchSecurity 参数。

验证

当您完成 RHCOS 的快速跟踪安装，以及 Ignition 在第一次引导时运行后，验证解密是否成功。

- 如果解密成功，您可以预期类似以下示例的输出：

输出示例

```
[    2.801433] systemd[1]: Starting coreos-ignition-setup-user.service - CoreOS Ignition
```

```
User Config Setup...

[    2.803959] coreos-secex-ignition-decrypt[731]: gpg: key <key_name>: public key
"Secure Execution (secex) 38.20230323.dev.0" imported
[    2.808874] coreos-secex-ignition-decrypt[740]: gpg: encrypted with rsa4096 key, ID
<key_name>, created <yyyy-mm-dd>
[  OK  ] Finished coreos-secex-igni…S Secex Ignition Config Decryptor.
```

- 如果解密失败，您可以预期类似以下示例的输出：

输出示例

```
Starting coreos-ignition-s…reOS Ignition User Config Setup...
[    2.863675] coreos-secex-ignition-decrypt[729]: gpg: key <key_name>: public key
"Secure Execution (secex) 38.20230323.dev.0" imported
[    2.869178] coreos-secex-ignition-decrypt[738]: gpg: encrypted with RSA key, ID
<key_name>
[    2.870347] coreos-secex-ignition-decrypt[738]: gpg: public key decryption failed: No
secret key
[    2.870371] coreos-secex-ignition-decrypt[738]: gpg: decryption failed: No secret key
```

其他资源

- Linux 的 IBM® 安全执行简介
- Linux 作为 IBM® Secure Execution 主机或客户机
- 在 IBM Z 上设置 IBM® Secure Execution

18.5.11.2. 在 IBM Z 或 IBM LinuxONE 环境中使用静态 IP 配置 NBDE

在 IBM Z® 或 IBM® LinuxONE 环境中启用 NBDE 磁盘加密需要额外的步骤，本节中详细介绍。

先决条件

- 您已设置了外部 Tang 服务器。具体步骤请查看 网络绑定磁盘加密。
- 您已安装了 with ane 实用程序。
- 您已查看如何使用 Butane 创建机器配置的说明。

流程

1. 为 control plane 和计算节点创建 Butane 配置文件。

   以下 control plane 节点的 Butane 配置示例为磁盘加密创建一个名为 master-storage.bu 的文件：

```
variant: openshift
version: 4.16.0
metadata:
  name: master-storage
  labels:
    machineconfiguration.openshift.io/role: master
storage:
  luks:
    - clevis:
        tang:
          - thumbprint: QcPr_NHFJammnRCA3fFMVdNBwjs
            url: http://clevis.example.com:7500
        options: 1
          - --cipher
          - aes-cbc-essiv:sha256
      device: /dev/disk/by-partlabel/root
      label: luks-root
      name: root
      wipe_volume: true
  filesystems:
    - device: /dev/mapper/root
      format: xfs
      label: root
      wipe_filesystem: true
openshift:
  fips: true 2
```

1. 只有在启用了 FIPS 模式时才需要 cipher 选项。如果禁用了 FIPS，则省略该条目。
2.

2. 运行以下命令，创建自定义 initramfs 文件来引导机器：

```
$ coreos-installer pxe customize \
    /root/rhcos-bootfiles/rhcos-<release>-live-initramfs.s390x.img \
    --dest-device /dev/disk/by-id/scsi-<serial_number> --dest-karg-append \
    ip=<ip_address>::<gateway_ip>:<subnet_mask>::<network_device>:none \
    --dest-karg-append nameserver=<nameserver_ip> \
    --dest-karg-append rd.neednet=1 -o \
    /root/rhcos-bootfiles/<node_name>-initramfs.s390x.img
```

注意

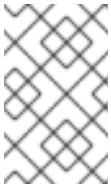
首次引导前，您必须为集群中的每个节点自定义 initramfs，并添加 PXE 内核参数。

3. 创建包含 ignition.platform.id=metal 和 ignition.firstboot 的参数文件。

```
rd.neednet=1 \
console=ttysclp0 \
ignition.firstboot ignition.platform.id=metal \
coreos.live.rootfs_url=http://<http_server>/rhcos-<version>-live-rootfs.<architecture>.img \ 1
coreos.inst.ignition_url=http://<http_server>/master.ign \ 2
ip=10.19.17.2::10.19.17.1:255.255.255.0::enbdd0:none nameserver=10.19.17.1 \
zfcp.allow_lun_scan=0 \
rd.znet=qeth,0.0.bdd0,0.0.bdd1,0.0.bdd2,layer2=1 \
rd.zfcp=0.0.5677,0x600606680g7f0056,0x034F000000000000
```

1 指定您要引导的 kernel 和 initramfs 的 rootfs 工件位置。仅支持 HTTP 和 HTTPS 协议。

2 指定 Ignition 配置文件的位置。使用 master.ign 或 worker.ign。仅支持 HTTP 和 HTTPS 协议。



注意

将参数文件中的所有选项写为一行，并确保您没有换行字符。

其他资源

- 使用 Butane 创建机器配置

18.5.11.3. 使用预打包的 QCOW2 磁盘镜像快速跟踪安装

完成以下步骤，在 Red Hat Enterprise Linux CoreOS(RHCOS)快速跟踪安装中创建机器，导入预打包的 Red Hat Enterprise Linux CoreOS(RHCOS)QEMU 写时复制(QCOW2)磁盘镜像。

先决条件

- 至少有一个 LPAR 在 KVM 的 RHEL 8.6 或更高版本中运行，在此过程中称为 RHEL KVM 主机。
- KVM/QEMU 管理程序安装在 RHEL KVM 主机上。
- 一个域名服务器(DNS)，它可以对节点执行主机名和反向查找。
- 提供 IP 地址的 DHCP 服务器。

流程

1. 从红帽客户门户网站的产品下载页或 RHCOS image mirror 页获得 RHEL QEMU copy-on-write (QCOW2) 磁盘镜像。

重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每个发行版本而改变。您必须下载最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。仅使用以下流程中描述的适当 RHCOS QCOW2 镜像。

2. 将 QCOW2 磁盘镜像和 Ignition 文件下载到 RHEL KVM 主机上的通用目录中。

例如： /var/lib/libvirt/images

注意

Ignition 文件由 OpenShift Container Platform 安装程序生成。

3. 为每个 KVM 客户机节点使用 QCOW2 磁盘镜像文件创建新磁盘镜像。

```
$ qemu-img create -f qcow2 -F qcow2 -b /var/lib/libvirt/images/{source_rhcos_qemu}
/var/lib/libvirt/images/{vmname}.qcow2 {size}
```

4. 使用 Ignition 文件和新磁盘镜像创建新的 KVM 客户机节点。

```
$ virt-install --noautoconsole \
  --connect qemu:///system \
  --name {vm_name} \
  --memory {memory} \
  --vcpus {vcpus} \
  --disk {disk} \
  --launchSecurity type="s390-pv" \ 1
  --import \
  --network network={network},mac={mac} \
  --disk path=
{ign_file},format=raw,readonly=on,serial=ignition,startup_policy=optional 2
```

1. 如果启用了 IBM® Secure Execution，请添加 launchSecurity type="s390-pv" 参数。

2. 如果启用了 IBM® Secure Execution，请将 serial=ignition 替换为 serial=ignition_crypted。

18.5.11.4. 在新 QCOW2 磁盘镜像上完全安装

完成以下步骤，在新的 QEMU 写时复制(QCOW2)磁盘镜像上在完整安装中创建机器。

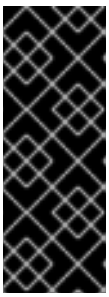
先决条件

- 至少有一个 LPAR 在 KVM 的 RHEL 8.6 或更高版本中运行，在此过程中称为 RHEL KVM 主机。

- KVM/QEMU 管理程序安装在 RHEL KVM 主机上。
- 一个域名服务器(DNS)，它可以对节点执行主机名和反向查找。
- 设置了 HTTP 或 HTTPS 服务器。

流程

1. 从红帽客户门户网站的产品下载页或 RHCOS image mirror 页获取 RHEL kernel, initramfs, 和 rootfs 文件。



重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每个发行版本而改变。您必须下载最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。仅使用以下流程中描述的适当 RHCOS QCOW2 镜像。

文件名包含 OpenShift Container Platform 版本号。它们类似以下示例：

- kernel: rhcos-<version>-live-kernel-<architecture>
- initramfs: rhcos-<version>-live-initramfs.<architecture>.img
- rootfs: rhcos-<version>-live-rootfs.<architecture>.img

2. 在启动 virt-install 前，将下载的 RHEL live kernel、initramfs 和 rootfs 以及 Ignition 文件移到 HTTP 或 HTTPS 服务器中。

注意

Ignition 文件由 OpenShift Container Platform 安装程序生成。

3. 使用 RHEL 内核、initramfs 和 Ignition 文件、新磁盘镜像并调整 parm 行参数，创建新的 KVM 客户机节点。

- 对于 --location，指定 kernel/initrd 在 HTTP 或 HTTPS 服务器上的位置。
- 对于 coreos.inst.ignition_url=，请为机器角色指定 Ignition 文件。使用 bootstrap.ign、master.ign 或 worker.ign。仅支持 HTTP 和 HTTPS 协议。
- 对于 coreos.live.rootfs_url=，请为您引导的内核和 initramfs 指定匹配的 rootfs 构件。仅支持 HTTP 和 HTTPS 协议。

```
$ virt-install \
  --connect qemu:///system \
  --name {vm_name} \
  --vcpus {vcpus} \
  --memory {memory_mb} \
  --disk {vm_name}.qcow2,size={image_size| default(10,true)} \
  --network network={virt_network_parm} \
  --boot hd \
  --location {media_location},kernel={rhcos_kernel},initrd={rhcos_initrd} \
  --extra-args "rd.neednet=1 coreos.inst.install_dev=/dev/vda
coreos.live.rootfs_url={rhcos_liveos} ip={ip}::{default_gateway}:
{subnet_mask_length}:{vm_name}:enc1:none:{MTU} nameserver={dns}
coreos.inst.ignition_url={rhcos_ign}" \
  --noautoconsole \
  --wait
```

18.5.11.5. 高级 RHCOS 安装参考

本节演示了网络配置和其他高级选项，允许您修改 Red Hat Enterprise Linux CoreOS(RHCOS)手动安装过程。下表描述了您可以用于 RHCOS live 安装程序和 coreos-installer 命令的内核参数和命令行选项。

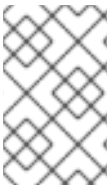
18.5.11.5.1. ISO 安装的网络选项

如果从 ISO 镜像安装 RHCOS，您可以在引导镜像时手动添加内核参数，以便为节点配置网络。如果没有指定网络参数，当 RHCOS 检测到需要网络来获取 Ignition 配置文件时，在 initramfs 中激活 DHCP。

重要

在手动添加网络参数时，还必须添加 rd.neednet=1 内核参数，以便在 initramfs 中启动网络。

以下信息提供了在 RHCOS 节点上为 ISO 安装配置网络的示例。示例描述了如何使用 ip= 和 nameserver= 内核参数。



注意

在添加内核参数时，顺序非常重要：ip= 和 nameserver=。

网络选项在系统引导过程中传递给 dracut 工具。有关由 dracut 支持的网络选项的更多信息，请参阅 dracut.cmdline 手册页。

以下示例是 ISO 安装的网络选项。

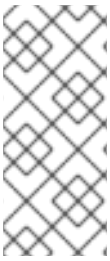
配置 DHCP 或静态 IP 地址

要配置 IP 地址，可使用 DHCP(ip=dhcp)或设置单独的静态 IP 地址(ip=<host_ip>)。如果设置静态 IP，则必须在每个节点上识别 DNS 服务器 IP 地址（名称服务器=<dns_ip>）。以下示例集：

- 节点的 IP 地址为 10.10.10.2
- 网关地址为 10.10.10.254
- 子网掩码为 255.255.255.0
- 到 core0.example.com 的主机名
- DNS 服务器地址为 4.4.4.41

- 自动配置值为 none。当以静态方式配置 IP 网络时，不需要自动配置。

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
nameserver=4.4.4.41
```



注意

当您使用 DHCP 为 RHCOS 机器配置 IP 寻址时，机器还通过 DHCP 获取 DNS 服务器信息。对于基于 DHCP 的部署，您可以通过 DHCP 服务器配置定义 RHCOS 节点使用的 DNS 服务器地址。

配置没有静态主机名的 IP 地址

您可以在不分配静态主机名的情况下配置 IP 地址。如果用户没有设置静态主机名，则会提取并通过反向 DNS 查找自动设置。要在没有静态主机名的情况下配置 IP 地址，请参考以下示例：

- 节点的 IP 地址为 10.10.10.2
- 网关地址为 10.10.10.254
- 子网掩码为 255.255.255.0
- DNS 服务器地址为 4.4.4.41
- 自动配置值为 none。当以静态方式配置 IP 网络时，不需要自动配置。

```
ip=10.10.10.2::10.10.10.254:255.255.255.0::enp1s0:none
nameserver=4.4.4.41
```

指定多个网络接口

您可以通过设置多个 ip= 条目来指定多个网络接口。

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
ip=10.10.10.3::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none
```

配置默认网关和路由

可选：您可以通过设置 a rd.route= 值来配置到额外网络的路由。

注意

当您配置一个或多个网络时，需要一个默认网关。如果额外网络网关与主要网络网关不同，则默认网关必须是主要网络网关。

- 运行以下命令来配置默认网关：

```
ip=::10.10.10.254::::
```

- 输入以下命令为额外网络配置路由：

```
rd.route=20.20.20.0/24:20.20.20.254:enp2s0
```

在单个接口中禁用 DHCP

您可以在单一接口中禁用 DHCP，例如当有两个或者多个网络接口时，且只有一个接口被使用。在示例中，enp1s0 接口具有一个静态网络配置，而 enp2s0 禁用了 DHCP，不使用它：

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
ip=::::core0.example.com:enp2s0:none
```

合并 DHCP 和静态 IP 配置

您可以将系统上的 DHCP 和静态 IP 配置与多个网络接口合并，例如：

```
ip=enp1s0:dhcp
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none
```

在独立接口上配置 VLAN

可选： 您可以使用 vlan= 参数在单个接口上配置 VLAN。

- 要在网络接口中配置 VLAN 并使用静态 IP 地址，请运行以下命令：

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0.100:none
vlan=enp2s0.100:enp2s0
```

- 要在网络接口中配置 VLAN 并使用 DHCP，请运行以下命令：

```
ip=enp2s0.100:dhcp
vlan=enp2s0.100:enp2s0
```

提供多个 DNS 服务器

您可以通过为每个服务器添加一个 nameserver= 条目来提供多个 DNS 服务器，例如

```
nameserver=1.1.1.1
nameserver=8.8.8.8
```

18.5.12. 等待 bootstrap 过程完成

OpenShift Container Platform bootstrap 过程在集群节点首次引导到安装到磁盘的持久 RHCOS 环境后开始。通过 Ignition 配置文件提供的配置信息用于初始化 bootstrap 过程并在机器上安装 OpenShift Container Platform。您必须等待 bootstrap 过程完成。

先决条件

- 已为集群创建 Ignition 配置文件。
- 您已配置了适当的网络、DNS 和负载平衡基础架构。
- 已获得安装程序，并为集群生成 Ignition 配置文件。
- 已在集群机器上安装 RHCOS，并提供 OpenShift Container Platform 安装程序生成的 Ignition 配置文件。

流程

1. 监控 bootstrap 过程：

```
$ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \ 1
    --log-level=info 2
```

1

对于 <installation_directory>，请指定安装文件保存到的目录的路径。

2

要查看不同的安装详情，请指定 warn、debug 或 error，而不是 info。

输出示例

```
INFO Waiting up to 30m0s for the Kubernetes API at
https://api.test.example.com:6443...
INFO API v1.29.4 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
```

当 Kubernetes API 服务器提示已在 control plane 机器上引导它时，该命令会成功。

2. bootstrap 过程完成后，从负载均衡器中删除 bootstrap 机器。

重要

此时您必须从负载均衡器中删除 bootstrap 机器。您还可以删除或重新格式化 bootstrap 机器本身。

18.5.13. 使用 CLI 登录集群

您可以通过导出集群 kubeconfig 文件，以默认系统用户身份登录集群。kubeconfig 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。

- 已安装 oc CLI。

流程

1. 导出 kubeadmin 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 <installation_directory>，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 oc 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

18.5.14. 批准机器的证书签名请求

当您将机器添加到集群时，会为您添加的每台机器生成两个待处理证书签名请求(CSR)。您必须确认这些 CSR 已获得批准，或根据需要自行批准。必须首先批准客户端请求，然后批准服务器请求。

先决条件

- 您已将机器添加到集群中。

流程

1. 确认集群可以识别这些机器：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0  Ready    master  63m  v1.29.4
master-1  Ready    master  63m  v1.29.4
master-2  Ready    master  64m  v1.29.4
```

输出中列出了您创建的所有机器。

注意

在有些 CSR 被批准前，前面的输出可能不包括计算节点（也称为 worker 节点）。

2. 检查待处理的 CSR，并确保添加到集群中的每台机器都有 Pending 或 Approved 状态的客户端请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE    REQUESTOR                                                   CONDITION
csr-8b2br   15m    system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
csr-8vnps   15m    system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
...
```

在本例中，两台机器加入集群。您可能会在列表中看到更多已批准的 CSR。

3. 如果 CSR 没有获得批准，在您添加的机器的所有待处理 CSR 都处于 Pending 状态后，请批准集群机器的 CSR：

注意

由于 CSR 会自动轮转，因此请在将机器添加到集群后一小时内批准您的 CSR。如果没有在一小时内批准它们，证书将会轮转，每个节点会存在多个证书。您必须批准所有这些证书。批准客户端 CSR 后，Kubelet 为服务证书创建一个二级 CSR，这需要手动批准。然后，如果 Kubelet 请求具有相同参数的新证书，则后续提供证书续订请求由 machine-approver 自动批准。

注意

对于在未启用机器 API 的平台上运行的集群，如裸机和其他用户置备的基础架构，您必须实施一种方法来自动批准 kubelet 提供证书请求(CSR)。如果没有批准请求，则 oc exec、ocrsh 和 oc logs 命令将无法成功，因为 API 服务器连接到 kubelet 时需要服务证书。与 Kubelet 端点联系的任何操作都需要此证书批准。该方法必须监视新的 CSR，确认 CSR 由 system: node 或 system:admin 组中的 node-bootstrapper 服务帐户提交，并确认节点的身份。

- 要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 <csr_name> 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}
{{"\n"}}{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```

注意

在有些 CSR 被批准前，一些 Operator 可能无法使用。

4. 现在，您的客户端请求已被批准，您必须查看添加到集群中的每台机器的服务器请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE     REQUESTOR                                                   CONDITION
csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal       Pending
csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal       Pending
...
```

5. 如果剩余的 CSR 没有被批准，且处于 Pending 状态，请批准集群机器的 CSR：

- 要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 <csr_name> 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}
{{"\n"}}{{end}}{{end}}' | xargs oc adm certificate approve
```

6. 批准所有客户端和服务器 CSR 后，机器将 处于 Ready 状态。运行以下命令验证：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0  Ready    master  73m  v1.29.4
master-1  Ready    master  73m  v1.29.4
master-2  Ready    master  74m  v1.29.4
worker-0  Ready    worker  11m  v1.29.4
worker-1  Ready    worker  11m  v1.29.4
```

注意

批准服务器 CSR 后可能需要几分钟时间让机器过渡到 Ready 状态。

其他信息

- 如需有关 CSR 的更多信息，请参阅 证书签名请求。

18.5.15. 初始 Operator 配置

在 control plane 初始化后，您必须立即配置一些 Operator，以便它们都可用。

先决条件

- 您的 control plane 已初始化。

流程

1. 观察集群组件上线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION   AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.16.0    True        False         False      19m
baremetal                                  4.16.0    True        False         False      37m
cloud-credential                           4.16.0    True        False         False      40m
cluster-autoscaler                         4.16.0    True        False         False      37m
```

```
config-operator                          4.16.0    True        False         False      38m
console                                  4.16.0    True        False         False      26m
csi-snapshot-controller                  4.16.0    True        False         False      37m
dns                                      4.16.0    True        False         False      37m
etcd                                     4.16.0    True        False         False      36m
image-registry                           4.16.0    True        False         False      31m
ingress                                  4.16.0    True        False         False      30m
insights                                 4.16.0    True        False         False      31m
kube-apiserver                           4.16.0    True        False         False      26m
kube-controller-manager                  4.16.0    True        False         False      36m
kube-scheduler                           4.16.0    True        False         False      36m
kube-storage-version-migrator            4.16.0    True        False         False      37m
machine-api                              4.16.0    True        False         False      29m
machine-approver                         4.16.0    True        False         False      37m
machine-config                           4.16.0    True        False         False      36m
marketplace                              4.16.0    True        False         False      37m
monitoring                               4.16.0    True        False         False      29m
network                                  4.16.0    True        False         False      38m
node-tuning                              4.16.0    True        False         False      37m
openshift-apiserver                      4.16.0    True        False         False      32m
openshift-controller-manager             4.16.0    True        False         False      30m
openshift-samples                        4.16.0    True        False         False      32m
operator-lifecycle-manager               4.16.0    True        False         False      37m
operator-lifecycle-manager-catalog       4.16.0    True        False         False      37m
operator-lifecycle-manager-packageserver 4.16.0    True        False         False      32m
service-ca                               4.16.0    True        False         False      38m
storage                                  4.16.0    True        False         False      37m
```

2. 配置不可用的 Operator。

18.5.15.1. 禁用默认的 OperatorHub 目录源

在 OpenShift Container Platform 安装过程中，默认为 OperatorHub 配置由红帽和社区项目提供的源内容的 operator 目录。在受限网络环境中，必须以集群管理员身份禁用默认目录。

流程

- 通过在 OperatorHub 对象中添加 disableAllDefaultSources: true 来 禁用默认目录的源：

```
$ oc patch OperatorHub cluster --type json \
    -p '[{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]'
```

提示

或者，您可以使用 Web 控制台管理目录源。在 Administration → Cluster Settings → Configuration → OperatorHub 页面中，点 Sources 选项卡，您可以在其中创建、更新、删除、禁用和启用单独的源。

18.5.15.2. 镜像 registry 存储配置

对于不提供默认存储的平台，Image Registry Operator 最初不可用。安装后，您必须将 registry 配置为使用存储，以便 Registry Operator 可用。

显示配置生产集群所需的持久性卷的说明。如果适用，显示有关将空目录配置为存储位置的说明，这仅适用于非生产集群。

提供了在升级过程中使用 Recreate rollout 策略来允许镜像 registry 使用块存储类型的说明。

18.5.15.2.1. 为 IBM Z 配置 registry 存储

作为集群管理员，在安装后需要配置 registry 来使用存储。

先决条件

- 您可以使用具有 cluster-admin 角色的用户访问集群。
- 在 IBM Z® 上有一个集群。
- 您已为集群置备持久性存储，如 Red Hat OpenShift Data Foundation。

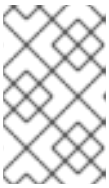
重要

当您只有一个副本时，OpenShift Container Platform 支持对镜像 registry 存储的 ReadWriteOnce 访问。ReadWriteOnce 访问还要求 registry 使用 Recreate rollout 策略。要部署支持高可用性的镜像 registry，需要两个或多个副本，ReadWriteMany 访问。

- 必须具有 100Gi 容量。

流程

1. 要将 registry 配置为使用存储，修改 configs.imageregistry/cluster 资源中的 spec.storage.pvc。



注意

使用共享存储时，请查看您的安全设置以防止外部访问。

2. 验证您没有 registry pod:

```
$ oc get pod -n openshift-image-registry -l docker-registry=default
```

输出示例

```
No resources found in openshift-image-registry namespace
```

注意

如果您的输出中有一个 registry pod，则不需要继续这个过程。

3. 检查 registry 配置：

```
$ oc edit configs.imageregistry.operator.openshift.io
```

输出示例

```
storage:
  pvc:
    claim:
```

将 claim 字段留空以允许自动创建 image-registry-storage PVC。

4. 检查 clusteroperator 状态：

```
$ oc get clusteroperator image-registry
```

输出示例

```
NAME             VERSION   AVAILABLE   PROGRESSING   DEGRADED   SINCE   MESSAGE
image-registry   4.16      True        False         False      6h50m
```

5. 确保 registry 设置为 managed，以启用镜像的构建和推送。

- 运行：

```
$ oc edit configs.imageregistry/cluster
```

然后，更改行

```
managementState: Removed
```

至

```
managementState: Managed
```

18.5.15.2.2. 在非生产集群中为镜像 registry 配置存储

您必须为 Image Registry Operator 配置存储。对于非生产集群，您可以将镜像 registry 设置为空目录。如果您这样做，重启 registry 时会丢失所有镜像。

流程

- 将镜像 registry 存储设置为空目录：

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch
'{"spec":{"storage":{"emptyDir":{}}}}'
```

警告

仅为非生产集群配置这个选项。

如果在 Image Registry Operator 初始化其组件前运行这个命令，oc patch 命令会失败并显示以下错误：

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster"
not found
```

等待几分钟，然后再次运行 命令。

18.5.16. 在用户置备的基础架构上完成安装

完成 Operator 配置后，可以在您提供的基础架构上完成集群安装。

先决条件

- 您的 control plane 已初始化。
- 已完成初始 Operator 配置。

流程

1. 使用以下命令确认所有集群组件都在线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION   AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.16.0    True        False         False      19m
baremetal                                  4.16.0    True        False         False      37m
cloud-credential                           4.16.0    True        False         False      40m
cluster-autoscaler                         4.16.0    True        False         False      37m
config-operator                            4.16.0    True        False         False      38m
console                                    4.16.0    True        False         False      26m
csi-snapshot-controller                    4.16.0    True        False         False      37m
dns                                        4.16.0    True        False         False      37m
etcd                                       4.16.0    True        False         False      36m
image-registry                             4.16.0    True        False         False      31m
ingress                                    4.16.0    True        False         False      30m
insights                                   4.16.0    True        False         False      31m
kube-apiserver                             4.16.0    True        False         False      26m
kube-controller-manager                    4.16.0    True        False         False      36m
kube-scheduler                             4.16.0    True        False         False      36m
kube-storage-version-migrator              4.16.0    True        False         False      37m
machine-api                                4.16.0    True        False         False      29m
machine-approver                           4.16.0    True        False         False      37m
machine-config                             4.16.0    True        False         False      36m
marketplace                                4.16.0    True        False         False      37m
monitoring                                 4.16.0    True        False         False      29m
network                                    4.16.0    True        False         False      38m
node-tuning                                4.16.0    True        False         False      37m
openshift-apiserver                        4.16.0    True        False         False      32m
openshift-controller-manager               4.16.0    True        False         False      30m
openshift-samples                          4.16.0    True        False         False      32m
operator-lifecycle-manager                 4.16.0    True        False         False      37m
operator-lifecycle-manager-catalog         4.16.0    True        False         False      37m
operator-lifecycle-manager-packageserver   4.16.0    True        False         False      32m
service-ca                                 4.16.0    True        False         False      38m
storage                                    4.16.0    True        False         False      37m
```

另外，当所有集群都可用时，以下命令会通知您。它还检索并显示凭证：

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

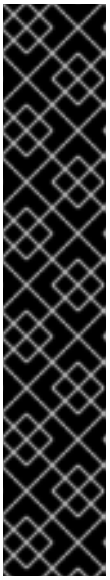
1

对于 <installation_directory>，请指定安装文件保存到的目录的路径。

输出示例

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

Cluster Version Operator 完成从 Kubernetes API 服务器部署 OpenShift Container Platform 集群时，该命令会成功。



重要

- 安装程序生成的 Ignition 配置文件包含 24 小时后过期的证书，然后在该时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 node-bootstrapper 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。

- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

2. 确认 Kubernetes API 服务器正在与 pod 通信。

   a. 要查看所有 pod 的列表，请使用以下命令：

```
$ oc get pods --all-namespaces
```

输出示例

```
NAMESPACE                  NAME                                  READY   STATUS
```

```
RESTARTS   AGE
openshift-apiserver-operator    openshift-apiserver-operator-85cb746d55-zqhs8
1/1    Running    1        9m
openshift-apiserver             apiserver-67b9g                          1/1    Running    0
3m
openshift-apiserver             apiserver-ljcmx                          1/1    Running    0
1m
openshift-apiserver             apiserver-z25h4                          1/1    Running    0
2m
openshift-authentication-operator authentication-operator-69d5d8bf84-vh2n8
1/1    Running    0        5m
...
```

b. 使用以下命令，查看上一命令的输出中所列 pod 的日志：

```
$ oc logs <pod_name> -n <namespace> 1
```

1 指定 pod 名称和命名空间，如上一命令的输出中所示。

如果 pod 日志显示，Kubernetes API 服务器可以与集群机器通信。

3. 对于使用光纤通道协议(FCP)的安装，还需要额外的步骤才能启用多路径。不要在安装过程中启用多路径。

如需更多信息，请参阅 *安装后机器配置任务* 文档中的"使用 RHCOS 上使用内核参数启用多路径"。

4. 在 Cluster registration 页面注册 您的集群。

其他资源

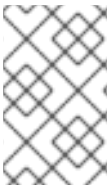
- 如何在没有 SSH 的 OpenShift Container Platform 版本 4 节点中生成 SOSREPORT。

18.5.17. 后续步骤

- 自定义集群。
- 如果您用来安装集群的镜像 registry 具有可信任的 CA，请通过 配置额外的信任存储将其添加到集群中。
- 如果需要，您可以选择 不使用远程健康报告。
- 如果需要，请参阅 注册断开连接的集群

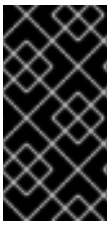
18.6. 在 IBM Z 和 IBM LINUXONE 的 LPAR 上安装集群

在 OpenShift Container Platform 版本 4.16 中，您可以在您置备的 IBM Z® 或 IBM® LinuxONE 基础架构上安装集群。



注意

虽然本文档只涉及 IBM Z®，但它的所有信息也适用于 IBM® LinuxONE。



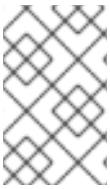
重要

非裸机平台还有其他注意事项。在安装 OpenShift Container Platform 集群前，请参阅有关在未经测试的平台上部署 OpenShift Container Platform 的指南 中的信息。

18.6.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读有关 选择集群安装方法的文档，并为用户准备它。
- 在开始安装过程前，您必须清理安装目录。这样可确保在安装过程中创建和更新所需的安装文件。
- 已为集群置备了 使用 OpenShift Data Foundation 或其他支持的存储协议的持久性存储。要部署私有镜像 registry，您必须使用 ReadWriteMany 访问设置持久性存储。

- 如果使用防火墙，则会 将其配置为允许集群需要访问的站点。



注意

如果要配置代理，请务必查看此站点列表。

18.6.2. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类型的基础架构上执行受限网络安装。在此过程中，您可以下载所需的内容，并使用它为镜像 registry 填充安装软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群前，您要更新镜像 registry 的内容。

18.6.3. 具有用户置备基础架构的集群的要求

对于包含用户置备的基础架构的集群，您必须部署所有所需的机器。

本节论述了在用户置备的基础架构上部署 OpenShift Container Platform 的要求。

18.6.3.1. 集群安装所需的机器

最小的 OpenShift Container Platform 集群需要以下主机：

表 18.76. 最低所需的主机

主机	描述
一个临时 bootstrap 机器	集群需要 bootstrap 机器在三台 control plane 机器上部署 OpenShift Container Platform 集群。您可在安装集群后删除 bootstrap 机器。
三台 control plane 机器	control plane 机器运行组成 control plane 的 Kubernetes 和 OpenShift Container Platform 服务。
至少两台计算机器，也称为 worker 机器。	OpenShift Container Platform 用户请求的工作负载在计算机器上运行。

重要

要保持集群的高可用性，请将独立的物理主机用于这些集群机器。

bootstrap 和 control plane 机器必须使用 Red Hat Enterprise Linux CoreOS(RHCOS)作为操作系统。但是，计算机器可以在 Red Hat Enterprise Linux CoreOS(RHCOS)、Red Hat Enterprise Linux(RHEL) 8.6 和更高的版本。

请注意，RHCOS 基于 Red Hat Enterprise Linux(RHEL) 9.2，并继承其所有硬件认证和要求。查看红帽企业 Linux 技术功能和限制。

18.6.3.2. 集群安装的最低资源要求

每台集群机器都必须满足以下最低要求：

表 18.77. 最低资源要求

机器	操作系统	vCPU [1]	虚拟内存	Storage	每秒输入/输出(IOPS)
bootstrap	RHCOS	4	16 GB	100 GB	N/A

机器	操作系统	vCPU [1]	虚拟内存	Storage	每秒输入/输出 (IOPS)
Control plane（控制平面）	RHCOS	4	16 GB	100 GB	N/A
Compute	RHCOS	2	8 GB	100 GB	N/A

1. 当启用 SMT-2 时，一个物理核心(IFL)提供两个逻辑核心（线程）。管理程序可以提供两个或多个 vCPU。

注意

从 OpenShift Container Platform 版本 4.13 开始，RHCOS 基于 RHEL 版本 9.2，它更新了微架构要求。以下列表包含每个架构需要的最小指令集架构 (ISA)：

- x86-64 体系结构需要 x86-64-v2 ISA
- ARM64 架构需要 ARMv8.0-A ISA
- IBM Power 架构需要 Power 9 ISA
- s390x 架构需要 z14 ISA

如需更多信息，请参阅 RHEL 架构。

如果平台的实例类型满足集群机器的最低要求，则 OpenShift Container Platform 支持使用它。

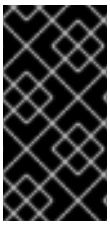
其他资源

- 优化存储

18.6.3.3. 最低 IBM Z 系统环境

您可以在以下 IBM® 硬件上安装 OpenShift Container Platform 版本 4.16：

- IBM® z16 （所有型号）、IBM® z15 （所有型号）、IBM® z14 （所有型号）
- IBM® LinuxONE 4 （所有型号）、IBM® LinuxONE （所有型号）、IBM® LinuxONE Emperor II、IBM® LinuxONE Rockhopper II

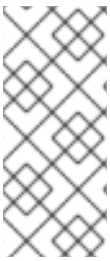


重要

在没有虚拟机监控程序的 IBM Z® 上运行 OpenShift Container Platform 时，请使用 Dynamic Partition Manager (DPM)来管理您的机器。

硬件要求

- Linux(IFL)等效的、启用了 SMT2 的 Linux 集成设施，每个群集都启用了 SMT2。
- 至少一个网络连接连接到 LoadBalancer 服务，并为集群外的流量提供数据。



注意

您可以使用专用或共享的 IFL 来分配足够的计算资源。资源共享是 IBM Z® 的关键优势之一。但是，您必须正确调整每个虚拟机监控程序层上的容量，并确保每个 OpenShift Container Platform 集群都有足够资源。

重要

由于集群的整体性能会受到影响，用于设置 OpenShift Container Platform 集群的 LPAR 必须提供足够的计算容量。就此而言，管理程序级别上的 LPAR 权重管理、授权和 CPU 共享扮演着重要角色。

操作系统要求

- 五个逻辑分区(LPAR)
  - 用于 OpenShift Container Platform control plane 机器的三个 LPAR

- 用于 OpenShift Container Platform 计算机器的两个 LPAR

- 一台用于临时 OpenShift Container Platform bootstrap 机器的机器

IBM Z 网络连接要求

要在 LPAR 中安装 IBM Z®，您需要：

- 直接附加的 OSA 或 RoCE 网络适配器

- 对于首选设置，请使用 OSA 链接聚合。

磁盘存储

- 附加了 FICON 的磁盘存储(DASD)。这些可以是必须格式化为 CDL 的专用 DASD，这是默认设置。要达到 Red Hat Enterprise Linux CoreOS(RHCOS)安装所需的最小 DASD 大小，您需要扩展地址卷(EAV)。如果可用，请使用 HyperPAV 来确保最佳性能。

- FCP 连接的磁盘存储

存储/主内存

- OpenShift Container Platform control plane 机器需要 16 GB

- OpenShift Container Platform 计算机器需要 8 GB

- 临时 OpenShift Container Platform bootstrap 机器需要 16 GB

其他资源

- IBM® 文档中的 处理器资源/系统管理器规划指南，了解 PR/SM 模式注意事项。

- IBM® 文档中的用于 DPM 模式的 IBM Dynamic partition Manager (DPM)指南。

- LPAR 权重管理和权利的性能的主题。
- IBM Z® 和 IBM® LinuxONE 环境的推荐主机实践

18.6.3.4. 首选 IBM Z 系统环境

硬件要求

- 三个 LPARS，每个都相当于 6 个 IFL（每个集群启用了 SMT2）。
- 两个网络连接连接到 LoadBalancer 服务，并为集群外的流量提供数据。
- 作为设备直接附加到节点的 HiperSockets。要将 HiperSockets 直接连接到节点，您必须通过 RHEL 8 客户机设置到外部网络的网关来桥接到 HiperSockets 网络。

操作系统要求

- 用于 OpenShift Container Platform control plane 机器的三个 LPAR。
- 至少 6 个 LPAR 用于 OpenShift Container Platform 计算机器。
- 一个机器或 LPAR 用于临时 OpenShift Container Platform bootstrap 机器。

IBM Z 网络连接要求

要在 LPAR 中安装 IBM Z®，您需要：

- 直接附加的 OSA 或 RoCE 网络适配器
- 对于首选设置，请使用 OSA 链接聚合。

磁盘存储

- 附加了 FICON 的磁盘存储(DASD)。这些可以是必须格式化为 CDL 的专用 DASD，这是默认设置。要达到 Red Hat Enterprise Linux CoreOS(RHCOS)安装所需的最小 DASD 大小，您

需要扩展地址卷(EAV)。如果可用，请使用 HyperPAV 来确保最佳性能。

- FCP 连接的磁盘存储

存储/主内存

- OpenShift Container Platform control plane 机器需要 16 GB
- OpenShift Container Platform 计算机器需要 8 GB
- 临时 OpenShift Container Platform bootstrap 机器需要 16 GB

18.6.3.5. 证书签名请求管理

在使用您置备的基础架构时，集群只能有限地访问自动机器管理，因此您必须提供一种在安装后批准集群证书签名请求 (CSR) 的机制。kube-controller-manager 只能批准 kubelet 客户端 CSR。machine-approver 无法保证使用 kubelet 凭证请求的提供证书的有效性，因为它不能确认是正确的机器发出了该请求。您必须决定并实施一种方法，以验证 kubelet 提供证书请求的有效性并进行批准。

18.6.3.6. 用户置备的基础架构对网络的要求

所有 Red Hat Enterprise Linux CoreOS(RHCOS)机器都需要在启动时在 initramfs 中配置联网，以获取它们的 Ignition 配置文件。

在初次启动过程中，机器需要 HTTP 或 HTTPS 服务器建立网络连接，以下载其 Ignition 配置文件。

机器配置有静态 IP 地址。不需要 DHCP 服务器。确保机器具有持久的 IP 地址和主机名。

Kubernetes API 服务器必须能够解析集群机器的节点名称。如果 API 服务器和 worker 节点位于不同的区域中，您可以配置默认 DNS 搜索区域，以允许 API 服务器解析节点名称。另一种支持的方法是始终通过节点对象和所有 DNS 请求中的完全限定域名引用主机。

18.6.3.6.1. 网络连接要求

您必须配置机器之间的网络连接，以允许 OpenShift Container Platform 集群组件进行通信。每台机器都必须能够解析集群中所有其他机器的主机名。

本节详细介绍了所需的端口。

重要

在连接的 OpenShift Container Platform 环境中，所有节点都需要访问互联网才能为平台容器拉取镜像，并向红帽提供遥测数据。

表 18.78. 用于全机器到所有机器通信的端口

协议	port	描述
ICMP	N/A	网络可访问性测试
TCP	**1936**	指标
	9000-9999	主机级别的服务，包括端口 9 **100-9101** 上的节点导出器，以及端口 **9099** 上的 Cluster Version Operator。
	10250-10259	Kubernetes 保留的默认端口
UDP	**4789**	VXLAN
	6081	Geneve
	9000-9999	主机级别的服务，包括端口 **9100**到**9101** 上的节点导出器。
	500	IPsec IKE 数据包
	4500	IPsec NAT-T 数据包
	123	UDP 端口 **123** 上的网络时间协议 (NTP) 如果配置了外部 NTP 时间服务器，需要打开 UDP 端口 **123**。
TCP/UDP	**30000-32767**	Kubernetes 节点端口
ESP	N/A	IPsec Encapsulating Security Payload(ESP)

表 18.79. 用于所有机器控制平面通信的端口

协议	port	描述
TCP	**6443**	Kubernetes API

表 18.80. control plane 机器用于 control plane 机器通信的端口

协议	port	描述
TCP	**2379-2380**	etcd 服务器和对等端口

用户置备的基础架构的 NTP 配置

OpenShift Container Platform 集群被配置为默认使用公共网络时间协议(NTP)服务器。如果要使用本地企业 NTP 服务器，或者集群部署在断开连接的网络中，您可以将集群配置为使用特定的时间服务器。如需更多信息，*请参阅配置 chrony 时间服务* 的文档。

其他资源

- **配置 chrony 时间服务**

18.6.3.7. 用户置备的 DNS 要求

在 OpenShift Container Platform 部署中，以下组件需要 DNS 名称解析：

- **The Kubernetes API**
- **OpenShift Container Platform 应用程序通配符**
- **bootstrap、control plane 和计算机器**

Kubernetes API、bootstrap 机器、control plane 机器和计算机器也需要反向 DNS 解析。

DNS A/AAAA 或 CNAME 记录用于名称解析，PTR 记录用于反向名称解析。反向记录很重要，因为 Red Hat Enterprise Linux CoreOS(RHCOS)使用反向记录为所有节点设置主机名，除非 DHCP 提供主机名。另外，反向记录用于生成 OpenShift Container Platform 需要操作的证书签名请求(CSR)。

用户置备的 OpenShift Container Platform 集群需要以下 DNS 记录，这些记录必须在安装前就位。

在每个记录中，<cluster_name> 是集群名称，<base_domain> 是您在 install-config.yaml 文件中指定的基域。完整的 DNS 记录采用以下形式： <component>.<cluster_name>.<base_domain>.。

表 18.81. 所需的 DNS 记录

组件	记录	描述
Kubernetes API	**api.<cluster_name>.<base_domain>.**	DNS A/AAAA 或 CNAME 记录，以及用于标识 API 负载均衡器的 DNS PTR 记录。这些记录必须由集群外的客户端和集群中的所有节点解析。
	api-int.<cluster_name>.<base_domain>.	DNS A/AAAA 或 CNAME 记录，以及用于内部标识 API 负载均衡器的 DNS PTR 记录。这些记录必须可以从集群中的所有节点解析。 **重要** API 服务器必须能够根据 Kubernetes 中记录的主机名解析 worker 节点。如果 API 服务器无法解析节点名称，则代理的 API 调用会失败，且您无法从 pod 检索日志。
Routes	***.apps.<cluster_name>.<base_domain>.**	通配符 DNS A/AAAA 或 CNAME 记录，指向应用程序入口负载均衡器。应用程序入口负载均衡器以运行 Ingress Controller Pod 的机器为目标。默认情况下，Ingress Controller Pod 在计算机器上运行。这些记录必须由集群外的客户端和集群中的所有节点解析。 例如，console **-openshift-console.apps.<cluster_name>.<base_domain>** 用作到 OpenShift Container Platform 控制台的通配符路由。
bootstrap 机器	**bootstrap.<cluster_name>.<base_domain>.**	DNS A/AAAA 或 CNAME 记录，以及用于标识 bootstrap 机器的 DNS PTR 记录。这些记录必须由集群中的节点解析。
control plane 机器	**<control_plane><n>.<cluster_name>.<base_domain>.**	DNS A/AAAA 或 CNAME 记录，以识别 control plane 节点的每台机器。这些记录必须由集群中的节点解析。
计算机器	**<compute><n>.<cluster_name>.<base_domain>.**	DNS A/AAAA 或 CNAME 记录，用于识别 worker 节点的每台机器。这些记录必须由集群中的节点解析。

注意

在 OpenShift Container Platform 4.4 及更新的版本中，您不需要在 DNS 配置中指定 etcd 主机和 SRV 记录。

提示

您可以使用 dig 命令验证名称和反向名称解析。如需了解详细的 *验证步骤，请参阅为用户置备的基础架构验证 DNS 解析* 一节。

18.6.3.7.1. 用户置备的集群的 DNS 配置示例

本节提供 A 和 PTR 记录配置示例，它们满足了在用户置备的基础架构上部署 OpenShift Container Platform 的 DNS 要求。样本不是为选择一个 DNS 解决方案提供建议。

在这个示例中，集群名称为 ocp4，基域是 example.com。

用户置备的集群的 DNS A 记录配置示例

以下示例是 BIND 区域文件，其中显示了用户置备的集群中名称解析的 A 记录示例。

例 18.13. DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
  2019070700 ; serial
  3H  ; refresh (3 hours)
  30M  ; retry (30 minutes)
  2W  ; expiry (2 weeks)
  1W )  ; minimum (1 week)
 IN NS ns1.example.com.
 IN MX 10 smtp.example.com.
;
;
ns1.example.com.  IN A 192.168.1.5
smtp.example.com.  IN A 192.168.1.5
;
helper.example.com.  IN A 192.168.1.5
helper.ocp4.example.com. IN A 192.168.1.5
;
api.ocp4.example.com.  IN A 192.168.1.5 1
api-int.ocp4.example.com. IN A 192.168.1.5 2
;
*.apps.ocp4.example.com. IN A 192.168.1.5 3
;
bootstrap.ocp4.example.com. IN A 192.168.1.96 4
;
control-plane0.ocp4.example.com. IN A 192.168.1.97 5
control-plane1.ocp4.example.com. IN A 192.168.1.98 6
control-plane2.ocp4.example.com. IN A 192.168.1.99 7
;
compute0.ocp4.example.com. IN A 192.168.1.11 8
```

```
compute1.ocp4.example.com. IN A 192.168.1.7 9
;
;EOF
```

1

为 Kubernetes API 提供名称解析。记录引用 API 负载均衡器的 IP 地址。

2

为 Kubernetes API 提供名称解析。记录引用 API 负载均衡器的 IP 地址，用于内部集群通信。

3

为通配符路由提供名称解析。记录引用应用程序入口负载均衡器的 IP 地址。应用程序入口负载均衡器以运行 Ingress Controller Pod 的机器为目标。默认情况下，Ingress Controller Pod 在计算机器上运行。

注意

在这个示例中，将相同的负载均衡器用于 Kubernetes API 和应用入口流量。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。

4

为 bootstrap 机器提供名称解析。

5 6 7

为 control plane 机器提供名称解析。

8 9

为计算机器提供名称解析。

用户置备的集群的 DNS PTR 记录配置示例

以下示例 BIND 区域文件显示了用户置备的集群中反向名称解析的 PTR 记录示例。

例 18.14. 反向记录的 DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
  2019070700 ; serial
  3H  ; refresh (3 hours)
  30M  ; retry (30 minutes)
  2W  ; expiry (2 weeks)
  1W )  ; minimum (1 week)
 IN NS ns1.example.com.
;
5.1.168.192.in-addr.arpa. IN PTR api.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. IN PTR api-int.ocp4.example.com. 2
;
96.1.168.192.in-addr.arpa. IN PTR bootstrap.ocp4.example.com. 3
;
97.1.168.192.in-addr.arpa. IN PTR control-plane0.ocp4.example.com. 4
98.1.168.192.in-addr.arpa. IN PTR control-plane1.ocp4.example.com. 5
99.1.168.192.in-addr.arpa. IN PTR control-plane2.ocp4.example.com. 6
;
11.1.168.192.in-addr.arpa. IN PTR compute0.ocp4.example.com. 7
7.1.168.192.in-addr.arpa. IN PTR compute1.ocp4.example.com. 8
;
;EOF
```

1

为 Kubernetes API 提供反向 DNS 解析。PTR 记录引用 API 负载均衡器的记录名称。

2

为 Kubernetes API 提供反向 DNS 解析。PTR 记录引用 API 负载均衡器的记录名称，用于内部集群通信。

3

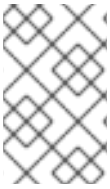
为 bootstrap 机器提供反向 DNS 解析。

4 5 6

为 control plane 机器提供反向 DNS 解析。

7 8

为计算机器提供反向 DNS 解析。

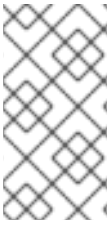


注意

OpenShift Container Platform 应用程序通配符不需要 **PTR** 记录。

18.6.3.8. 用户置备的基础架构的负载均衡要求

在安装 **OpenShift Container Platform** 前，您必须置备 **API** 和应用程序入口负载均衡基础架构。在生产环境中，您可以单独部署 **API** 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。



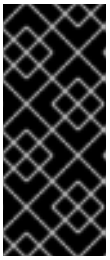
注意

如果要使用 **Red Hat Enterprise Linux (RHEL)** 实例部署 **API** 和应用程序入口负载均衡器，您必须单独购买 **RHEL** 订阅。

负载平衡基础架构必须满足以下要求：

1. **API** 负载均衡器 ：提供一个通用端点，供用户（包括人工和机器）与平台交互和配置。配置以下条件：

   - 仅第 4 层负载均衡.这可被称为 **Raw TCP** 或 **SSL Passthrough** 模式。
   - 无状态负载平衡算法。这些选项根据负载均衡器的实施而有所不同。



重要

不要为 **API** 负载均衡器配置会话持久性。为 **Kubernetes API** 服务器配置会话持久性可能会导致出现过量 **OpenShift Container Platform** 集群应用程序流量，以及过量的在集群中运行的 **Kubernetes API**。

在负载均衡器的前端和后端配置以下端口：

表 18.82. API 负载均衡器

port	后端机器（池成员）	internal	外部	描述
6443	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。您必须为 API 服务器健康检查探测配置 **/readyz** 端点。	X	X	Kubernetes API 服务器
22623	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。	X		机器配置服务器

注意

负载均衡器必须配置为，从 API 服务器关闭 **/readyz** 端点到从池中移除 API 服务器实例时最多需要 30 秒。在 **/readyz** 返回错误或健康后的时间范围内，端点必须被删除或添加。每 5 秒或 10 秒探测一次，有两个成功请求处于健康状态，三个成为不健康的请求是经过良好测试的值。

2. 应用程序入口负载均衡器 ：为应用程序流量从集群外部流提供入口点。OpenShift Container Platform 集群需要正确配置入口路由器。

配置以下条件：

- 仅第 4 层负载均衡.这可被称为 **Raw TCP 或 SSL Passthrough 模式。**
- 建议根据可用选项以及平台上托管的应用程序类型，使用基于连接的或基于会话的持久性。

提示

如果应用程序入口负载均衡器可以看到客户端的真实 IP 地址，启用基于 IP 的会话持久性可以提高使用端到端 TLS 加密的应用程序的性能。

在负载均衡器的前端和后端配置以下端口：

表 18.83. 应用程序入口负载均衡器

port	后端机器（池成员）	internal	外部	描述
443	默认情况下，运行 Ingress Controller Pod、计算或 worker 的机器。	X	X	HTTPS 流量
80	默认情况下，运行 Ingress Controller Pod、计算或 worker 的机器。	X	X	HTTP 流量

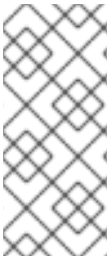
注意

如果要部署一个带有零计算节点的三节点集群，Ingress Controller Pod 在 control plane 节点上运行。在三节点集群部署中，您必须配置应用程序入口负载均衡器，将 HTTP 和 HTTPS 流量路由到 control plane 节点。

18.6.3.8.1. 用户置备的集群的负载均衡器配置示例

本节提供了一个满足用户置备集群的负载均衡要求的 API 和应用程序入口负载均衡器配置示例。示例是 HAProxy 负载均衡器的 /etc/haproxy/haproxy.cfg 配置。这个示例不是为选择一个负载平衡解决方案提供建议。

在这个示例中，将相同的负载均衡器用于 Kubernetes API 和应用入口流量。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。



注意

如果您使用 HAProxy 作为负载均衡器，并且 SELinux 设置为 enforcing，您必须通过运行 setsebool -P haproxy_connect_any=1 来确保 HAProxy 服务可以绑定到配置的 TCP 端口。

例 18.15. API 和应用程序入口负载均衡器配置示例

```
global
  log         127.0.0.1 local2
  pidfile     /var/run/haproxy.pid
  maxconn     4000
  daemon
defaults
  mode                    http
  log                     global
  option                  dontlognull
  option http-server-close
  option                  redispatch
  retries                 3
  timeout http-request    10s
```

```
  timeout queue           1m
  timeout connect         10s
  timeout client          1m
  timeout server          1m
  timeout http-keep-alive 10s
  timeout check           10s
  maxconn                 3000
listen api-server-6443 1
  bind *:6443
  mode tcp
  option  httpchk GET /readyz HTTP/1.0
  option  log-health-checks
  balance roundrobin
  server bootstrap bootstrap.ocp4.example.com:6443 verify none check check-ssl inter 10s
fall 2 rise 3 backup 2
  server master0 master0.ocp4.example.com:6443 weight 1 verify none check check-ssl
inter 10s fall 2 rise 3
  server master1 master1.ocp4.example.com:6443 weight 1 verify none check check-ssl
inter 10s fall 2 rise 3
  server master2 master2.ocp4.example.com:6443 weight 1 verify none check check-ssl
inter 10s fall 2 rise 3
listen machine-config-server-22623 3
  bind *:22623
  mode tcp
  server bootstrap bootstrap.ocp4.example.com:22623 check inter 1s backup 4
  server master0 master0.ocp4.example.com:22623 check inter 1s
  server master1 master1.ocp4.example.com:22623 check inter 1s
  server master2 master2.ocp4.example.com:22623 check inter 1s
listen ingress-router-443 5
  bind *:443
  mode tcp
  balance source
  server compute0 compute0.ocp4.example.com:443 check inter 1s
  server compute1 compute1.ocp4.example.com:443 check inter 1s
listen ingress-router-80 6
  bind *:80
  mode tcp
  balance source
  server compute0 compute0.ocp4.example.com:80 check inter 1s
  server compute1 compute1.ocp4.example.com:80 check inter 1s
```

1

端口 6443 处理 Kubernetes API 流量并指向 control plane 机器。

2 4

bootstrap 条目必须在 OpenShift Container Platform 集群安装前就位，且必须在 bootstrap 过程完成后删除它们。

3

5

端口 443 处理 HTTPS 流量，并指向运行 Ingress Controller pod 的机器。默认情况下，Ingress Controller Pod 在计算机器上运行。

6

端口 80 处理 HTTP 流量，并指向运行 Ingress Controller pod 的机器。默认情况下，Ingress Controller Pod 在计算机器上运行。

注意

如果要部署一个带有零计算节点的三节点集群，Ingress Controller Pod 在 control plane 节点上运行。在三节点集群部署中，您必须配置应用程序入口负载均衡器，将 HTTP 和 HTTPS 流量路由到 control plane 节点。

提示

如果您使用 HAProxy 作为负载均衡器，您可以通过在 HAProxy 节点上运行 netstat -nltupe 来检查 haproxy 进程是否在侦听端口 6443、22623、443 和 80。

18.6.4. 准备用户置备的基础架构

在用户置备的基础架构上安装 OpenShift Container Platform 之前，您必须准备底层基础架构。

本节详细介绍了设置集群基础架构以准备 OpenShift Container Platform 安装所需的高级别步骤。这包括为集群节点配置 IP 网络和网络连接，为 Ignition 文件准备 Web 服务器，通过防火墙启用所需的端口，以及设置所需的 DNS 和负载均衡基础架构。

准备后，集群基础架构必须满足 *带有用户置备的基础架构部分的集群要求*。

先决条件

- 您已参阅 OpenShift Container Platform 4.x Tested Integrations 页面。

- 您已查看了 *具有用户置备基础架构的集群要求部分中详述的基础架构* 要求。

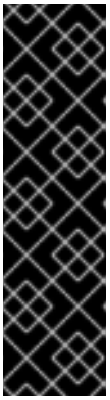
流程

1. 设置静态 IP 地址。

2. 设置 HTTP 或 HTTPS 服务器，为集群节点提供 Ignition 文件。

3. 确保您的网络基础架构提供集群组件之间所需的网络连接。有关 *要求的详情，请参阅用户置备的基础架构* 的网络要求部分。

4. 将防火墙配置为启用 OpenShift Container Platform 集群组件进行通信所需的端口。*如需有关所需端口的详细信息，请参阅用户置备的基础架构* 部分的网络要求。



重要

默认情况下，OpenShift Container Platform 集群可以访问端口 1936，因为每个 control plane 节点都需要访问此端口。

避免使用 Ingress 负载均衡器公开此端口，因为这样做可能会导致公开敏感信息，如统计信息和指标（与 Ingress Controller 相关的统计信息和指标）。

5. 为集群设置所需的 DNS 基础架构。

   a. 为 Kubernetes API、应用程序通配符、bootstrap 机器、control plane 机器和计算机器配置 DNS 名称解析。

   b. 为 Kubernetes API、bootstrap 机器、control plane 机器和计算机器配置反向 DNS 解析。

   如需有关 *OpenShift Container Platform DNS 要求的更多信息，请参阅用户置备* DNS 要求部分。

6. 验证您的 DNS 配置。

   a. 从安装节点，针对 Kubernetes API 的记录名称、通配符路由和集群节点运行 DNS 查找。验证响应中的 IP 地址是否与正确的组件对应。

   b. 从安装节点，针对负载均衡器和集群节点的 IP 地址运行反向 DNS 查找。验证响应中的记录名称是否与正确的组件对应。

   有关详细的 *DNS 验证步骤，请参阅用户置备的基础架构* 验证 DNS 解析部分。

7. 置备所需的 API 和应用程序入口负载平衡基础架构。有关 *要求的更多信息，请参阅用户置备的基础架构的负载平衡* 要求部分。

注意

某些负载平衡解决方案要求在初始化负载平衡之前，对群集节点进行 DNS 名称解析。

18.6.5. 验证用户置备的基础架构的 DNS 解析

您可以在在用户置备的基础架构上安装 OpenShift Container Platform 前验证 DNS 配置。

重要

本节中详述的验证步骤必须在安装集群前成功。

先决条件

- 已为您的用户置备的基础架构配置了所需的 DNS 记录。

流程

1. 从安装节点，针对 Kubernetes API 的记录名称、通配符路由和集群节点运行 DNS 查找。验证响应中包含的 IP 地址是否与正确的组件对应。

   a. 对 Kubernetes API 记录名称执行查询。检查结果是否指向 API 负载均衡器的 IP 地

址：

```
$ dig +noall +answer @<nameserver_ip> api.<cluster_name>.<base_domain> 1
```

1

将 <nameserver_ip> 替换为 nameserver 的 IP 地址，<cluster_name> 替换为您的集群名称，<base_domain> 替换为您的基本域名。

输出示例

```
api.ocp4.example.com.  604800 IN A 192.168.1.5
```

b.

对 Kubernetes 内部 API 记录名称执行查询。检查结果是否指向 API 负载均衡器的 IP 地址：

```
$ dig +noall +answer @<nameserver_ip> api-int.<cluster_name>.<base_domain>
```

输出示例

```
api-int.ocp4.example.com.  604800 IN A 192.168.1.5
```

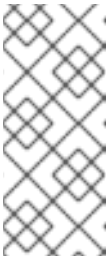
c.

测试 *.apps.<cluster_name>.<base_domain> DNS 通配符查找示例。所有应用程序通配符查询都必须解析为应用程序入口负载均衡器的 IP 地址：

```
$ dig +noall +answer @<nameserver_ip> random.apps.<cluster_name>.
<base_domain>
```

输出示例

```
random.apps.ocp4.example.com.  604800 IN A 192.168.1.5
```



注意

在示例中，将相同的负载均衡器用于 Kubernetes API 和应用程序入口流量。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。

您可以使用另一个通配符值替换 random。例如，您可以查询到 OpenShift Container Platform 控制台的路由：

```
$ dig +noall +answer @<nameserver_ip> console-openshift-console.apps.
<cluster_name>.<base_domain>
```

输出示例

```
console-openshift-console.apps.ocp4.example.com. 604800 IN A 192.168.1.5
```

d. 针对 bootstrap DNS 记录名称运行查询。检查结果是否指向 bootstrap 节点的 IP 地址：

```
$ dig +noall +answer @<nameserver_ip> bootstrap.<cluster_name>.
<base_domain>
```

输出示例

```
bootstrap.ocp4.example.com.  604800 IN A 192.168.1.96
```

e. 使用此方法对 control plane 和计算节点的 DNS 记录名称执行查找。检查结果是否与每个节点的 IP 地址对应。

2. 从安装节点，针对负载均衡器和集群节点的 IP 地址运行反向 DNS 查找。验证响应中包含的记录名称是否与正确的组件对应。

a. 对 API 负载均衡器的 IP 地址执行反向查找。检查响应是否包含 Kubernetes API 和 Kubernetes 内部 API 的记录名称：

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.5
```

输出示例

```
5.1.168.192.in-addr.arpa. 604800 IN PTR api-int.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. 604800 IN PTR api.ocp4.example.com. 2
```

1 为 Kubernetes 内部 API 提供记录名称。

2 为 Kubernetes API 提供记录名称。

注意

OpenShift Container Platform 应用程序通配符不需要 PTR 记录。针对应用程序入口负载均衡器的 IP 地址解析反向 DNS 解析不需要验证步骤。

b. 对 bootstrap 节点的 IP 地址执行反向查找。检查结果是否指向 bootstrap 节点的 DNS 记录名称：

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.96
```

输出示例

```
96.1.168.192.in-addr.arpa. 604800 IN PTR bootstrap.ocp4.example.com.
```

c. 使用此方法对 control plane 和计算节点的 IP 地址执行反向查找。检查结果是否与每个节点的 DNS 记录名称对应。

18.6.6. 为集群节点 SSH 访问生成密钥对

在 OpenShift Container Platform 安装过程中，您可以为安装程序提供 SSH 公钥。密钥通过它们的 Ignition 配置文件传递给 Red Hat Enterprise Linux CoreOS(RHCOS)节点，用于验证对节点的 SSH 访问。密钥添加到每个节点上 core 用户的 ~/.ssh/authorized_keys 列表中，这将启用免密码身份验证。

将密钥传递给节点后，您可以使用密钥对作为用户 核心 通过 SSH 连接到 RHCOS 节点。若要通过 SSH 访问节点，必须由 SSH 为您的本地用户管理私钥身份。

如果要通过 SSH 连接到集群节点来执行安装调试或灾难恢复，则必须在安装过程中提供 SSH 公钥。./openshift-install gather 命令还需要在集群节点上设置 SSH 公钥。

重要

不要在生产环境中跳过这个过程，在生产环境中需要灾难恢复和调试。

流程

1. 如果您在本地计算机上没有可用于在集群节点上进行身份验证的现有 SSH 密钥对，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 ~/.ssh/id_ed25519。如果您已有密钥对，请确保您的公钥位于 ~/.ssh 目录中。

注意

如果您计划在 x86_64、ppc64le 和 s390x 架构上安装使用 RHEL 加密库（这些加密库已提交给 NIST 用于 FIPS 140-2/140-3 验证）的 OpenShift Container Platform 集群，则不要创建使用 ed25519 算法的密钥。相反，创建一个使用 rsa 或 ecdsa 算法的密钥。

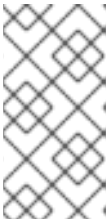
2. 查看公共 SSH 密钥：

```
$ cat <path>/<file_name>.pub
```

例如，运行以下命令来查看 ~/.ssh/id_ed25519.pub 公钥：

```
$ cat ~/.ssh/id_ed25519.pub
```

3. 将 SSH 私钥身份添加到本地用户的 SSH 代理（如果尚未添加）。在集群节点上，或者要使用 ./openshift-install gather 命令，需要对该密钥进行 SSH 代理管理，才能在集群节点上进行免密码 SSH 身份验证。



注意

在某些发行版中，自动管理默认 SSH 私钥身份，如 ~/.ssh/id_rsa 和 ~/.ssh/id_dsa。

a. 如果 ssh-agent 进程尚未为您的本地用户运行，请将其作为后台任务启动：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果集群处于 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

4. 将 SSH 私钥添加到 ssh-agent ：

```
$ ssh-add <path>/<file_name> 1
```

1 指定 SSH 私钥的路径和文件名，如 ~/.ssh/id_ed25519.pub

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

后续步骤

- 安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

18.6.7. 获取安装程序

在安装 OpenShift Container Platform 前，将安装文件下载到您用于安装的主机上。

先决条件

- 您有一台运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB。

流程

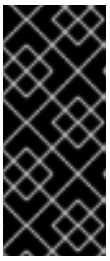
1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐户，请使用您的凭证登录。如果没有，请创建一个帐户。

2. 选择您的基础架构供应商。

3. 进入到安装类型的页面，下载与您的主机操作系统和架构对应的安装程序，并将该文件放在您要存储安装配置文件的目录中。

重要

安装程序会在用来安装集群的计算机上创建几个文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。



重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，请为特定云供应商完成 OpenShift Container Platform 卸载流程。

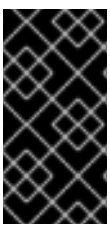
4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret。此 pull secret 允许您与所含授权机构提供的服务进行身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

18.6.8. 安装 OpenShift CLI

您可以安装 OpenShift CLI(oc)来使用命令行界面与 OpenShift Container Platform 进行交互。您可以在 Linux、Windows 或 macOS 上安装 oc。



重要

如果安装了旧版本的 oc，则可能无法使用 OpenShift Container Platform 4.16 中的所有命令。下载并安装新版本的 oc。

在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI(oc)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 产品变体 下拉列表中选择架构。
3. 从 版本 下拉列表中选择适当的版本。
4. 点 OpenShift v4.16 Linux Client 条目旁的 Download Now 来保存文件。
5. 解包存档：

```
$ tar xvf <file>
```

6. 将 oc 二进制文件放到 PATH 中的目录中。

   要查看您的 PATH，请执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 oc 命令：

```
$ oc <command>
```

在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI(oc)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 版本 下拉列表中选择适当的版本。
3. 点 OpenShift v4.16 Windows Client 条目旁的 Download Now 来保存文件。
4. 使用 ZIP 程序解压存档。
5. 将 oc 二进制文件移到 PATH 中的目录中。

   要查看您的 PATH，请打开命令提示并执行以下命令：

   ```
   C:\> path
   ```

验证

- 安装 OpenShift CLI 后，可以使用 oc 命令：

  ```
  C:\> oc <command>
  ```

在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI(oc)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 版本 下拉列表中选择适当的版本。
3. 点 OpenShift v4.16 macOS Client 条目旁的 Download Now 来保存文件。

注意

对于 **macOS arm64**，请选择 **OpenShift v4.16 macOS arm64 Client** 条目。

4. 解包和解压存档。

5. 将 **oc** 二进制文件移到 **PATH** 的目录中。

要查看您的 **PATH**，请打开终端并执行以下命令：

```
$ echo $PATH
```

验证

- 安装 **OpenShift CLI** 后，可以使用 **oc** 命令：

```
$ oc <command>
```

18.6.9. 手动创建安装配置文件

安装集群要求您手动创建安装配置文件。

先决条件

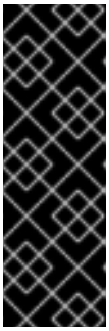
- 您在本地机器上有一个 **SSH** 公钥来提供给安装程序。该密钥将用于在集群节点上进行 **SSH** 身份验证，以进行调试和灾难恢复。

- 已获取 **OpenShift Container Platform** 安装程序和集群的 **pull secret**。

流程

1. 创建一个安装目录来存储所需的安装资产：

```
$ mkdir <installation_directory>
```



重要

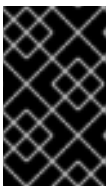
您必须创建一个目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

2. 自定义提供的 install-config.yaml 文件模板示例，并将其保存在 <installation_directory> 中。

注意

此配置文件必须命名为 install-config.yaml。

3. 备份 install-config.yaml 文件，以便您可以使用它安装多个集群。



重要

install-config.yaml 文件会在安装过程的下一步中使用。现在必须备份它。

其他资源

- IBM Z® 的安装配置参数

18.6.9.1. IBM Z 的 install-config.yaml 文件示例

您可以自定义 install-config.yaml 文件，以指定有关 OpenShift Container Platform 集群平台的更多详情，或修改所需参数的值。

```
apiVersion: v1
baseDomain: example.com (1)
compute: (2)
- hyperthreading: Enabled (3)
  name: worker
  replicas: 0 (4)
  architecture: s390x
controlPlane: (5)
  hyperthreading: Enabled (6)
```

```
  name: master
  replicas: 3 7
  architecture: s390x
metadata:
  name: test 8
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14 9
    hostPrefix: 23 10
  networkType: OVNKubernetes 11
  serviceNetwork: 12
  - 172.30.0.0/16
platform:
  none: {} 13
fips: false 14
pullSecret: '{"auths": ...}' 15
sshKey: 'ssh-ed25519 AAAA...' 16
```

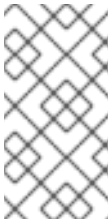
1

2 5

controlPlane 部分是一个单个映射，但 compute 部分是一系列映射。为满足不同数据结构的要求，compute 部分的第一行必须以连字符 - 开头，controlPlane 部分 的第一行则不以连字符开头。仅使用一个 control plane 池。

3 6

指定要启用或禁用并发多线程(SMT)还是超线程。默认情况下，启用 SMT 可提高机器中内核的性能。您可以通过将 参数值设置为 Disabled 来禁用它。如果禁用 SMT，则必须在所有集群机器中禁用它；这包括 control plane 和计算机器。



注意

默认启用并发多线程(SMT)。如果 OpenShift Container Platform 节点上没有 SMT，超线程 参数无效。

重要

如果您禁用 超线程，无论是在 OpenShift Container Platform 节点上，还是在 install-config.yaml 文件中，请确保您的容量规划考虑机器性能显著降低的情况。

4

在用户置备的基础架构上安装 OpenShift Container Platform 时，必须将这个值设置为 0。在安装程序置备的安装中，参数控制集群为您创建和管理的计算机器数量。在用户置备的安装中，您必

须在完成集群安装前手动部署计算机器。

注意

如果要安装一个三节点集群，在安装 Red Hat Enterprise Linux CoreOS(RHCOS)机器时不要部署任何计算机器。

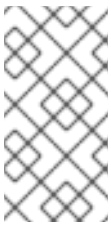
7

您添加到集群的 control plane 机器数量。由于集群使用这些值作为集群中的 etcd 端点数量，所以该值必须与您部署的 control plane 机器数量匹配。

8

9

从中分配 Pod IP 地址的 IP 地址块。此块不得与现有物理网络重叠。这些 IP 地址用于 pod 网络。如果需要从外部网络访问 pod，您必须配置负载均衡器和路由器来管理流量。



注意

类 E CIDR 范围被保留以供以后使用。要使用 Class E CIDR 范围，您必须确保您的网络环境接受 Class E CIDR 范围内的 IP 地址。

10

分配给每个节点的子网前缀长度。例如，如果 hostPrefix 设为 23，则每个节点从 given cidr 中分配 a /23 子网，这样就能有 510（2^(32 - 23)- 2）个 pod IP 地址。如果需要从外部网络访问节点，请配置负载均衡器和路由器来管理流量。

11

要安装的集群网络插件。默认值 OVNKubernetes 是唯一支持的值。

12

用于服务 IP 地址的 IP 地址池。您只能输入一个 IP 地址池。此块不得与现有物理网络重叠。如果您需要从外部网络访问服务，请配置负载均衡器和路由器来管理流量。

13

您必须将平台设置为 none。您无法为 IBM Z® 基础架构提供额外的平台配置变量。

重要

使用平台类型 none 安装的集群无法使用一些功能，如使用 Machine API 管理计算机器。即使附加到集群的计算机器安装在通常支持该功能的平台上，也会应用这个限制。在安装后无法更改此参数。

14

是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 Red Hat Enterprise Linux (RHEL) 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅在 FIPS 模式中安装该系统。

当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS)时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。

15

Red Hat OpenShift Cluster Manager 的 pull secret。此 pull secret 允许您与所含授权机构提供的服务进行身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

16

Red Hat Enterprise Linux CoreOS(RHCOS)中 core 用户的 SSH 公钥。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 ssh-agent 进程使用的 SSH 密钥。

18.6.9.2. 在安装过程中配置集群范围的代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 install-

config.yaml 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 install-config.yaml 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 Proxy 对象的 spec.noProxy 字段中添加站点来绕过代理。

注意

Proxy 对象 status.noProxy 字段使用安装配置中的 networking.machineNetwork[].cidr、networking.clusterNetwork[].cidr 和 networking.serviceNetwork[] 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，Proxy 对象 status.noProxy 字段也会使用实例元数据端点填充(169.254.169.254)。

流程

1. 编辑 install-config.yaml 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 http。

2.

3

要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 . 以仅匹配子域。例如，.y.com 匹配 x.y.com，但不匹配 y.com。使用 * 绕过所有目的地的代理。

4

如果提供，安装程序会在 openshift-config 命名空间中生成名为 user-ca-bundle 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 trusted-ca-bundle 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， Proxy 对象的 trustedCA 字段中也会引用此配置映射。additionalTrustBundle 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

5

可选：决定 Proxy 对象的配置以引用 trustedCA 字段中 user-ca-bundle 配置映射的策略。允许的值是 Proxyonly 和 Always。仅在配置了 http/https 代理时，使用 Proxyonly 引用 user-ca-bundle 配置映射。使用 Always 始终引用 user-ca-bundle 配置映射。默认值为 Proxyonly。

注意

安装程序不支持代理的 readinessEndpoints 字段。

注意

如果安装程序超时，重启并使用安装程序的 wait-for 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. 保存该文件并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 cluster 的集群范围代理，该代理 使用 提供的 install-config.yaml 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 cluster Proxy 对象，但它会有一个空 spec。

注意

只支持名为 cluster 的 Proxy 对象，且无法创建额外的代理。

18.6.9.3. 配置三节点集群

另外，您可以在由三台 control plane 机器组成的最少三个节点集群中部署零台计算机器。这为集群管理员和开发人员提供了更小、效率更高的集群，用于测试、开发和生产。

在三节点 OpenShift Container Platform 环境中，三台 control plane 机器可以调度，这意味着应用程序工作负载被调度到它们上运行。

先决条件

- 您有一个现有的 install-config.yaml 文件。

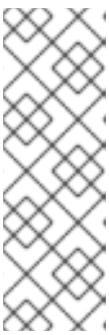
流程

- 确保 install-config.yaml 文件中的计算副本数量设置为 0，如以下 计算 小节所示：

```
compute:
- name: worker
  platform: {}
  replicas: 0
```

注意

在用户置备的基础架构上安装 OpenShift Container Platform 时，无论您要部署的计算机器数量有多少，您必须将计算机器的 replicas 参数值设置为 0。在安装程序置备的安装中，参数控制集群为您创建和管理的计算机器数量。这不适用于手动部署计算机器的用户置备安装。



注意

control plane 节点的首选资源是 6 个 vCPU 和 21 GB。对于三个 control plane 节点，这是相当于至少五节点集群的内存 + vCPU。您应该为三个节点提供支持，每个节点安装在一个 120 GB 的磁盘上，并且启用了三个 SMT2 的 IFL。测试最小的设置是每个 control plane 节点在 120 GB 磁盘上的三个 vCPU 和 10 GB。

对于三节点集群安装，请按照以下步骤执行：

- 如果要部署一个带有零计算节点的三节点集群，Ingress Controller Pod 在 control plane 节点上运行。在三节点集群部署中，您必须配置应用程序入口负载均衡器，将 HTTP 和 HTTPS 流量路由到 control plane 节点。如需更多信息，*请参阅用户置备的基础架构的负载平衡要求* 部分。

- 在以下步骤中创建 Kubernetes 清单文件时，请确保 <installation_directory>/manifests/cluster-scheduler-02-config.yml 文件中的 mastersSchedulable 参数被设置为 true。这可让应用程序工作负载在 control plane 节点上运行。

- 在创建 Red Hat Enterprise Linux CoreOS(RHCOS)机器时，不要部署任何计算节点。

18.6.10. Cluster Network Operator 配置

集群网络的配置作为 Cluster Network Operator(CNO)配置的一部分指定，并存储在名为 cluster 的自定义资源(CR)对象中。CR 指定 operator.openshift.io API 组中的 Network API 的字段。

CNO 配置在集群安装过程中从 Network.config.openshift.io API 组中的 Network API 继承以下字段：

clusterNetwork

从中分配 Pod IP 地址的 IP 地址池。

serviceNetwork

服务的 IP 地址池.

defaultNetwork.type

集群网络插件。OVNKubernetes 是安装期间唯一支持的插件。

您可以通过在名为 cluster 的 CNO 对象中设置 defaultNetwork 对象的字段来为集群指定集群网络插件配置。

18.6.10.1. Cluster Network Operator 配置对象

下表中描述了 Cluster Network Operator(CNO)的字段：

表 18.84. Cluster Network Operator 配置对象

字段	类型	描述
metadata.name	**字符串**	CNO 对象的名称。这个名称始终是 **集群**。

spec.clusterNetwork	**array**	用于指定从哪些 IP 地址块分配 Pod IP 地址以及集群中每个节点的子网前缀长度的列表。例如：
		`spec:`
		`  clusterNetwork:`
		`  - cidr: 10.128.0.0/19`
		`  hostPrefix: 23`
		`  - cidr: 10.128.32.0/19`
		`  hostPrefix: 23`

spec.serviceNetwork	**array**	服务的 IP 地址块。OpenShift SDN 和 OVN-Kubernetes 网络插件只支持服务网络的一个 IP 地址块。例如：
		`spec:`
		`  serviceNetwork:`
		`  - 172.30.0.0/14`
		您只能在创建清单前在 **install-config.yaml** 文件中自定义此字段。该值在清单文件中是只读的。

spec.defaultNetwork	**object**	为集群网络配置网络插件。
spec.kubeProxyConfig	**object**	此对象的字段指定 kube-proxy 配置。如果使用 OVN-Kubernetes 集群网络供应商，则 kube-proxy 配置不会起作用。

defaultNetwork 对象配置

下表列出了 **defaultNetwork** 对象的值：

表 18.85. defaultNetwork 对象

字段	类型	描述

字段	类型	描述
type	**字符串**	**OVNKubernetes**。Red Hat OpenShift Networking 网络插件在安装过程中被选择。此值在集群安装后无法更改。 **注意** OpenShift Container Platform 默认使用 OVN-Kubernetes 网络插件。OpenShift SDN 不再作为新集群的安装选择提供。
ovnKubernetesConfig	**object**	此对象仅对 OVN-Kubernetes 网络插件有效。

配置 OVN-Kubernetes 网络插件

下表描述了 OVN-Kubernetes 网络插件的配置字段：

表 18.86. ovnKubernetesConfig object

字段	类型	描述
mtu	**integer**	Geneve（通用网络虚拟化封装）覆盖网络的最大传输单元 (MTU)。这根据主网络接口的 MTU 自动探测。您通常不需要覆盖检测到的 MTU。 如果自动探测的值不是您期望的值，请确认节点上主网络接口上的 MTU 是否正确。您不能使用这个选项更改节点上主网络接口的 MTU 值。 如果集群中不同节点需要不同的 MTU 值，则必须将此值设置为**比** 集群中的最低 MTU 值小 100。例如，如果集群中的某些节点的 MTU 为 **9001**，而某些节点的 MTU 为 **1500**，则必须将此值设置为 **1400**。
genevePort	**integer**	用于所有 Geneve 数据包的端口。默认值为 **6081**。此值在集群安装后无法更改。
ipsecConfig	**object**	指定用于自定义 IPsec 配置的配置对象。
ipv4	**object**	为 IPv4 设置指定配置对象。
ipv6	**object**	为 IPv6 设置指定配置对象。
policyAuditConfig	**object**	指定用于自定义网络策略审计日志的配置对象。如果未设置，则使用默认的审计日志设置。

字段	类型	描述
gatewayConfig	**object**	可选：指定一个配置对象来自定义如何将出口流量发送到节点网关。  **注意** 在迁移出口流量时，工作负载和服务流量会受到一定影响，直到 Cluster Network Operator (CNO) 成功推出更改。

表 18.87. ovnKubernetesConfig.ipv4 object

字段	类型	描述
internalTransitSwitchSubnet	字符串	如果您的现有网络基础架构与 **100.88.0.0/16** IPv4 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。启用东西流量的分布式传输交换机的子网。此子网不能与 OVN-Kubernetes 或主机本身使用的任何其他子网重叠。必须足够大，以适应集群中的每个节点一个 IP 地址。 默认值为 **100.88.0.0/16**。
internalJoinSubnet	字符串	如果您的现有网络基础架构与 **100.64.0.0/16** IPv4 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。您必须确保 IP 地址范围没有与 OpenShift Container Platform 安装使用的任何其他子网重叠。IP 地址范围必须大于可添加到集群的最大节点数。例如，如果 **clusterNetwork.cidr** 值为 **10.128.0.0/14**，并且 **clusterNetwork.hostPrefix** 值为 **/23**，则最大节点数量为 **2^(23-14)=512**。 默认值为 **100.64.0.0/16**。

表 18.88. ovnKubernetesConfig.ipv6 object

字段	类型	描述
internalTransitSwitchSubnet	字符串	如果您的现有网络基础架构与 **fd97::/64** IPv6 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。启用东西流量的分布式传输交换机的子网。此子网不能与 OVN-Kubernetes 或主机本身使用的任何其他子网重叠。必须足够大，以适应集群中的每个节点一个 IP 地址。 默认值为 **fd97::/64**。

字段	类型	描述
internalJoinSubnet	字符串	如果您的现有网络基础架构与 **fd98::/64** IPv6 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。您必须确保 IP 地址范围没有与 OpenShift Container Platform 安装使用的任何其他子网重叠。IP 地址范围必须大于可添加到集群的最大节点数。 默认值为 **fd98::/64**。

表 18.89. policyAuditConfig object

字段	类型	描述
rateLimit	整数	每个节点每秒生成一次的消息数量上限。默认值为每秒 **20** 条消息。
maxFileSize	整数	审计日志的最大大小，以字节为单位。默认值为 **50000000** 或 50 MB。
maxLogFiles	整数	保留的日志文件的最大数量。
目的地	字符串	以下附加审计日志目标之一： **libc** 主机上的 journald 进程的 libc **syslog（）** 函数。 **UDP:<host>:<port>** 一个 syslog 服务器。将 **<host>:<port>** 替换为 **syslog 服务器的主机** 和端口。 **Unix:<file>** 由 **<file>** 指定的 Unix 域套接字文件。 **null** 不要将审计日志发送到任何其他目标。
syslogFacility	字符串	syslog 工具，如 as **kern**，如 RFC5424 定义。默认值为 **local0。**

表 18.90. gatewayConfig object

字段	类型	描述

字段	类型	描述
routingViaHost	**布尔值**	将此字段设置为 **true**，将来自 pod 的出口流量发送到主机网络堆栈。对于依赖于在内核路由表中手动配置路由的高级别安装和应用程序，您可能需要将出口流量路由到主机网络堆栈。默认情况下，出口流量在 OVN 中进行处理以退出集群，不受内核路由表中的特殊路由的影响。默认值为 **false**。 此字段与 Open vSwitch 硬件卸载功能有交互。如果将此字段设置为 **true**，则不会获得卸载的性能优势，因为主机网络堆栈会处理出口流量。
ipForwarding	**object**	您可以使用 **Network** 资源中的 **ipForwarding** 规格来控制 OVN-Kubernetes 管理接口上所有流量的 IP 转发。指定 **Restricted** 只允许 Kubernetes 相关流量的 IP 转发。指定 **Global** 以允许转发所有 IP 流量。对于新安装，默认值为 **Restricted**。对于到 OpenShift Container Platform 4.14 或更高版本的更新，默认值为 **Global**。
ipv4	**object**	可选：指定一个对象来为主机配置内部 OVN-Kubernetes 伪装地址，以服务 IPv4 地址的流量。
ipv6	**object**	可选：指定一个对象来为主机配置内部 OVN-Kubernetes 伪装地址，以服务 IPv6 地址的流量。

表 18.91. gatewayConfig.ipv4 对象

字段	类型	描述
internalMasqueradeSubnet	**字符串**	内部使用的伪装 IPv4 地址，以启用主机服务流量。主机配置了这些 IP 地址和共享网关网桥接口。默认值为 **169.254.169.0/29**。

表 18.92. gatewayConfig.ipv6 对象

字段	类型	描述
internalMasqueradeSubnet	**字符串**	内部使用的伪装 IPv6 地址，以启用主机服务流量。主机配置了这些 IP 地址和共享网关网桥接口。默认值为 **fd69::/125**。

表 18.93. ipsecConfig 对象

字段	类型	描述

字段	类型	描述
模式	**字符串**	指定 IPsec 实现的行为。必须是以下值之一： • **Disabled**: 在集群节点上不启用 IPsec。 • **External** ：对于带有外部主机的网络流量，启用 IPsec。 • **Full**: IPsec 为带有外部主机的 pod 流量和网络流量启用 IPsec。

启用 IPSec 的 OVN-Kubernetes 配置示例

```
defaultNetwork:
  type: OVNKubernetes
  ovnKubernetesConfig:
    mtu: 1400
    genevePort: 6081
      ipsecConfig:
        mode: Full
```

重要

使用 OVNKubernetes 可能会导致 IBM Power® 上的堆栈耗尽问题。

kubeProxyConfig 对象配置（仅限 OpenShiftSDN 容器网络接口）

kubeProxyConfig 对象的值在下表中定义：

表 18.94. kubeProxyConfig object

字段	类型	描述

字段	类型	描述
iptablesSyncPeriod	**字符串**	**iptables** 规则的刷新周期。默认值为 **30s**。有效的后缀包括 **s**、**m** 和 **h**，具体参见 Go 时间 包 文档。 **注意** 由于 OpenShift Container Platform 4.3 及更高版本中引进了性能改进，不再需要调整 **iptablesSyncPeriod** 参数。
proxyArguments.iptables-min-sync-period	**array**	刷新 **iptables** 规则前的最短持续时间。此字段确保刷新的频率不会过于频繁。有效的后缀包括 **s**、**m** 和 **h**，具体参见 Go **time** 软件包。默认值为： kubeProxyConfig: proxyArguments: iptables-min-sync-period: - 0s

18.6.11. 创建 Kubernetes 清单和 Ignition 配置文件

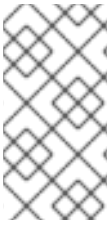
由于您必须修改一些集群定义文件并手动启动集群机器，因此您必须生成 Kubernetes 清单和 Ignition 配置文件来配置机器。

安装配置文件转换为 Kubernetes 清单。清单嵌套到 Ignition 配置文件中，稍后用于配置集群机器。

重要

- OpenShift Container Platform 安装程序生成的 Ignition 配置文件包含 24 小时后过期的证书，然后在该时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 node-bootstrapper 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。

- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。



注意

生成清单和 Ignition 文件的安装程序是特定的架构，可以从 客户端镜像镜像获取。安装程序的 Linux 版本仅在 s390x 上运行。此安装程序也可用作 Mac OS 版本。

先决条件

- 已获得 OpenShift Container Platform 安装程序。
- 已创建 install-config.yaml 安装配置文件。

流程

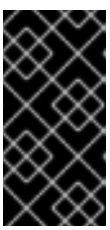
1. 进入包含 OpenShift Container Platform 安装程序的目录，并为集群生成 Kubernetes 清单：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 对于 <installation_directory>，请指定包含您创建的 install-config.yaml 文件的安装目录。

警告

如果您要安装一个三节点集群，请跳过以下步骤，以便可以调度 control plane 节点。



重要

当您将 control plane 节点从默认的不可调度配置为可以调度时，需要额外的订阅。这是因为 control plane 节点变为计算节点。

2. 检查 <installation_directory>/manifests/cluster-scheduler-02-config.yml Kubernetes 清单文件中的 mastersSchedulable 参数是否已设置为 false。此设置可防止在 control plane 机

器上调度 pod：

a. 打开 <installation_directory>/manifests/cluster-scheduler-02-config.yml 文件。

b. 找到 mastersSchedulable 参数，并确保它被设置为 false。

c. 保存并退出 文件。

3. 要创建 Ignition 配置文件，请从包含安装程序的目录运行以下命令：

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

1 对于 <installation_directory>，请指定相同的安装目录。

为安装目录中的 bootstrap、control plane 和计算节点创建 Ignition 配置文件。kubeadmin-password 和 kubeconfig 文件在 ./<installation_directory>/auth 目录中创建：

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

18.6.12. 在 IBM Z 或 IBM LinuxONE 环境中使用静态 IP 配置 NBDE

在 IBM Z® 或 IBM® LinuxONE 环境中启用 NBDE 磁盘加密需要额外的步骤，本节中详细介绍。

先决条件

- 您已设置了外部 Tang 服务器。具体步骤请查看 网络绑定磁盘加密。

- 您已安装了 with ane 实用程序。
- 您已查看如何使用 Butane 创建机器配置的说明。

流程

1. 为 control plane 和计算节点创建 Butane 配置文件。

   以下 control plane 节点的 Butane 配置示例为磁盘加密创建一个名为 master-storage.bu 的文件：

```
variant: openshift
version: 4.16.0
metadata:
  name: master-storage
  labels:
    machineconfiguration.openshift.io/role: master
storage:
  luks:
    - clevis:
        tang:
          - thumbprint: QcPr_NHFJammnRCA3fFMVdNBwjs
            url: http://clevis.example.com:7500
        options: 1
          - --cipher
          - aes-cbc-essiv:sha256
      device: /dev/disk/by-partlabel/root 2
      label: luks-root
      name: root
      wipe_volume: true
  filesystems:
    - device: /dev/mapper/root
      format: xfs
      label: root
      wipe_filesystem: true
openshift:
  fips: true 3
```

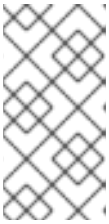
1. 只有在启用了 FIPS 模式时才需要 cipher 选项。如果禁用了 FIPS，则省略该条目。
2. 对于在 DASD 类型磁盘中安装，使用 device: /dev/disk/by-label/root 替换。

3

是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

2. 运行以下命令，创建自定义 initramfs 文件来引导机器：

```
$ coreos-installer pxe customize \
  /root/rhcos-bootfiles/rhcos-<release>-live-initramfs.s390x.img \
  --dest-device /dev/disk/by-id/scsi-<serial_number> --dest-karg-append \
  ip=<ip_address>::<gateway_ip>:<subnet_mask>::<network_device>:none \
  --dest-karg-append nameserver=<nameserver_ip> \
  --dest-karg-append rd.neednet=1 -o \
  /root/rhcos-bootfiles/<node_name>-initramfs.s390x.img
```



注意

首次引导前，您必须为集群中的每个节点自定义 initramfs，并添加 PXE 内核参数。

3. 创建包含 ignition.platform.id=metal 和 ignition.firstboot 的参数文件。

control plane 机器的内核参数文件示例：

```
rd.neednet=1 \
console=ttysclp0 \
coreos.inst.install_dev=/dev/dasda \1
ignition.firstboot ignition.platform.id=metal \
coreos.live.rootfs_url=http://<http_server>/rhcos-<version>-live-rootfs.
<architecture>.img \2
coreos.inst.ignition_url=http://<http_server>/master.ign \3
ip=10.19.17.2::10.19.17.1:255.255.255.0::enbdd0:none nameserver=10.19.17.1 \
zfcp.allow_lun_scan=0 \4
rd.znet=qeth,0.0.bdd0,0.0.bdd1,0.0.bdd2,layer2=1 \
rd.zfcp=0.0.5677,0x600606680g7f0056,0x034F000000000000 \5
```

1

2

指定您要引导的 kernel 和 initramfs 的 rootfs 工件位置。仅支持 HTTP 和 HTTPS 协议。

3

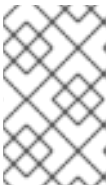
指定 Ignition 配置文件的位置。使用 master.ign 或 worker.ign。仅支持 HTTP 和 HTTPS 协议。

4

对于在 FCP 类型磁盘中安装，请添加 zfcp.allow_lun_scan=0。为 DASD 类型磁盘省略这个值。

5

对于在 DASD 类型磁盘中安装，使用 rd.dasd=0.0.3490 替换来指定 DASD 设备。



注意

将参数文件中的所有选项写为一行，并确保您没有换行字符。

其他资源

- 使用 Butane 创建机器配置

18.6.13. 安装 RHCOS 并启动 OpenShift Container Platform bootstrap 过程

要在您置备的 IBM Z® 基础架构上安装 OpenShift Container Platform，您必须在 LPAR 中安装 Red Hat Enterprise Linux CoreOS (RHCOS)。安装 RHCOS 时，您必须为您要安装的机器类型提供 OpenShift Container Platform 安装程序生成的 Ignition 配置文件。如果您配置了合适的网络、DNS 和负载均衡基础架构，OpenShift Container Platform bootstrap 过程会在 RHCOS 客户机机器重启后自动启动。

完成以下步骤以创建机器。

先决条件

- 在置备机器上运行的 HTTP 或 HTTPS 服务器，可供您创建的机器访问。

- 如果要启用安全引导，需要已获取了适当的红帽产品签名密钥，并在 IBM 文档中的 IBM Z 和 IBM LinuxONE 上读取安全引导。

流程

1. 在您的置备机器上登录到 Linux。

2. 从 RHCOS 镜像镜像获取 Red Hat Enterprise Linux CoreOS(RHCOS)内核、initramfs 和 rootfs 文件。

重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每个发行版本而改变。您必须下载最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。仅使用以下步骤中描述的适当 kernel、initramfs 和 rootfs 工件。

文件名包含 OpenShift Container Platform 版本号。它们类似以下示例：

- kernel: rhcos-<version>-live-kernel-<architecture>
- initramfs: rhcos-<version>-live-initramfs.<architecture>.img
- rootfs: rhcos-<version>-live-rootfs.<architecture>.img

注意

FCP 和 DASD 的 rootfs 镜像相同。

3. 创建参数文件。以下参数特定于特定虚拟机：

- 对于 ip=，请指定以下七项：

  i. 计算机的 IP 地址。

  ii. 个空字符串。

  iii. 网关

  iv. 子网掩码.

  v. hostname.domainname 格式的机器主机和域名。省略这个值可让 RHCOS 决定。

  vi. 网络接口名称。省略这个值可让 RHCOS 决定。

  vii. 如果使用静态 IP 地址，请指定 none。

- 对于 coreos.inst.ignition_url=，请为机器角色指定 Ignition 文件。使用 bootstrap.ign、master.ign 或 worker.ign。仅支持 HTTP 和 HTTPS 协议。

- 对于 coreos.live.rootfs_url=，请为您引导的内核和 initramfs 指定匹配的 rootfs 构件。仅支持 HTTP 和 HTTPS 协议。

- 可选：要启用安全引导，请添加 coreos.inst.secure_ipl

- 对于在 DASD 类型磁盘中安装，请完成以下任务：

  i. 对于 coreos.inst.install_dev=，请指定 /dev/dasda。

ii. Userd .dasd= 指定安装 RHCOS 的 DASD。

iii. 所有其他参数保持不变。

bootstrap 机器的参数文件示例：bootstrap-0.parm ：

```
rd.neednet=1 \
console=ttysclp0 \
coreos.inst.install_dev=/dev/dasda \
coreos.live.rootfs_url=http://<http_server>/rhcos-<version>-live-rootfs.
<architecture>.img \ 1
coreos.inst.ignition_url=http://<http_server>/bootstrap.ign \ 2
coreos.inst.secure_ipl \ 3
ip=172.18.78.2::172.18.78.1:255.255.255.0:::none nameserver=172.18.78.1 \
rd.znet=qeth,0.0.bdf0,0.0.bdf1,0.0.bdf2,layer2=1,portno=0 \
zfcp.allow_lun_scan=0 \
rd.dasd=0.0.3490
```

1 指定您要引导的 kernel 和 initramfs 的 rootfs 工件位置。仅支持 HTTP 和 HTTPS 协议。

2 指定 Ignition 配置文件的位置。使用 bootstrap.ign、master.ign 或 worker.ign。仅支持 HTTP 和 HTTPS 协议。

3 可选： 要启用安全引导，请添加 coreos.inst.secure_ipl。

将参数文件中的所有选项写为一行，并确保您没有换行字符。

- 对于在 FCP 类型磁盘中安装，请完成以下任务：

i. User d.zfcp=<adapter>,<wwpn>,<lun> 以指定要安装 RHCOS 的 FCP 磁盘。对于多路径，为每个额外路径重复此步骤。

注意

当使用多个路径安装时，您必须在安装后直接启用多路径，而不是在以后启用多路径，因为这可能导致问题。

ii. **将安装设备设置为：coreos.inst.install_dev=/dev/disk/by-id/scsi-<serial_number>。**

注意

如果使用 NPIV 配置额外的 LUN，FCP 需要 zfcp.allow_lun_scan=0。如果必须启用 zfcp.allow_lun_scan=1，因为您使用 CSI 驱动程序，则必须配置 NPIV，以便每个节点无法访问另一个节点的引导分区。

iii. **所有其他参数保持不变。**

重要

需要额外的安装后步骤才能完全启用多路径。如需更多信息，请参阅 *安装后机器配置任务* 中的"使用 RHCOS 上内核参数启用多路径"。

以下是使用多路径的计算节点的 worker-1.parm 示例参数文件：

```
rd.neednet=1 \
console=ttysclp0 \
coreos.inst.install_dev=/dev/disk/by-id/scsi-<serial_number> \
coreos.live.rootfs_url=http://<http_server>/rhcos-<version>-live-rootfs.
<architecture>.img \
coreos.inst.ignition_url=http://<http_server>/worker.ign \
ip=172.18.78.2::172.18.78.1:255.255.255.0:::none nameserver=172.18.78.1 \
rd.znet=qeth,0.0.bdf0,0.0.bdf1,0.0.bdf2,layer2=1,portno=0 \
zfcp.allow_lun_scan=0 \
rd.zfcp=0.0.1987,0x50050763070bc5e3,0x4008400B00000000 \
rd.zfcp=0.0.19C7,0x50050763070bc5e3,0x4008400B00000000 \
rd.zfcp=0.0.1987,0x50050763071bc5e3,0x4008400B00000000 \
rd.zfcp=0.0.19C7,0x50050763071bc5e3,0x4008400B00000000
```

将参数文件中的所有选项写为一行，并确保您没有换行字符。

4. 将 initramfs、内核、参数文件和 RHCOS 镜像传送到 LPAR，例如使用 FTP。有关如何使用 FTP 和引导传输文件的详情，请参考在 LPAR 中安装。

5. 引导机器

6. 对集群中的其他机器重复此步骤。

18.6.13.1. 高级 RHCOS 安装参考

本节演示了网络配置和其他高级选项，允许您修改 Red Hat Enterprise Linux CoreOS(RHCOS)手动安装过程。下表描述了您可以用于 RHCOS live 安装程序和 coreos-installer 命令的内核参数和命令行选项。

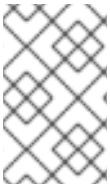
18.6.13.1.1. ISO 安装的网络和绑定选项

如果从 ISO 镜像安装 RHCOS，您可以在引导镜像时手动添加内核参数，以便为节点配置网络。如果没有指定网络参数，当 RHCOS 检测到需要网络来获取 Ignition 配置文件时，在 initramfs 中激活 DHCP。

重要

在手动添加网络参数时，还必须添加 rd.neednet=1 内核参数，以便在 initramfs 中启动网络。

以下信息提供了在 RHCOS 节点上为 ISO 安装配置网络和绑定的示例。示例描述了如何使用 ip=、name server = 和 bond= 内核参数。



注意

添加内核参数时顺序非常重要： ip=、name server=，然后 bond=。

网络选项在系统引导过程中传递给 dracut 工具。有关 dracut 支持的网络选项的更多信息，请参阅 dracut.cmdline 手册页。

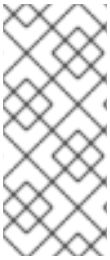
以下示例是 ISO 安装的网络选项。

配置 DHCP 或静态 IP 地址

要配置 IP 地址，可使用 DHCP(ip=dhcp)或设置单独的静态 IP 地址(ip=<host_ip>)。如果设置静态 IP，则必须在每个节点上识别 DNS 服务器 IP 地址（名称服务器=<dns_ip>）。以下示例集：

- 节点的 IP 地址为 10.10.10.2
- 网关地址为 10.10.10.254
- 子网掩码为 255.255.255.0
- 到 core0.example.com 的主机名
- DNS 服务器地址为 4.4.4.41
- 自动配置值为 none。当以静态方式配置 IP 网络时，不需要自动配置。

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
nameserver=4.4.4.41
```



注意

当您使用 DHCP 为 RHCOS 机器配置 IP 寻址时，机器还通过 DHCP 获取 DNS 服务器信息。对于基于 DHCP 的部署，您可以通过 DHCP 服务器配置定义 RHCOS 节点使用的 DNS 服务器地址。

配置没有静态主机名的 IP 地址

您可以在不分配静态主机名的情况下配置 IP 地址。如果用户没有设置静态主机名，则会提取并通过反向 DNS 查找自动设置。要在没有静态主机名的情况下配置 IP 地址，请参考以下示例：

- 节点的 IP 地址为 10.10.10.2

- 网关地址为 10.10.10.254
- 子网掩码为 255.255.255.0
- DNS 服务器地址为 4.4.4.41
- 自动配置值为 none。当以静态方式配置 IP 网络时，不需要自动配置。

```
ip=10.10.10.2::10.10.10.254:255.255.255.0::enp1s0:none
nameserver=4.4.4.41
```

指定多个网络接口

您可以通过设置多个 ip= 条目来指定多个网络接口。

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
ip=10.10.10.3::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none
```

配置默认网关和路由

可选：您可以通过设置 a rd.route= 值来配置到额外网络的路由。

注意

当您配置一个或多个网络时，需要一个默认网关。如果额外网络网关与主要网络网关不同，则默认网关必须是主要网络网关。

- 运行以下命令来配置默认网关：

```
ip=::10.10.10.254::::
```

- 输入以下命令为额外网络配置路由：

```
rd.route=20.20.20.0/24:20.20.20.254:enp2s0
```

在单个接口中禁用 DHCP

您可以在单一接口中禁用 DHCP，例如当有两个或者多个网络接口时，且只有一个接口被使用。在示

例中，enp1s0 接口具有一个静态网络配置，而 enp2s0 禁用了 DHCP，不使用它：

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
ip=::::core0.example.com:enp2s0:none
```

合并 DHCP 和静态 IP 配置

您可以将系统上的 DHCP 和静态 IP 配置与多个网络接口合并，例如：

```
ip=enp1s0:dhcp
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none
```

在独立接口上配置 VLAN

可选：您可以使用 vlan= 参数在单个接口上配置 VLAN。

- 要在网络接口中配置 VLAN 并使用静态 IP 地址，请运行以下命令：

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0.100:none
vlan=enp2s0.100:enp2s0
```

- 要在网络接口中配置 VLAN 并使用 DHCP，请运行以下命令：

```
ip=enp2s0.100:dhcp
vlan=enp2s0.100:enp2s0
```

提供多个 DNS 服务器

您可以通过为每个服务器添加一个 nameserver= 条目来提供多个 DNS 服务器，例如

```
nameserver=1.1.1.1
nameserver=8.8.8.8
```

将多个网络接口绑定到一个接口

可选：您可以使用 bond= 选项将多个网络接口绑定到一个接口。请参见以下示例：

- 配置绑定接口的语法为：bond=<name>[:<network_interfaces>][:options]

  <name> 是绑定设备名称 (bond0)、<network_interfaces> 代表以逗号分隔的物理（以太网）接口列表(em1,em2)，*options* 是用逗号分开的绑定选项列表。输入 modinfo bonding 查看可用选项。

- 当使用 bond= 创建绑定接口时，您必须指定如何分配 IP 地址以及绑定接口的其他信息。

  - 要将绑定接口配置为使用 DHCP，请将绑定的 IP 地址设置为 dhcp。例如：

```
bond=bond0:em1,em2:mode=active-backup
ip=bond0:dhcp
```

  - 要将绑定接口配置为使用静态 IP 地址，请输入您需要的特定 IP 地址和相关信息。例如：

```
bond=bond0:em1,em2:mode=active-backup,fail_over_mac=1
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0:none
```

在 active-backup 模式中始终设置 fail_over_mac=1 选项，以避免使用共享 OSA/RoCE 卡时出现问题。

将多个网络接口绑定到一个接口

可选：您可以使用 vlan= 参数并在绑定接口上配置 VLAN，并使用 DHCP，例如：

```
ip=bond0.100:dhcp
bond=bond0:em1,em2:mode=active-backup
vlan=bond0.100:bond0
```

使用以下示例配置带有 VLAN 的绑定接口并使用静态 IP 地址：

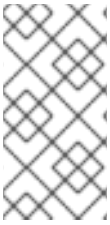
```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0.100:none
bond=bond0:em1,em2:mode=active-backup
vlan=bond0.100:bond0
```

使用网络团队

可选：您可以使用 team= 参数来将网络团队用作绑定的替代选择：

- 配置组接口的语法为：team=name[:network_interfaces]

  name 是组设备名称(team0)，*network_interfaces* 代表以逗号分隔的物理（以太网）接口（em1、em2）列表。



注意

当 RHCOS 切换到即将推出的 RHEL 版本时，团队(team)功能被计划弃用。如需更多信息，请参阅红帽知识库文章。

使用以下示例配置网络团队：

```
team=team0:em1,em2
ip=team0:dhcp
```

18.6.14. 等待 bootstrap 过程完成

OpenShift Container Platform bootstrap 过程在集群节点首次引导到安装到磁盘的持久 RHCOS 环境后开始。通过 Ignition 配置文件提供的配置信息用于初始化 bootstrap 过程并在机器上安装 OpenShift Container Platform。您必须等待 bootstrap 过程完成。

先决条件

- 已为集群创建 Ignition 配置文件。
- 您已配置了适当的网络、DNS 和负载平衡基础架构。
- 已获得安装程序，并为集群生成 Ignition 配置文件。
- 已在集群机器上安装 RHCOS，并提供 OpenShift Container Platform 安装程序生成的 Ignition 配置文件。
- 您的机器可以直接访问互联网，或者有 HTTP 或 HTTPS 代理可用。

流程

1. 监控 bootstrap 过程：

```
$ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \ 1
    --log-level=info 2
```

1

对于 <installation_directory>，请指定安装文件保存到的目录的路径。

2

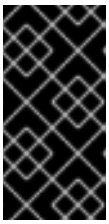
要查看不同的安装详情，请指定 warn、debug 或 error，而不是 info。

输出示例

```
INFO Waiting up to 30m0s for the Kubernetes API at
https://api.test.example.com:6443...
INFO API v1.29.4 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
```

当 Kubernetes API 服务器提示已在 control plane 机器上引导它时，该命令会成功。

2. bootstrap 过程完成后，从负载均衡器中删除 bootstrap 机器。



重要

此时您必须从负载均衡器中删除 bootstrap 机器。您还可以删除或重新格式化 bootstrap 机器本身。

18.6.15. 使用 CLI 登录集群

您可以通过导出集群 kubeconfig 文件，以默认系统用户身份登录集群。kubeconfig 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。

- 已安装 oc CLI。

流程

1. 导出 kubeadmin 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 <installation_directory>，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 oc 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

18.6.16. 批准机器的证书签名请求

当您将机器添加到集群时，会为您添加的每台机器生成两个待处理证书签名请求(CSR)。您必须确认这些 CSR 已获得批准，或根据需要自行批准。必须首先批准客户端请求，然后批准服务器请求。

先决条件

- 您已将机器添加到集群中。

流程

1. 确认集群可以识别这些机器：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0  Ready    master  63m  v1.29.4
master-1  Ready    master  63m  v1.29.4
master-2  Ready    master  64m  v1.29.4
```

输出中列出了您创建的所有机器。

注意

在有些 CSR 被批准前，前面的输出可能不包括计算节点（也称为 worker 节点）。

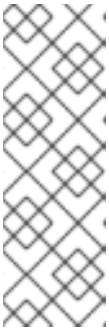
2. 检查待处理的 CSR，并确保添加到集群中的每台机器都有 Pending 或 Approved 状态的客户端请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE   REQUESTOR                              CONDITION
csr-mddf5   20m   system:node:master-01.example.com   Approved,Issued
csr-z5rln   16m   system:node:worker-21.example.com   Approved,Issued
```

3. 如果 CSR 没有获得批准，在您添加的机器的所有待处理 CSR 都处于 Pending 状态 后，请批准集群机器的 CSR：



注意

由于 CSR 会自动轮转，因此请在将机器添加到集群后一小时内批准您的 CSR。如果没有在一小时内批准它们，证书将会轮转，每个节点会存在多个证书。您必须批准所有这些证书。批准客户端 CSR 后，Kubelet 为服务证书创建一个二级 CSR，这需要手动批准。然后，如果 Kubelet 请求具有相同参数的新证书，则后续提供证书续订请求由 machine-approver 自动批准。

注意

对于在未启用机器 API 的平台上运行的集群，如裸机和其他用户置备的基础架构，您必须实施一种方法来自动批准 kubelet 提供证书请求(CSR)。如果没有批准请求，则 oc exec、ocrsh 和 oc logs 命令将无法成功，因为 API 服务器连接到 kubelet 时需要服务证书。与 Kubelet 端点联系的任何操作都需要此证书批准。该方法必须监视新的 CSR，确认 CSR 由 system: node 或 system:admin 组中的 node-bootstrapper 服务帐户提交，并确认节点的身份。

- 要单独批准，请对每个有效的 CSR 运行以下命令：

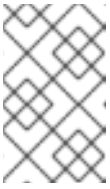
```
$ oc adm certificate approve <csr_name> 1
```

1

<csr_name> 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}
{{"\n"}}{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```



注意

在有些 CSR 被批准前，一些 Operator 可能无法使用。

4. 现在，您的客户端请求已被批准，您必须查看添加到集群中的每台机器的服务器请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE     REQUESTOR                                                   CONDITION
csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal       Pending
csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal       Pending
...
```

5. 如果剩余的 CSR 没有被批准，且处于 Pending 状态，请批准集群机器的 CSR：

- 要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 <csr_name> 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}
{{"\n"}}{{end}}{{end}}' | xargs oc adm certificate approve
```

6. 批准所有客户端和服务器 CSR 后，机器将 处于 Ready 状态。运行以下命令验证：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0  Ready    master  73m  v1.29.4
master-1  Ready    master  73m  v1.29.4
master-2  Ready    master  74m  v1.29.4
worker-0  Ready    worker  11m  v1.29.4
worker-1  Ready    worker  11m  v1.29.4
```

注意

批准服务器 CSR 后可能需要几分钟时间让机器过渡到 Ready 状态。

其他信息

- 如需有关 CSR 的更多信息，请参阅 证书签名请求。

18.6.17. 初始 Operator 配置

在 control plane 初始化后，您必须立即配置一些 Operator，以便它们都可用。

先决条件

- 您的 control plane 已初始化。

流程

1. 观察集群组件上线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION   AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.16.0    True        False         False      19m
baremetal                                  4.16.0    True        False         False      37m
cloud-credential                           4.16.0    True        False         False      40m
cluster-autoscaler                         4.16.0    True        False         False      37m
config-operator                            4.16.0    True        False         False      38m
console                                    4.16.0    True        False         False      26m
csi-snapshot-controller                    4.16.0    True        False         False      37m
dns                                        4.16.0    True        False         False      37m
etcd                                       4.16.0    True        False         False      36m
image-registry                             4.16.0    True        False         False      31m
ingress                                    4.16.0    True        False         False      30m
```

```
insights                                   4.16.0    True        False         False      31m
kube-apiserver                             4.16.0    True        False         False      26m
kube-controller-manager                    4.16.0    True        False         False      36m
kube-scheduler                             4.16.0    True        False         False      36m
kube-storage-version-migrator              4.16.0    True        False         False      37m
machine-api                                4.16.0    True        False         False      29m
machine-approver                           4.16.0    True        False         False      37m
machine-config                             4.16.0    True        False         False      36m
marketplace                                4.16.0    True        False         False      37m
monitoring                                 4.16.0    True        False         False      29m
network                                    4.16.0    True        False         False      38m
node-tuning                                4.16.0    True        False         False      37m
openshift-apiserver                        4.16.0    True        False         False      32m
openshift-controller-manager               4.16.0    True        False         False      30m
openshift-samples                          4.16.0    True        False         False      32m
operator-lifecycle-manager                 4.16.0    True        False         False      37m
operator-lifecycle-manager-catalog         4.16.0    True        False         False      37m
operator-lifecycle-manager-packageserver   4.16.0    True        False         False      32m
service-ca                                 4.16.0    True        False         False      38m
storage                                    4.16.0    True        False         False      37m
```

2. 配置不可用的 Operator。

18.6.17.1. 镜像 registry 存储配置

对于不提供默认存储的平台，Image Registry Operator 最初不可用。安装后，您必须将 registry 配置为使用存储，以便 Registry Operator 可用。

显示配置生产集群所需的持久性卷的说明。如果适用，显示有关将空目录配置为存储位置的说明，这仅适用于非生产集群。

提供了在升级过程中使用 Recreate rollout 策略来允许镜像 registry 使用块存储类型的说明。

18.6.17.1.1. 为 IBM Z 配置 registry 存储

作为集群管理员，在安装后需要配置 registry 来使用存储。

先决条件

- 您可以使用具有 cluster-admin 角色的用户访问集群。
- 在 IBM Z® 上有一个集群。
- 您已为集群置备持久性存储，如 Red Hat OpenShift Data Foundation。

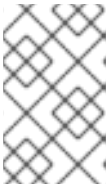
重要

当您只有一个副本时，OpenShift Container Platform 支持对镜像 registry 存储的 ReadWriteOnce 访问。ReadWriteOnce 访问还要求 registry 使用 Recreate rollout 策略。要部署支持高可用性的镜像 registry，需要两个或多个副本，ReadWriteMany 访问。

- 必须具有 100Gi 容量。

流程

1. 要将 registry 配置为使用存储，修改 configs.imageregistry/cluster 资源中的 spec.storage.pvc。



注意

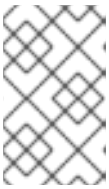
使用共享存储时，请查看您的安全设置以防止外部访问。

2. 验证您没有 registry pod：

```
$ oc get pod -n openshift-image-registry -l docker-registry=default
```

输出示例

```
No resources found in openshift-image-registry namespace
```



注意

如果您的输出中有一个 registry pod，则不需要继续这个过程。

3. 检查 **registry** 配置：

```
$ oc edit configs.imageregistry.operator.openshift.io
```

输出示例

```
storage:
  pvc:
    claim:
```

将 claim 字段留空以允许自动创建 image-registry-storage PVC。

4. 检查 **clusteroperator** 状态：

```
$ oc get clusteroperator image-registry
```

输出示例

```
NAME            VERSION      AVAILABLE   PROGRESSING   DEGRADED   SINCE   MESSAGE
image-registry   4.16         True        False         False      6h50m
```

5. **确保 registry** 设置为 **managed**，以启用镜像的构建和推送。

- 运行：

```
$ oc edit configs.imageregistry/cluster
```

然后，更改行

```
managementState: Removed
```

至

```
managementState: Managed
```

18.6.17.1.2. 在非生产集群中为镜像 registry 配置存储

您必须为 Image Registry Operator 配置存储。对于非生产集群，您可以将镜像 registry 设置为空目录。如果您这样做，重启 registry 时会丢失所有镜像。

流程

- 将镜像 registry 存储设置为空目录：

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch
'{"spec":{"storage":{"emptyDir":{}}}}'
```

警告

仅为非生产集群配置这个选项。

如果在 Image Registry Operator 初始化其组件前运行这个命令，oc patch 命令会失败并显示以下错误：

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster"
not found
```

等待几分钟，然后再次运行 命令。

18.6.18. 在用户置备的基础架构上完成安装

完成 Operator 配置后，可以在您提供的基础架构上完成集群安装。

先决条件

- 您的 control plane 已初始化。
- 已完成初始 Operator 配置。

流程

1. 使用以下命令确认所有集群组件都在线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION   AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.16.0    True        False         False      19m
baremetal                                  4.16.0    True        False         False      37m
cloud-credential                           4.16.0    True        False         False      40m
cluster-autoscaler                         4.16.0    True        False         False      37m
config-operator                            4.16.0    True        False         False      38m
console                                    4.16.0    True        False         False      26m
csi-snapshot-controller                    4.16.0    True        False         False      37m
dns                                        4.16.0    True        False         False      37m
etcd                                       4.16.0    True        False         False      36m
image-registry                             4.16.0    True        False         False      31m
ingress                                    4.16.0    True        False         False      30m
insights                                   4.16.0    True        False         False      31m
kube-apiserver                             4.16.0    True        False         False      26m
kube-controller-manager                    4.16.0    True        False         False      36m
kube-scheduler                             4.16.0    True        False         False      36m
kube-storage-version-migrator              4.16.0    True        False         False      37m
machine-api                                4.16.0    True        False         False      29m
machine-approver                           4.16.0    True        False         False      37m
machine-config                             4.16.0    True        False         False      36m
marketplace                                4.16.0    True        False         False      37m
monitoring                                 4.16.0    True        False         False      29m
network                                    4.16.0    True        False         False      38m
node-tuning                                4.16.0    True        False         False      37m
openshift-apiserver                        4.16.0    True        False         False      32m
```

```
openshift-controller-manager             4.16.0    True        False         False      30m
openshift-samples                        4.16.0    True        False         False      32m
operator-lifecycle-manager               4.16.0    True        False         False      37m
operator-lifecycle-manager-catalog       4.16.0    True        False         False      37m
operator-lifecycle-manager-packageserver 4.16.0    True        False         False      32m
service-ca                               4.16.0    True        False         False      38m
storage                                  4.16.0    True        False         False      37m
```

另外，当所有集群都可用时，以下命令会通知您。它还检索并显示凭证：

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

1

对于 <installation_directory>，请指定安装文件保存到的目录的路径。

输出示例

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

Cluster Version Operator 完成从 Kubernetes API 服务器部署 OpenShift Container Platform 集群时，该命令会成功。

重要

- 安装程序生成的 Ignition 配置文件包含 24 小时后过期的证书，然后在该时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 node-bootstrapper 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。

- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

2. 确认 Kubernetes API 服务器正在与 pod 通信。

a. 要查看所有 pod 的列表，请使用以下命令：

```
$ oc get pods --all-namespaces
```

输出示例

```
NAMESPACE                       NAME                                        READY   STATUS
RESTARTS   AGE
openshift-apiserver-operator    openshift-apiserver-operator-85cb746d55-zqhs8
1/1     Running   1          9m
openshift-apiserver             apiserver-67b9g                             1/1     Running   0
3m
openshift-apiserver             apiserver-ljcmx                             1/1     Running   0
1m
openshift-apiserver             apiserver-z25h4                             1/1     Running   0
2m
openshift-authentication-operator authentication-operator-69d5d8bf84-vh2n8
1/1     Running   0          5m
...
```

b. 使用以下命令，查看上一命令的输出中所列 pod 的日志：

```
$ oc logs <pod_name> -n <namespace> 1
```

1

指定 pod 名称和命名空间，如上一命令的输出中所示。

如果 pod 日志显示，Kubernetes API 服务器可以与集群机器通信。

3. 对于使用光纤通道协议(FCP)的安装，还需要额外的步骤才能启用多路径。不要在安装过程中启用多路径。

如需更多信息，请参阅 *安装后机器配置任务* 文档中的"使用 RHCOS 上使用内核参数启用多路径"。

验证

如果您在 OpenShift Container Platform bootstrap 过程中启用了安全引导，则需要以下验证步骤：

1. 运行以下命令来调试节点：

```
$ oc debug node/<node_name>
chroot /host
```

2. 运行以下命令确认启用了安全引导：

```
$ cat /sys/firmware/ipl/secure
```

输出示例

```
1 1
```

1

如果启用了安全引导，则值为 1，如果未启用安全引导，则值为 0。

3. 运行以下命令列出 re-IPL 配置：

```
# lsreipl
```

FCP 磁盘的输出示例

```
Re-IPL type: fcp
WWPN: 0x500507630400d1e3
LUN: 0x4001400e00000000
Device: 0.0.810e
bootprog: 0
br_lba: 0
Loadparm: ""
Bootparms: ""
clear: 0
```

DASD 磁盘输出示例

```
for DASD output:
Re-IPL type: ccw
Device: 0.0.525d
Loadparm: ""
clear: 0
```

4. 运行以下命令来关闭节点：

```
sudo shutdown -h
```

5. 从 LPAR 从硬件管理控制台 (HMC) 启动引导。请参阅 IBM 文档中的 从 LPAR 启动安全引导。

6. 当节点恢复时，再次检查安全引导状态。

18.6.19. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

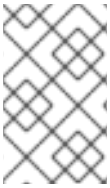
- 有关 Telemetry 服务的更多信息，请参阅关于 远程健康监控
- 如何在没有 SSH 的情况下在 OpenShift4 节点中生成 SOSREPORT。

18.6.20. 后续步骤

- 在 RHCOS 上启用带有内核参数的多路径。
- 自定义集群。
- 如果需要，您可以选择 不使用远程健康报告。

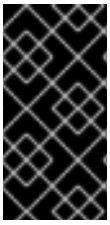
18.7. 在受限网络中的 IBM Z 和 IBM LINUXONE 上的 LPAR 上安装集群

在 OpenShift Container Platform 版本 4.16 中，您可以在受限网络中置备的 IBM Z® 或 IBM® LinuxONE 基础架构上安装集群。



注意

虽然本文档只涉及 IBM Z®，但它的所有信息也适用于 IBM® LinuxONE。

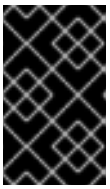


重要

非裸机平台还有其他注意事项。在安装 OpenShift Container Platform 集群前，请参阅有关在未经测试的平台上部署 OpenShift Container Platform 的指南 中的信息。

18.7.1. 先决条件

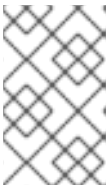
- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读有关 选择集群安装方法的文档，并为用户准备它。
- 您 创建了用于在受限网络中安装的镜像 registry，并获取您的 OpenShift Container Platform 版本的 imageContentSources 数据。
- 在开始安装过程前，您必须移动或删除任何现有的安装文件。这样可确保在安装过程中创建和更新所需的安装文件。



重要

确保从可访问安装介质的机器中执行安装步骤。

- 已为集群置备了 使用 OpenShift Data Foundation 或其他支持的存储协议的持久性存储。要部署私有镜像 registry，您必须使用 ReadWriteMany 访问设置持久性存储。
- 如果您使用防火墙并计划使用 Telemetry 服务，则将防火墙配置为允许集群需要访问的站点。



注意

如果要配置代理，请务必查看此站点列表。

18.7.2. 关于在受限网络中安装

在 OpenShift Container Platform 4.16 中，可以执行不需要有效的互联网连接来获取软件组件的安装。受限网络安装可以使用安装程序置备的基础架构或用户置备的基础架构完成，具体取决于您要安装集

群的云平台。

如果您选择在云平台中执行受限网络安装，您仍需要访问其云 API。有些云功能，比如 Amazon Web Service 的 Route 53 DNS 和 IAM 服务，需要访问互联网。根据您的网络，在裸机硬件、Nutanix 或 VMware vSphere 上安装可能需要较少的互联网访问。

要完成受限网络安装，您必须创建一个 registry，以镜像 OpenShift 镜像 registry 的内容并包含安装介质。您可以在镜像主机上创建此 registry，该主机可同时访问互联网和您的封闭网络，也可以使用满足您的限制条件的其他方法。

重要

由于用户置备安装配置的复杂性，在尝试使用用户置备的基础架构受限网络安装前，请考虑完成标准用户置备的基础架构安装。完成此测试安装后，您可以更轻松地隔离和排除在受限网络中安装过程中可能出现的任何问题。

18.7.2.1. 其他限制

受限网络中的集群有以下额外限制和限制：

- ClusterVersion 状态包含一个 Unable to retrieve available updates 错误。
- 默认情况下，您无法使用 Developer Catalog 的内容，因为您无法访问所需的镜像流标签。

18.7.3. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来获得用来安装集群的镜像。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。

- 获取执行集群更新所需的软件包。

18.7.4. 具有用户置备基础架构的集群的要求

对于包含用户置备的基础架构的集群，您必须部署所有所需的机器。

本节论述了在用户置备的基础架构上部署 OpenShift Container Platform 的要求。

18.7.4.1. 集群安装所需的机器

最小的 OpenShift Container Platform 集群需要以下主机：

表 18.95. 最低所需的主机

主机	描述
一个临时 bootstrap 机器	集群需要 bootstrap 机器在三台 control plane 机器上部署 OpenShift Container Platform 集群。您可在安装集群后删除 bootstrap 机器。
三台 control plane 机器	control plane 机器运行组成 control plane 的 Kubernetes 和 OpenShift Container Platform 服务。
至少两台计算机器，也称为 worker 机器。	OpenShift Container Platform 用户请求的工作负载在计算机器上运行。

重要

要保持集群的高可用性，请将独立的物理主机用于这些集群机器。

bootstrap 和 control plane 机器必须使用 Red Hat Enterprise Linux CoreOS(RHCOS)作为操作系统。但是，计算机器可以在 Red Hat Enterprise Linux CoreOS(RHCOS)、Red Hat Enterprise Linux(RHEL) 8.6 和更高的版本。

请注意，RHCOS 基于 Red Hat Enterprise Linux(RHEL) 9.2，并继承其所有硬件认证和要求。查看红帽企业 Linux 技术功能和限制。

18.7.4.2. 集群安装的最低资源要求

每台集群机器都必须满足以下最低要求：

表 18.96. 最低资源要求

机器	操作系统	vCPU [1]	虚拟内存	Storage	每秒输入/输出 (IOPS)
bootstrap	RHCOS	4	16 GB	100 GB	N/A
Control plane（控制平面）	RHCOS	4	16 GB	100 GB	N/A
Compute	RHCOS	2	8 GB	100 GB	N/A

1. 当启用 SMT-2 时，一个物理核心(IFL)提供两个逻辑核心（线程）。管理程序可以提供两个或多个 vCPU。

注意

从 OpenShift Container Platform 版本 4.13 开始，RHCOS 基于 RHEL 版本 9.2，它更新了微架构要求。以下列表包含每个架构需要的最小指令集架构 (ISA)：

- x86-64 体系结构需要 x86-64-v2 ISA
- ARM64 架构需要 ARMv8.0-A ISA
- IBM Power 架构需要 Power 9 ISA
- s390x 架构需要 z14 ISA

如需更多信息，请参阅 RHEL 架构。

如果平台的实例类型满足集群机器的最低要求，则 OpenShift Container Platform 支持使用它。

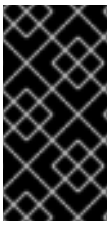
其他资源

- 优化存储

18.7.4.3. 最低 IBM Z 系统环境

您可以在以下 IBM® 硬件上安装 OpenShift Container Platform 版本 4.16：

- IBM® z16 （所有型号）、IBM® z15 （所有型号）、IBM® z14 （所有型号）
- IBM® LinuxONE 4 （所有型号）、IBM® LinuxONE （所有型号）、IBM® LinuxONE Emperor II、IBM® LinuxONE Rockhopper II



重要

在没有虚拟机监控程序的 IBM Z® 上运行 OpenShift Container Platform 时，请使用 Dynamic Partition Manager (DPM)来管理您的机器。

硬件要求

- Linux(IFL)等效的、启用了 SMT2 的 Linux 集成设施，每个群集都启用了 SMT2。
- 至少一个网络连接连接到 LoadBalancer 服务，并为集群外的流量提供数据。

注意

您可以使用专用或共享的 IFL 来分配足够的计算资源。资源共享是 IBM Z® 的关键优势之一。但是，您必须正确调整每个虚拟机监控程序层上的容量，并确保每个 OpenShift Container Platform 集群都有足够资源。

重要

由于集群的整体性能会受到影响，用于设置 OpenShift Container Platform 集群的 LPAR 必须提供足够的计算容量。就此而言，管理程序级别上的 LPAR 权重管理、授权和 CPU 共享扮演着重要角色。

操作系统要求

- 五个逻辑分区(LPAR)
  - 用于 OpenShift Container Platform control plane 机器的三个 LPAR
  - 用于 OpenShift Container Platform 计算机器的两个 LPAR
- 一台用于临时 OpenShift Container Platform bootstrap 机器的机器

IBM Z 网络连接要求

要在 LPAR 中安装 IBM Z®，您需要：

- 直接附加的 OSA 或 RoCE 网络适配器
- 对于首选设置，请使用 OSA 链接聚合。

磁盘存储

- 附加了 FICON 的磁盘存储(DASD)。这些可以是必须格式化为 CDL 的专用 DASD，这是默认设置。要达到 Red Hat Enterprise Linux CoreOS(RHCOS)安装所需的最小 DASD 大小，您需要扩展地址卷(EAV)。如果可用，请使用 HyperPAV 来确保最佳性能。
- FCP 连接的磁盘存储

存储/主内存

- OpenShift Container Platform control plane 机器需要 16 GB
- OpenShift Container Platform 计算机器需要 8 GB
- 临时 OpenShift Container Platform bootstrap 机器需要 16 GB

其他资源

- IBM® 文档中的 处理器资源/系统管理器规划指南，了解 PR/SM 模式注意事项。
- IBM® 文档中的用于 DPM 模式的 IBM Dynamic partition Manager (DPM)指南。
- LPAR 权重管理和权利的性能的主题。
- IBM Z® 和 IBM® LinuxONE 环境的推荐主机实践

18.7.4.4. 首选 IBM Z 系统环境

硬件要求

- 三个 LPARS，每个都相当于 6 个 IFL（每个集群启用了 SMT2）。
- 两个网络连接连接到 LoadBalancer 服务，并为集群外的流量提供数据。
- 作为设备直接附加到节点的 HiperSockets。要将 HiperSockets 直接连接到节点，您必须通过 RHEL 8 客户机设置到外部网络的网关来桥接到 HiperSockets 网络。

操作系统要求

- 用于 OpenShift Container Platform control plane 机器的三个 LPAR。
- 至少 6 个 LPAR 用于 OpenShift Container Platform 计算机器。
- 一个机器或 LPAR 用于临时 OpenShift Container Platform bootstrap 机器。

IBM Z 网络连接要求

要在 LPAR 中安装 IBM Z®，您需要：

- 直接附加的 OSA 或 RoCE 网络适配器

- 对于首选设置，请使用 OSA 链接聚合。

磁盘存储

- 附加了 FICON 的磁盘存储(DASD)。这些可以是必须格式化为 CDL 的专用 DASD，这是默认设置。要达到 Red Hat Enterprise Linux CoreOS(RHCOS)安装所需的最小 DASD 大小，您需要扩展地址卷(EAV)。如果可用，请使用 HyperPAV 来确保最佳性能。

- FCP 连接的磁盘存储

存储/主内存

- OpenShift Container Platform control plane 机器需要 16 GB

- OpenShift Container Platform 计算机器需要 8 GB

- 临时 OpenShift Container Platform bootstrap 机器需要 16 GB

18.7.4.5. 证书签名请求管理

在使用您置备的基础架构时，集群只能有限地访问自动机器管理，因此您必须提供一种在安装后批准集群证书签名请求 (CSR) 的机制。kube-controller-manager 只能批准 kubelet 客户端 CSR。machine-approver 无法保证使用 kubelet 凭证请求的提供证书的有效性，因为它不能确认是正确的机器发出了该请求。您必须决定并实施一种方法，以验证 kubelet 提供证书请求的有效性并进行批准。

18.7.4.6. 用户置备的基础架构对网络的要求

所有 Red Hat Enterprise Linux CoreOS(RHCOS)机器都需要在启动时在 initramfs 中配置联网，以获取它们的 Ignition 配置文件。

在初次启动过程中，机器需要 IP 地址配置，该配置通过 DHCP 服务器或静态设置，提供所需的引导选项。建立网络连接后，机器会从 HTTP 或 HTTPS 服务器下载 Ignition 配置文件。然后，Ignition 配置文件用于设置每台机器的确切状态。Machine Config Operator 在安装后完成对机器的更多更改，如应用新证书或密钥。

建议使用 DHCP 服务器对集群机器进行长期管理。确保 DHCP 服务器已配置为向集群机器提供持久的 IP 地址、DNS 服务器信息和主机名。

注意

如果用户置备的基础架构没有 DHCP 服务，您可以在 RHCOS 安装时向节点提供 IP 网络配置和 DNS 服务器地址。如果要从 ISO 镜像安装，这些参数可作为引导参数传递。如需有关静态 IP 置备和高级网络选项的更多信息，请参阅 *安装 RHCOS 并启动 OpenShift Container Platform bootstrap 过程* 部分。

Kubernetes API 服务器必须能够解析集群机器的节点名称。如果 API 服务器和 worker 节点位于不同的区域中，您可以配置默认 DNS 搜索区域，以允许 API 服务器解析节点名称。另一种支持的方法是始终通过节点对象和所有 DNS 请求中的完全限定域名引用主机。

18.7.4.6.1. 通过 DHCP 设置集群节点主机名

在 Red Hat Enterprise Linux CoreOS(RHCOS)机器上，主机名是通过 NetworkManager 设置的。默认情况下，机器通过 DHCP 获取其主机名。如果主机名不是由 DHCP 提供，请通过内核参数或者其它方法进行静态设置，请通过反向 DNS 查找获取。反向 DNS 查找在网络初始化后进行，可能需要一些时间来解决。其他系统服务可以在此之前启动，并将主机名检测为 localhost 或类似的内容。您可以使用 DHCP 为每个集群节点提供主机名来避免这种情况。

另外，通过 DHCP 设置主机名可以绕过实施 DNS split-horizon 的环境中的手动 DNS 记录名称配置错误。

18.7.4.6.2. 网络连接要求

您必须配置机器之间的网络连接，以允许 OpenShift Container Platform 集群组件进行通信。每台机器都必须能够解析集群中所有其他机器的主机名。

本节详细介绍了所需的端口。

表 18.97. 用于全机器到所有机器通信的端口

协议	port	描述
ICMP	N/A	网络可访问性测试
TCP	**1936**	指标
	9000-**9999**	主机级别的服务，包括端口 9**100**-**9101** 上的节点导出器，以及端口 **9099** 上的 Cluster Version Operator。

协议	port	描述
	10250-10259	Kubernetes 保留的默认端口
UDP	**4789**	VXLAN
	6081	Geneve
	9000-9999	主机级别的服务，包括端口 **9100**到**9101** 上的节点导出器。
	500	IPsec IKE 数据包
	4500	IPsec NAT-T 数据包
	123	UDP 端口 **123** 上的网络时间协议 (NTP) 如果配置了外部 NTP 时间服务器，需要打开 UDP 端口 **123**。
TCP/UDP	**30000-32767**	Kubernetes 节点端口
ESP	N/A	IPsec Encapsulating Security Payload(ESP)

表 18.98. 用于所有机器控制平面通信的端口

协议	port	描述
TCP	**6443**	Kubernetes API

表 18.99. control plane 机器用于 control plane 机器通信的端口

协议	port	描述
TCP	**2379-2380**	etcd 服务器和对等端口

用户置备的基础架构的 NTP 配置

OpenShift Container Platform 集群被配置为默认使用公共网络时间协议(NTP)服务器。如果要使用本地企业 NTP 服务器，或者集群部署在断开连接的网络中，您可以将集群配置为使用特定的时间服务器。如需更多信息，*请参阅配置 chrony 时间服务* 的文档。

其他资源

- 配置 chrony 时间服务

18.7.4.7. 用户置备的 DNS 要求

在 OpenShift Container Platform 部署中，以下组件需要 DNS 名称解析：

- The Kubernetes API
- OpenShift Container Platform 应用程序通配符
- bootstrap、control plane 和计算机器

Kubernetes API、bootstrap 机器、control plane 机器和计算机器也需要反向 DNS 解析。

DNS A/AAAA 或 CNAME 记录用于名称解析，PTR 记录用于反向名称解析。反向记录很重要，因为 Red Hat Enterprise Linux CoreOS(RHCOS)使用反向记录为所有节点设置主机名，除非 DHCP 提供主机名。另外，反向记录用于生成 OpenShift Container Platform 需要操作的证书签名请求(CSR)。

用户置备的 OpenShift Container Platform 集群需要以下 DNS 记录，这些记录必须在安装前就位。在每个记录中，<cluster_name> 是集群名称，<base_domain> 是您在 install-config.yaml 文件中指定的基域。完整的 DNS 记录采用以下形式：<component>.<cluster_name>.<base_domain>.。

表 18.100. 所需的 DNS 记录

组件	记录	描述
Kubernetes API	**api.<cluster_name>.<base_domain>.**	DNS A/AAAA 或 CNAME 记录，以及用于标识 API 负载均衡器的 DNS PTR 记录。这些记录必须由集群外的客户端和集群中的所有节点解析。
	api-int.<cluster_name>.<base_domain>.	DNS A/AAAA 或 CNAME 记录，以及用于内部标识 API 负载均衡器的 DNS PTR 记录。这些记录必须可以从集群中的所有节点解析。  **重要** API 服务器必须能够根据 Kubernetes 中记录的主机名解析 worker 节点。如果 API 服务器无法解析节点名称，则代理的 API 调用会失败，且您无法从 pod 检索日志。

组件	记录	描述
Routes	***.apps.<cluster_name>.<base_domain>.**	通配符 DNS A/AAAA 或 CNAME 记录，指向应用程序入口负载均衡器。应用程序入口负载均衡器以运行 Ingress Controller Pod 的机器为目标。默认情况下，Ingress Controller Pod 在计算机器上运行。这些记录必须由集群外的客户端和集群中的所有节点解析。 例如，console **-openshift-console.apps.<cluster_name>.<base_domain>** 用作到 OpenShift Container Platform 控制台的通配符路由。
bootstrap 机器	**bootstrap.<cluster_name>.<base_domain>.**	DNS A/AAAA 或 CNAME 记录，以及用于标识 bootstrap 机器的 DNS PTR 记录。这些记录必须由集群中的节点解析。
control plane 机器	**<control_plane><n>.<cluster_name>.<base_domain>.**	DNS A/AAAA 或 CNAME 记录，以识别 control plane 节点的每台机器。这些记录必须由集群中的节点解析。
计算机器	**<compute><n>.<cluster_name>.<base_domain>.**	DNS A/AAAA 或 CNAME 记录，用于识别 worker 节点的每台机器。这些记录必须由集群中的节点解析。

注意

在 OpenShift Container Platform 4.4 及更新的版本中，您不需要在 DNS 配置中指定 etcd 主机和 SRV 记录。

提示

您可以使用 dig 命令验证名称和反向名称解析。如需了解详细的 *验证步骤，请参阅为用户置备的基础架构验证 DNS 解析* 一节。

18.7.4.7.1. 用户置备的集群的 DNS 配置示例

本节提供 A 和 PTR 记录配置示例，它们满足了在用户置备的基础架构上部署 OpenShift Container Platform 的 DNS 要求。样本不是为选择一个 DNS 解决方案提供建议。

在这个示例中，集群名称为 ocp4，基域是 example.com。

用户置备的集群的 DNS A 记录配置示例

以下示例是 BIND 区域文件，其中显示了用户置备的集群中名称解析的 A 记录示例。

例 18.16. DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
  2019070700 ; serial
  3H  ; refresh (3 hours)
  30M  ; retry (30 minutes)
  2W  ; expiry (2 weeks)
  1W )  ; minimum (1 week)
 IN NS ns1.example.com.
 IN MX 10 smtp.example.com.
;
;
ns1.example.com.  IN A 192.168.1.5
smtp.example.com.  IN A 192.168.1.5
;
helper.example.com.  IN A 192.168.1.5
helper.ocp4.example.com. IN A 192.168.1.5
;
api.ocp4.example.com.  IN A 192.168.1.5 1
api-int.ocp4.example.com. IN A 192.168.1.5 2
;
*.apps.ocp4.example.com. IN A 192.168.1.5 3
;
bootstrap.ocp4.example.com. IN A 192.168.1.96 4
;
control-plane0.ocp4.example.com. IN A 192.168.1.97 5
control-plane1.ocp4.example.com. IN A 192.168.1.98 6
control-plane2.ocp4.example.com. IN A 192.168.1.99 7
;
compute0.ocp4.example.com. IN A 192.168.1.11 8
compute1.ocp4.example.com. IN A 192.168.1.7 9
;
;EOF
```

1 为 Kubernetes API 提供名称解析。记录引用 API 负载均衡器的 IP 地址。

2 为 Kubernetes API 提供名称解析。记录引用 API 负载均衡器的 IP 地址，用于内部集群通信。

3 为通配符路由提供名称解析。记录引用应用程序入口负载均衡器的 IP 地址。应用程序入口负载均衡器以运行 Ingress Controller Pod 的机器为目标。默认情况下，Ingress Controller Pod 在计算机器上运行。

注意

在这个示例中，将相同的负载均衡器用于 Kubernetes API 和应用入口流量。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。

4

为 bootstrap 机器提供名称解析。

5 6 7

为 control plane 机器提供名称解析。

8 9

为计算机器提供名称解析。

用户置备的集群的 DNS PTR 记录配置示例

以下示例 BIND 区域文件显示了用户置备的集群中反向名称解析的 PTR 记录示例。

例 18.17. 反向记录的 DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
  2019070700 ; serial
  3H  ; refresh (3 hours)
  30M  ; retry (30 minutes)
  2W  ; expiry (2 weeks)
  1W )  ; minimum (1 week)
 IN NS ns1.example.com.
;
5.1.168.192.in-addr.arpa. IN PTR api.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. IN PTR api-int.ocp4.example.com. 2
;
96.1.168.192.in-addr.arpa. IN PTR bootstrap.ocp4.example.com. 3
;
97.1.168.192.in-addr.arpa. IN PTR control-plane0.ocp4.example.com. 4
98.1.168.192.in-addr.arpa. IN PTR control-plane1.ocp4.example.com. 5
99.1.168.192.in-addr.arpa. IN PTR control-plane2.ocp4.example.com. 6
;
11.1.168.192.in-addr.arpa. IN PTR compute0.ocp4.example.com. 7
```

```
7.1.168.192.in-addr.arpa. IN PTR compute1.ocp4.example.com. 8
;
;EOF
```

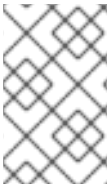
1 为 Kubernetes API 提供反向 DNS 解析。PTR 记录引用 API 负载均衡器的记录名称。

2 为 Kubernetes API 提供反向 DNS 解析。PTR 记录引用 API 负载均衡器的记录名称，用于内部集群通信。

3 为 bootstrap 机器提供反向 DNS 解析。

4 5 6 为 control plane 机器提供反向 DNS 解析。

7 8 为计算机器提供反向 DNS 解析。



注意

OpenShift Container Platform 应用程序通配符不需要 PTR 记录。

18.7.4.8. 用户置备的基础架构的负载均衡要求

在安装 OpenShift Container Platform 前，您必须置备 API 和应用程序入口负载均衡基础架构。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。

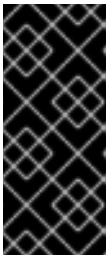
注意

如果要使用 Red Hat Enterprise Linux (RHEL) 实例部署 API 和应用程序入口负载均衡器，您必须单独购买 RHEL 订阅。

负载平衡基础架构必须满足以下要求：

1. **API** 负载均衡器 ：提供一个通用端点，供用户（包括人工和机器）与平台交互和配置。配置以下条件：

   - 仅第 4 层负载均衡.这可被称为 **Raw TCP** 或 **SSL Passthrough** 模式。
   - 无状态负载平衡算法。这些选项根据负载均衡器的实施而有所不同。



重要

不要为 **API** 负载均衡器配置会话持久性。为 **Kubernetes API** 服务器配置会话持久性可能会导致出现过量 **OpenShift Container Platform** 集群应用程序流量，以及过量的在集群中运行的 **Kubernetes API**。

在负载均衡器的前端和后端配置以下端口：

表 18.101. API 负载均衡器

port	后端机器（池成员）	internal	外部	描述
6443	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。您必须为 API 服务器健康检查探测配置 **/readyz** 端点。	X	X	Kubernetes API 服务器
22623	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。	X		机器配置服务器

注意

负载均衡器必须配置为，从 **API** 服务器关闭 **/readyz** 端点到从池中移除 **API** 服务器实例时最多需要 **30** 秒。在 **/readyz** 返回错误或健康后的时间范围内，端点必须被删除或添加。每 **5** 秒或 **10** 秒探测一次，有两个成功请求处于健康状态，三个成为不健康的请求是经过良好测试的值。

2. 应用程序入口负载均衡器 ：为应用程序流量从集群外部流提供入口点。OpenShift Container Platform 集群需要正确配置入口路由器。

配置以下条件：

- 仅第 4 层负载均衡.这可被称为 Raw TCP 或 SSL Passthrough 模式。
- 建议根据可用选项以及平台上托管的应用程序类型，使用基于连接的或基于会话的持久性。

提示

如果应用程序入口负载均衡器可以看到客户端的真实 IP 地址，启用基于 IP 的会话持久性可以提高使用端到端 TLS 加密的应用程序的性能。

在负载均衡器的前端和后端配置以下端口：

表 18.102. 应用程序入口负载均衡器

port	后端机器（池成员）	internal	外部	描述
443	默认情况下，运行 Ingress Controller Pod、计算或 worker 的机器。	X	X	HTTPS 流量
80	默认情况下，运行 Ingress Controller Pod、计算或 worker 的机器。	X	X	HTTP 流量

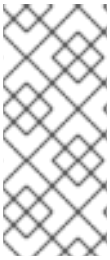
注意

如果要部署一个带有零计算节点的三节点集群，Ingress Controller Pod 在 control plane 节点上运行。在三节点集群部署中，您必须配置应用程序入口负载均衡器，将 HTTP 和 HTTPS 流量路由到 control plane 节点。

18.7.4.8.1. 用户置备的集群的负载均衡器配置示例

本节提供了一个满足用户置备集群的负载均衡要求的 API 和应用程序入口负载均衡器配置示例。示例是 HAProxy 负载均衡器的 /etc/haproxy/haproxy.cfg 配置。这个示例不是为选择一个负载平衡解决方案提供建议。

在这个示例中，将相同的负载均衡器用于 Kubernetes API 和应用入口流量。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。



注意

如果您使用 HAProxy 作为负载均衡器，并且 SELinux 设置为 enforcing，您必须通过运行 setsebool -P haproxy_connect_any=1 来确保 HAProxy 服务可以绑定到配置的 TCP 端口。

例 18.18. API 和应用程序入口负载均衡器配置示例

```
global
  log         127.0.0.1 local2
  pidfile     /var/run/haproxy.pid
  maxconn     4000
  daemon
defaults
  mode                    http
  log                     global
  option                  dontlognull
  option http-server-close
  option                  redispatch
  retries                 3
  timeout http-request    10s
  timeout queue           1m
  timeout connect         10s
  timeout client          1m
  timeout server          1m
  timeout http-keep-alive 10s
  timeout check           10s
  maxconn                 3000
listen api-server-6443 1
  bind *:6443
  mode tcp
  option  httpchk GET /readyz HTTP/1.0
  option  log-health-checks
  balance roundrobin
  server bootstrap bootstrap.ocp4.example.com:6443 verify none check check-ssl inter 10s
fall 2 rise 3 backup 2
  server master0 master0.ocp4.example.com:6443 weight 1 verify none check check-ssl
inter 10s fall 2 rise 3
  server master1 master1.ocp4.example.com:6443 weight 1 verify none check check-ssl
inter 10s fall 2 rise 3
  server master2 master2.ocp4.example.com:6443 weight 1 verify none check check-ssl
inter 10s fall 2 rise 3
listen machine-config-server-22623 3
  bind *:22623
  mode tcp
  server bootstrap bootstrap.ocp4.example.com:22623 check inter 1s backup 4
  server master0 master0.ocp4.example.com:22623 check inter 1s
```

```
  server master1 master1.ocp4.example.com:22623 check inter 1s
  server master2 master2.ocp4.example.com:22623 check inter 1s
listen ingress-router-443 5
  bind *:443
  mode tcp
  balance source
  server compute0 compute0.ocp4.example.com:443 check inter 1s
  server compute1 compute1.ocp4.example.com:443 check inter 1s
listen ingress-router-80 6
  bind *:80
  mode tcp
  balance source
  server compute0 compute0.ocp4.example.com:80 check inter 1s
  server compute1 compute1.ocp4.example.com:80 check inter 1s
```

1

端口 6443 处理 Kubernetes API 流量并指向 control plane 机器。

2 4

bootstrap 条目必须在 OpenShift Container Platform 集群安装前就位，且必须在 bootstrap 过程完成后删除它们。

3

端口 22623 处理机器配置服务器流量并指向 control plane 机器。

5

端口 443 处理 HTTPS 流量，并指向运行 Ingress Controller pod 的机器。默认情况下，Ingress Controller Pod 在计算机器上运行。

6

端口 80 处理 HTTP 流量，并指向运行 Ingress Controller pod 的机器。默认情况下，Ingress Controller Pod 在计算机器上运行。

注意

如果要部署一个带有零计算节点的三节点集群，Ingress Controller Pod 在 control plane 节点上运行。在三节点集群部署中，您必须配置应用程序入口负载均衡器，将 HTTP 和 HTTPS 流量路由到 control plane 节点。

提示

如果您使用 HAProxy 作为负载均衡器，您可以通过在 HAProxy 节点上运行 **netstat -nltupe** 来检查 **haproxy** 进程是否在侦听端口 6443、22623、443 和 80。

18.7.5. 准备用户置备的基础架构

在用户置备的基础架构上安装 OpenShift Container Platform 之前，您必须准备底层基础架构。

本节详细介绍了设置集群基础架构以准备 OpenShift Container Platform 安装所需的高级别步骤。这包括为集群节点配置 IP 网络和网络连接，为 Ignition 文件准备 Web 服务器，通过防火墙启用所需的端口，以及设置所需的 DNS 和负载均衡基础架构。

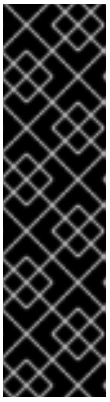
准备后，集群基础架构必须满足 *带有用户置备的基础架构部分的集群要求*。

先决条件

- 您已参阅 OpenShift Container Platform 4.x Tested Integrations 页面。
- 您已查看了 *具有用户置备基础架构的集群要求部分中详述的基础架构* 要求。

流程

1. 设置静态 IP 地址。
2. 设置 HTTP 或 HTTPS 服务器，为集群节点提供 Ignition 文件。
3. 确保您的网络基础架构提供集群组件之间所需的网络连接。有关 *要求的详情，请参阅用户置备的基础架构* 的网络要求部分。
4. 将防火墙配置为启用 OpenShift Container Platform 集群组件进行通信所需的端口。*如需有关所需端口的详细信息，请参阅用户置备的基础架构* 部分的网络要求。



重要

默认情况下，OpenShift Container Platform 集群可以访问端口 1936，因为每个 control plane 节点都需要访问此端口。

避免使用 Ingress 负载均衡器公开此端口，因为这样做可能会导致公开敏感信息，如统计信息和指标（与 Ingress Controller 相关的统计信息和指标）。

5. 为集群设置所需的 DNS 基础架构。

   a. 为 Kubernetes API、应用程序通配符、bootstrap 机器、control plane 机器和计算机器配置 DNS 名称解析。

   b. 为 Kubernetes API、bootstrap 机器、control plane 机器和计算机器配置反向 DNS 解析。

   如需有关 *OpenShift Container Platform DNS 要求的更多信息，请参阅用户置备* DNS 要求部分。

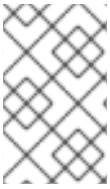
6. 验证您的 DNS 配置。

   a. 从安装节点，针对 Kubernetes API 的记录名称、通配符路由和集群节点运行 DNS 查找。验证响应中的 IP 地址是否与正确的组件对应。

   b. 从安装节点，针对负载均衡器和集群节点的 IP 地址运行反向 DNS 查找。验证响应中的记录名称是否与正确的组件对应。

   有关详细的 *DNS 验证步骤，请参阅用户置备的基础架构* 验证 DNS 解析部分。

7. 置备所需的 API 和应用程序入口负载平衡基础架构。有关 *要求的更多信息，请参阅用户置备的基础架构的负载平衡* 要求部分。

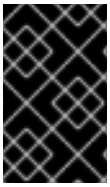


注意

某些负载平衡解决方案要求在初始化负载平衡之前，对群集节点进行 DNS 名称解析。

18.7.6. 验证用户置备的基础架构的 DNS 解析

您可以在在用户置备的基础架构上安装 OpenShift Container Platform 前验证 DNS 配置。



重要

本节中详述的验证步骤必须在安装集群前成功。

先决条件

- 已为您的用户置备的基础架构配置了所需的 DNS 记录。

流程

1. 从安装节点，针对 Kubernetes API 的记录名称、通配符路由和集群节点运行 DNS 查找。验证响应中包含的 IP 地址是否与正确的组件对应。

   a. 对 Kubernetes API 记录名称执行查询。检查结果是否指向 API 负载均衡器的 IP 地址：

   ```
   $ dig +noall +answer @<nameserver_ip> api.<cluster_name>.<base_domain> 1
   ```

   1 将 <nameserver_ip> 替换为 nameserver 的 IP 地址，<cluster_name> 替换为您的集群名称，<base_domain> 替换为您的基本域名。

   输出示例

   ```
   api.ocp4.example.com.  604800 IN A 192.168.1.5
   ```

b.

对 Kubernetes 内部 API 记录名称执行查询。检查结果是否指向 API 负载均衡器的 IP 地址：

```
$ dig +noall +answer @<nameserver_ip> api-int.<cluster_name>.<base_domain>
```

输出示例

```
api-int.ocp4.example.com.  604800 IN A 192.168.1.5
```

c.

测试 *.apps.<cluster_name>.<base_domain> DNS 通配符查找示例。所有应用程序通配符查询都必须解析为应用程序入口负载均衡器的 IP 地址：

```
$ dig +noall +answer @<nameserver_ip> random.apps.<cluster_name>.
<base_domain>
```

输出示例

```
random.apps.ocp4.example.com.  604800 IN A 192.168.1.5
```

注意

在示例中，将相同的负载均衡器用于 Kubernetes API 和应用程序入口流量。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。

您可以使用另一个通配符值替换 random。例如，您可以查询到 OpenShift Container Platform 控制台的路由：

```
$ dig +noall +answer @<nameserver_ip> console-openshift-console.apps.
<cluster_name>.<base_domain>
```

输出示例

```
console-openshift-console.apps.ocp4.example.com. 604800 IN A 192.168.1.5
```

d. 针对 bootstrap DNS 记录名称运行查询。检查结果是否指向 bootstrap 节点的 IP 地址：

```
$ dig +noall +answer @<nameserver_ip> bootstrap.<cluster_name>.
<base_domain>
```

输出示例

```
bootstrap.ocp4.example.com.  604800 IN A 192.168.1.96
```

e. 使用此方法对 control plane 和计算节点的 DNS 记录名称执行查找。检查结果是否与每个节点的 IP 地址对应。

2. 从安装节点，针对负载均衡器和集群节点的 IP 地址运行反向 DNS 查找。验证响应中包含的记录名称是否与正确的组件对应。

a. 对 API 负载均衡器的 IP 地址执行反向查找。检查响应是否包含 Kubernetes API 和 Kubernetes 内部 API 的记录名称：

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.5
```

输出示例

```
5.1.168.192.in-addr.arpa. 604800 IN PTR api-int.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. 604800 IN PTR api.ocp4.example.com. 2
```

1

为 Kubernetes 内部 API 提供记录名称。

2

为 Kubernetes API 提供记录名称。

注意

OpenShift Container Platform 应用程序通配符不需要 PTR 记录。针对应用程序入口负载均衡器的 IP 地址解析反向 DNS 解析不需要验证步骤。

b.

对 bootstrap 节点的 IP 地址执行反向查找。检查结果是否指向 bootstrap 节点的 DNS 记录名称：

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.96
```

输出示例

```
96.1.168.192.in-addr.arpa. 604800 IN PTR bootstrap.ocp4.example.com.
```

c.

使用此方法对 control plane 和计算节点的 IP 地址执行反向查找。检查结果是否与每个节点的 DNS 记录名称对应。

18.7.7. 为集群节点 SSH 访问生成密钥对

在 OpenShift Container Platform 安装过程中，您可以为安装程序提供 SSH 公钥。密钥通过它们的 Ignition 配置文件传递给 Red Hat Enterprise Linux CoreOS(RHCOS)节点，用于验证对节点的 SSH 访问。密钥添加到每个节点上 core 用户的 ~/.ssh/authorized_keys 列表中，这将启用免密码身份验证。

将密钥传递给节点后，您可以使用密钥对作为用户 核心 通过 SSH 连接到 RHCOS 节点。若要通过

SSH 访问节点，必须由 SSH 为您的本地用户管理私钥身份。

如果要通过 SSH 连接到集群节点来执行安装调试或灾难恢复，则必须在安装过程中提供 SSH 公钥。./openshift-install gather 命令还需要在集群节点上设置 SSH 公钥。

重要

不要在生产环境中跳过这个过程，在生产环境中需要灾难恢复和调试。

流程

1. 如果您在本地计算机上没有可用于在集群节点上进行身份验证的现有 SSH 密钥对，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 ~/.ssh/id_ed25519。如果您已有密钥对，请确保您的公钥位于 ~/.ssh 目录中。

注意

如果您计划在 x86_64、ppc64le 和 s390x 架构上安装使用 RHEL 加密库（这些加密库已提交给 NIST 用于 FIPS 140-2/140-3 验证）的 OpenShift Container Platform 集群，则不要创建使用 ed25519 算法的密钥。相反，创建一个使用 rsa 或 ecdsa 算法的密钥。

2. 查看公共 SSH 密钥：

```
$ cat <path>/<file_name>.pub
```

例如，运行以下命令来查看 ~/.ssh/id_ed25519.pub 公钥：

```
$ cat ~/.ssh/id_ed25519.pub
```

3. 将 SSH 私钥身份添加到本地用户的 SSH 代理（如果尚未添加）。在集群节点上，或者要使

用 ./openshift-install gather 命令，需要对该密钥进行 SSH 代理管理，才能在集群节点上进行免密码 SSH 身份验证。

注意

在某些发行版中，自动管理默认 SSH 私钥身份，如 ~/.ssh/id_rsa 和 ~/.ssh/id_dsa。

a. 如果 ssh-agent 进程尚未为您的本地用户运行，请将其作为后台任务启动：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果集群处于 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

4. 将 SSH 私钥添加到 ssh-agent ：

```
$ ssh-add <path>/<file_name> (1)
```

(1) 指定 SSH 私钥的路径和文件名，如 ~/.ssh/id_ed25519.pub

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

后续步骤

- 安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

18.7.8. 手动创建安装配置文件

安装集群要求您手动创建安装配置文件。

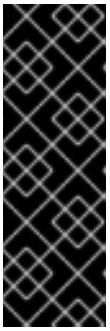
先决条件

- 您在本地机器上有一个 SSH 公钥来提供给安装程序。该密钥将用于在集群节点上进行 SSH 身份验证，以进行调试和灾难恢复。
- 已获取 OpenShift Container Platform 安装程序和集群的 pull secret。

流程

1. 创建一个安装目录来存储所需的安装资产：

```
$ mkdir <installation_directory>
```



重要

您必须创建一个目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

2. 自定义提供的 install-config.yaml 文件模板示例，并将其保存在 <installation_directory> 中。

注意

此配置文件必须命名为 **install-config.yaml**。

3. 备份 **install-config.yaml** 文件，以便您可以使用它安装多个集群。

重要

install-config.yaml 文件会在安装过程的下一步中使用。现在必须备份它。

其他资源

- IBM Z® 的安装配置参数

18.7.8.1. IBM Z 的 install-config.yaml 文件示例

您可以自定义 **install-config.yaml** 文件，以指定有关 OpenShift Container Platform 集群平台的更多详情，或修改所需参数的值。

```
apiVersion: v1
baseDomain: example.com (1)
compute: (2)
- hyperthreading: Enabled (3)
  name: worker
  replicas: 0 (4)
  architecture: s390x
controlPlane: (5)
  hyperthreading: Enabled (6)
  name: master
  replicas: 3 (7)
  architecture: s390x
metadata:
  name: test (8)
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14 (9)
    hostPrefix: 23 (10)
  networkType: OVNKubernetes (11)
  serviceNetwork: (12)
  - 172.30.0.0/16
platform:
  none: {} (13)
fips: false (14)
```

```
pullSecret: '{"auths":{"<local_registry>": {"auth": "<credentials>","email":
"you@example.com"}}}' 15
sshKey: 'ssh-ed25519 AAAA...' 16
additionalTrustBundle: | 17
  -----BEGIN CERTIFICATE-----
  ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
  -----END CERTIFICATE-----
imageContentSources: 18
- mirrors:
  - <local_repository>/ocp4/openshift4
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <local_repository>/ocp4/openshift4
  source: quay.io/openshift-release-dev/ocp-v4.0-art-dev
```

1

集群的基域。所有 DNS 记录都必须是这个基域的子域，并包含集群名称。

2 5

controlPlane 部分是一个单个映射，但 compute 部分是一系列映射。为满足不同数据结构的要求，compute 部分的第一行必须以连字符 - 开头，controlPlane 部分 的第一行则不以连字符开头。仅使用一个 control plane 池。

3 6

指定要启用或禁用并发多线程(SMT)还是超线程。默认情况下，启用 SMT 可提高机器中内核的性能。您可以通过将 参数值设置为 Disabled 来禁用它。如果禁用 SMT，则必须在所有集群机器中禁用它；这包括 control plane 和计算机器。

注意

默认启用并发多线程(SMT)。如果 OpenShift Container Platform 节点上没有 SMT，超线程 参数无效。

重要

如果您禁用 超线程，无论是在 OpenShift Container Platform 节点上，还是在 install-config.yaml 文件中，请确保您的容量规划考虑机器性能显著降低的情况。

4

在用户置备的基础架构上安装 OpenShift Container Platform 时，必须将这个值设置为 0。在安装程序置备的安装中，参数控制集群为您创建和管理的计算机器数量。在用户置备的安装中，您必须在完成集群安装前手动部署计算机器。

注意

如果要安装一个三节点集群，在安装 Red Hat Enterprise Linux CoreOS(RHCOS)机器时不要部署任何计算机器。

7

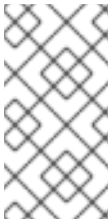
您添加到集群的 control plane 机器数量。由于集群使用这些值作为集群中的 etcd 端点数量，所以该值必须与您部署的 control plane 机器数量匹配。

8

您在 DNS 记录中指定的集群名称。

9

从中分配 Pod IP 地址的 IP 地址块。此块不得与现有物理网络重叠。这些 IP 地址用于 pod 网络。如果需要从外部网络访问 pod，您必须配置负载均衡器和路由器来管理流量。



注意

类 E CIDR 范围被保留以供以后使用。要使用 Class E CIDR 范围，您必须确保您的网络环境接受 Class E CIDR 范围内的 IP 地址。

10

分配给每个节点的子网前缀长度。例如，如果 hostPrefix 设为 23，则每个节点从 given cidr 中分配 a /23 子网，这样就能有 510（2^(32 - 23)- 2）个 pod IP 地址。如果需要从外部网络访问节点，请配置负载均衡器和路由器来管理流量。

11

要安装的集群网络插件。默认值 OVNKubernetes 是唯一支持的值。

12

用于服务 IP 地址的 IP 地址池。您只能输入一个 IP 地址池。此块不得与现有物理网络重叠。如果您需要从外部网络访问服务，请配置负载均衡器和路由器来管理流量。

13

您必须将平台设置为 none。您无法为 IBM Z® 基础架构提供额外的平台配置变量。

重要

使用平台类型 none 安装的集群无法使用一些功能，如使用 Machine API 管理计算机器。即使附加到集群的计算机器安装在通常支持该功能的平台上，也会应用这个限制。在安装后无法更改此参数。

14

是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 Red Hat Enterprise Linux (RHEL) 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅在 FIPS 模式中安装该系统。

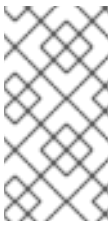
当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS)时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。

15

对于 <local_registry>，请指定 registry 域名，以及您的镜像 registry 用来提供内容的可选端口。例如： registry.example.com 或 registry.example.com:5000。对于 <credentials>，请为您的镜像 registry 指定 base64 编码的用户名和密码。

16

Red Hat Enterprise Linux CoreOS(RHCOS)中 core 用户的 SSH 公钥。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 ssh-agent 进程使用的 SSH 密钥。

17

添加 additionalTrustBundle 参数和值。该值必须是您用于镜像 registry 的证书文件内容。证书文件可以是现有的可信证书颁发机构，也可以是您为镜像 registry 生成的自签名证书。

18

根据您用来镜像存储库的命令输出提供 imageContentSources 部分。

重要

- 使用 oc adm release mirror 命令时，请使用 imageContentSources 部分中的输出。
- 使用 oc mirror 命令时，请使用运行该命令结果的 ImageContentSourcePolicy 文件的 repositoryDigestMirrors 部分。
- ImageContentSourcePolicy 已被弃用。如需更多信息，请参阅*配置镜像 registry 存储库镜像*。

18.7.8.2. 在安装过程中配置集群范围的代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 install-config.yaml 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 install-config.yaml 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 Proxy 对象的 spec.noProxy 字段中添加站点来绕过代理。

注意

Proxy 对象 status.noProxy 字段使用安装配置中的 networking.machineNetwork[].cidr、networking.clusterNetwork[].cidr 和 networking.serviceNetwork[] 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，Proxy 对象 status.noProxy 字段也会使用实例元数据端点填充(169.254.169.254)。

流程

1. 编辑 install-config.yaml 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 http。

2. 用于创建集群外 HTTPS 连接的代理 URL。

3. 要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 . 以仅匹配子域。例如，.y.com 匹配 x.y.com，但不匹配 y.com。使用 * 绕过所有目的地的代理。

4. 如果提供，安装程序会在 openshift-config 命名空间中生成名为 user-ca-bundle 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 trusted-ca-bundle 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， Proxy 对象的 trustedCA 字段中

也会引用此配置映射。additionalTrustBundle 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

5 可选：决定 Proxy 对象的配置以引用 trustedCA 字段中 user-ca-bundle 配置映射的策略。允许的值是 Proxyonly 和 Always。仅在配置了 http/https 代理时，使用 Proxyonly 引用 user-ca-bundle 配置映射。使用 Always 始终引用 user-ca-bundle 配置映射。默认值为 Proxyonly。

注意

安装程序不支持代理的 readinessEndpoints 字段。

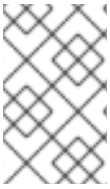
注意

如果安装程序超时，重启并使用安装程序的 wait-for 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. 保存该文件并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 cluster 的集群范围代理，该代理 使用 提供的 install-config.yaml 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 cluster Proxy 对象，但它会有一个空 spec。



注意

只支持名为 cluster 的 Proxy 对象，且无法创建额外的代理。

18.7.8.3. 配置三节点集群

另外，您可以在只由三台 control plane 机器组成的裸机集群中部署零台计算机器。这为集群管理员和开发人员提供了更小、效率更高的集群，用于测试、开发和生产。

在三节点 OpenShift Container Platform 环境中，三台 control plane 机器可以调度，这意味着应用程序工作负载被调度到它们上运行。

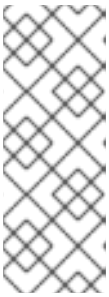
先决条件

- 您有一个现有的 **install-config.yaml** 文件。

流程

- 确保 **install-config.yaml** 文件中的计算副本数量设置为 **0**，如以下 **计算** 小节所示：

```
compute:
- name: worker
  platform: {}
  replicas: 0
```



注意

在用户置备的基础架构上安装 **OpenShift Container Platform** 时，无论您要部署的计算机器数量有多少，您必须将计算机器的 **replicas** 参数值设置为 **0**。在安装程序置备的安装中，参数控制集群为您创建和管理的计算机器数量。这不适用于手动部署计算机器的用户置备安装。

对于三节点集群安装，请按照以下步骤执行：

- 如果要部署一个带有零计算节点的三节点集群，**Ingress Controller** Pod 在 **control plane** 节点上运行。在三节点集群部署中，您必须配置应用程序入口负载均衡器，将 **HTTP** 和 **HTTPS** 流量路由到 **control plane** 节点。如需更多信息，*请参阅用户置备的基础架构的负载平衡要求* 部分。

- 在以下步骤中创建 **Kubernetes** 清单文件时，请确保 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** 文件中的 **mastersSchedulable** 参数被设置为 **true**。这可让应用程序工作负载在 **control plane** 节点上运行。

- 在创建 **Red Hat Enterprise Linux CoreOS(RHCOS)**机器时，不要部署任何计算节点。

18.7.9. Cluster Network Operator 配置

集群网络的配置作为 **Cluster Network Operator(CNO)**配置的一部分指定，并存储在名为 **cluster** 的自定义资源**(CR)**对象中。**CR** 指定 **operator.openshift.io API** 组中的 **Network API** 的字段。

CNO 配置在集群安装过程中从 Network.config.openshift.io API 组中的 Network API 继承以下字段：

clusterNetwork

从中分配 Pod IP 地址的 IP 地址池。

serviceNetwork

服务的 IP 地址池.

defaultNetwork.type

集群网络插件。OVNKubernetes 是安装期间唯一支持的插件。

您可以通过在名为 cluster 的 CNO 对象中设置 defaultNetwork 对象的字段来为集群指定集群网络插件配置。

18.7.9.1. Cluster Network Operator 配置对象

下表中描述了 Cluster Network Operator(CNO)的字段：

表 18.103. Cluster Network Operator 配置对象

字段	类型	描述
metadata.name	**字符串**	CNO 对象的名称。这个名称始终是 **集群**。
spec.clusterNetwork	**array**	用于指定从哪些 IP 地址块分配 Pod IP 地址以及集群中每个节点的子网前缀长度的列表。例如： spec: clusterNetwork: - cidr: 10.128.0.0/19 hostPrefix: 23 - cidr: 10.128.32.0/19 hostPrefix: 23

字段	类型	描述
spec.serviceNetwork	**array**	服务的 IP 地址块。OpenShift SDN 和 OVN-Kubernetes 网络插件只支持服务网络的一个 IP 地址块。例如： spec: serviceNetwork: - 172.30.0.0/14 您只能在创建清单前在 **install-config.yaml** 文件中自定义此字段。该值在清单文件中是只读的。
spec.defaultNetwork	**object**	为集群网络配置网络插件。
spec.kubeProxyConfig	**object**	此对象的字段指定 kube-proxy 配置。如果使用 OVN-Kubernetes 集群网络供应商，则 kube-proxy 配置不会起作用。

defaultNetwork 对象配置

下表列出了 defaultNetwork 对象的值：

表 18.104. defaultNetwork 对象

字段	类型	描述
type	**字符串**	**OVNKubernetes**。Red Hat OpenShift Networking 网络插件在安装过程中被选择。此值在集群安装后无法更改。 **注意** OpenShift Container Platform 默认使用 OVN-Kubernetes 网络插件。OpenShift SDN 不再作为新集群的安装选择提供。
ovnKubernetesConfig	**object**	此对象仅对 OVN-Kubernetes 网络插件有效。

配置 OVN-Kubernetes 网络插件

下表描述了 OVN-Kubernetes 网络插件的配置字段：

表 18.105. ovnKubernetesConfig object

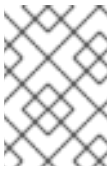
字段	类型	描述
mtu	**integer**	Geneve（通用网络虚拟化封装）覆盖网络的最大传输单元 (MTU)。这根据主网络接口的 MTU 自动探测。您通常不需要覆盖检测到的 MTU。 如果自动探测的值不是您期望的值，请确认节点上主网络接口上的 MTU 是否正确。您不能使用这个选项更改节点上主网络接口的 MTU 值。 如果集群中不同节点需要不同的 MTU 值，则必须将此值设置为**比** 集群中的最低 MTU 值小 100。例如，如果集群中的某些节点的 MTU 为 **9001**，而某些节点的 MTU 为 **1500**，则必须将此值设置为 **1400**。
genevePort	**integer**	用于所有 Geneve 数据包的端口。默认值为 **6081**。此值在集群安装后无法更改。
ipsecConfig	**object**	指定用于自定义 IPsec 配置的配置对象。
ipv4	**object**	为 IPv4 设置指定配置对象。
ipv6	**object**	为 IPv6 设置指定配置对象。
policyAuditConfig	**object**	指定用于自定义网络策略审计日志的配置对象。如果未设置，则使用默认的审计日志设置。
gatewayConfig	**object**	可选：指定一个配置对象来自定义如何将出口流量发送到节点网关。  **注意** 在迁移出口流量时，工作负载和服务流量会受到一定影响，直到 Cluster Network Operator (CNO) 成功推出更改。

表 18.106. ovnKubernetesConfig.ipv4 object

字段	类型	描述
internalTransitSwitchSubnet	字符串	如果您的现有网络基础架构与 **100.88.0.0/16** IPv4 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。启用东西流量的分布式传输交换机的子网。此子网不能与 OVN-Kubernetes 或主机本身使用的任何其他子网重叠。必须足够大，以适应集群中的每个节点一个 IP 地址。 默认值为 **100.88.0.0/16**。

字段	类型	描述
internalJoinSubnet	字符串	如果您的现有网络基础架构与 **100.64.0.0/16** IPv4 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。您必须确保 IP 地址范围没有与 OpenShift Container Platform 安装使用的任何其他子网重叠。IP 地址范围必须大于可添加到集群的最大节点数。例如，如果 **clusterNetwork.cidr** 值为 **10.128.0.0/14**，并且 **clusterNetwork.hostPrefix** 值为 **/23**，则最大节点数量为 **2^(23-14)=512**。 默认值为 **100.64.0.0/16**。

表 18.107. ovnKubernetesConfig.ipv6 object

字段	类型	描述
internalTransitSwitchSubnet	字符串	如果您的现有网络基础架构与 **fd97::/64** IPv6 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。启用东西流量的分布式传输交换机的子网。此子网不能与 OVN-Kubernetes 或主机本身使用的任何其他子网重叠。必须足够大，以适应集群中的每个节点一个 IP 地址。 默认值为 **fd97::/64**。
internalJoinSubnet	字符串	如果您的现有网络基础架构与 **fd98::/64** IPv6 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。您必须确保 IP 地址范围没有与 OpenShift Container Platform 安装使用的任何其他子网重叠。IP 地址范围必须大于可添加到集群的最大节点数。 默认值为 **fd98::/64**。

表 18.108. policyAuditConfig object

字段	类型	描述
rateLimit	整数	每个节点每秒生成一次的消息数量上限。默认值为每秒 **20** 条消息。
maxFileSize	整数	审计日志的最大大小，以字节为单位。默认值为 **50000000** 或 50 MB。
maxLogFiles	整数	保留的日志文件的最大数量。

字段	类型	描述
目的地	字符串	以下附加审计日志目标之一： **libc** 主机上的 journald 进程的 libc **syslog（）** 函数。 **UDP:<host>:<port>** 一个 syslog 服务器。将 **<host>:<port>** 替换为 **syslog 服务器的主机** 和端口。 **Unix:<file>** 由 **<file>** 指定的 Unix 域套接字文件。 **null** 不要将审计日志发送到任何其他目标。
syslogFacility	字符串	syslog 工具，如 as **kern**，如 RFC5424 定义。默认值为 **local0。**

表 18.109. gatewayConfig object

字段	类型	描述
routingViaHost	**布尔值**	将此字段设置为 **true**，将来自 pod 的出口流量发送到主机网络堆栈。对于依赖于在内核路由表中手动配置路由的高级别安装和应用程序，您可能需要将出口流量路由到主机网络堆栈。默认情况下，出口流量在 OVN 中进行处理以退出集群，不受内核路由表中的特殊路由的影响。默认值为 **false**。 此字段与 Open vSwitch 硬件卸载功能有交互。如果将此字段设置为 **true**，则不会获得卸载的性能优势，因为主机网络堆栈会处理出口流量。
ipForwarding	**object**	您可以使用 **Network** 资源中的 **ipForwarding** 规格来控制 OVN-Kubernetes 管理接口上所有流量的 IP 转发。指定 **Restricted** 只允许 Kubernetes 相关流量的 IP 转发。指定 **Global** 以允许转发所有 IP 流量。对于新安装，默认值为 **Restricted**。对于到 OpenShift Container Platform 4.14 或更高版本的更新，默认值为 **Global**。
ipv4	**object**	可选：指定一个对象来为主机配置内部 OVN-Kubernetes 伪装地址，以服务 IPv4 地址的流量。
ipv6	**object**	可选：指定一个对象来为主机配置内部 OVN-Kubernetes 伪装地址，以服务 IPv6 地址的流量。

表 18.110. gatewayConfig.ipv4 对象

字段	类型	描述
internalMasqueradeSubnet	**字符串**	内部使用的伪装 IPv4 地址，以启用主机服务流量。主机配置了这些 IP 地址和共享网关网桥接口。默认值为 **169.254.169.0/29**。

表 18.111. gatewayConfig.ipv6 对象

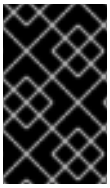
字段	类型	描述
internalMasqueradeSubnet	**字符串**	内部使用的伪装 IPv6 地址，以启用主机服务流量。主机配置了这些 IP 地址和共享网关网桥接口。默认值为 **fd69::/125**。

表 18.112. ipsecConfig 对象

字段	类型	描述
模式	**字符串**	指定 IPsec 实现的行为。必须是以下值之一： • **Disabled**: 在集群节点上不启用 IPsec。 • **External** ：对于带有外部主机的网络流量，启用 IPsec。 • **Full**: IPsec 为带有外部主机的 pod 流量和网络流量启用 IPsec。

启用 IPSec 的 OVN-Kubernetes 配置示例

```
defaultNetwork:
  type: OVNKubernetes
  ovnKubernetesConfig:
    mtu: 1400
    genevePort: 6081
      ipsecConfig:
        mode: Full
```



重要

使用 OVNKubernetes 可能会导致 IBM Power® 上的堆栈耗尽问题。

kubeProxyConfig 对象配置（仅限 OpenShiftSDN 容器网络接口）

kubeProxyConfig 对象的值在下表中定义：

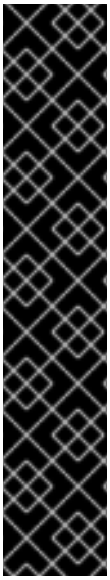
表 18.113. kubeProxyConfig object

字段	类型	描述
iptablesSyncPeriod	**字符串**	**iptables** 规则的刷新周期。默认值为 **30s**。有效的后缀包括 **s**、**m** 和 **h**，具体参见 Go 时间 包 文档。 **注意** 由于 OpenShift Container Platform 4.3 及更高版本中引进了性能改进，不再需要调整 **iptablesSyncPeriod** 参数。
proxyArguments.iptables-min-sync-period	**array**	刷新 **iptables** 规则前的最短持续时间。此字段确保刷新的频率不会过于频繁。有效的后缀包括 **s**、**m** 和 **h**，具体参见 Go **time** 软件包。默认值为： kubeProxyConfig: proxyArguments: iptables-min-sync-period: - 0s

18.7.10. 创建 Kubernetes 清单和 Ignition 配置文件

由于您必须修改一些集群定义文件并手动启动集群机器，因此您必须生成 Kubernetes 清单和 Ignition 配置文件来配置机器。

安装配置文件转换为 Kubernetes 清单。清单嵌套到 Ignition 配置文件中，稍后用于配置集群机器。



重要

- OpenShift Container Platform 安装程序生成的 Ignition 配置文件包含 24 小时后过期的证书，然后在该时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 node-bootstrapper 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。

- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

注意

生成清单和 Ignition 文件的安装程序是特定的架构，可以从 客户端镜像镜像获取。安装程序的 Linux 版本仅在 s390x 上运行。此安装程序也可用作 Mac OS 版本。

先决条件

- 已获得 OpenShift Container Platform 安装程序。对于受限网络安装，这些文件位于您的镜像主机上。

- 已创建 install-config.yaml 安装配置文件。

流程

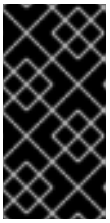
1. 进入包含 OpenShift Container Platform 安装程序的目录，并为集群生成 Kubernetes 清单：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 对于 <installation_directory>，请指定包含您创建的 install-config.yaml 文件的安装目录。

警告

如果您要安装一个三节点集群，请跳过以下步骤，以便可以调度 control plane 节点。



重要

当您将 control plane 节点从默认的不可调度配置为可以调度时，需要额外的订阅。这是因为 control plane 节点变为计算节点。

2. 检查 <installation_directory>/manifests/cluster-scheduler-02-config.yml Kubernetes 清单文件中的 mastersSchedulable 参数是否已设置为 false。此设置可防止在 control plane 机器上调度 pod：

   a. 打开 <installation_directory>/manifests/cluster-scheduler-02-config.yml 文件。

   b. 找到 mastersSchedulable 参数，并确保它被设置为 false。

   c. 保存并退出 文件。

3. 要创建 Ignition 配置文件，请从包含安装程序的目录运行以下命令：

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

1 对于 <installation_directory>，请指定相同的安装目录。

为安装目录中的 bootstrap、control plane 和计算节点创建 Ignition 配置文件。kubeadmin-password 和 kubeconfig 文件在 ./<installation_directory>/auth 目录中创建：

```
.
```

```
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

18.7.11. 在 IBM Z 或 IBM LinuxONE 环境中使用静态 IP 配置 NBDE

在 IBM Z® 或 IBM® LinuxONE 环境中启用 NBDE 磁盘加密需要额外的步骤，本节中详细介绍。

先决条件

- 您已设置了外部 Tang 服务器。具体步骤请查看 网络绑定磁盘加密。
- 您已安装了 with ane 实用程序。
- 您已查看如何使用 Butane 创建机器配置的说明。

流程

1. 为 control plane 和计算节点创建 Butane 配置文件。

   以下 control plane 节点的 Butane 配置示例为磁盘加密创建一个名为 master-storage.bu 的文件：

```
variant: openshift
version: 4.16.0
metadata:
  name: master-storage
  labels:
    machineconfiguration.openshift.io/role: master
storage:
  luks:
    - clevis:
        tang:
          - thumbprint: QcPr_NHFJammnRCA3fFMVdNBwjs
            url: http://clevis.example.com:7500
        options: 1
          - --cipher
          - aes-cbc-essiv:sha256
      device: /dev/disk/by-partlabel/root 2
```

```
    label: luks-root
    name: root
    wipe_volume: true
  filesystems:
   - device: /dev/mapper/root
    format: xfs
    label: root
    wipe_filesystem: true
openshift:
  fips: true 3
```

1

只有在启用了 FIPS 模式时才需要 cipher 选项。如果禁用了 FIPS，则省略该条目。

2

对于在 DASD 类型磁盘中安装，使用 device: /dev/disk/by-label/root 替换。

3

是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

2. 运行以下命令，创建自定义 initramfs 文件来引导机器：

```
$ coreos-installer pxe customize \
  /root/rhcos-bootfiles/rhcos-<release>-live-initramfs.s390x.img \
  --dest-device /dev/disk/by-id/scsi-<serial_number> --dest-karg-append \
  ip=<ip_address>::<gateway_ip>:<subnet_mask>::<network_device>:none \
  --dest-karg-append nameserver=<nameserver_ip> \
  --dest-karg-append rd.neednet=1 -o \
  /root/rhcos-bootfiles/<node_name>-initramfs.s390x.img
```

注意

首次引导前，您必须为集群中的每个节点自定义 initramfs，并添加 PXE 内核参数。

3. 创建包含 ignition.platform.id=metal 和 ignition.firstboot 的参数文件。

control plane 机器的内核参数文件示例：

```
rd.neednet=1 \
console=ttysclp0 \
coreos.inst.install_dev=/dev/dasda \(1)
ignition.firstboot ignition.platform.id=metal \
coreos.live.rootfs_url=http://<http_server>/rhcos-<version>-live-rootfs.
<architecture>.img \(2)
coreos.inst.ignition_url=http://<http_server>/master.ign \(3)
ip=10.19.17.2::10.19.17.1:255.255.255.0::enbdd0:none nameserver=10.19.17.1 \
zfcp.allow_lun_scan=0 \(4)
rd.znet=qeth,0.0.bdd0,0.0.bdd1,0.0.bdd2,layer2=1 \
rd.zfcp=0.0.5677,0x600606680g7f0056,0x034F000000000000 \(5)
```

(1) 对于在 DASD 类型磁盘中安装，请添加 coreos.inst.install_dev=/dev/dasda。为 FCP 类型磁盘省略这个值。

(2) 指定您要引导的 kernel 和 initramfs 的 rootfs 工件位置。仅支持 HTTP 和 HTTPS 协议。

(3) 指定 Ignition 配置文件的位置。使用 master.ign 或 worker.ign。仅支持 HTTP 和 HTTPS 协议。

(4) 对于在 FCP 类型磁盘中安装，请添加 zfcp.allow_lun_scan=0。为 DASD 类型磁盘省略这个值。

(5) 对于在 DASD 类型磁盘中安装，使用 rd.dasd=0.0.3490 替换来指定 DASD 设备。

注意

将参数文件中的所有选项写为一行，并确保您没有换行字符。

其他资源

- 使用 Butane 创建机器配置

18.7.12. 安装 RHCOS 并启动 OpenShift Container Platform bootstrap 过程

要在您置备的 IBM Z® 基础架构上安装 OpenShift Container Platform，您必须在 LPAR 中安装 Red Hat Enterprise Linux CoreOS (RHCOS)。安装 RHCOS 时，您必须为您要安装的机器类型提供 OpenShift Container Platform 安装程序生成的 Ignition 配置文件。如果您配置了合适的网络、DNS 和负载均衡基础架构，OpenShift Container Platform bootstrap 过程会在 RHCOS 客户机机器重启后自动启动。

完成以下步骤以创建机器。

先决条件

- 在置备机器上运行的 HTTP 或 HTTPS 服务器，可供您创建的机器访问。
- 如果要启用安全引导，需要已获取了适当的红帽产品签名密钥，并在 IBM 文档中的 IBM Z 和 IBM LinuxONE 上读取安全引导。

流程

1. 在您的置备机器上登录到 Linux。
2. 从 RHCOS 镜像镜像获取 Red Hat Enterprise Linux CoreOS(RHCOS)内核、initramfs 和 rootfs 文件。

重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每个发行版本而改变。您必须下载最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。仅使用以下步骤中描述的适当 kernel、initramfs 和 rootfs 工件。

文件名包含 OpenShift Container Platform 版本号。它们类似以下示例：

- **kernel: rhcos-<version>-live-kernel-<architecture>**
- **initramfs: rhcos-<version>-live-initramfs.<architecture>.img**
- **rootfs: rhcos-<version>-live-rootfs.<architecture>.img**

注意

FCP 和 DASD 的 rootfs 镜像相同。

3. 创建参数文件。以下参数特定于特定虚拟机：

- 对于 **ip=**，请指定以下七项：
  i. 计算机的 **IP** 地址。
  ii. 个空字符串。
  iii. 网关
  iv. 子网掩码.
  v. **hostname.domainname** 格式的机器主机和域名。省略这个值可让 **RHCOS** 决定。
  vi. 网络接口名称。省略这个值可让 **RHCOS** 决定。
  vii. 如果使用静态 **IP** 地址，请指定 **none**。
-

对于 coreos.inst.ignition_url=，请为机器角色指定 Ignition 文件。使用 bootstrap.ign、master.ign 或 worker.ign。仅支持 HTTP 和 HTTPS 协议。

- 对于 coreos.live.rootfs_url=，请为您引导的内核和 initramfs 指定匹配的 rootfs 构件。仅支持 HTTP 和 HTTPS 协议。

- 可选：要启用安全引导，请添加 coreos.inst.secure_ipl

- 对于在 DASD 类型磁盘中安装，请完成以下任务：

  i. 对于 coreos.inst.install_dev=，请指定 /dev/dasda。

  ii. Userd .dasd= 指定安装 RHCOS 的 DASD。

  iii. 所有其他参数保持不变。

  bootstrap 机器的参数文件示例：bootstrap-0.parm ：

```
rd.neednet=1 \
console=ttysclp0 \
coreos.inst.install_dev=/dev/dasda \
coreos.live.rootfs_url=http://<http_server>/rhcos-<version>-live-rootfs.
<architecture>.img \ 1
coreos.inst.ignition_url=http://<http_server>/bootstrap.ign \ 2
coreos.inst.secure_ipl \ 3
ip=172.18.78.2::172.18.78.1:255.255.255.0:::none nameserver=172.18.78.1 \
rd.znet=qeth,0.0.bdf0,0.0.bdf1,0.0.bdf2,layer2=1,portno=0 \
zfcp.allow_lun_scan=0 \
rd.dasd=0.0.3490
```

1 指定您要引导的 kernel 和 initramfs 的 rootfs 工件位置。仅支持 HTTP 和 HTTPS 协议。

2 指定 Ignition 配置文件的位置。使用 bootstrap.ign、master.ign 或 worker.ign。仅支持 HTTP 和 HTTPS 协议。

3

可选：要启用安全引导，请添加 coreos.inst.secure_ipl。

将参数文件中的所有选项写为一行，并确保您没有换行字符。

- 对于在 FCP 类型磁盘中安装，请完成以下任务：

i. User d.zfcp=<adapter>,<wwpn>,<lun> 以指定要安装 RHCOS 的 FCP 磁盘。对于多路径，为每个额外路径重复此步骤。

注意

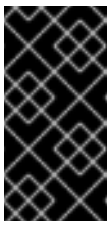
当使用多个路径安装时，您必须在安装后直接启用多路径，而不是在以后启用多路径，因为这可能导致问题。

ii. 将安装设备设置为：coreos.inst.install_dev=/dev/disk/by-id/scsi-<serial_number>。

注意

如果使用 NPIV 配置额外的 LUN，FCP 需要 zfcp.allow_lun_scan=0。如果必须启用 zfcp.allow_lun_scan=1，因为您使用 CSI 驱动程序，则必须配置 NPIV，以便每个节点无法访问另一个节点的引导分区。

iii. 所有其他参数保持不变。



重要

需要额外的安装后步骤才能完全启用多路径。如需更多信息，请参阅 *安装后机器配置任务* 中的"使用 RHCOS 上内核参数启用多路径"。

以下是使用多路径的计算节点的 worker-1.parm 示例参数文件：

```
rd.neednet=1 \
console=ttysclp0 \
coreos.inst.install_dev=/dev/disk/by-id/scsi-<serial_number> \
coreos.live.rootfs_url=http://<http_server>/rhcos-<version>-live-rootfs.
<architecture>.img \
coreos.inst.ignition_url=http://<http_server>/worker.ign \
ip=172.18.78.2::172.18.78.1:255.255.255.0:::none nameserver=172.18.78.1 \
rd.znet=qeth,0.0.bdf0,0.0.bdf1,0.0.bdf2,layer2=1,portno=0 \
zfcp.allow_lun_scan=0 \
rd.zfcp=0.0.1987,0x50050763070bc5e3,0x4008400B00000000 \
rd.zfcp=0.0.19C7,0x50050763070bc5e3,0x4008400B00000000 \
rd.zfcp=0.0.1987,0x50050763071bc5e3,0x4008400B00000000 \
rd.zfcp=0.0.19C7,0x50050763071bc5e3,0x4008400B00000000
```

将参数文件中的所有选项写为一行，并确保您没有换行字符。

4. 将 initramfs、内核、参数文件和 RHCOS 镜像传送到 LPAR，例如使用 FTP。有关如何使用 FTP 和引导传输文件的详情，请参考在 LPAR 中安装。

5. 引导机器

6. 对集群中的其他机器重复此步骤。

18.7.12.1. 高级 RHCOS 安装参考

本节演示了网络配置和其他高级选项，允许您修改 Red Hat Enterprise Linux CoreOS(RHCOS)手动安装过程。下表描述了您可以用于 RHCOS live 安装程序和 coreos-installer 命令的内核参数和命令行选项。

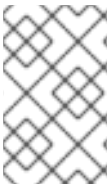
18.7.12.1.1. ISO 安装的网络和绑定选项

如果从 ISO 镜像安装 RHCOS，您可以在引导镜像时手动添加内核参数，以便为节点配置网络。如果没有指定网络参数，当 RHCOS 检测到需要网络来获取 Ignition 配置文件时，在 initramfs 中激活 DHCP。

重要

在手动添加网络参数时，还必须添加 rd.neednet=1 内核参数，以便在 initramfs 中启动网络。

以下信息提供了在 RHCOS 节点上为 ISO 安装配置网络和绑定的示例。示例描述了如何使用 ip=、name server = 和 bond= 内核参数。



注意

添加内核参数时顺序非常重要： ip=、 name server=，然后 bond=。

网络选项在系统引导过程中传递给 dracut 工具。有关 dracut 支持的网络选项的更多信息，请参阅 dracut.cmdline 手册页。

以下示例是 ISO 安装的网络选项。

配置 DHCP 或静态 IP 地址

要配置 IP 地址，可使用 DHCP(ip=dhcp)或设置单独的静态 IP 地址(ip=<host_ip>)。如果设置静态 IP，则必须在每个节点上识别 DNS 服务器 IP 地址（名称服务器=<dns_ip>）。以下示例集：

- 节点的 IP 地址为 10.10.10.2
- 网关地址为 10.10.10.254
- 子网掩码为 255.255.255.0
- 到 core0.example.com 的主机名
- DNS 服务器地址为 4.4.4.41
- 自动配置值为 none。当以静态方式配置 IP 网络时，不需要自动配置。

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
nameserver=4.4.4.41
```

注意

当您使用 DHCP 为 RHCOS 机器配置 IP 寻址时，机器还通过 DHCP 获取 DNS 服务器信息。对于基于 DHCP 的部署，您可以通过 DHCP 服务器配置定义 RHCOS 节点使用的 DNS 服务器地址。

配置没有静态主机名的 IP 地址

您可以在不分配静态主机名的情况下配置 IP 地址。如果用户没有设置静态主机名，则会提取并通过反向 DNS 查找自动设置。要在没有静态主机名的情况下配置 IP 地址，请参考以下示例：

- **节点的 IP 地址为 10.10.10.2**
- **网关地址为 10.10.10.254**
- **子网掩码为 255.255.255.0**
- **DNS 服务器地址为 4.4.4.41**
- **自动配置值为 none。当以静态方式配置 IP 网络时，不需要自动配置。**

```
ip=10.10.10.2::10.10.10.254:255.255.255.0::enp1s0:none
nameserver=4.4.4.41
```

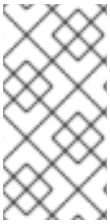
指定多个网络接口

您可以通过设置多个 ip= 条目来指定多个网络接口。

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
ip=10.10.10.3::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none
```

配置默认网关和路由

可选：您可以通过设置 a rd.route= 值来配置到额外网络的路由。



注意

当您配置一个或多个网络时，需要一个默认网关。如果额外网络网关与主要网络网关不同，则默认网关必须是主要网络网关。

- 运行以下命令来配置默认网关：

```
ip=::10.10.10.254::::
```

- 输入以下命令为额外网络配置路由：

```
rd.route=20.20.20.0/24:20.20.20.20.254:enp2s0
```

在单个接口中禁用 DHCP

您可以在单一接口中禁用 DHCP，例如当有两个或者多个网络接口时，且只有一个接口被使用。在示例中，enp1s0 接口具有一个静态网络配置，而 enp2s0 禁用了 DHCP，不使用它：

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
ip=::::core0.example.com:enp2s0:none
```

合并 DHCP 和静态 IP 配置

您可以将系统上的 DHCP 和静态 IP 配置与多个网络接口合并，例如：

```
ip=enp1s0:dhcp
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none
```

在独立接口上配置 VLAN

可选：您可以使用 vlan= 参数在单个接口上配置 VLAN。

- 要在网络接口中配置 VLAN 并使用静态 IP 地址，请运行以下命令：

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0.100:none
vlan=enp2s0.100:enp2s0
```

- 要在网络接口中配置 VLAN 并使用 DHCP，请运行以下命令：

```
ip=enp2s0.100:dhcp
vlan=enp2s0.100:enp2s0
```

-

提供多个 DNS 服务器

您可以通过为每个服务器添加一个 nameserver= 条目来提供多个 DNS 服务器，例如

```
nameserver=1.1.1.1
nameserver=8.8.8.8
```

将多个网络接口绑定到一个接口

可选：您可以使用 bond= 选项将多个网络接口绑定到一个接口。请参见以下示例：

- 配置绑定接口的语法为：bond=<name>[:<network_interfaces>][:options]

  <name> 是绑定设备名称 (bond0)、<network_interfaces> 代表以逗号分隔的物理（以太网）接口列表(em1,em2)，*options* 是用逗号分开的绑定选项列表。输入 modinfo bonding 查看可用选项。

- 当使用 bond= 创建绑定接口时，您必须指定如何分配 IP 地址以及绑定接口的其他信息。

  - 要将绑定接口配置为使用 DHCP，请将绑定的 IP 地址设置为 dhcp。例如：

    ```
    bond=bond0:em1,em2:mode=active-backup
    ip=bond0:dhcp
    ```

  - 要将绑定接口配置为使用静态 IP 地址，请输入您需要的特定 IP 地址和相关信息。例如：

```
bond=bond0:em1,em2:mode=active-backup,fail_over_mac=1
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0:none
```

在 active-backup 模式中始终设置 fail_over_mac=1 选项，以避免使用共享 OSA/RoCE 卡时出现问题。

将多个网络接口绑定到一个接口

可选：您可以使用 vlan= 参数并在绑定接口上配置 VLAN，并使用 DHCP，例如：

```
ip=bond0.100:dhcp
bond=bond0:em1,em2:mode=active-backup
vlan=bond0.100:bond0
```

–

使用以下示例配置带有 VLAN 的绑定接口并使用静态 IP 地址：

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0.100:none
bond=bond0:em1,em2:mode=active-backup
vlan=bond0.100:bond0
```

使用网络团队

可选：您可以使用 team= 参数来将网络团队用作绑定的替代选择：

- 配置组接口的语法为：team=name[:network_interfaces]

  name 是组设备名称(team0)，*network_interfaces* 代表以逗号分隔的物理（以太网）接口（em1、em2）列表。

注意

当 RHCOS 切换到即将推出的 RHEL 版本时，团队(team)功能被计划弃用。如需更多信息，请参阅红帽知识库文章。

使用以下示例配置网络团队：

```
team=team0:em1,em2
ip=team0:dhcp
```

18.7.13. 等待 bootstrap 过程完成

OpenShift Container Platform bootstrap 过程在集群节点首次引导到安装到磁盘的持久 RHCOS 环境后开始。通过 Ignition 配置文件提供的配置信息用于初始化 bootstrap 过程并在机器上安装 OpenShift Container Platform。您必须等待 bootstrap 过程完成。

先决条件

- 已为集群创建 Ignition 配置文件。
- 您已配置了适当的网络、DNS 和负载平衡基础架构。

- 已获得安装程序，并为集群生成 Ignition 配置文件。
- 已在集群机器上安装 RHCOS，并提供 OpenShift Container Platform 安装程序生成的 Ignition 配置文件。

流程

1. 监控 bootstrap 过程：

```
$ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \ 1
    --log-level=info 2
```

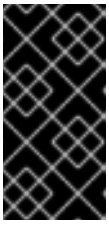
1. 对于 <installation_directory>，请指定安装文件保存到的目录的路径。
2. 要查看不同的安装详情，请指定 warn、debug 或 error，而不是 info。

输出示例

```
INFO Waiting up to 30m0s for the Kubernetes API at
https://api.test.example.com:6443...
INFO API v1.29.4 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
```

当 Kubernetes API 服务器提示已在 control plane 机器上引导它时，该命令会成功。

2. bootstrap 过程完成后，从负载均衡器中删除 bootstrap 机器。



重要

此时您必须从负载均衡器中删除 bootstrap 机器。您还可以删除或重新格式化 bootstrap 机器本身。

18.7.14. 使用 CLI 登录集群

您可以通过导出集群 kubeconfig 文件，以默认系统用户身份登录集群。kubeconfig 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。
- 已安装 oc CLI。

流程

1. 导出 kubeadmin 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 <installation_directory>，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 oc 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

18.7.15. 批准机器的证书签名请求

当您将机器添加到集群时，会为您添加的每台机器生成两个待处理证书签名请求(CSR)。您必须确认这些 CSR 已获得批准，或根据需要自行批准。必须首先批准客户端请求，然后批准服务器请求。

先决条件

- 您已将机器添加到集群中。

流程

1. 确认集群可以识别这些机器：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0  Ready    master  63m  v1.29.4
master-1  Ready    master  63m  v1.29.4
master-2  Ready    master  64m  v1.29.4
```

输出中列出了您创建的所有机器。

注意

在有些 CSR 被批准前，前面的输出可能不包括计算节点（也称为 worker 节点）。

2. 检查待处理的 CSR，并确保添加到集群中的每台机器都有 Pending 或 Approved 状态的客户端请求：

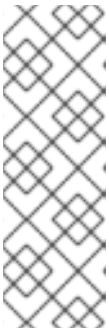
```
$ oc get csr
```

输出示例

```
NAME        AGE    REQUESTOR                                                                   CONDITION
csr-8b2br   15m    system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
csr-8vnps   15m    system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
...
```

在本例中，两台机器加入集群。您可能会在列表中看到更多已批准的 CSR。

3. 如果 CSR 没有获得批准，在您添加的机器的所有待处理 CSR 都处于 Pending 状态后，请批准集群机器的 CSR：



注意

由于 CSR 会自动轮转，因此请在将机器添加到集群后一小时内批准您的 CSR。如果没有在一小时内批准它们，证书将会轮转，每个节点会存在多个证书。您必须批准所有这些证书。批准客户端 CSR 后，Kubelet 为服务证书创建一个二级 CSR，这需要手动批准。然后，如果 Kubelet 请求具有相同参数的新证书，则后续提供证书续订请求由 machine-approver 自动批准。

注意

对于在未启用机器 API 的平台上运行的集群，如裸机和其他用户置备的基础架构，您必须实施一种方法来自动批准 kubelet 提供证书请求(CSR)。如果没有批准请求，则 oc exec、ocrsh 和 oc logs 命令将无法成功，因为 API 服务器连接到 kubelet 时需要服务证书。与 Kubelet 端点联系的任何操作都需要此证书批准。该方法必须监视新的 CSR，确认 CSR 由 system: node 或 system:admin 组中的 node-bootstrapper 服务帐户提交，并确认节点的身份。

- 要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1

<csr_name> 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}
{{"\n"}}{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```

注意

在有些 CSR 被批准前，一些 Operator 可能无法使用。

4. 现在，您的客户端请求已被批准，您必须查看添加到集群中的每台机器的服务器请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE     REQUESTOR                                                   CONDITION
csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

5. 如果剩余的 CSR 没有被批准，且处于 Pending 状态，请批准集群机器的 CSR：

- 要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1

<csr_name> 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}
{{"\n"}}{{end}}{{end}}' | xargs oc adm certificate approve
```

6. 批准所有客户端和服务器 CSR 后，机器将处于 Ready 状态。运行以下命令验证：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0  Ready    master  73m  v1.29.4
master-1  Ready    master  73m  v1.29.4
master-2  Ready    master  74m  v1.29.4
worker-0  Ready    worker  11m  v1.29.4
worker-1  Ready    worker  11m  v1.29.4
```

注意

批准服务器 CSR 后可能需要几分钟时间让机器过渡到 Ready 状态。

其他信息

- 如需有关 CSR 的更多信息，请参阅 证书签名请求。

18.7.16. 初始 Operator 配置

在 control plane 初始化后，您必须立即配置一些 Operator，以便它们都可用。

先决条件

- 您的 control plane 已初始化。

流程

1. 观察集群组件上线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION   AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.16.0    True        False         False      19m
baremetal                                  4.16.0    True        False         False      37m
cloud-credential                           4.16.0    True        False         False      40m
cluster-autoscaler                         4.16.0    True        False         False      37m
config-operator                            4.16.0    True        False         False      38m
console                                    4.16.0    True        False         False      26m
csi-snapshot-controller                    4.16.0    True        False         False      37m
dns                                        4.16.0    True        False         False      37m
etcd                                       4.16.0    True        False         False      36m
image-registry                             4.16.0    True        False         False      31m
ingress                                    4.16.0    True        False         False      30m
insights                                   4.16.0    True        False         False      31m
kube-apiserver                             4.16.0    True        False         False      26m
kube-controller-manager                    4.16.0    True        False         False      36m
kube-scheduler                             4.16.0    True        False         False      36m
kube-storage-version-migrator              4.16.0    True        False         False      37m
machine-api                                4.16.0    True        False         False      29m
machine-approver                           4.16.0    True        False         False      37m
machine-config                             4.16.0    True        False         False      36m
marketplace                                4.16.0    True        False         False      37m
monitoring                                 4.16.0    True        False         False      29m
network                                    4.16.0    True        False         False      38m
node-tuning                                4.16.0    True        False         False      37m
openshift-apiserver                        4.16.0    True        False         False      32m
openshift-controller-manager               4.16.0    True        False         False      30m
openshift-samples                          4.16.0    True        False         False      32m
operator-lifecycle-manager                 4.16.0    True        False         False      37m
operator-lifecycle-manager-catalog         4.16.0    True        False         False      37m
operator-lifecycle-manager-packageserver   4.16.0    True        False         False      32m
service-ca                                 4.16.0    True        False         False      38m
storage                                    4.16.0    True        False         False      37m
```

2. 配置不可用的 Operator。

18.7.16.1. 禁用默认的 OperatorHub 目录源

在 OpenShift Container Platform 安装过程中，默认为 OperatorHub 配置由红帽和社区项目提供的源内容的 operator 目录。在受限网络环境中，必须以集群管理员身份禁用默认目录。

流程

- 通过在 OperatorHub 对象中添加 disableAllDefaultSources: true 来 禁用默认目录的源：

```
$ oc patch OperatorHub cluster --type json \
    -p '[{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]'
```

提示

或者，您可以使用 Web 控制台管理目录源。在 Administration → Cluster Settings → Configuration → OperatorHub 页面中，点 Sources 选项卡，您可以在其中创建、更新、删除、禁用和启用单独的源。

18.7.16.2. 镜像 registry 存储配置

对于不提供默认存储的平台，Image Registry Operator 最初不可用。安装后，您必须将 registry 配置为使用存储，以便 Registry Operator 可用。

显示配置生产集群所需的持久性卷的说明。如果适用，显示有关将空目录配置为存储位置的说明，这仅适用于非生产集群。

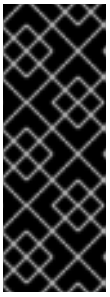
提供了在升级过程中使用 Recreate rollout 策略来允许镜像 registry 使用块存储类型的说明。

18.7.16.2.1. 为 IBM Z 配置 registry 存储

作为集群管理员，在安装后需要配置 registry 来使用存储。

先决条件

- 您可以使用具有 cluster-admin 角色的用户访问集群。
- 在 IBM Z® 上有一个集群。
- 您已为集群置备持久性存储，如 Red Hat OpenShift Data Foundation。



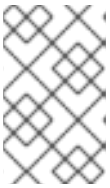
重要

当您只有一个副本时，OpenShift Container Platform 支持对镜像 registry 存储的 ReadWriteOnce 访问。ReadWriteOnce 访问还要求 registry 使用 Recreate rollout 策略。要部署支持高可用性的镜像 registry，需要两个或多个副本，ReadWriteMany 访问。

- 必须具有 100Gi 容量。

流程

1. 要将 registry 配置为使用存储，修改 configs.imageregistry/cluster 资源中的 spec.storage.pvc。



注意

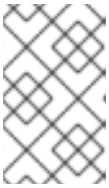
使用共享存储时，请查看您的安全设置以防止外部访问。

2. 验证您没有 registry pod:

```
$ oc get pod -n openshift-image-registry -l docker-registry=default
```

输出示例

```
No resources found in openshift-image-registry namespace
```



注意

如果您的输出中有一个 registry pod，则不需要继续这个过程。

3. 检查 **registry** 配置：

```
$ oc edit configs.imageregistry.operator.openshift.io
```

输出示例

```
storage:
  pvc:
    claim:
```

将 claim 字段留空以允许自动创建 image-registry-storage PVC。

4. 检查 **clusteroperator** 状态：

```
$ oc get clusteroperator image-registry
```

输出示例

```
NAME            VERSION      AVAILABLE   PROGRESSING   DEGRADED   SINCE   MESSAGE
image-registry   4.16         True        False         False      6h50m
```

5. **确保 registry** 设置为 **managed，以启用镜像的构建和推送。**

- 运行：

```
$ oc edit configs.imageregistry/cluster
```

然后，更改行

```
managementState: Removed
```

至

```
managementState: Managed
```

18.7.16.2.2. 在非生产集群中为镜像 registry 配置存储

您必须为 Image Registry Operator 配置存储。对于非生产集群，您可以将镜像 registry 设置为空目录。如果您这样做，重启 registry 时会丢失所有镜像。

流程

- 将镜像 registry 存储设置为空目录：

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch
'{"spec":{"storage":{"emptyDir":{}}}}'
```

警告

仅为非生产集群配置这个选项。

如果在 Image Registry Operator 初始化其组件前运行这个命令，oc patch 命令会失败并显示以下错误：

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster"
not found
```

等待几分钟，然后再次运行 命令。

18.7.17. 在用户置备的基础架构上完成安装

完成 Operator 配置后，可以在您提供的基础架构上完成集群安装。

先决条件

- 您的 control plane 已初始化。
- 已完成初始 Operator 配置。

流程

1. 使用以下命令确认所有集群组件都在线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION   AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.16.0    True        False         False      19m
baremetal                                  4.16.0    True        False         False      37m
cloud-credential                           4.16.0    True        False         False      40m
cluster-autoscaler                         4.16.0    True        False         False      37m
config-operator                            4.16.0    True        False         False      38m
console                                    4.16.0    True        False         False      26m
csi-snapshot-controller                    4.16.0    True        False         False      37m
dns                                        4.16.0    True        False         False      37m
etcd                                       4.16.0    True        False         False      36m
image-registry                             4.16.0    True        False         False      31m
ingress                                    4.16.0    True        False         False      30m
insights                                   4.16.0    True        False         False      31m
kube-apiserver                             4.16.0    True        False         False      26m
kube-controller-manager                    4.16.0    True        False         False      36m
kube-scheduler                             4.16.0    True        False         False      36m
kube-storage-version-migrator              4.16.0    True        False         False      37m
machine-api                                4.16.0    True        False         False      29m
machine-approver                           4.16.0    True        False         False      37m
machine-config                             4.16.0    True        False         False      36m
marketplace                                4.16.0    True        False         False      37m
monitoring                                 4.16.0    True        False         False      29m
network                                    4.16.0    True        False         False      38m
node-tuning                                4.16.0    True        False         False      37m
openshift-apiserver                        4.16.0    True        False         False      32m
```

```
openshift-controller-manager             4.16.0    True        False         False      30m
openshift-samples                        4.16.0    True        False         False      32m
operator-lifecycle-manager               4.16.0    True        False         False      37m
operator-lifecycle-manager-catalog       4.16.0    True        False         False      37m
operator-lifecycle-manager-packageserver 4.16.0    True        False         False      32m
service-ca                               4.16.0    True        False         False      38m
storage                                  4.16.0    True        False         False      37m
```

另外，当所有集群都可用时，以下命令会通知您。它还检索并显示凭证：

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

1

对于 <installation_directory>，请指定安装文件保存到的目录的路径。

输出示例

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

Cluster Version Operator 完成从 Kubernetes API 服务器部署 OpenShift Container Platform 集群时，该命令会成功。

重要

- 安装程序生成的 Ignition 配置文件包含 24 小时后过期的证书，然后在该时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 node-bootstrapper 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。

- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

2. 确认 Kubernetes API 服务器正在与 pod 通信。

a. 要查看所有 pod 的列表，请使用以下命令：

```
$ oc get pods --all-namespaces
```

输出示例

```
NAMESPACE                       NAME                                        READY   STATUS
RESTARTS   AGE
openshift-apiserver-operator    openshift-apiserver-operator-85cb746d55-zqhs8
1/1     Running     1          9m
openshift-apiserver             apiserver-67b9g                             1/1     Running     0
3m
openshift-apiserver             apiserver-ljcmx                             1/1     Running     0
1m
openshift-apiserver             apiserver-z25h4                             1/1     Running     0
2m
openshift-authentication-operator authentication-operator-69d5d8bf84-vh2n8
1/1     Running     0          5m
...
```

b. 使用以下命令，查看上一命令的输出中所列 pod 的日志：

```
$ oc logs <pod_name> -n <namespace> 1
```

1

指定 pod 名称和命名空间，如上一命令的输出中所示。

如果 pod 日志显示，Kubernetes API 服务器可以与集群机器通信。

3. 对于使用光纤通道协议(FCP)的安装，还需要额外的步骤才能启用多路径。不要在安装过程中启用多路径。

   如需更多信息，请参阅 *安装后机器配置任务* 文档中的"使用 RHCOS 上使用内核参数启用多路径"。

4. 在 Cluster registration 页面注册 您的集群。

验证

如果您在 OpenShift Container Platform bootstrap 过程中启用了安全引导，则需要以下验证步骤：

1. 运行以下命令来调试节点：

```
$ oc debug node/<node_name>
chroot /host
```

2. 运行以下命令确认启用了安全引导：

```
$ cat /sys/firmware/ipl/secure
```

输出示例

```
1 1
```

1

如果启用了安全引导，则值为 1，如果未启用安全引导，则值为 0。

3. 运行以下命令列出 re-IPL 配置：

```
# lsreipl
```

FCP 磁盘的输出示例

```
Re-IPL type: fcp
WWPN: 0x500507630400d1e3
LUN: 0x4001400e00000000
Device: 0.0.810e
bootprog: 0
br_lba: 0
Loadparm: ""
Bootparms: ""
clear: 0
```

DASD 磁盘输出示例

```
for DASD output:
Re-IPL type: ccw
Device: 0.0.525d
Loadparm: ""
clear: 0
```

4. 运行以下命令来关闭节点：

```
sudo shutdown -h
```

5. 从 LPAR 从硬件管理控制台 (HMC) 启动引导。请参阅 IBM 文档中的 从 LPAR 启动安全引导。

6. 当节点恢复时，再次检查安全引导状态。

其他资源

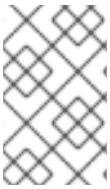
- 如何在没有 SSH 的 OpenShift Container Platform 版本 4 节点中生成 SOSREPORT。

18.7.18. 后续步骤

- 自定义集群。
- 如果您用来安装集群的镜像 registry 具有可信任的 CA，请通过 配置额外的信任存储将其添加到集群中。
- 如果需要，您可以选择 不使用远程健康报告。
- 如果需要，请参阅 注册断开连接的集群

18.8. IBM Z 和 IBM LINUXONE 的安装配置参数

在部署 OpenShift Container Platform 集群前，您可以提供一个自定义 的 install-config.yaml 安装配置文件，该文件描述了您的环境的详情。

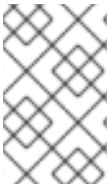


注意

虽然本文档只涉及 IBM Z®，但它的所有信息也适用于 IBM® LinuxONE。

18.8.1. IBM Z 可用的安装配置参数

下表指定您可以在安装过程中设置所需的、可选和 IBM Z 特定安装配置参数。



注意

安装后，您无法在 install-config.yaml 文件中修改这些参数。

18.8.1.1. 所需的配置参数

下表描述了所需的安装配置参数：

表 18.114. 所需的参数

参数	描述	值
apiVersion:	**install-config.yaml** 内容的 API 版本。当前版本为 **v1**。安装程序可能还支持旧的 API 版本。	字符串
baseDomain:	云供应商的基域。基域用于创建到 OpenShift Container Platform 集群组件的路由。集群的完整 DNS 名称是 **baseDomain** 和 **metadata.name** 参数值的组合，其格式为 **<metadata.name>.<baseDomain>**。	完全限定域名或子域名，如 **example.com**。
metadata:	Kubernetes 资源 **ObjectMeta**，其中只消耗 **name** 参数。	对象
metadata: name:	集群的名称。集群的 DNS 记录是 **{{.metadata.name}}.{{.baseDomain}}** 的子域。	小写字母、连字符(**-**)和句点(**.**)字符串，如 **dev**。
platform:	对于特定平台的配置取决于执行安装的环境： **aws**, **baremetal**, **azure**, **gcp**, **ibmcloud**, **nutanix**, **openstack**, **powervs**, **vsphere**, 或 **{}**.有关 **platform.<platform>** 参数的更多信息，请参考下表中您的特定平台。	对象

参数	描述	值
pullSecret:	从 Red Hat OpenShift Cluster Manager 获取 pull secret，验证从 Quay.io 等服务中下载 OpenShift Container Platform 组件的容器镜像。	{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }

18.8.1.2. 网络配置参数

您可以根据现有网络基础架构的要求自定义安装配置。例如，您可以扩展集群网络的 IP 地址块，或者提供不同于默认值的不同 IP 地址块。

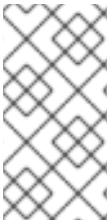
- 如果使用 Red Hat OpenShift Networking OVN-Kubernetes 网络插件，则支持 IPv4 和 IPv6 地址系列。

如果将集群配置为使用两个 IP 地址系列，请查看以下要求：

- 两个 IP 系列都必须将相同的网络接口用于默认网关。
- 两个 IP 系列都必须具有默认网关。
- 您必须为所有网络配置参数指定 IPv4 和 IPv6 地址。例如，以下配置 IPv4 地址列在 IPv6 地址的前面。

```
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  - cidr: fd00:10:128::/56
    hostPrefix: 64
```

```
serviceNetwork:
- 172.30.0.0/16
- fd00:172:16::/112
```



注意

Red Hat OpenShift Data Foundation 灾难恢复解决方案不支持 Globalnet。对于区域灾难恢复场景，请确保为每个集群中的集群和服务网络使用非重叠的专用 IP 地址。

表 18.115. 网络参数

参数	描述	值
networking:	集群网络的配置。	对象  **注意** 您无法在安装后修改 **网络** 对象指定的参数。
networking: networkType:	要安装的 Red Hat OpenShift Networking 网络插件。	**OVNKubernetes**。**OVNKubernetes** 是 Linux 网络和包含 Linux 和 Windows 服务器的混合网络的 CNI 插件。默认值为 **OVNKubernetes**。

networking:
clusterNetwork:
	pod 的 IP 地址块。
	默认值为 **10.128.0.0/14**，主机前缀为 **/23**。
	如果您指定了多个 IP 地址块，块不得重叠。
		对象数组。例如：
		`networking:`
		`clusterNetwork:`
		`- cidr: 10.128.0.0/14`
		`hostPrefix: 23`

networking: clusterNetwork: cidr:	使用 **networking.clusterNetwork** 时需要此项。IP 地址块。 IPv4 网络。	无类别域间路由(CIDR)表示法中的 IP 地址块。IPv4 块的前缀长度介于 **0 到 32** 之间。
networking: clusterNetwork: hostPrefix:	分配给每个节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从 given **cidr** 中分配 a/**23** 子网。**hostPrefix** 值 **23** 提供 510（2^(32 - 23)- 2）pod IP 地址。	子网前缀。 默认值为 **23**。

参数	描述	值
networking: serviceNetwork:	服务的 IP 地址块。默认值为 **172.30.0.0/16**。 OVN-Kubernetes 网络插件只支持服务网络的一个 IP 地址块。	CIDR 格式具有 IP 地址块的数组。例如： networking: serviceNetwork: - 172.30.0.0/16
networking: machineNetwork:	机器的 IP 地址块。 如果您指定了多个 IP 地址块，块不得重叠。 如果您指定了多个 IP 内核参数，**machineNetwork.cidr** 值必须是主网络的 CIDR。	对象数组。例如： networking: machineNetwork: - cidr: 10.0.0.0/16
networking: machineNetwork: cidr:	使用 **networking.machineNetwork** 时需要此项。IP 地址块。对于 libvirt 和 IBM Power® Virtual Server 以外的所有平台，默认值为 **10.0.0.0/16**。对于 libvirt，默认值 为 **192.168.126.0/24**。对于 IBM Power® Virtual Server，默认值为 **192.168.0.0/24**。	CIDR 表示法中的 IP 网络块。 例如： **10.0.0.0/16**。 **注意** 将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

18.8.1.3. 可选的配置参数

下表描述了可选的安装配置参数：

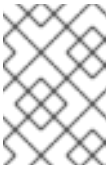
表 18.116. 可选参数

参数	描述	值
additionalTrustBundle:	添加到节点可信证书存储中的 PEM 编码 X.509 证书捆绑包。配置了代理时，也可以使用此信任捆绑包。	字符串

参数	描述	值
capabilities:	控制可选核心组件的安装。您可以通过禁用可选组件来减少 OpenShift Container Platform 集群的空间。如需更多信息，请参阅*安装*中的"集群功能"页面。	字符串数组
capabilities: baselineCapabilitySet:	选择要启用的一组初始可选功能。有效值为 **None**、**v4.11**、**v4.12** 和 **vCurrent**。默认值为 **vCurrent**。	字符串
capabilities: additionalEnabledCapabilities:	将可选功能集合扩展到您在 **baselineCapabilitySet** 中指定的范围。您可以在此参数中指定多个功能。	字符串数组
cpuPartitioningMode:	启用工作负载分区，它会隔离 OpenShift Container Platform 服务、集群管理工作负载和基础架构 pod，以便在保留的一组 CPU 上运行。工作负载分区只能在安装过程中启用，且在安装后无法禁用。虽然此字段启用工作负载分区，但它不会将工作负载配置为使用特定的 CPU。如需更多信息，请参阅 *Scalability and Performance* 部分中的 *Workload partitioning* 页面。	**None** 或 **AllNodes**.**None** 是默认值。
compute:	组成计算节点的机器的配置。	**MachinePool** 对象的数组。
compute: architecture:	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **s390x** （默认值）。	字符串
compute: hyperthreading:	是否在计算机器上启用或禁用并发多**线程或超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。	**enabled** 或 **Disabled**

参数	描述	值
compute: name:	使用 **compute** 时需要此项。机器池的名称。	**worker**
compute: platform:	使用 **compute** 时需要此项。使用此参数指定托管 worker 机器的云供应商。此参数值必须与 **controlPlane.platform** 参数值匹配。	**aws**, **azure**, **gcp**, **ibmcloud**, **nutanix**, **openstack**, **powervs**, **vsphere**, 或 **{}**
compute: replicas:	要置备的计算机器数量，也称为 worker 机器。	大于或等于 **2** 的正整数。默认值为 **3**。
featureSet:	为功能集启用集群。功能集是 OpenShift Container Platform 功能的集合，默认情况下不启用。有关在安装过程中启用功能集的更多信息，请参阅"使用功能门启用功能"。	字符串.要启用的功能集的名称，如 **TechPreviewNoUpgrade**。
controlPlane:	组成 control plane 的机器的配置。	**MachinePool** 对象的数组。
controlPlane: architecture:	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **s390x** （默认值）。	字符串
controlPlane: hyperthreading:	是否在 control plane 机器上启用或禁用并发多 **线程或超线程**。默认情况下，启用并发多线程以提高机器内核的性能。 **重要** 如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。	**enabled** 或 **Disabled**
controlPlane: name:	使用 **controlPlane** 时需要此项。机器池的名称。	**master**

参数	描述	值
controlPlane: platform:	使用 **controlPlane** 时需要此项。使用此参数指定托管 control plane 机器的云供应商。此参数值必须与 **compute.platform** 参数值匹配。	**aws**, **azure**, **gcp**, **ibmcloud**, **nutanix**, **openstack**, **powervs**, **vsphere**, 或 **{}**
controlPlane: replicas:	要置备的 control plane 机器数量。	部署单节点 OpenShift 时支持的值为 **3** 或 **1**。
credentialsMode:	Cloud Credential Operator(CCO)模式。如果没有指定模式，CCO 会动态尝试决定提供的凭证的功能，在支持多个模式的平台上首选 mint 模式。	**Mint**、**Passthrough**、**Manual** 或空字符串(**""**)。[1]

参数	描述	值
fips:	启用或禁用 FIPS 模式。默认值为 **false**（禁用）。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。  **重要** 要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 Red Hat Enterprise Linux (RHEL) 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅在 FIPS 模式中安装该系统。 当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS) 时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。  **注意** 如果使用 Azure File 存储，则无法启用 FIPS 模式。	**false** 或 **true**
imageContentSources:	release-image 内容的源和存储库。	对象数组。包括一个 **source** 以及可选的 **mirrors**，如本表的以下行所述。
imageContentSources: source:	使用 **imageContentSources** 时需要此项。指定用户在镜像拉取规格中引用的存储库。	字符串

参数	描述	值
imageContentSources: mirrors:	指定可能还包含同一镜像的一个或多个仓库。	字符串数组
publish:	如何发布或公开集群的面向用户的端点，如 Kubernetes API、OpenShift 路由。	**内部或外部 .**默认值为 **External**。 在非云平台上不支持将此字段设置为 **Internal**。 **重要** 如果将字段的值设为 **Internal**，集群将无法运行。如需更多信息，请参阅 BZ#1953035。
sshKey:	用于验证对集群机器的访问的 SSH 密钥。 **注意** 对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。	例如，**sshKey: ssh-ed25519 AAAA..**。

1. **不是所有 CCO 模式都支持所有云供应商。有关 CCO 模式的更多信息，请参阅*身份验证和授权*内容中的"管理云供应商凭证"条目。**

第 19 章 在 IBM POWER 上安装

19.1. 准备在 IBM POWER 上安装

19.1.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读有关 选择集群安装方法的文档，并为用户准备它。

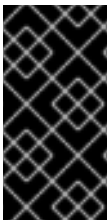
19.1.2. 选择在 IBM Power 上安装 OpenShift Container Platform 的方法

您可以使用以下方法之一在您置备的 IBM Power® 基础架构上安装集群：

- 在 IBM Power® 上安装集群 ：您可以在您置备的 IBM Power® 基础架构上安装 OpenShift Container Platform。
- 在受限网络中的 IBM Power® 上安装集群 ：您可以使用安装发行内容的内部镜像在受限或断开连接的网络中置备的 IBM Power® 基础架构上安装 OpenShift Container Platform。您可以使用此方法安装不需要活跃互联网连接的集群来获取软件组件。您还可以使用此安装方法来确保集群只使用满足您组织对外部内容控制的容器镜像。

19.2. 在 IBM POWER 上安装集群

在 OpenShift Container Platform 版本 4.16 中，您可以在您置备的 IBM Power® 基础架构上安装集群。



重要

非裸机平台还有其他注意事项。在安装 OpenShift Container Platform 集群前，请参阅有关在未经测试的平台上部署 OpenShift Container Platform 的指南 中的信息。

19.2.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。

- 您可以阅读有关 选择集群安装方法的文档，并为用户准备它。

- 在开始安装过程前，您必须清理安装目录。这样可确保在安装过程中创建和更新所需的安装文件。

- 已为集群置备了 使用 OpenShift Data Foundation 或其他支持的存储协议的持久性存储。要部署私有镜像 registry，您必须使用 ReadWriteMany 访问设置持久性存储。

- 如果使用防火墙，则会 将其配置为允许集群需要访问的站点。

注意

如果要配置代理，请务必查看此站点列表。

19.2.2. OpenShift Container Platform 互联网访问

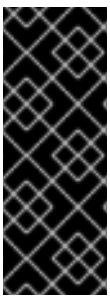
在 OpenShift Container Platform 4.16 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。

- 访问 Quay.io，以获取安装集群所需的软件包。

- 获取执行集群更新所需的软件包。



重要

如果您的集群无法直接访问互联网，则可以在置备的某些类型的基础架构上执行受限网络安装。在此过程中，您可以下载所需的内容，并使用它为镜像 registry 填充安装软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群前，您要更新镜像 registry 的内容。

19.2.3. 具有用户置备基础架构的集群的要求

对于包含用户置备的基础架构的集群，您必须部署所有所需的机器。

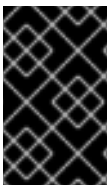
本节论述了在用户置备的基础架构上部署 OpenShift Container Platform 的要求。

19.2.3.1. 集群安装所需的机器

最小的 OpenShift Container Platform 集群需要以下主机：

表 19.1. 最低所需的主机

主机	描述
一个临时 bootstrap 机器	集群需要 bootstrap 机器在三台 control plane 机器上部署 OpenShift Container Platform 集群。您可在安装集群后删除 bootstrap 机器。
三台 control plane 机器	control plane 机器运行组成 control plane 的 Kubernetes 和 OpenShift Container Platform 服务。
至少两台计算机器，也称为 worker 机器。	OpenShift Container Platform 用户请求的工作负载在计算机器上运行。



重要

要保持集群的高可用性，请将独立的物理主机用于这些集群机器。

bootstrap、control plane 和计算机器必须使用 Red Hat Enterprise Linux CoreOS(RHCOS)作为操作系统。

请注意，RHCOS 基于 Red Hat Enterprise Linux(RHEL) 9.2，并继承其所有硬件认证和要求。查看红帽企业 Linux 技术功能和限制。

19.2.3.2. 集群安装的最低资源要求

每台集群机器都必须满足以下最低要求：

表 19.2. 最低资源要求

机器	操作系统	vCPU [1]	虚拟内存	Storage	每秒输入/输出 (IOPS) [2]
bootstrap	RHCOS	2	16 GB	100 GB	300
Control plane（控制平面）	RHCOS	2	16 GB	100 GB	300
Compute	RHCOS	2	8 GB	100 GB	300

1. 当未启用并发多线程(SMT)或超线程时，一个 vCPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比例：（每个内核数的线程）× sockets = vCPU。

2. OpenShift Container Platform 和 Kubernetes 对磁盘性能敏感，建议使用更快的存储，特别是 control plane 节点上的 etcd。请注意，在许多云平台上，存储大小和 IOPS 可一起扩展，因此您可能需要过度分配存储卷来获取足够的性能。

注意

从 OpenShift Container Platform 版本 4.13 开始，RHCOS 基于 RHEL 版本 9.2，它更新了微架构要求。以下列表包含每个架构需要的最小指令集架构 (ISA)：

- x86-64 体系结构需要 x86-64-v2 ISA
- ARM64 架构需要 ARMv8.0-A ISA
- IBM Power 架构需要 Power 9 ISA
- s390x 架构需要 z14 ISA

如需更多信息，请参阅 RHEL 架构。

如果平台的实例类型满足集群机器的最低要求，则 OpenShift Container Platform 支持使用它。

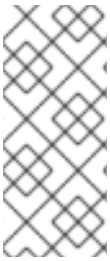
其他资源

- 优化存储

19.2.3.3. 最低 IBM Power 要求

您可以在以下 IBM® 硬件上安装 OpenShift Container Platform 版本 4.16：

- IBM Power®9 或 IBM Power®10 处理器系统



注意

OpenShift Container Platform 4.16 中弃用了所有 IBM Power®8 模型、IBM Power® AC922、IBM Power® IC922 和 IBM Power® LC922 的 RHCOS 功能。红帽建议您使用后续的硬件模型。

硬件要求

- 跨多个 PowerVM 服务器的六个逻辑分区(LPAR)

操作系统要求

- IBM Power®9 或 Power10 处理器系统的一个实例

在 IBM Power® 实例中设置：

- 用于 OpenShift Container Platform control plane 机器的三个 LPAR
- 用于 OpenShift Container Platform 计算机器的两个 LPAR
- 一个 LPAR 用于临时 OpenShift Container Platform bootstrap 机器

IBM Power 客户机虚拟机的磁盘存储

- 本地存储或由虚拟 I/O 服务器使用 vSCSI、NPIV(N-Port ID Virtualization)或 SSP（共享存

储池）置备的存储

PowerVM 客户机虚拟机的网络

- 专用物理适配器或 SR-IOV 虚拟功能
- 虚拟 I/O 服务器使用共享以太网适配器提供
- 虚拟 I/O 服务器使用 IBM® vNIC 虚拟化

存储/主内存

- OpenShift Container Platform control plane 机器需要 100 GB / 16 GB
- OpenShift Container Platform 计算机器需要 100 GB / 8 GB
- 临时 OpenShift Container Platform bootstrap 机器需要 100 GB / 16 GB

19.2.3.4. 推荐的 IBM Power 系统要求

硬件要求

- 跨多个 PowerVM 服务器的六个 LPAR

操作系统要求

- IBM Power®9 或 IBM Power®10 处理器的系统的一个实例

在 IBM Power® 实例中设置：

- 用于 OpenShift Container Platform control plane 机器的三个 LPAR
- 用于 OpenShift Container Platform 计算机器的两个 LPAR

- 一个 LPAR 用于临时 OpenShift Container Platform bootstrap 机器

IBM Power 客户机虚拟机的磁盘存储

- 本地存储或由虚拟 I/O 服务器使用 vSCSI、NPIV(N-Port ID Virtualization)或 SSP（共享存储池）置备的存储

PowerVM 客户机虚拟机的网络

- 专用物理适配器或 SR-IOV 虚拟功能
- 虚拟 I/O 服务器使用共享以太网适配器虚拟化
- 虚拟 I/O 服务器使用 IBM® vNIC 虚拟化

存储/主内存

- OpenShift Container Platform control plane 机器需要 120 GB / 32 GB
- OpenShift Container Platform 计算机器需要 120 GB / 32 GB
- 临时 OpenShift Container Platform bootstrap 机器需要 120 GB / 16 GB

19.2.3.5. 证书签名请求管理

在使用您置备的基础架构时，集群只能有限地访问自动机器管理，因此您必须提供一种在安装后批准集群证书签名请求 (CSR) 的机制。kube-controller-manager 只能批准 kubelet 客户端 CSR。machine-approver 无法保证使用 kubelet 凭证请求的提供证书的有效性，因为它不能确认是正确的机器发出了该请求。您必须决定并实施一种方法，以验证 kubelet 提供证书请求的有效性并进行批准。

19.2.3.6. 用户置备的基础架构对网络的要求

所有 Red Hat Enterprise Linux CoreOS(RHCOS)机器都需要在启动时在 initramfs 中配置联网，以获取它们的 Ignition 配置文件。

在初次启动过程中，机器需要 IP 地址配置，该配置通过 DHCP 服务器或静态设置，提供所需的引导

选项。建立网络连接后，机器会从 HTTP 或 HTTPS 服务器下载 Ignition 配置文件。然后，Ignition 配置文件用于设置每台机器的确切状态。Machine Config Operator 在安装后完成对机器的更多更改，如应用新证书或密钥。

建议使用 DHCP 服务器对集群机器进行长期管理。确保 DHCP 服务器已配置为向集群机器提供持久的 IP 地址、DNS 服务器信息和主机名。

注意

如果用户置备的基础架构没有 DHCP 服务，您可以在 RHCOS 安装时向节点提供 IP 网络配置和 DNS 服务器地址。如果要从 ISO 镜像安装，这些参数可作为引导参数传递。如需有关静态 IP 置备和高级网络选项的更多信息，请参阅 *安装 RHCOS 并启动 OpenShift Container Platform bootstrap 过程* 部分。

Kubernetes API 服务器必须能够解析集群机器的节点名称。如果 API 服务器和 worker 节点位于不同的区域中，您可以配置默认 DNS 搜索区域，以允许 API 服务器解析节点名称。另一种支持的方法是始终通过节点对象和所有 DNS 请求中的完全限定域名引用主机。

19.2.3.6.1. 通过 DHCP 设置集群节点主机名

在 Red Hat Enterprise Linux CoreOS(RHCOS)机器上，主机名是通过 NetworkManager 设置的。默认情况下，机器通过 DHCP 获取其主机名。如果主机名不是由 DHCP 提供，请通过内核参数或者其它方法进行静态设置，请通过反向 DNS 查找获取。反向 DNS 查找在网络初始化后进行，可能需要一些时间来解决。其他系统服务可以在此之前启动，并将主机名检测为 localhost 或类似的内容。您可以使用 DHCP 为每个集群节点提供主机名来避免这种情况。

另外，通过 DHCP 设置主机名可以绕过实施 DNS split-horizon 的环境中的手动 DNS 记录名称配置错误。

19.2.3.6.2. 网络连接要求

您必须配置机器之间的网络连接，以允许 OpenShift Container Platform 集群组件进行通信。每台机器都必须能够解析集群中所有其他机器的主机名。

本节详细介绍了所需的端口。

重要

在连接的 OpenShift Container Platform 环境中，所有节点都需要访问互联网才能为平台容器拉取镜像，并向红帽提供遥测数据。

表 19.3. 用于全机器到所有机器通信的端口

协议	port	描述
ICMP	N/A	网络可访问性测试
TCP	**1936**	指标
	9000-**9999**	主机级别的服务，包括端口 9**100**-**9101** 上的节点导出器，以及端口 **9099** 上的 Cluster Version Operator。
	10250-**10259**	Kubernetes 保留的默认端口
UDP	**4789**	VXLAN
	6081	Geneve
	9000-**9999**	主机级别的服务，包括端口 **9100**到**9101** 上的节点导出器。
	500	IPsec IKE 数据包
	4500	IPsec NAT-T 数据包
	123	UDP 端口 **123** 上的网络时间协议 (NTP) 如果配置了外部 NTP 时间服务器，需要打开 UDP 端口 **123**。
TCP/UDP	**30000**-**32767**	Kubernetes 节点端口
ESP	N/A	IPsec Encapsulating Security Payload(ESP)

表 19.4. 用于所有机器控制平面通信的端口

协议	port	描述
TCP	**6443**	Kubernetes API

表 19.5. control plane 机器用于 control plane 机器通信的端口

协议	port	描述
TCP	**2379-2380**	etcd 服务器和对等端口

用户置备的基础架构的 NTP 配置

OpenShift Container Platform 集群被配置为默认使用公共网络时间协议(NTP)服务器。如果要使用本地企业 NTP 服务器，或者集群部署在断开连接的网络中，您可以将集群配置为使用特定的时间服务器。如需更多信息，*请参阅配置 chrony 时间服务* 的文档。

如果 DHCP 服务器提供 NTP 服务器信息，Red Hat Enterprise Linux CoreOS(RHCOS)机器上的 chrony 时间服务会读取信息，并可以把时钟与 NTP 服务器同步。

其他资源

- **配置 chrony 时间服务**

19.2.3.7. 用户置备的 DNS 要求

在 OpenShift Container Platform 部署中，以下组件需要 DNS 名称解析：

- **The Kubernetes API**
- **OpenShift Container Platform 应用程序通配符**
- **bootstrap、control plane 和计算机器**

Kubernetes API、bootstrap 机器、control plane 机器和计算机器也需要反向 DNS 解析。

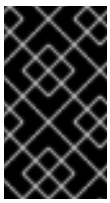
DNS A/AAAA 或 CNAME 记录用于名称解析，PTR 记录用于反向名称解析。反向记录很重要，因为 Red Hat Enterprise Linux CoreOS(RHCOS)使用反向记录为所有节点设置主机名，除非 DHCP 提供主机名。另外，反向记录用于生成 OpenShift Container Platform 需要操作的证书签名请求(CSR)。

注意

建议使用 DHCP 服务器为每个群集节点提供主机名。如需更多信息，*请参阅用户置备的基础架构部分的 DHCP 建议*。

用户置备的 OpenShift Container Platform 集群需要以下 DNS 记录，这些记录必须在安装前就位。在每个记录中，<cluster_name> 是集群名称，<base_domain> 是您在 install-config.yaml 文件中指定的基域。完整的 DNS 记录采用以下形式： <component>.<cluster_name>.<base_domain>.。

表 19.6. 所需的 DNS 记录

组件	记录	描述
Kubernetes API	**api.<cluster_name>. <base_domain>.**	DNS A/AAAA 或 CNAME 记录，以及用于标识 API 负载均衡器的 DNS PTR 记录。这些记录必须由集群外的客户端和集群中的所有节点解析。
	api-int.<cluster_name>. <base_domain>.	DNS A/AAAA 或 CNAME 记录，以及用于内部标识 API 负载均衡器的 DNS PTR 记录。这些记录必须可以从集群中的所有节点解析。  **重要** API 服务器必须能够根据 Kubernetes 中记录的主机名解析 worker 节点。如果 API 服务器无法解析节点名称，则代理的 API 调用会失败，且您无法从 pod 检索日志。
Routes	***.apps.<cluster_name>. <base_domain>.**	通配符 DNS A/AAAA 或 CNAME 记录，指向应用程序入口负载均衡器。应用程序入口负载均衡器以运行 Ingress Controller Pod 的机器为目标。默认情况下，Ingress Controller Pod 在计算机器上运行。这些记录必须由集群外的客户端和集群中的所有节点解析。 例如，console **-openshift-console.apps. <cluster_name>.<base_domain>** 用作到 OpenShift Container Platform 控制台的通配符路由。
bootstrap 机器	**bootstrap.<cluster_name>. <base_domain>.**	DNS A/AAAA 或 CNAME 记录，以及用于标识 bootstrap 机器的 DNS PTR 记录。这些记录必须由集群中的节点解析。
control plane 机器	**<control_plane><n>. <cluster_name>. <base_domain>.**	DNS A/AAAA 或 CNAME 记录，以识别 control plane 节点的每台机器。这些记录必须由集群中的节点解析。
计算机器	**<compute><n>. <cluster_name>. <base_domain>.**	DNS A/AAAA 或 CNAME 记录，用于识别 worker 节点的每台机器。这些记录必须由集群中的节点解析。

注意

在 OpenShift Container Platform 4.4 及更新的版本中，您不需要在 DNS 配置中指定 etcd 主机和 SRV 记录。

提示

您可以使用 dig 命令验证名称和反向名称解析。如需了解详细的 *验证步骤，请参阅为用户置备的基础架构验证 DNS 解析* 一节。

19.2.3.7.1. 用户置备的集群的 DNS 配置示例

本节提供 A 和 PTR 记录配置示例，它们满足了在用户置备的基础架构上部署 OpenShift Container Platform 的 DNS 要求。样本不是为选择一个 DNS 解决方案提供建议。

在这个示例中，集群名称为 ocp4，基域是 example.com。

用户置备的集群的 DNS A 记录配置示例

以下示例是 BIND 区域文件，其中显示了用户置备的集群中名称解析的 A 记录示例。

例 19.1. DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
  2019070700 ; serial
  3H  ; refresh (3 hours)
  30M  ; retry (30 minutes)
  2W  ; expiry (2 weeks)
  1W )  ; minimum (1 week)
 IN NS ns1.example.com.
 IN MX 10 smtp.example.com.
;
;
ns1.example.com.  IN A 192.168.1.5
smtp.example.com.  IN A 192.168.1.5
;
helper.example.com.  IN A 192.168.1.5
helper.ocp4.example.com. IN A 192.168.1.5
;
api.ocp4.example.com.  IN A 192.168.1.5 1
api-int.ocp4.example.com. IN A 192.168.1.5 2
;
*.apps.ocp4.example.com. IN A 192.168.1.5 3
;
```

```
bootstrap.ocp4.example.com. IN A 192.168.1.96 4
;
control-plane0.ocp4.example.com. IN A 192.168.1.97 5
control-plane1.ocp4.example.com. IN A 192.168.1.98 6
control-plane2.ocp4.example.com. IN A 192.168.1.99 7
;
compute0.ocp4.example.com. IN A 192.168.1.11 8
compute1.ocp4.example.com. IN A 192.168.1.7 9
;
;EOF
```

1

为 Kubernetes API 提供名称解析。记录引用 API 负载均衡器的 IP 地址。

2

为 Kubernetes API 提供名称解析。记录引用 API 负载均衡器的 IP 地址，用于内部集群通信。

3

为通配符路由提供名称解析。记录引用应用程序入口负载均衡器的 IP 地址。应用程序入口负载均衡器以运行 Ingress Controller Pod 的机器为目标。默认情况下，Ingress Controller Pod 在计算机器上运行。

注意

在这个示例中，将相同的负载均衡器用于 Kubernetes API 和应用入口流量。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。

4

为 bootstrap 机器提供名称解析。

5 6 7

为 control plane 机器提供名称解析。

8 9

为计算机器提供名称解析。

用户置备的集群的 DNS PTR 记录配置示例

以下示例 BIND 区域文件显示了用户置备的集群中反向名称解析的 PTR 记录示例。

例 19.2. 反向记录的 DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
  2019070700 ; serial
  3H  ; refresh (3 hours)
  30M  ; retry (30 minutes)
  2W  ; expiry (2 weeks)
  1W )  ; minimum (1 week)
 IN NS ns1.example.com.
;
5.1.168.192.in-addr.arpa. IN PTR api.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. IN PTR api-int.ocp4.example.com. 2
;
96.1.168.192.in-addr.arpa. IN PTR bootstrap.ocp4.example.com. 3
;
97.1.168.192.in-addr.arpa. IN PTR control-plane0.ocp4.example.com. 4
98.1.168.192.in-addr.arpa. IN PTR control-plane1.ocp4.example.com. 5
99.1.168.192.in-addr.arpa. IN PTR control-plane2.ocp4.example.com. 6
;
11.1.168.192.in-addr.arpa. IN PTR compute0.ocp4.example.com. 7
7.1.168.192.in-addr.arpa. IN PTR compute1.ocp4.example.com. 8
;
;EOF
```

1 为 Kubernetes API 提供反向 DNS 解析。PTR 记录引用 API 负载均衡器的记录名称。

2 为 Kubernetes API 提供反向 DNS 解析。PTR 记录引用 API 负载均衡器的记录名称，用于内部集群通信。

3 为 bootstrap 机器提供反向 DNS 解析。

4 5 6 为 control plane 机器提供反向 DNS 解析。

7 8

为计算机器提供反向 DNS 解析。

注意

OpenShift Container Platform 应用程序通配符不需要 PTR 记录。

19.2.3.8. 用户置备的基础架构的负载均衡要求

在安装 OpenShift Container Platform 前，您必须置备 API 和应用程序入口负载均衡基础架构。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。

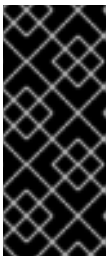
注意

如果要使用 Red Hat Enterprise Linux (RHEL) 实例部署 API 和应用程序入口负载均衡器，您必须单独购买 RHEL 订阅。

负载平衡基础架构必须满足以下要求：

1. API 负载均衡器 ：提供一个通用端点，供用户（包括人工和机器）与平台交互和配置。配置以下条件：

   - 仅第 4 层负载均衡.这可被称为 Raw TCP 或 SSL Passthrough 模式。
   - 无状态负载平衡算法。这些选项根据负载均衡器的实施而有所不同。



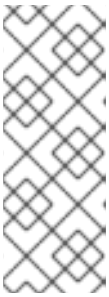
重要

不要为 API 负载均衡器配置会话持久性。为 Kubernetes API 服务器配置会话持久性可能会导致出现过量 OpenShift Container Platform 集群应用程序流量，以及过量的在集群中运行的 Kubernetes API。

在负载均衡器的前端和后端配置以下端口：

表 19.7. API 负载均衡器

port	后端机器（池成员）	internal	外部	描述
6443	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。您必须为 API 服务器健康检查探测配置 **/readyz** 端点。	X	X	Kubernetes API 服务器
22623	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。	X		机器配置服务器



注意

负载均衡器必须配置为，从 API 服务器关闭 **/readyz** 端点到从池中移除 API 服务器实例时最多需要 30 秒。在 **/readyz** 返回错误或健康后的时间范围内，端点必须被删除或添加。每 5 秒或 10 秒探测一次，有两个成功请求处于健康状态，三个成为不健康的请求是经过良好测试的值。

2. 应用程序入口负载均衡器 ：为应用程序流量从集群外部流提供入口点。OpenShift Container Platform 集群需要正确配置入口路由器。

配置以下条件：

- 仅第 4 层负载均衡.这可被称为 **Raw TCP 或 SSL Passthrough 模式。**
- 建议根据可用选项以及平台上托管的应用程序类型，使用基于连接的或基于会话的持久性。

提示

如果应用程序入口负载均衡器可以看到客户端的真实 IP 地址，启用基于 IP 的会话持久性可以提高使用端到端 TLS 加密的应用程序的性能。

在负载均衡器的前端和后端配置以下端口：

表 19.8. 应用程序入口负载均衡器

port	后端机器（池成员）	internal	外部	描述
443	默认情况下，运行 Ingress Controller Pod、计算或 worker 的机器。	X	X	HTTPS 流量
80	默认情况下，运行 Ingress Controller Pod、计算或 worker 的机器。	X	X	HTTP 流量

注意

如果要部署一个带有零计算节点的三节点集群，**Ingress Controller Pod** 在 **control plane** 节点上运行。在三节点集群部署中，您必须配置应用程序入口负载均衡器，将 **HTTP** 和 **HTTPS** 流量路由到 **control plane** 节点。

19.2.3.8.1. 用户置备的集群的负载均衡器配置示例

本节提供了一个满足用户置备集群的负载均衡要求的 **API** 和应用程序入口负载均衡器配置示例。示例是 **HAProxy** 负载均衡器的 **/etc/haproxy/haproxy.cfg** 配置。这个示例不是为选择一个负载平衡解决方案提供建议。

在这个示例中，将相同的负载均衡器用于 **Kubernetes API** 和应用入口流量。在生产环境中，您可以单独部署 **API** 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。

注意

如果您使用 **HAProxy** 作为负载均衡器，并且 **SELinux** 设置为 **enforcing**，您必须通过运行 **setsebool -P haproxy_connect_any=1** 来确保 **HAProxy** 服务可以绑定到配置的 **TCP** 端口。

例 19.3. API 和应用程序入口负载均衡器配置示例

```
global
  log         127.0.0.1 local2
  pidfile     /var/run/haproxy.pid
  maxconn     4000
  daemon
defaults
  mode                    http
  log                     global
```

```
  option                  dontlognull
  option http-server-close
  option                  redispatch
  retries                 3
  timeout http-request    10s
  timeout queue           1m
  timeout connect         10s
  timeout client          1m
  timeout server          1m
  timeout http-keep-alive 10s
  timeout check           10s
  maxconn                 3000
listen api-server-6443 (1)
  bind *:6443
  mode tcp
  option  httpchk GET /readyz HTTP/1.0
  option  log-health-checks
  balance roundrobin
  server bootstrap bootstrap.ocp4.example.com:6443 verify none check check-ssl inter 10s
fall 2 rise 3 backup (2)
  server master0 master0.ocp4.example.com:6443 weight 1 verify none check check-ssl
inter 10s fall 2 rise 3
  server master1 master1.ocp4.example.com:6443 weight 1 verify none check check-ssl
inter 10s fall 2 rise 3
  server master2 master2.ocp4.example.com:6443 weight 1 verify none check check-ssl
inter 10s fall 2 rise 3
listen machine-config-server-22623 (3)
  bind *:22623
  mode tcp
  server bootstrap bootstrap.ocp4.example.com:22623 check inter 1s backup (4)
  server master0 master0.ocp4.example.com:22623 check inter 1s
  server master1 master1.ocp4.example.com:22623 check inter 1s
  server master2 master2.ocp4.example.com:22623 check inter 1s
listen ingress-router-443 (5)
  bind *:443
  mode tcp
  balance source
  server compute0 compute0.ocp4.example.com:443 check inter 1s
  server compute1 compute1.ocp4.example.com:443 check inter 1s
listen ingress-router-80 (6)
  bind *:80
  mode tcp
  balance source
  server compute0 compute0.ocp4.example.com:80 check inter 1s
  server compute1 compute1.ocp4.example.com:80 check inter 1s
```

(1)

端口 6443 处理 Kubernetes API 流量并指向 control plane 机器。

(2) (4)

bootstrap 条目必须在 OpenShift Container Platform 集群安装前就位，且必须在 bootstrap 过程完成后删除它们。

3

端口 22623 处理机器配置服务器流量并指向 control plane 机器。

5

端口 443 处理 HTTPS 流量，并指向运行 Ingress Controller pod 的机器。默认情况下，Ingress Controller Pod 在计算机器上运行。

6

端口 80 处理 HTTP 流量，并指向运行 Ingress Controller pod 的机器。默认情况下，Ingress Controller Pod 在计算机器上运行。

注意

如果要部署一个带有零计算节点的三节点集群，Ingress Controller Pod 在 control plane 节点上运行。在三节点集群部署中，您必须配置应用程序入口负载均衡器，将 HTTP 和 HTTPS 流量路由到 control plane 节点。

提示

如果您使用 HAProxy 作为负载均衡器，您可以通过在 HAProxy 节点上运行 **netstat -nltupe** 来检查 haproxy 进程是否在侦听端口 6443、22623、443 和 80。

19.2.4. 准备用户置备的基础架构

在用户置备的基础架构上安装 OpenShift Container Platform 之前，您必须准备底层基础架构。

本节详细介绍了设置集群基础架构以准备 OpenShift Container Platform 安装所需的高级别步骤。这包括为您的集群节点配置 IP 网络和网络连接，通过防火墙启用所需的端口，以及设置所需的 DNS 和负载均衡基础架构。

准备后，集群基础架构必须满足 *带有用户置备的基础架构部分的集群要求*。

先决条件

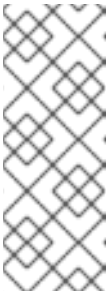
- 您已参阅 OpenShift Container Platform 4.x Tested Integrations 页面。
- 您已查看了 *具有用户置备基础架构的集群要求部分中详述的基础架构* 要求。

流程

1. 如果您使用 DHCP 向集群节点提供 IP 网络配置，请配置 DHCP 服务。

   a. 将节点的持久 IP 地址添加到您的 DHCP 服务器配置。在您的配置中，将相关网络接口的 MAC 地址与每个节点的预期 IP 地址匹配。

   b. 当您使用 DHCP 为集群机器配置 IP 寻址时，机器还通过 DHCP 获取 DNS 服务器信息。定义集群节点通过 DHCP 服务器配置使用的持久性 DNS 服务器地址。

   注意

   如果没有使用 DHCP 服务，则必须在 RHCOS 安装时为节点提供 IP 网络配置和 DNS 服务器地址。如果要从 ISO 镜像安装，这些参数可作为引导参数传递。如需有关静态 IP 置备和高级网络选项的更多信息，请参阅 *安装 RHCOS 并启动 OpenShift Container Platform bootstrap 过程* 部分。

   c. 在 DHCP 服务器配置中定义集群节点的主机名。有关 *主机名注意事项的详情，请参阅通过 DHCP 设置集群节点* 主机名部分。

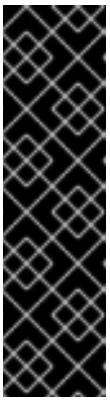
   

   注意

   如果没有使用 DHCP 服务，集群节点可以通过反向 DNS 查找来获取其主机名。

2. 确保您的网络基础架构提供集群组件之间所需的网络连接。有关 *要求的详情，请参阅用户置备的基础架构* 的网络要求部分。

3. 将防火墙配置为启用 OpenShift Container Platform 集群组件进行通信所需的端口。*如需有关所需端口的详细信息，请参阅用户置备的基础架构* 部分的网络要求。



重要

默认情况下，OpenShift Container Platform 集群可以访问端口 1936，因为每个 control plane 节点都需要访问此端口。

避免使用 Ingress 负载均衡器公开此端口，因为这样做可能会导致公开敏感信息，如统计信息和指标（与 Ingress Controller 相关的统计信息和指标）。

4. 为集群设置所需的 DNS 基础架构。

   a. 为 Kubernetes API、应用程序通配符、bootstrap 机器、control plane 机器和计算机器配置 DNS 名称解析。

   b. 为 Kubernetes API、bootstrap 机器、control plane 机器和计算机器配置反向 DNS 解析。

   如需有关 *OpenShift Container Platform DNS 要求的更多信息，请参阅用户置备* DNS 要求部分。

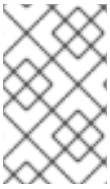
5. 验证您的 DNS 配置。

   a. 从安装节点，针对 Kubernetes API 的记录名称、通配符路由和集群节点运行 DNS 查找。验证响应中的 IP 地址是否与正确的组件对应。

   b. 从安装节点，针对负载均衡器和集群节点的 IP 地址运行反向 DNS 查找。验证响应中的记录名称是否与正确的组件对应。

   有关详细的 *DNS 验证步骤，请参阅用户置备的基础架构*验证 DNS 解析部分。

6. 置备所需的 API 和应用程序入口负载平衡基础架构。有关 *要求的更多信息，请参阅用户置备的基础架构的负载平衡* 要求部分。

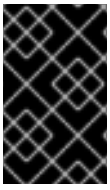


注意

某些负载平衡解决方案要求在初始化负载平衡之前，对群集节点进行 DNS 名称解析。

19.2.5. 验证用户置备的基础架构的 DNS 解析

您可以在在用户置备的基础架构上安装 OpenShift Container Platform 前验证 DNS 配置。



重要

本节中详述的验证步骤必须在安装集群前成功。

先决条件

- 已为您的用户置备的基础架构配置了所需的 DNS 记录。

流程

1. 从安装节点，针对 Kubernetes API 的记录名称、通配符路由和集群节点运行 DNS 查找。验证响应中包含的 IP 地址是否与正确的组件对应。

   a. 对 Kubernetes API 记录名称执行查询。检查结果是否指向 API 负载均衡器的 IP 地址：

   ```
   $ dig +noall +answer @<nameserver_ip> api.<cluster_name>.<base_domain> 1
   ```

   1 将 <nameserver_ip> 替换为 nameserver 的 IP 地址，<cluster_name> 替换为您的集群名称，<base_domain> 替换为您的基本域名。

   输出示例

   ```
   api.ocp4.example.com.  604800 IN A 192.168.1.5
   ```

b.

对 Kubernetes 内部 API 记录名称执行查询。检查结果是否指向 API 负载均衡器的 IP 地址：

```
$ dig +noall +answer @<nameserver_ip> api-int.<cluster_name>.<base_domain>
```

输出示例

```
api-int.ocp4.example.com.  604800 IN A 192.168.1.5
```

c.

测试 *.apps.<cluster_name>.<base_domain> DNS 通配符查找示例。所有应用程序通配符查询都必须解析为应用程序入口负载均衡器的 IP 地址：

```
$ dig +noall +answer @<nameserver_ip> random.apps.<cluster_name>.
<base_domain>
```

输出示例

```
random.apps.ocp4.example.com.  604800 IN A 192.168.1.5
```

注意

在示例中，将相同的负载均衡器用于 Kubernetes API 和应用程序入口流量。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。

您可以使用另一个通配符值替换 random。例如，您可以查询到 OpenShift Container Platform 控制台的路由：

```
$ dig +noall +answer @<nameserver_ip> console-openshift-console.apps.
<cluster_name>.<base_domain>
```

输出示例

```
console-openshift-console.apps.ocp4.example.com. 604800 IN A 192.168.1.5
```

d. 针对 bootstrap DNS 记录名称运行查询。检查结果是否指向 bootstrap 节点的 IP 地址：

```
$ dig +noall +answer @<nameserver_ip> bootstrap.<cluster_name>.
<base_domain>
```

输出示例

```
bootstrap.ocp4.example.com.  604800 IN A 192.168.1.96
```

e. 使用此方法对 control plane 和计算节点的 DNS 记录名称执行查找。检查结果是否与每个节点的 IP 地址对应。

2. 从安装节点，针对负载均衡器和集群节点的 IP 地址运行反向 DNS 查找。验证响应中包含的记录名称是否与正确的组件对应。

a. 对 API 负载均衡器的 IP 地址执行反向查找。检查响应是否包含 Kubernetes API 和 Kubernetes 内部 API 的记录名称：

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.5
```

输出示例

```
5.1.168.192.in-addr.arpa. 604800 IN PTR api-int.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. 604800 IN PTR api.ocp4.example.com. 2
```

1

为 Kubernetes 内部 API 提供记录名称。

2

为 Kubernetes API 提供记录名称。

注意

OpenShift Container Platform 应用程序通配符不需要 PTR 记录。针对应用程序入口负载均衡器的 IP 地址解析反向 DNS 解析不需要验证步骤。

b. 对 bootstrap 节点的 IP 地址执行反向查找。检查结果是否指向 bootstrap 节点的 DNS 记录名称：

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.96
```

输出示例

```
96.1.168.192.in-addr.arpa. 604800 IN PTR bootstrap.ocp4.example.com.
```

c. 使用此方法对 control plane 和计算节点的 IP 地址执行反向查找。检查结果是否与每个节点的 DNS 记录名称对应。

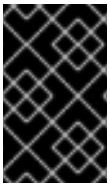
19.2.6. 为集群节点 SSH 访问生成密钥对

在 OpenShift Container Platform 安装过程中，您可以为安装程序提供 SSH 公钥。密钥通过它们的 Ignition 配置文件传递给 Red Hat Enterprise Linux CoreOS(RHCOS)节点，用于验证对节点的 SSH 访问。密钥添加到每个节点上 core 用户的 ~/.ssh/authorized_keys 列表中，这将启用免密码身份验证。

将密钥传递给节点后，您可以使用密钥对作为用户 核心 通过 SSH 连接到 RHCOS 节点。若要通过

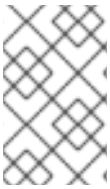
SSH 访问节点，必须由 SSH 为您的本地用户管理私钥身份。

如果要通过 SSH 连接到集群节点来执行安装调试或灾难恢复，则必须在安装过程中提供 SSH 公钥。./openshift-install gather 命令还需要在集群节点上设置 SSH 公钥。



重要

不要在生产环境中跳过这个过程，在生产环境中需要灾难恢复和调试。



注意

您必须使用本地密钥，而不是使用特定平台方法配置 的密钥，如 AWS 密钥对。

流程

1. 如果您在本地计算机上没有可用于在集群节点上进行身份验证的现有 SSH 密钥对，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 ~/.ssh/id_ed25519。如果您已有密钥对，请确保您的公钥位于 ~/.ssh 目录中。

注意

如果您计划在 x86_64、ppc64le 和 s390x 架构上安装使用 RHEL 加密库（这些加密库已提交给 NIST 用于 FIPS 140-2/140-3 验证）的 OpenShift Container Platform 集群，则不要创建使用 ed25519 算法的密钥。相反，创建一个使用 rsa 或 ecdsa 算法的密钥。

2. 查看公共 SSH 密钥：

```
$ cat <path>/<file_name>.pub
```

例如，运行以下命令来查看 ~/.ssh/id_ed25519.pub 公钥：

```
$ cat ~/.ssh/id_ed25519.pub
```

3. 将 SSH 私钥身份添加到本地用户的 SSH 代理（如果尚未添加）。在集群节点上，或者要使用 ./openshift-install gather 命令，需要对该密钥进行 SSH 代理管理，才能在集群节点上进行免密码 SSH 身份验证。

注意

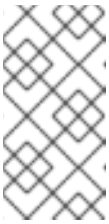
在某些发行版中，自动管理默认 SSH 私钥身份，如 ~/.ssh/id_rsa 和 ~/.ssh/id_dsa。

a. 如果 ssh-agent 进程尚未为您的本地用户运行，请将其作为后台任务启动：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```



注意

如果集群处于 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

4. 将 SSH 私钥添加到 ssh-agent ：

```
$ ssh-add <path>/<file_name> 1
```

1 指定 SSH 私钥的路径和文件名，如 ~/.ssh/id_ed25519.pub

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

后续步骤

- 安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

19.2.7. 获取安装程序

在安装 OpenShift Container Platform 前，将安装文件下载到您用于安装的主机上。

先决条件

- 您有一台运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB。

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐户，请使用您的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入到安装类型的页面，下载与您的主机操作系统和架构对应的安装程序，并将该文件放在您要存储安装配置文件的目录中。

重要

安装程序会在用来安装集群的计算机上创建几个文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。

重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，请为特定云供应商完成 OpenShift Container Platform 卸载流程。

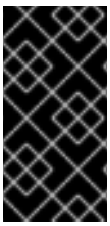
4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret。此 pull secret 允许您与所含授权机构提供的服务进行身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

19.2.8. 安装 OpenShift CLI

您可以安装 OpenShift CLI(oc)来使用命令行界面与 OpenShift Container Platform 进行交互。您可以在 Linux、Windows 或 macOS 上安装 oc。



重要

如果安装了旧版本的 oc，则可能无法使用 OpenShift Container Platform 4.16 中的所有命令。下载并安装新版本的 oc。

在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI(oc)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 产品变体 下拉列表中选择架构。
3. 从 版本 下拉列表中选择适当的版本。

4. 点 OpenShift v4.16 Linux Client 条目旁的 Download Now 来保存文件。

5. 解包存档：

```
$ tar xvf <file>
```

6. 将 oc 二进制文件放到 PATH 中的目录中。

   要查看您的 PATH，请执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 oc 命令：

```
$ oc <command>
```

在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI(oc)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。

2. 从 版本 下拉列表中选择适当的版本。

3. 点 OpenShift v4.16 Windows Client 条目旁的 Download Now 来保存文件。

4. 使用 ZIP 程序解压存档。

5. 将 oc 二进制文件移到 PATH 中的目录中。

要查看您的 PATH，请打开命令提示并执行以下命令：

```
C:\> path
```

验证

- 安装 OpenShift CLI 后，可以使用 oc 命令：

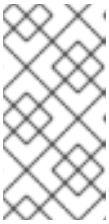
```
C:\> oc <command>
```

在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI(oc)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 版本 下拉列表中选择适当的版本。
3. 点 OpenShift v4.16 macOS Client 条目旁的 Download Now 来保存文件。



注意

对于 macOS arm64，请选择 OpenShift v4.16 macOS arm64 Client 条目。

4. 解包和解压存档。
5. 将 oc 二进制文件移到 PATH 的目录中。

要查看您的 PATH，请打开终端并执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 oc 命令：

```
$ oc <command>
```

19.2.9. 手动创建安装配置文件

安装集群要求您手动创建安装配置文件。

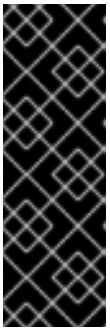
先决条件

- 您在本地机器上有一个 SSH 公钥来提供给安装程序。该密钥将用于在集群节点上进行 SSH 身份验证，以进行调试和灾难恢复。
- 已获取 OpenShift Container Platform 安装程序和集群的 pull secret。

流程

1. 创建一个安装目录来存储所需的安装资产：

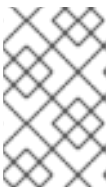
```
$ mkdir <installation_directory>
```



重要

您必须创建一个目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

2. 自定义提供的 install-config.yaml 文件模板示例，并将其保存在 <installation_directory> 中。



注意

此配置文件必须命名为 install-config.yaml。

3. 备份 install-config.yaml 文件，以便您可以使用它安装多个集群。

重要

install-config.yaml 文件会在安装过程的下一步中使用。现在必须备份它。

其他资源

- IBM Power® 的安装配置参数。

19.2.9.1. IBM Power 的 install-config.yaml 文件示例

您可以自定义 install-config.yaml 文件，以指定有关 OpenShift Container Platform 集群平台的更多详情，或修改所需参数的值。

```
apiVersion: v1
baseDomain: example.com (1)
compute: (2)
- hyperthreading: Enabled (3)
  name: worker
  replicas: 0 (4)
  architecture: ppc64le
controlPlane: (5)
  hyperthreading: Enabled (6)
  name: master
  replicas: 3 (7)
  architecture: ppc64le
metadata:
  name: test (8)
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14 (9)
    hostPrefix: 23 (10)
  networkType: OVNKubernetes (11)
  serviceNetwork: (12)
  - 172.30.0.0/16
platform:
  none: {} (13)
fips: false (14)
pullSecret: '{"auths": ...}' (15)
sshKey: 'ssh-ed25519 AAAA...' (16)
```

(1)

集群的基域。所有 DNS 记录都必须是这个基域的子域，并包含集群名称。

2 5

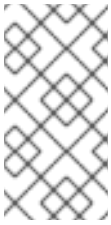
controlPlane 部分是一个单个映射，但 compute 部分是一系列映射。为满足不同数据结构的要求，compute 部分的第一行必须以连字符 - 开头，controlPlane 部分 的第一行则不以连字符开头。仅使用一个 control plane 池。

3 6

不支持并发多线程 (SMT)。

4

在用户置备的基础架构上安装 OpenShift Container Platform 时，必须将这个值设置为 0。在安装程序置备的安装中，参数控制集群为您创建和管理的计算机器数量。在用户置备的安装中，您必须在完成集群安装前手动部署计算机器。



注意

如果要安装一个三节点集群，在安装 Red Hat Enterprise Linux CoreOS(RHCOS)机器时不要部署任何计算机器。

7

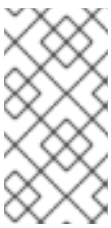
您添加到集群的 control plane 机器数量。由于集群使用这些值作为集群中的 etcd 端点数量，所以该值必须与您部署的 control plane 机器数量匹配。

8

您在 DNS 记录中指定的集群名称。

9

从中分配 Pod IP 地址的 IP 地址块。此块不得与现有物理网络重叠。这些 IP 地址用于 pod 网络。如果需要从外部网络访问 pod，您必须配置负载均衡器和路由器来管理流量。



注意

类 E CIDR 范围被保留以供以后使用。要使用 Class E CIDR 范围，您必须确保您的网络环境接受 Class E CIDR 范围内的 IP 地址。

10

分配给每个节点的子网前缀长度。例如，如果 hostPrefix 设为 23，则每个节点从 given cidr 中分配 a /23 子网，这样就能有 510（2^(32 - 23)- 2）个 pod IP 地址。如果需要从外部网络访问节点，请配置负载均衡器和路由器来管理流量。

11

要安装的集群网络插件。默认值 OVNKubernetes 是唯一支持的值。

12

用于服务 IP 地址的 IP 地址池。您只能输入一个 IP 地址池。此块不得与现有物理网络重叠。如果您需要从外部网络访问服务，请配置负载均衡器和路由器来管理流量。

13

您必须将平台设置为 none。您无法为 IBM Power® 基础架构提供额外的平台配置变量。

重要

使用平台类型 none 安装的集群无法使用一些功能，如使用 Machine API 管理计算机器。即使附加到集群的计算机器安装在通常支持该功能的平台上，也会应用这个限制。在安装后无法更改此参数。

14

是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 Red Hat Enterprise Linux (RHEL) 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅在 FIPS 模式中安装该系统。

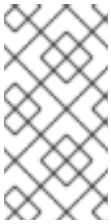
当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS)时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。

15

Red Hat OpenShift Cluster Manager 的 pull secret。此 pull secret 允许您与所含授权机构提供的服务进行身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

16

Red Hat Enterprise Linux CoreOS(RHCOS)中 core 用户的 SSH 公钥。



注意

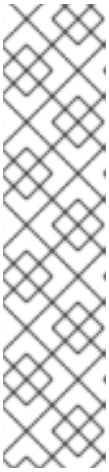
对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 ssh-agent 进程使用的 SSH 密钥。

19.2.9.2. 在安装过程中配置集群范围的代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 install-config.yaml 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 install-config.yaml 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 Proxy 对象的 spec.noProxy 字段中添加站点来绕过代理。



注意

Proxy 对象 status.noProxy 字段使用安装配置中的 networking.machineNetwork[].cidr、networking.clusterNetwork[].cidr 和 networking.serviceNetwork[] 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，Proxy 对象 status.noProxy 字段也会使用实例元数据端点填充(169.254.169.254)。

流程

1. 编辑 install-config.yaml 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

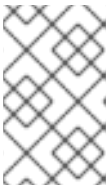
1 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 http。

2 用于创建集群外 HTTPS 连接的代理 URL。

3 要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 . 以仅匹配子域。例如，.y.com 匹配 x.y.com，但不匹配 y.com。使用 * 绕过所有目的地的代理。

4 如果提供，安装程序会在 openshift-config 命名空间中生成名为 user-ca-bundle 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 trusted-ca-bundle 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， Proxy 对象的 trustedCA 字段中也会引用此配置映射。additionalTrustBundle 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

5 可选：决定 Proxy 对象的配置以引用 trustedCA 字段中 user-ca-bundle 配置映射的策略。允许的值是 Proxyonly 和 Always。仅在配置了 http/https 代理时，使用 Proxyonly 引用 user-ca-bundle 配置映射。使用 Always 始终引用 user-ca-bundle 配置映射。默认值为 Proxyonly。



注意

安装程序不支持代理的 readinessEndpoints 字段。

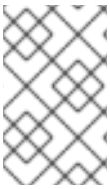
注意

如果安装程序超时，重启并使用安装程序的 wait-for 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. **保存该文件并在安装 OpenShift Container Platform 时引用。**

安装程序会创建一个名为 cluster 的集群范围代理，该代理 使用 提供的 install-config.yaml 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 cluster Proxy 对象，但它会有一个空 spec。



注意

只支持名为 cluster 的 Proxy 对象，且无法创建额外的代理。

19.2.9.3. 配置三节点集群

另外，您可以在只由三台 control plane 机器组成的裸机集群中部署零台计算机器。这为集群管理员和开发人员提供了更小、效率更高的集群，用于测试、开发和生产。

在三节点 OpenShift Container Platform 环境中，三台 control plane 机器可以调度，这意味着应用程序工作负载被调度到它们上运行。

先决条件

- **您有一个现有的 install-config.yaml 文件。**

流程

- **确保 install-config.yaml 文件中的计算副本数量设置为 0，如以下 计算 小节所示：**

```
compute:
- name: worker
  platform: {}
  replicas: 0
```

注意

在用户置备的基础架构上安装 OpenShift Container Platform 时，无论您要部署的计算机器数量有多少，您必须将计算机器的 replicas 参数值设置为 0。在安装程序置备的安装中，参数控制集群为您创建和管理的计算机器数量。这不适用于手动部署计算机器的用户置备安装。

对于三节点集群安装，请按照以下步骤执行：

- 如果要部署一个带有零计算节点的三节点集群，Ingress Controller Pod 在 control plane 节点上运行。在三节点集群部署中，您必须配置应用程序入口负载均衡器，将 HTTP 和 HTTPS 流量路由到 control plane 节点。如需更多信息，*请参阅用户置备的基础架构的负载平衡要求* 部分。

- 在以下步骤中创建 Kubernetes 清单文件时，请确保 <installation_directory>/manifests/cluster-scheduler-02-config.yml 文件中的 mastersSchedulable 参数被设置为 true。这可让应用程序工作负载在 control plane 节点上运行。

- 在创建 Red Hat Enterprise Linux CoreOS(RHCOS)机器时，不要部署任何计算节点。

19.2.10. Cluster Network Operator 配置

集群网络的配置作为 Cluster Network Operator(CNO)配置的一部分指定，并存储在名为 cluster 的自定义资源(CR)对象中。CR 指定 operator.openshift.io API 组中的 Network API 的字段。

CNO 配置在集群安装过程中从 Network.config.openshift.io API 组中的 Network API 继承以下字段：

clusterNetwork

从中分配 Pod IP 地址的 IP 地址池。

serviceNetwork

服务的 IP 地址池.

defaultNetwork.type

集群网络插件。OVNKubernetes 是安装期间唯一支持的插件。

您可以通过在名为 cluster 的 CNO 对象中设置 defaultNetwork 对象的字段来为集群指定集群网络插件配置。

19.2.10.1. Cluster Network Operator 配置对象

下表中描述了 Cluster Network Operator(CNO)的字段：

表 19.9. Cluster Network Operator 配置对象

字段	类型	描述
metadata.name	**字符串**	CNO 对象的名称。这个名称始终是 **集群**。

spec.clusterNetwork	**array**	用于指定从哪些 IP 地址块分配 Pod IP 地址以及集群中每个节点的子网前缀长度的列表。例如：
		`spec:`
		`  clusterNetwork:`
		`  - cidr: 10.128.0.0/19`
		`  hostPrefix: 23`
		`  - cidr: 10.128.32.0/19`
		`  hostPrefix: 23`

spec.serviceNetwork	**array**	服务的 IP 地址块。OpenShift SDN 和 OVN-Kubernetes 网络插件只支持服务网络的一个 IP 地址块。例如：
		`spec:`
		`  serviceNetwork:`
		`  - 172.30.0.0/14`
		您只能在创建清单前在 **install-config.yaml** 文件中自定义此字段。该值在清单文件中是只读的。

spec.defaultNetwork	**object**	为集群网络配置网络插件。
spec.kubeProxyConfig	**object**	此对象的字段指定 kube-proxy 配置。如果使用 OVN-Kubernetes 集群网络供应商，则 kube-proxy 配置不会起作用。

defaultNetwork 对象配置

下表列出了 defaultNetwork 对象的值：

表 19.10. defaultNetwork 对象

字段	类型	描述
type	**字符串**	**OVNKubernetes**。Red Hat OpenShift Networking 网络插件在安装过程中被选择。此值在集群安装后无法更改。 **注意** OpenShift Container Platform 默认使用 OVN-Kubernetes 网络插件。OpenShift SDN 不再作为新集群的安装选择提供。
ovnKubernetesConfig	**object**	此对象仅对 OVN-Kubernetes 网络插件有效。

配置 OVN-Kubernetes 网络插件

下表描述了 OVN-Kubernetes 网络插件的配置字段：

表 19.11. ovnKubernetesConfig object

字段	类型	描述
mtu	**integer**	Geneve（通用网络虚拟化封装）覆盖网络的最大传输单元 (MTU)。这根据主网络接口的 MTU 自动探测。您通常不需要覆盖检测到的 MTU。 如果自动探测的值不是您期望的值，请确认节点上主网络接口上的 MTU 是否正确。您不能使用这个选项更改节点上主网络接口的 MTU 值。 如果集群中不同节点需要不同的 MTU 值，则必须将此值设置为**比** 集群中的最低 MTU 值小 100。例如，如果集群中的某些节点的 MTU 为 **9001**，而某些节点的 MTU 为 **1500**，则必须将此值设置为 **1400**。
genevePort	**integer**	用于所有 Geneve 数据包的端口。默认值为 **6081**。此值在集群安装后无法更改。
ipsecConfig	**object**	指定用于自定义 IPsec 配置的配置对象。
ipv4	**object**	为 IPv4 设置指定配置对象。

字段	类型	描述
ipv6	**object**	为 IPv6 设置指定配置对象。
policyAuditConfig	**object**	指定用于自定义网络策略审计日志的配置对象。如果未设置，则使用默认的审计日志设置。
gatewayConfig	**object**	可选：指定一个配置对象来自定义如何将出口流量发送到节点网关。 **注意** 在迁移出口流量时，工作负载和服务流量会受到一定影响，直到 Cluster Network Operator (CNO) 成功推出更改。

表 19.12. ovnKubernetesConfig.ipv4 object

字段	类型	描述
internalTransitSwitchSubnet	字符串	如果您的现有网络基础架构与 **100.88.0.0/16** IPv4 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。启用东西流量的分布式传输交换机的子网。此子网不能与 OVN-Kubernetes 或主机本身使用的任何其他子网重叠。必须足够大，以适应集群中的每个节点一个 IP 地址。 默认值为 **100.88.0.0/16**。
internalJoinSubnet	字符串	如果您的现有网络基础架构与 **100.64.0.0/16** IPv4 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。您必须确保 IP 地址范围没有与 OpenShift Container Platform 安装使用的任何其他子网重叠。IP 地址范围必须大于可添加到集群的最大节点数。例如，如果 **clusterNetwork.cidr** 值为 **10.128.0.0/14**，并且 **clusterNetwork.hostPrefix** 值为 **/23**，则最大节点数量为 **2^(23-14)=512**。 默认值为 **100.64.0.0/16**。

表 19.13. ovnKubernetesConfig.ipv6 object

字段	类型	描述
internalTransitSwitchSubnet	字符串	如果您的现有网络基础架构与 **fd97::/64** IPv6 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。启用东西流量的分布式传输交换机的子网。此子网不能与 OVN-Kubernetes 或主机本身使用的任何其他子网重叠。必须足够大，以适应集群中的每个节点一个 IP 地址。 默认值为 **fd97::/64**。

字段	类型	描述
internalJoinSubnet	字符串	如果您的现有网络基础架构与 **fd98::/64** IPv6 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。您必须确保 IP 地址范围没有与 OpenShift Container Platform 安装使用的任何其他子网重叠。IP 地址范围必须大于可添加到集群的最大节点数。 默认值为 **fd98::/64**。

表 19.14. policyAuditConfig object

字段	类型	描述
rateLimit	整数	每个节点每秒生成一次的消息数量上限。默认值为每秒 **20** 条消息。
maxFileSize	整数	审计日志的最大大小，以字节为单位。默认值为 **50000000** 或 50 MB。
maxLogFiles	整数	保留的日志文件的最大数量。
目的地	字符串	以下附加审计日志目标之一： **libc** 主机上的 journald 进程的 libc **syslog（）** 函数。 **UDP:<host>:<port>** 一个 syslog 服务器。将 **<host>:<port>** 替换为 **syslog 服务器的主机** 和端口。 **Unix:<file>** 由 **<file>** 指定的 Unix 域套接字文件。 **null** 不要将审计日志发送到任何其他目标。
syslogFacility	字符串	syslog 工具，如 as **kern**，如 RFC5424 定义。默认值为 **local0**。

表 19.15. gatewayConfig object

字段	类型	描述

字段	类型	描述
routingViaHost	**布尔值**	将此字段设置为 **true**，将来自 pod 的出口流量发送到主机网络堆栈。对于依赖于在内核路由表中手动配置路由的高级别安装和应用程序，您可能需要将出口流量路由到主机网络堆栈。默认情况下，出口流量在 OVN 中进行处理以退出集群，不受内核路由表中的特殊路由的影响。默认值为 **false**。 此字段与 Open vSwitch 硬件卸载功能有交互。如果将此字段设置为 **true**，则不会获得卸载的性能优势，因为主机网络堆栈会处理出口流量。
ipForwarding	**object**	您可以使用 **Network** 资源中的 **ipForwarding** 规格来控制 OVN-Kubernetes 管理接口上所有流量的 IP 转发。指定 **Restricted** 只允许 Kubernetes 相关流量的 IP 转发。指定 **Global** 以允许转发所有 IP 流量。对于新安装，默认值为 **Restricted**。对于到 OpenShift Container Platform 4.14 或更高版本的更新，默认值为 **Global**。
ipv4	**object**	可选：指定一个对象来为主机配置内部 OVN-Kubernetes 伪装地址，以服务 IPv4 地址的流量。
ipv6	**object**	可选：指定一个对象来为主机配置内部 OVN-Kubernetes 伪装地址，以服务 IPv6 地址的流量。

表 19.16. gatewayConfig.ipv4 对象

字段	类型	描述
internalMasqueradeSubnet	**字符串**	内部使用的伪装 IPv4 地址，以启用主机服务流量。主机配置了这些 IP 地址和共享网关网桥接口。默认值为 **169.254.169.0/29**。

表 19.17. gatewayConfig.ipv6 对象

字段	类型	描述
internalMasqueradeSubnet	**字符串**	内部使用的伪装 IPv6 地址，以启用主机服务流量。主机配置了这些 IP 地址和共享网关网桥接口。默认值为 **fd69::/125**。

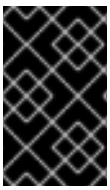
表 19.18. ipsecConfig 对象

字段	类型	描述

字段	类型	描述
模式	**字符串**	指定 IPsec 实现的行为。必须是以下值之一： • **Disabled**: 在集群节点上不启用 IPsec。 • **External** ：对于带有外部主机的网络流量，启用 IPsec。 • **Full**: IPsec 为带有外部主机的 pod 流量和网络流量启用 IPsec。

启用 IPSec 的 OVN-Kubernetes 配置示例

```
defaultNetwork:
  type: OVNKubernetes
  ovnKubernetesConfig:
    mtu: 1400
    genevePort: 6081
      ipsecConfig:
        mode: Full
```



重要

使用 OVNKubernetes 可能会导致 IBM Power® 上的堆栈耗尽问题。

kubeProxyConfig 对象配置（仅限 OpenShiftSDN 容器网络接口）

kubeProxyConfig 对象的值在下表中定义：

表 19.19. kubeProxyConfig object

字段	类型	描述

字段	类型	描述
iptablesSyncPeriod	**字符串**	**iptables** 规则的刷新周期。默认值为 **30s**。有效的后缀包括 **s**、**m** 和 **h**，具体参见 Go 时间 包 文档。 **注意** 由于 OpenShift Container Platform 4.3 及更高版本中引进了性能改进，不再需要调整 **iptablesSyncPeriod** 参数。
proxyArguments.iptables-min-sync-period	**array**	刷新 **iptables** 规则前的最短持续时间。此字段确保刷新的频率不会过于频繁。有效的后缀包括 **s**、**m** 和 **h**，具体参见 Go **time** 软件包。默认值为： kubeProxyConfig: proxyArguments: iptables-min-sync-period: - 0s

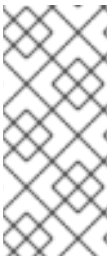
19.2.11. 创建 Kubernetes 清单和 Ignition 配置文件

由于您必须修改一些集群定义文件并手动启动集群机器，因此您必须生成 Kubernetes 清单和 Ignition 配置文件来配置机器。

安装配置文件转换为 Kubernetes 清单。清单嵌套到 Ignition 配置文件中，稍后用于配置集群机器。

重要

- OpenShift Container Platform 安装程序生成的 Ignition 配置文件包含 24 小时后过期的证书，然后在该时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 node-bootstrapper 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。



注意

生成清单和 Ignition 文件的安装程序是特定的架构，可以从 客户端镜像镜像获取。安装程序的 Linux 版本（没有构架 postfix）仅在 ppc64le 上运行。此安装程序也可用作 Mac OS 版本。

先决条件

- 已获得 OpenShift Container Platform 安装程序。
- 已创建 install-config.yaml 安装配置文件。

流程

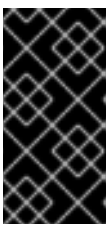
1. 进入包含 OpenShift Container Platform 安装程序的目录，并为集群生成 Kubernetes 清单：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 对于 <installation_directory>，请指定包含您创建的 install-config.yaml 文件的安装目录。

警告

如果您要安装一个三节点集群，请跳过以下步骤，以便可以调度 control plane 节点。



重要

当您将 control plane 节点从默认的不可调度配置为可以调度时，需要额外的订阅。这是因为 control plane 节点变为计算节点。

2. 检查 <installation_directory>/manifests/cluster-scheduler-02-config.yml Kubernetes

清单文件中的 mastersSchedulable 参数是否已设置为 false。此设置可防止在 control plane 机器上调度 pod：

a. 打开 <installation_directory>/manifests/cluster-scheduler-02-config.yml 文件。

b. 找到 mastersSchedulable 参数，并确保它被设置为 false。

c. 保存并退出 文件。

3. 要创建 Ignition 配置文件，请从包含安装程序的目录运行以下命令：

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

1 对于 <installation_directory>，请指定相同的安装目录。

为安装目录中的 bootstrap、control plane 和计算节点创建 Ignition 配置文件。kubeadmin-password 和 kubeconfig 文件在 ./<installation_directory>/auth 目录中创建：

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

19.2.12. 安装 RHCOS 并启动 OpenShift Container Platform bootstrap 过程

要在您置备的 IBM Power® 基础架构上安装 OpenShift Container Platform，您必须在机器上安装 Red Hat Enterprise Linux CoreOS (RHCOS)。安装 RHCOS 时，您必须为您要安装的机器类型提供 OpenShift Container Platform 安装程序生成的 Ignition 配置文件。如果您配置了适当的网络、DNS 和负载均衡基础架构，OpenShift Container Platform bootstrap 过程会在 RHCOS 机器重启后自动启动。

按照以下步骤使用 ISO 镜像或网络 PXE 引导在机器上安装 RHCOS。

19.2.12.1. 使用 ISO 镜像安装 RHCOS

您可以使用 ISO 镜像在机器上安装 RHCOS。

先决条件

- 已为集群创建 Ignition 配置文件。
- 您已配置了适当的网络、DNS 和负载平衡基础架构。
- 您有一个可以从计算机以及您创建的机器访问的 HTTP 服务器。
- 您已参阅 *Advanced RHCOS 安装配置* 部分来了解配置功能的不同方法，如网络和磁盘分区。

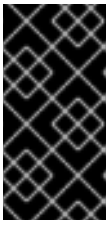
流程

1. 获取每个 Ignition 配置文件的 SHA512 摘要。例如，您可以在运行 Linux 的系统上使用以下内容来获取 bootstrap.ign Ignition 配置文件的 SHA512 摘要：

```
$ sha512sum <installation_directory>/bootstrap.ign
```

后续步骤中会向 coreos-installer 提供摘要，以验证集群节点上 Ignition 配置文件的真实性。

2. 将安装程序创建的 bootstrap、control plane 和计算节点 Ignition 配置文件上传到 HTTP 服务器。注意这些文件的 URL。



重要

您可以在 Ignition 配置中添加或更改配置设置，然后将其保存到 HTTP 服务器。如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

3. 从安装主机上，验证 Ignition 配置文件是否在 URL 上可用。以下示例获取 bootstrap 节点的 Ignition 配置文件：

```
$ curl -k http://<HTTP_server>/bootstrap.ign 1
```

输出示例

```
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0{"ignition":
{"version":"3.2.0"},"passwd":{"users":[{"name":"core","sshAuthorizedKeys":["ssh-
rsa...
```

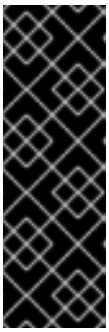
在命令中将 bootstrap.ign 替换为 master.ign 或 worker.ign，以验证 control plane 和计算节点的 Ignition 配置文件是否可用。

4. 虽然可以从 RHCOS 镜像镜像页面获取您选择的操作系统实例安装方法所需的 RHCOS 镜像，但推荐的方法是从 openshift-install 命令的输出获取 RHCOS 镜像的正确版本：

```
$ openshift-install coreos print-stream-json | grep '\.iso[^.]'
```

输出示例

```
"location": "<url>/art/storage/releases/rhcos-4.16-aarch64/<release>/aarch64/rhcos-
<release>-live.aarch64.iso",
"location": "<url>/art/storage/releases/rhcos-4.16-ppc64le/<release>/ppc64le/rhcos-
<release>-live.ppc64le.iso",
"location": "<url>/art/storage/releases/rhcos-4.16-s390x/<release>/s390x/rhcos-
<release>-live.s390x.iso",
"location": "<url>/art/storage/releases/rhcos-4.16/<release>/x86_64/rhcos-<release>-
live.x86_64.iso",
```



重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每个发行版本而改变。您必须下载最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。如果可用，请使用与 OpenShift Container Platform 版本匹配的镜像版本。这个过程只使用 ISO 镜像。此安装类型不支持 RHCOS qcow2 镜像。

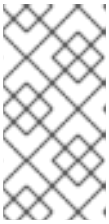
ISO 文件名类似以下示例：

```
rhcos-<version>-live.<architecture>.iso
```

5. 使用 ISO 启动 RHCOS 安装。使用以下安装选项之一：

- 将 ISO 映像刻录到磁盘并直接启动。
- 使用 light-out 管理(LOM)接口使用 ISO 重定向。

6. 在不指定任何选项或中断实时引导序列的情况下引导 RHCOS ISO 镜像。等待安装程序在 RHCOS live 环境中引导进入 shell 提示符。



注意

可以中断 RHCOS 安装引导过程来添加内核参数。但是，在这个 ISO 过程中，您应该使用以下步骤中所述的 coreos-installer 命令，而不是添加内核参数。

7. 运行 coreos-installer 命令并指定满足您的安装要求的选项。您至少必须指定指向节点类型的 Ignition 配置文件的 URL，以及您要安装到的设备：

```
$ sudo coreos-installer install --ignition-url=http://<HTTP_server>/<node_type>.ign
<device> --ignition-hash=sha512-<digest> 1 2
```

1 1 您必须使用 sudo 运行 coreos-installer 命令，因为 core 用户没有执行安装所需的 root 权限。

2

当 Ignition 配置文件通过 HTTP URL 获取时，需要 --ignition-hash 选项来验证集群节点上 Ignition 配置文件的真实性。<digest> 是上一步中获取的 Ignition 配置文件 SHA512 摘要。

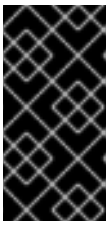
注意

如果要通过使用 TLS 的 HTTPS 服务器提供 Ignition 配置文件，您可以在运行 coreos-installer 前将内部证书颁发机构(CA)添加到系统信任存储中。

以下示例将引导节点安装初始化到 /dev/sda 设备。bootstrap 节点的 Ignition 配置文件从 IP 地址 192.168.1.2 的 HTTP Web 服务器获取：

```
$ sudo coreos-installer install --ignition-
url=http://192.168.1.2:80/installation_directory/bootstrap.ign /dev/sda --ignition-
hash=sha512-
a5a2d43879223273c9b60af66b44202a1d1248fc01cf156c46d4a79f552b6bad47bc8cc78dd
f0116e80c59d2ea9e32ba53bc807afbca581aa059311def2c3e3b
```

8. 在机器的控制台上监控 RHCOS 安装的进度。



重要

在开始安装 OpenShift Container Platform 之前，确保每个节点中安装成功。观察安装过程可以帮助确定可能会出现 RHCOS 安装问题的原因。

9. 安装 RHCOS 后，您必须重启系统。系统重启过程中，它会应用您指定的 Ignition 配置文件。

10. 检查控制台输出，以验证 Ignition 是否运行。

示例命令

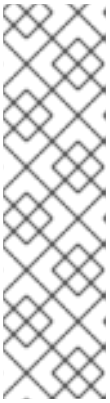
```
Ignition: ran on 2022/03/14 14:48:33 UTC (this boot)
Ignition: user-provided config was applied
```

11. 继续为集群创建其他机器。

重要

此时您必须创建 bootstrap 和 control plane 机器。如果 control plane 机器不可调度，请在安装 OpenShift Container Platform 前至少创建两台计算机器。

如果存在所需的网络、DNS 和负载均衡器基础架构，OpenShift Container Platform bootstrap 过程会在 RHCOS 节点重启后自动启动。



注意

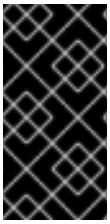
RHCOS 节点不包含 core 用户的默认密码。您可以使用可访问 SSH 私钥的用户的身份运行 ssh core@<node>.<cluster_name>.<base_domain > 来访问节点，该私钥与您在 install_config.yaml 文件中指定的公钥配对。运行 RHCOS 的 OpenShift Container Platform 4 集群节点不可变，它依赖于 Operator 来应用集群更改。不建议使用 SSH 访问集群节点。但是，当调查安装问题时，如果 OpenShift Container Platform API 不可用，或者 kubelet 在目标节点上无法正常工作，则调试或灾难恢复可能需要 SSH 访问。

19.2.12.1.1. 高级 RHCOS 安装参考

本节演示了网络配置和其他高级选项，允许您修改 Red Hat Enterprise Linux CoreOS(RHCOS)手动安装过程。下表描述了您可以用于 RHCOS live 安装程序和 coreos-installer 命令的内核参数和命令行选项。

19.2.12.1.1.1. ISO 安装的网络和绑定选项

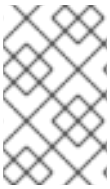
如果从 ISO 镜像安装 RHCOS，您可以在引导镜像时手动添加内核参数，以便为节点配置网络。如果没有指定网络参数，当 RHCOS 检测到需要网络来获取 Ignition 配置文件时，在 initramfs 中激活 DHCP。



重要

在手动添加网络参数时，还必须添加 rd.neednet=1 内核参数，以便在 initramfs 中启动网络。

以下信息提供了在 RHCOS 节点上为 ISO 安装配置网络和绑定的示例。示例描述了如何使用 ip=、name server = 和 bond= 内核参数。



注意

添加内核参数时顺序非常重要： ip=、 name server=，然后 bond=。

网络选项在系统引导过程中传递给 dracut 工具。有关 dracut 支持的网络选项的更多信息，请参阅 dracut.cmdline 手册页。

以下示例是 ISO 安装的网络选项。

配置 DHCP 或静态 IP 地址

要配置 IP 地址，可使用 DHCP(ip=dhcp)或设置单独的静态 IP 地址(ip=<host_ip>)。如果设置静态 IP，则必须在每个节点上识别 DNS 服务器 IP 地址（名称服务器=<dns_ip>）。以下示例集：

- 节点的 IP 地址为 10.10.10.2
- 网关地址为 10.10.10.254
- 子网掩码为 255.255.255.0
- 到 core0.example.com 的主机名
- DNS 服务器地址为 4.4.4.41

- 自动配置值为 none。当以静态方式配置 IP 网络时，不需要自动配置。

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
nameserver=4.4.4.41
```

注意

当您使用 DHCP 为 RHCOS 机器配置 IP 寻址时，机器还通过 DHCP 获取 DNS 服务器信息。对于基于 DHCP 的部署，您可以通过 DHCP 服务器配置定义 RHCOS 节点使用的 DNS 服务器地址。

配置没有静态主机名的 IP 地址

您可以在不分配静态主机名的情况下配置 IP 地址。如果用户没有设置静态主机名，则会提取并通过反向 DNS 查找自动设置。要在没有静态主机名的情况下配置 IP 地址，请参考以下示例：

- 节点的 IP 地址为 10.10.10.2
- 网关地址为 10.10.10.254
- 子网掩码为 255.255.255.0
- DNS 服务器地址为 4.4.4.41
- 自动配置值为 none。当以静态方式配置 IP 网络时，不需要自动配置。

```
ip=10.10.10.2::10.10.10.254:255.255.255.0::enp1s0:none
nameserver=4.4.4.41
```

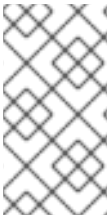
指定多个网络接口

您可以通过设置多个 ip= 条目来指定多个网络接口。

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
ip=10.10.10.3::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none
```

配置默认网关和路由

可选：您可以通过设置 a rd.route= 值来配置到额外网络的路由。



注意

当您配置一个或多个网络时，需要一个默认网关。如果额外网络网关与主要网络网关不同，则默认网关必须是主要网络网关。

- 运行以下命令来配置默认网关：

```
ip=::10.10.10.254::::
```

- 输入以下命令为额外网络配置路由：

```
rd.route=20.20.20.0/24:20.20.20.254:enp2s0
```

在单个接口中禁用 DHCP

您可以在单一接口中禁用 DHCP，例如当有两个或者多个网络接口时，且只有一个接口被使用。在示例中，enp1s0 接口具有一个静态网络配置，而 enp2s0 禁用了 DHCP，不使用它：

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
ip=::::core0.example.com:enp2s0:none
```

合并 DHCP 和静态 IP 配置

您可以将系统上的 DHCP 和静态 IP 配置与多个网络接口合并，例如：

```
ip=enp1s0:dhcp
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none
```

在独立接口上配置 VLAN

可选： 您可以使用 vlan= 参数在单个接口上配置 VLAN。

- 要在网络接口中配置 VLAN 并使用静态 IP 地址，请运行以下命令：

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0.100:none
vlan=enp2s0.100:enp2s0
```

- 要在网络接口中配置 VLAN 并使用 DHCP，请运行以下命令：

```
ip=enp2s0.100:dhcp
vlan=enp2s0.100:enp2s0
```

提供多个 DNS 服务器

您可以通过为每个服务器添加一个 nameserver= 条目来提供多个 DNS 服务器，例如

```
nameserver=1.1.1.1
nameserver=8.8.8.8
```

将多个网络接口绑定到一个接口

可选：您可以使用 bond= 选项将多个网络接口绑定到一个接口。请参见以下示例：

- 配置绑定接口的语法为：bond=<name>[:<network_interfaces>][:options]

  <name> 是绑定设备名称 (bond0)、<network_interfaces> 代表以逗号分隔的物理（以太网）接口列表(em1,em2)，*options* 是用逗号分开的绑定选项列表。输入 modinfo bonding 查看可用选项。

- 当使用 bond= 创建绑定接口时，您必须指定如何分配 IP 地址以及绑定接口的其他信息。

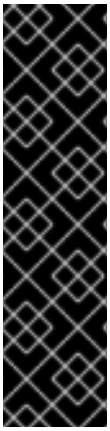
  - 要将绑定接口配置为使用 DHCP，请将绑定的 IP 地址设置为 dhcp。例如：

    ```
    bond=bond0:em1,em2:mode=active-backup
    ip=bond0:dhcp
    ```

  - 要将绑定接口配置为使用静态 IP 地址，请输入您需要的特定 IP 地址和相关信息。例如：

    ```
    bond=bond0:em1,em2:mode=active-backup
    ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0:none
    ```

将多个 SR-IOV 网络接口绑定到双端口 NIC 接口



重要

支持与为 SR-IOV 设备启用 NIC 分区关联的第 1 天操作只是一个技术预览功能。技术预览功能不受红帽产品服务等级协议（SLA）支持，且功能可能并不完整。红帽不推荐在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。

有关红帽技术预览功能支持范围的更多信息，请参阅技术预览功能支持范围。

可选： 您可以使用 bond= 选项将多个 SR-IOV 网络接口绑定到双端口 NIC 接口。

在每个节点上，您必须执行以下任务：

1. 按照管理 SR-IOV 设备中的指导创建 SR-IOV 虚拟功能(VF)。按照"将 SR-IOV 网络设备附加到虚拟机"部分中的步骤操作。

2. 创建绑定，将所需的 VF 附加到绑定，并根据配置网络绑定的指导设置绑定链接状态。按照任何描述的步骤创建绑定。

以下示例演示了您必须使用的语法：

- 配置绑定接口的语法为：bond=<name>[:<network_interfaces>][:options]

  <name> 是绑定设备名称 (bond0)、<network_interfaces> 由内核中已知的名称来代表虚拟功能(VF)，并显示在 ip link 命令的输出中 (eno1f0,eno2f0)， *options* 是以逗号分隔的绑定选项列表。输入 modinfo bonding 查看可用选项。

- 当使用 bond= 创建绑定接口时，您必须指定如何分配 IP 地址以及绑定接口的其他信息。

  - 要将绑定接口配置为使用 DHCP，请将绑定的 IP 地址设置为 dhcp。例如：

    ```
    bond=bond0:eno1f0,eno2f0:mode=active-backup
    ip=bond0:dhcp
    ```

  -

要将绑定接口配置为使用静态 IP 地址，请输入您需要的特定 IP 地址和相关信息。例如：

```
bond=bond0:eno1f0,eno2f0:mode=active-backup
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0:none
```

19.2.12.2. 使用 PXE 引导安装 RHCOS

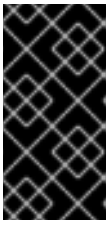
您可以使用 PXE 引导在机器上安装 RHCOS。

先决条件

- 已为集群创建 Ignition 配置文件。
- 您已配置了适当的网络、DNS 和负载平衡基础架构。
- 您已配置了适当的 PXE 基础架构。
- 您有一个可以从计算机以及您创建的机器访问的 HTTP 服务器。
- 您已参阅 *Advanced RHCOS 安装配置* 部分来了解配置功能的不同方法，如网络和磁盘分区。

流程

1. 将安装程序创建的 bootstrap、control plane 和计算节点 Ignition 配置文件上传到 HTTP 服务器。注意这些文件的 URL。



重要

您可以在 Ignition 配置中添加或更改配置设置，然后将其保存到 HTTP 服务器。如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

2. 从安装主机上，验证 Ignition 配置文件是否在 URL 上可用。以下示例获取 bootstrap 节点的 Ignition 配置文件：

-

```
$ curl -k http://<HTTP_server>/bootstrap.ign 1
```

输出示例

```
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0{"ignition":
{"version":"3.2.0"},"passwd":{"users":[{"name":"core","sshAuthorizedKeys":["ssh-
rsa...
```

在命令中将 bootstrap.ign 替换为 master.ign 或 worker.ign，以验证 control plane 和计算节点的 Ignition 配置文件是否可用。

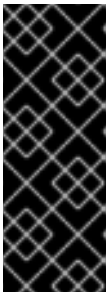
3. 虽然可以从 RHCOS image mirror 页面获取您选择的操作系统实例所需的 RHCOS kernel、initramfs 和 rootfs 文件，但推荐的方法是从 openshift-install 命令的输出中获取 RHCOS 文件的正确版本：

```
$ openshift-install coreos print-stream-json | grep -Eo '"https.*(kernel-
|initramfs.|rootfs.)\w+(\.img)?"'
```

输出示例

```
"<url>/art/storage/releases/rhcos-4.16-aarch64/<release>/aarch64/rhcos-<release>-live-
kernel-aarch64"
"<url>/art/storage/releases/rhcos-4.16-aarch64/<release>/aarch64/rhcos-<release>-live-
initramfs.aarch64.img"
"<url>/art/storage/releases/rhcos-4.16-aarch64/<release>/aarch64/rhcos-<release>-live-
rootfs.aarch64.img"
"<url>/art/storage/releases/rhcos-4.16-ppc64le/49.84.202110081256-0/ppc64le/rhcos-
<release>-live-kernel-ppc64le"
"<url>/art/storage/releases/rhcos-4.16-ppc64le/<release>/ppc64le/rhcos-<release>-live-
initramfs.ppc64le.img"
"<url>/art/storage/releases/rhcos-4.16-ppc64le/<release>/ppc64le/rhcos-<release>-live-
rootfs.ppc64le.img"
"<url>/art/storage/releases/rhcos-4.16-s390x/<release>/s390x/rhcos-<release>-live-
kernel-s390x"
"<url>/art/storage/releases/rhcos-4.16-s390x/<release>/s390x/rhcos-<release>-live-
initramfs.s390x.img"
"<url>/art/storage/releases/rhcos-4.16-s390x/<release>/s390x/rhcos-<release>-live-
rootfs.s390x.img"
"<url>/art/storage/releases/rhcos-4.16/<release>/x86_64/rhcos-<release>-live-kernel-
```

```
x86_64"
"<url>/art/storage/releases/rhcos-4.16/<release>/x86_64/rhcos-<release>-live-
initramfs.x86_64.img"
"<url>/art/storage/releases/rhcos-4.16/<release>/x86_64/rhcos-<release>-live-
rootfs.x86_64.img"
```



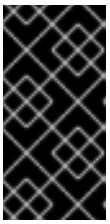
重要

RHCOS 工件可能不会随着 OpenShift Container Platform 的每个发行版本而改变。您必须下载最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。这个过程只使用下面描述的适当 kernel、initram fs 和 rootfs 工件。此安装类型不支持 RHCOS QCOW2 镜像。

文件名包含 OpenShift Container Platform 版本号。它们类似以下示例：

- kernel:rhcos-<version>-live-kernel-<architecture>
- initramfs: rhcos-<version>-live-initramfs.<architecture>.img
- rootfs: rhcos-<version>-live-rootfs.<architecture>.img

4. 将 rootfs、kernel 和 initramfs 文件上传到 HTTP 服务器。



重要

如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

5. 配置网络引导基础架构，以便在安装 RHCOS 后机器从本地磁盘启动。

6. 为 RHCOS 镜像配置 PXE 安装并开始安装。

为您的环境修改以下示例菜单条目，并验证能否正确访问镜像和 Ignition 文件：

```
DEFAULT pxeboot
TIMEOUT 20
PROMPT 0
LABEL pxeboot
    KERNEL http://<HTTP_server>/rhcos-<version>-live-kernel-<architecture> (1)
    APPEND initrd=http://<HTTP_server>/rhcos-<version>-live-initramfs.<architecture>.img
coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-rootfs.<architecture>.img
coreos.inst.install_dev=/dev/sda coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign
(2) (3)
```

(1) (1)

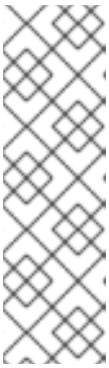
指定上传到 HTTP 服务器的 live kernel 文件位置。URL 必须是 HTTP、TFTP 或 FTP；不支持 HTTPS 和 NFS。

(2)

如果您使用多个 NIC，请在 ip 选项中指定一个接口。例如，要在名为 eno1 的 NIC 上使用 DHCP，请设置 ip=eno1:dhcp。

(3)

指定上传到 HTTP 服务器的 RHCOS 文件的位置。initrd 参数值是 initramfs 文件的位置，coreos.live.rootfs_url 参数值是 rootfs 文件的位置，coreos.inst.ignition_url 参数值则是 bootstrap Ignition 配置文件的位置。您还可以在 APPEND 行中添加更多内核参数来配置联网或其他引导选项。



注意

此配置不会在图形控制台的机器上启用串行控制台访问。要配置不同的控制台，请在 APPEND 行中添加一个或多个 console= 参数。例如，添加 console=tty0 console=ttyS0 以将第一个 PC 串口设置为主控制台，并将图形控制台设置为二级控制台。如需更多信息，请参阅如何在 Red Hat Enterprise Linux 中设置串行终端和/或控制台？ 和"启用 PXE 和 ISO 安装的串行控制台"部分。

7. 在机器的控制台上监控 RHCOS 安装的进度。

重要

在开始安装 OpenShift Container Platform 之前，确保每个节点中安装成功。观察安装过程可以帮助确定可能会出现 RHCOS 安装问题的原因。

8. 安装 RHCOS 后，系统会重启。在重启过程中，系统会应用您指定的 Ignition 配置文件。

9. 检查控制台输出，以验证 Ignition 是否运行。

示例命令

```
Ignition: ran on 2022/03/14 14:48:33 UTC (this boot)
Ignition: user-provided config was applied
```

10. 继续为集群创建机器。

重要

此时您必须创建 bootstrap 和 control plane 机器。如果 control plane 机器不可调度，请在安装集群前至少创建两台计算机器。

如果存在所需的网络、DNS 和负载均衡器基础架构，OpenShift Container Platform bootstrap 过程会在 RHCOS 节点重启后自动启动。

注意

RHCOS 节点不包含 core 用户的默认密码。您可以使用可访问 SSH 私钥的用户的身份运行 ssh core@<node>.<cluster_name>.<base_domain > 来访问节点，该私钥与您在 install_config.yaml 文件中指定的公钥配对。运行 RHCOS 的 OpenShift Container Platform 4 集群节点不可变，它依赖于 Operator 来应用集群更改。不建议使用 SSH 访问集群节点。但是，当调查安装问题时，如果 OpenShift Container Platform API 不可用，或者 kubelet 在目标节点上无法正常工作，则调试或灾难恢复可能需要 SSH 访问。

19.2.12.3. 在 RHCOS 上启用带有内核参数的多路径

在 OpenShift Container Platform 版本 4.16 中，您可以在安装过程中为置备的节点启用多路径。RHCOS 支持主磁盘上的多路径。多路径为硬件故障提供更强大的弹性，以实现更高的主机可用性。

在初始集群创建过程中，您可能需要在所有 master 节点或 worker 节点中添加内核参数。要在 master 节点或 worker 节点中添加内核参数，您可以创建一个 MachineConfig 对象，并将该对象注入 Ignition 在集群设置过程中使用的清单文件集合中。

流程

1. 进入包含安装程序的目录，并为集群生成 Kubernetes 清单：

```
$ ./openshift-install create manifests --dir <installation_directory>
```

2. 决定您要将内核参数添加到 worker 节点还是 control plane 节点。

- 创建机器配置文件。例如，创建一个 99-master-kargs-mpath.yaml 来指示集群添加 master 标签并识别多路径内核参数：

```
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: "master"
  name: 99-master-kargs-mpath
spec:
  kernelArguments:
    - 'rd.multipath=default'
    - 'root=/dev/disk/by-label/dm-mpath-root'
```

3. 在 worker 节点上启用多路径：

- 创建机器配置文件。例如，创建一个 99-worker-kargs-mpath.yaml 来指示集群添加 worker 标签并识别多路径内核参数：

```
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: "worker"
```

```
  name: 99-worker-kargs-mpath
spec:
  kernelArguments:
    - 'rd.multipath=default'
    - 'root=/dev/disk/by-label/dm-mpath-root'
```

现在，您可以继续创建集群。

重要

需要额外的安装后步骤才能完全启用多路径。如需更多信息，请参阅 *安装后机器配置任务* 中的"使用 RHCOS 上内核参数启用多路径"。

如果 MPIO 失败，请使用 bootlist 命令使用其他逻辑设备名称更新引导设备列表。命令显示引导列表，它会在系统以正常模式引导时指定可能的引导设备。

a. 要显示引导列表并在系统以正常模式引导时指定可能的引导设备，请输入以下命令：

```
$ bootlist -m normal -o
sda
```

b. 要为常规模式更新引导列表并添加备用设备名称，请输入以下命令：

```
$ bootlist -m normal -o /dev/sdc /dev/sdd /dev/sde
sdc
sdd
sde
```

如果原始引导磁盘路径停机，节点会从在普通引导设备列表中注册的替代设备重启。

19.2.13. 等待 bootstrap 过程完成

OpenShift Container Platform bootstrap 过程在集群节点首次引导到安装到磁盘的持久 RHCOS 环境后开始。通过 Ignition 配置文件提供的配置信息用于初始化 bootstrap 过程并在机器上安装 OpenShift Container Platform。您必须等待 bootstrap 过程完成。

先决条件

- 已为集群创建 Ignition 配置文件。
- 您已配置了适当的网络、DNS 和负载平衡基础架构。
- 已获得安装程序，并为集群生成 Ignition 配置文件。
- 已在集群机器上安装 RHCOS，并提供 OpenShift Container Platform 安装程序生成的 Ignition 配置文件。
- 您的机器可以直接访问互联网，或者有 HTTP 或 HTTPS 代理可用。

流程

1. 监控 bootstrap 过程：

```
$ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \ 1
    --log-level=info 2
```

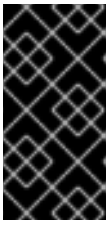
1. 对于 <installation_directory>，请指定安装文件保存到的目录的路径。
2. 要查看不同的安装详情，请指定 warn、debug 或 error，而不是 info。

输出示例

```
INFO Waiting up to 30m0s for the Kubernetes API at
https://api.test.example.com:6443...
INFO API v1.29.4 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
```

当 Kubernetes API 服务器提示已在 control plane 机器上引导它时，该命令会成功。

2. bootstrap 过程完成后，从负载均衡器中删除 bootstrap 机器。



重要

此时您必须从负载均衡器中删除 bootstrap 机器。您还可以删除或重新格式化 bootstrap 机器本身。

19.2.14. 使用 CLI 登录集群

您可以通过导出集群 kubeconfig 文件，以默认系统用户身份登录集群。kubeconfig 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。
- 已安装 oc CLI。

流程

1. 导出 kubeadmin 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 <installation_directory>，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 oc 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

19.2.15. 批准机器的证书签名请求

当您将机器添加到集群时，会为您添加的每台机器生成两个待处理证书签名请求(CSR)。您必须确认这些 CSR 已获得批准，或根据需要自行批准。必须首先批准客户端请求，然后批准服务器请求。

先决条件

- 您已将机器添加到集群中。

流程

1. 确认集群可以识别这些机器：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0  Ready    master  63m  v1.29.4
master-1  Ready    master  63m  v1.29.4
master-2  Ready    master  64m  v1.29.4
```

输出中列出了您创建的所有机器。

注意

在有些 CSR 被批准前，前面的输出可能不包括计算节点（也称为 worker 节点）。

2. 检查待处理的 CSR，并确保添加到集群中的每台机器都有 Pending 或 Approved 状态的客户端请求：

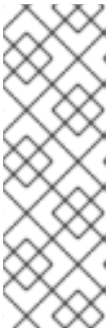
```
$ oc get csr
```

输出示例

```
NAME        AGE    REQUESTOR                                                   CONDITION
csr-8b2br   15m    system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
csr-8vnps   15m    system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
...
```

在本例中，两台机器加入集群。您可能会在列表中看到更多已批准的 CSR。

3. 如果 CSR 没有获得批准，在您添加的机器的所有待处理 CSR 都处于 Pending 状态后，请批准集群机器的 CSR：



注意

由于 CSR 会自动轮转，因此请在将机器添加到集群后一小时内批准您的 CSR。如果没有在一小时内批准它们，证书将会轮转，每个节点会存在多个证书。您必须批准所有这些证书。批准客户端 CSR 后，Kubelet 为服务证书创建一个二级 CSR，这需要手动批准。然后，如果 Kubelet 请求具有相同参数的新证书，则后续提供证书续订请求由 machine-approver 自动批准。

注意

对于在未启用机器 API 的平台上运行的集群，如裸机和其他用户置备的基础架构，您必须实施一种方法来自动批准 kubelet 提供证书请求(CSR)。如果没有批准请求，则 oc exec、ocrsh 和 oc logs 命令将无法成功，因为 API 服务器连接到 kubelet 时需要服务证书。与 Kubelet 端点联系的任何操作都需要此证书批准。该方法必须监视新的 CSR，确认 CSR 由 system: node 或 system:admin 组中的 node-bootstrapper 服务帐户提交，并确认节点的身份。

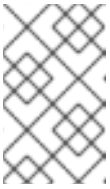
- 要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 <csr_name> 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}
{{"\n"}}{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```



注意

在有些 CSR 被批准前，一些 Operator 可能无法使用。

4. 现在，您的客户端请求已被批准，您必须查看添加到集群中的每台机器的服务器请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE     REQUESTOR                                                   CONDITION
csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal       Pending
csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal       Pending
...
```

5. 如果剩余的 CSR 没有被批准，且处于 Pending 状态，请批准集群机器的 CSR：

- 要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1

<csr_name> 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

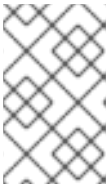
```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}
{{"\n"}}{{end}}{{end}}' | xargs oc adm certificate approve
```

6. 批准所有客户端和服务器 CSR 后，机器将处于 Ready 状态。运行以下命令验证：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0  Ready    master  73m  v1.29.4
master-1  Ready    master  73m  v1.29.4
master-2  Ready    master  74m  v1.29.4
worker-0  Ready    worker  11m  v1.29.4
worker-1  Ready    worker  11m  v1.29.4
```



注意

批准服务器 CSR 后可能需要几分钟时间让机器过渡到 Ready 状态。

其他信息

- 如需有关 CSR 的更多信息，请参阅 证书签名请求。

19.2.16. 初始 Operator 配置

在 control plane 初始化后，您必须立即配置一些 Operator，以便它们都可用。

先决条件

- 您的 control plane 已初始化。

流程

1. 观察集群组件上线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION   AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.16.0    True        False         False      19m
baremetal                                  4.16.0    True        False         False      37m
cloud-credential                           4.16.0    True        False         False      40m
cluster-autoscaler                         4.16.0    True        False         False      37m
config-operator                            4.16.0    True        False         False      38m
console                                    4.16.0    True        False         False      26m
csi-snapshot-controller                    4.16.0    True        False         False      37m
dns                                        4.16.0    True        False         False      37m
etcd                                       4.16.0    True        False         False      36m
image-registry                             4.16.0    True        False         False      31m
ingress                                    4.16.0    True        False         False      30m
insights                                   4.16.0    True        False         False      31m
kube-apiserver                             4.16.0    True        False         False      26m
kube-controller-manager                    4.16.0    True        False         False      36m
kube-scheduler                             4.16.0    True        False         False      36m
kube-storage-version-migrator              4.16.0    True        False         False      37m
machine-api                                4.16.0    True        False         False      29m
machine-approver                           4.16.0    True        False         False      37m
machine-config                             4.16.0    True        False         False      36m
marketplace                                4.16.0    True        False         False      37m
monitoring                                 4.16.0    True        False         False      29m
network                                    4.16.0    True        False         False      38m
node-tuning                                4.16.0    True        False         False      37m
openshift-apiserver                        4.16.0    True        False         False      32m
openshift-controller-manager               4.16.0    True        False         False      30m
openshift-samples                          4.16.0    True        False         False      32m
operator-lifecycle-manager                 4.16.0    True        False         False      37m
operator-lifecycle-manager-catalog         4.16.0    True        False         False      37m
operator-lifecycle-manager-packageserver   4.16.0    True        False         False      32m
service-ca                                 4.16.0    True        False         False      38m
storage                                    4.16.0    True        False         False      37m
```

2. 配置不可用的 Operator。

19.2.16.1. 镜像 registry 存储配置

对于不提供默认存储的平台，Image Registry Operator 最初不可用。安装后，您必须将 registry 配置为使用存储，以便 Registry Operator 可用。

显示配置生产集群所需的持久性卷的说明。如果适用，显示有关将空目录配置为存储位置的说明，这仅适用于非生产集群。

提供了在升级过程中使用 Recreate rollout 策略来允许镜像 registry 使用块存储类型的说明。

19.2.16.1.1. 为 IBM Power 配置 registry 存储

作为集群管理员，在安装后需要配置 registry 来使用存储。

先决条件

- 您可以使用具有 cluster-admin 角色的用户访问集群。
- 在 IBM Power® 上有一个集群。
- 您已为集群置备持久性存储，如 Red Hat OpenShift Data Foundation。

重要

当您只有一个副本时，OpenShift Container Platform 支持对镜像 registry 存储的 ReadWriteOnce 访问。ReadWriteOnce 访问还要求 registry 使用 Recreate rollout 策略。要部署支持高可用性的镜像 registry，需要两个或多个副本，ReadWriteMany 访问。

- 必须具有 100Gi 容量。

流程

1. **要将 registry 配置为使用存储，修改 configs.imageregistry/cluster 资源中的 spec.storage.pvc。**

注意

使用共享存储时，请查看您的安全设置以防止外部访问。

2. 验证您没有 **registry pod:**

```
$ oc get pod -n openshift-image-registry -l docker-registry=default
```

输出示例

```
No resources found in openshift-image-registry namespace
```

注意

如果您的输出中有一个 registry pod，则不需要继续这个过程。

3. 检查 **registry 配置：**

```
$ oc edit configs.imageregistry.operator.openshift.io
```

输出示例

```
storage:
  pvc:
    claim:
```

将 claim 字段留空以允许自动创建 image-registry-storage PVC。

4. 检查 clusteroperator 状态：

```
$ oc get clusteroperator image-registry
```

输出示例

```
NAME             VERSION   AVAILABLE   PROGRESSING   DEGRADED   SINCE   MESSAGE
image-registry   4.16      True        False         False      6h50m
```

5. 确保 registry 设置为 managed，以启用镜像的构建和推送。

- 运行：

```
$ oc edit configs.imageregistry/cluster
```

然后，更改行

```
managementState: Removed
```

至

```
managementState: Managed
```

19.2.16.1.2. 在非生产集群中为镜像 registry 配置存储

您必须为 Image Registry Operator 配置存储。对于非生产集群，您可以将镜像 registry 设置为空目录。如果您这样做，重启 registry 时会丢失所有镜像。

流程

- 将镜像 registry 存储设置为空目录：

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch
'{"spec":{"storage":{"emptyDir":{}}}}'
```

警告

仅为非生产集群配置这个选项。

如果在 Image Registry Operator 初始化其组件前运行这个命令，oc patch 命令会失败并显示以下错误：

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster"
not found
```

等待几分钟，然后再次运行 命令。

19.2.17. 在用户置备的基础架构上完成安装

完成 Operator 配置后，可以在您提供的基础架构上完成集群安装。

先决条件

- 您的 control plane 已初始化。
- 已完成初始 Operator 配置。

流程

1. 使用以下命令确认所有集群组件都在线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION   AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.16.0    True        False         False      19m
baremetal                                  4.16.0    True        False         False      37m
cloud-credential                           4.16.0    True        False         False      40m
cluster-autoscaler                         4.16.0    True        False         False      37m
config-operator                            4.16.0    True        False         False      38m
console                                    4.16.0    True        False         False      26m
csi-snapshot-controller                    4.16.0    True        False         False      37m
dns                                        4.16.0    True        False         False      37m
etcd                                       4.16.0    True        False         False      36m
image-registry                             4.16.0    True        False         False      31m
ingress                                    4.16.0    True        False         False      30m
insights                                   4.16.0    True        False         False      31m
kube-apiserver                             4.16.0    True        False         False      26m
kube-controller-manager                    4.16.0    True        False         False      36m
kube-scheduler                             4.16.0    True        False         False      36m
kube-storage-version-migrator              4.16.0    True        False         False      37m
machine-api                                4.16.0    True        False         False      29m
machine-approver                           4.16.0    True        False         False      37m
machine-config                             4.16.0    True        False         False      36m
marketplace                                4.16.0    True        False         False      37m
monitoring                                 4.16.0    True        False         False      29m
network                                    4.16.0    True        False         False      38m
node-tuning                                4.16.0    True        False         False      37m
openshift-apiserver                        4.16.0    True        False         False      32m
openshift-controller-manager               4.16.0    True        False         False      30m
openshift-samples                          4.16.0    True        False         False      32m
operator-lifecycle-manager                 4.16.0    True        False         False      37m
operator-lifecycle-manager-catalog         4.16.0    True        False         False      37m
operator-lifecycle-manager-packageserver   4.16.0    True        False         False      32m
service-ca                                 4.16.0    True        False         False      38m
storage                                    4.16.0    True        False         False      37m
```

另外，当所有集群都可用时，以下命令会通知您。它还检索并显示凭证：

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

1

对于 <installation_directory>，请指定安装文件保存到的目录的路径。

输出示例

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

Cluster Version Operator 完成从 Kubernetes API 服务器部署 OpenShift Container Platform 集群时，该命令会成功。

重要

- 安装程序生成的 Ignition 配置文件包含 24 小时后过期的证书，然后在该时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 node-bootstrapper 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。

- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

2. 确认 Kubernetes API 服务器正在与 pod 通信。

   a. 要查看所有 pod 的列表，请使用以下命令：

   ```
   $ oc get pods --all-namespaces
   ```

   输出示例

   ```
   NAMESPACE                  NAME                                        READY   STATUS
   RESTARTS   AGE
   openshift-apiserver-operator    openshift-apiserver-operator-85cb746d55-zqhs8
   1/1   Running   1        9m
   openshift-apiserver        apiserver-67b9g                             1/1     Running    0
   3m
   openshift-apiserver        apiserver-ljcmx                             1/1     Running    0
   1m
   openshift-apiserver        apiserver-z25h4                             1/1     Running    0
   ```

```
2m
openshift-authentication-operator authentication-operator-69d5d8bf84-vh2n8
1/1     Running     0          5m
...
```

b. 使用以下命令，查看上一命令的输出中所列 pod 的日志：

```
$ oc logs <pod_name> -n <namespace> 1
```

1 指定 pod 名称和命名空间，如上一命令的输出中所示。

如果 pod 日志显示，Kubernetes API 服务器可以与集群机器通信。

3. 启用多路径需要额外的步骤。不要在安装过程中启用多路径。

如需更多信息，请参阅 *安装后机器配置任务* 文档中的"使用 RHCOS 上使用内核参数启用多路径"。

19.2.18. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于 远程健康监控

19.2.19. 后续步骤

- 在 RHCOS 上启用带有内核参数的多路径。
- 自定义集群。
- 如果需要，您可以选择 不使用远程健康报告。

19.3. 在受限网络中的 IBM POWER 上安装集群

在 OpenShift Container Platform 版本 4.16 中，您可以在受限网络中置备的 IBM Power® 基础架构上安装集群。

重要

非裸机平台还有其他注意事项。在安装 OpenShift Container Platform 集群前，请参阅有关在未经测试的平台上部署 OpenShift Container Platform 的指南 中的信息。

19.3.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读有关 选择集群安装方法的文档，并为用户准备它。
- 您 创建了用于在受限网络中安装的镜像 registry，并获取您的 OpenShift Container Platform 版本的 imageContentSources 数据。
- 在开始安装过程前，您必须移动或删除任何现有的安装文件。这样可确保在安装过程中创建和更新所需的安装文件。

重要

确定在可访问安装介质的机器上执行安装步骤。

- 已为集群置备了 使用 OpenShift Data Foundation 或其他支持的存储协议的持久性存储。

要部署私有镜像 registry，您必须使用 ReadWriteMany 访问设置持久性存储。

- 如果您使用防火墙并计划使用 Telemetry 服务，则将防火墙配置为允许集群需要访问的站点。

注意

如果要配置代理，请务必查看此站点列表。

19.3.2. 关于在受限网络中安装

在 OpenShift Container Platform 4.16 中，可以执行不需要有效的互联网连接来获取软件组件的安装。受限网络安装可以使用安装程序置备的基础架构或用户置备的基础架构完成，具体取决于您要安装集群的云平台。

要完成受限网络安装，您必须创建一个 registry，以镜像 OpenShift 镜像 registry 的内容并包含安装介质。您可以在镜像主机上创建此 registry，该主机可同时访问互联网和您的封闭网络，也可以使用满足您的限制条件的其他方法。

重要

由于用户置备安装配置的复杂性，在尝试使用用户置备的基础架构受限网络安装前，请考虑完成标准用户置备的基础架构安装。完成此测试安装后，您可以更轻松地隔离和排除在受限网络中安装过程中可能出现的任何问题。

19.3.2.1. 其他限制

受限网络中的集群有以下额外限制和限制：

- ClusterVersion 状态包含一个 Unable to retrieve available updates 错误。
- 默认情况下，您无法使用 Developer Catalog 的内容，因为您无法访问所需的镜像流标签。

19.3.3. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来获得用来安装集群的镜像。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

19.3.4. 具有用户置备基础架构的集群的要求

对于包含用户置备的基础架构的集群，您必须部署所有所需的机器。

本节论述了在用户置备的基础架构上部署 OpenShift Container Platform 的要求。

19.3.4.1. 集群安装所需的机器

最小的 OpenShift Container Platform 集群需要以下主机：

表 19.20. 最低所需的主机

主机	描述
一个临时 bootstrap 机器	集群需要 bootstrap 机器在三台 control plane 机器上部署 OpenShift Container Platform 集群。您可在安装集群后删除 bootstrap 机器。
三台 control plane 机器	control plane 机器运行组成 control plane 的 Kubernetes 和 OpenShift Container Platform 服务。
至少两台计算机器，也称为 worker 机器。	OpenShift Container Platform 用户请求的工作负载在计算机器上运行。

重要

要保持集群的高可用性，请将独立的物理主机用于这些集群机器。

bootstrap、control plane 和计算机器必须使用 Red Hat Enterprise Linux CoreOS(RHCOS)作为操作系统。

请注意，RHCOS 基于 Red Hat Enterprise Linux(RHEL) 9.2，并继承其所有硬件认证和要求。查看红帽企业 Linux 技术功能和限制。

19.3.4.2. 集群安装的最低资源要求

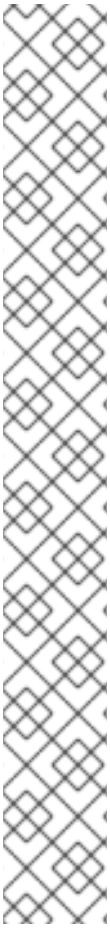
每台集群机器都必须满足以下最低要求：

表 19.21. 最低资源要求

机器	操作系统	vCPU [1]	虚拟内存	Storage	每秒输入/输出 (IOPS) [2]
bootstrap	RHCOS	2	16 GB	100 GB	300
Control plane（控制平面）	RHCOS	2	16 GB	100 GB	300
Compute	RHCOS	2	8 GB	100 GB	300

1. 当未启用并发多线程(SMT)或超线程时，一个 vCPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比例：（每个内核数的线程）× sockets = vCPU。

2. OpenShift Container Platform 和 Kubernetes 对磁盘性能敏感，建议使用更快的存储，特别是 control plane 节点上的 etcd。请注意，在许多云平台上，存储大小和 IOPS 可一起扩展，因此您可能需要过度分配存储卷来获取足够的性能。



注意

从 OpenShift Container Platform 版本 4.13 开始，RHCOS 基于 RHEL 版本 9.2，它更新了微架构要求。以下列表包含每个架构需要的最小指令集架构 (ISA)：

- x86-64 体系结构需要 x86-64-v2 ISA
- ARM64 架构需要 ARMv8.0-A ISA
- IBM Power 架构需要 Power 9 ISA
- s390x 架构需要 z14 ISA

如需更多信息，请参阅 RHEL 架构。

如果平台的实例类型满足集群机器的最低要求，则 OpenShift Container Platform 支持使用它。

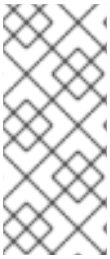
其他资源

- 优化存储

19.3.4.3. 最低 IBM Power 要求

您可以在以下 IBM® 硬件上安装 OpenShift Container Platform 版本 4.16：

- IBM Power®9 或 IBM Power®10 处理器系统



注意

OpenShift Container Platform 4.16 中弃用了所有 IBM Power®8 模型、IBM Power® AC922、IBM Power® IC922 和 IBM Power® LC922 的 RHCOS 功能。红帽建议您使用后续的硬件模型。

硬件要求

- 跨多个 PowerVM 服务器的六个逻辑分区(LPAR)

操作系统要求

- IBM Power®9 或 Power10 处理器系统的一个实例

在 IBM Power® 实例中设置：

- 用于 OpenShift Container Platform control plane 机器的三个 LPAR
- 用于 OpenShift Container Platform 计算机器的两个 LPAR
- 一个 LPAR 用于临时 OpenShift Container Platform bootstrap 机器

IBM Power 客户机虚拟机的磁盘存储

- 本地存储或由虚拟 I/O 服务器使用 vSCSI、NPIV(N-Port ID Virtualization)或 SSP（共享存储池）置备的存储

PowerVM 客户机虚拟机的网络

- 专用物理适配器或 SR-IOV 虚拟功能
- 虚拟 I/O 服务器使用共享以太网适配器提供
- 虚拟 I/O 服务器使用 IBM® vNIC 虚拟化

存储/主内存

- OpenShift Container Platform control plane 机器需要 100 GB / 16 GB
- OpenShift Container Platform 计算机器需要 100 GB / 8 GB

- 临时 OpenShift Container Platform bootstrap 机器需要 100 GB / 16 GB

19.3.4.4. 推荐的 IBM Power 系统要求

硬件要求

- 跨多个 PowerVM 服务器的六个 LPAR

操作系统要求

- IBM Power®9 或 IBM Power®10 处理器的系统的一个实例

在 IBM Power® 实例中设置：

- 用于 OpenShift Container Platform control plane 机器的三个 LPAR
- 用于 OpenShift Container Platform 计算机器的两个 LPAR
- 一个 LPAR 用于临时 OpenShift Container Platform bootstrap 机器

IBM Power 客户机虚拟机的磁盘存储

- 本地存储或由虚拟 I/O 服务器使用 vSCSI、NPIV(N-Port ID Virtualization)或 SSP（共享存储池）置备的存储

PowerVM 客户机虚拟机的网络

- 专用物理适配器或 SR-IOV 虚拟功能
- 虚拟 I/O 服务器使用共享以太网适配器虚拟化
- 虚拟 I/O 服务器使用 IBM® vNIC 虚拟化

存储/主内存

- OpenShift Container Platform control plane 机器需要 120 GB / 32 GB
- OpenShift Container Platform 计算机器需要 120 GB / 32 GB
- 临时 OpenShift Container Platform bootstrap 机器需要 120 GB / 16 GB

19.3.4.5. 证书签名请求管理

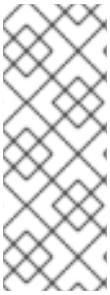
在使用您置备的基础架构时，集群只能有限地访问自动机器管理，因此您必须提供一种在安装后批准集群证书签名请求 (CSR) 的机制。kube-controller-manager 只能批准 kubelet 客户端 CSR。machine-approver 无法保证使用 kubelet 凭证请求的提供证书的有效性，因为它不能确认是正确的机器发出了该请求。您必须决定并实施一种方法，以验证 kubelet 提供证书请求的有效性并进行批准。

19.3.4.6. 用户置备的基础架构对网络的要求

所有 Red Hat Enterprise Linux CoreOS(RHCOS)机器都需要在启动时在 initramfs 中配置联网，以获取它们的 Ignition 配置文件。

在初次启动过程中，机器需要 IP 地址配置，该配置通过 DHCP 服务器或静态设置，提供所需的引导选项。建立网络连接后，机器会从 HTTP 或 HTTPS 服务器下载 Ignition 配置文件。然后，Ignition 配置文件用于设置每台机器的确切状态。Machine Config Operator 在安装后完成对机器的更多更改，如应用新证书或密钥。

建议使用 DHCP 服务器对集群机器进行长期管理。确保 DHCP 服务器已配置为向集群机器提供持久的 IP 地址、DNS 服务器信息和主机名。



注意

如果用户置备的基础架构没有 DHCP 服务，您可以在 RHCOS 安装时向节点提供 IP 网络配置和 DNS 服务器地址。如果要从 ISO 镜像安装，这些参数可作为引导参数传递。如需有关静态 IP 置备和高级网络选项的更多信息，请参阅 *安装 RHCOS 并启动 OpenShift Container Platform bootstrap 过程* 部分。

Kubernetes API 服务器必须能够解析集群机器的节点名称。如果 API 服务器和 worker 节点位于不同的区域中，您可以配置默认 DNS 搜索区域，以允许 API 服务器解析节点名称。另一种支持的方法是始终通过节点对象和所有 DNS 请求中的完全限定域名引用主机。

19.3.4.6.1. 通过 DHCP 设置集群节点主机名

在 Red Hat Enterprise Linux CoreOS(RHCOS)机器上，主机名是通过 NetworkManager 设置的。默认情况下，机器通过 DHCP 获取其主机名。如果主机名不是由 DHCP 提供，请通过内核参数或者其它方法进行静态设置，请通过反向 DNS 查找获取。反向 DNS 查找在网络初始化后进行，可能需要一些时间来解决。其他系统服务可以在此之前启动，并将主机名检测为 localhost 或类似的内容。您可以使用 DHCP 为每个集群节点提供主机名来避免这种情况。

另外，通过 DHCP 设置主机名可以绕过实施 DNS split-horizon 的环境中的手动 DNS 记录名称配置错误。

19.3.4.6.2. 网络连接要求

您必须配置机器之间的网络连接，以允许 OpenShift Container Platform 集群组件进行通信。每台机器都必须能够解析集群中所有其他机器的主机名。

本节详细介绍了所需的端口。

表 19.22. 用于全机器到所有机器通信的端口

协议	port	描述
ICMP	N/A	网络可访问性测试
TCP	**1936**	指标
	9000-**9999**	主机级别的服务，包括端口 9**100**-**9101** 上的节点导出器，以及端口 **9099** 上的 Cluster Version Operator。
	10250-**10259**	Kubernetes 保留的默认端口
UDP	**4789**	VXLAN
	6081	Geneve
	9000-**9999**	主机级别的服务，包括端口 **9100**到**9101** 上的节点导出器。
	500	IPsec IKE 数据包
	4500	IPsec NAT-T 数据包

协议	port	描述
	123	UDP 端口 **123** 上的网络时间协议 (NTP) 如果配置了外部 NTP 时间服务器，需要打开 UDP 端口 **123**。
TCP/UDP	**30000-32767**	Kubernetes 节点端口
ESP	N/A	IPsec Encapsulating Security Payload(ESP)

表 19.23. 用于所有机器控制平面通信的端口

协议	port	描述
TCP	**6443**	Kubernetes API

表 19.24. control plane 机器用于 control plane 机器通信的端口

协议	port	描述
TCP	**2379-2380**	etcd 服务器和对等端口

用户置备的基础架构的 NTP 配置

OpenShift Container Platform 集群被配置为默认使用公共网络时间协议(NTP)服务器。如果要使用本地企业 NTP 服务器，或者集群部署在断开连接的网络中，您可以将集群配置为使用特定的时间服务器。如需更多信息，*请参阅配置 chrony 时间服务* 的文档。

如果 DHCP 服务器提供 NTP 服务器信息，Red Hat Enterprise Linux CoreOS(RHCOS)机器上的 chrony 时间服务会读取信息，并可以把时钟与 NTP 服务器同步。

其他资源

- **配置 chrony 时间服务**

19.3.4.7. 用户置备的 DNS 要求

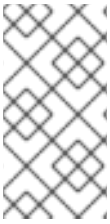
在 OpenShift Container Platform 部署中，以下组件需要 DNS 名称解析：

- **The Kubernetes API**

- OpenShift Container Platform 应用程序通配符
- bootstrap、control plane 和计算机器

Kubernetes API、bootstrap 机器、control plane 机器和计算机器也需要反向 DNS 解析。

DNS A/AAAA 或 CNAME 记录用于名称解析，PTR 记录用于反向名称解析。反向记录很重要，因为 Red Hat Enterprise Linux CoreOS(RHCOS)使用反向记录为所有节点设置主机名，除非 DHCP 提供主机名。另外，反向记录用于生成 OpenShift Container Platform 需要操作的证书签名请求(CSR)。



注意

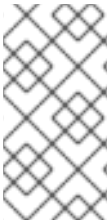
建议使用 DHCP 服务器为每个群集节点提供主机名。如需更多信息，*请参阅用户置备的基础架构部分的 DHCP 建议*。

用户置备的 OpenShift Container Platform 集群需要以下 DNS 记录，这些记录必须在安装前就位。在每个记录中，<cluster_name> 是集群名称，<base_domain> 是您在 install-config.yaml 文件中指定的基域。完整的 DNS 记录采用以下形式： <component>.<cluster_name>.<base_domain>.。

表 19.25. 所需的 DNS 记录

组件	记录	描述
Kubernetes API	**api.<cluster_name>.<base_domain>.**	DNS A/AAAA 或 CNAME 记录，以及用于标识 API 负载均衡器的 DNS PTR 记录。这些记录必须由集群外的客户端和集群中的所有节点解析。
	api-int.<cluster_name>.<base_domain>.	DNS A/AAAA 或 CNAME 记录，以及用于内部标识 API 负载均衡器的 DNS PTR 记录。这些记录必须可以从集群中的所有节点解析。  **重要** API 服务器必须能够根据 Kubernetes 中记录的主机名解析 worker 节点。如果 API 服务器无法解析节点名称，则代理的 API 调用会失败，且您无法从 pod 检索日志。

组件	记录	描述
Routes	***.apps.<cluster_name>.<base_domain>.**	通配符 DNS A/AAAA 或 CNAME 记录，指向应用程序入口负载均衡器。应用程序入口负载均衡器以运行 Ingress Controller Pod 的机器为目标。默认情况下，Ingress Controller Pod 在计算机器上运行。这些记录必须由集群外的客户端和集群中的所有节点解析。 例如，console **-openshift-console.apps.<cluster_name>.<base_domain>** 用作到 OpenShift Container Platform 控制台的通配符路由。
bootstrap 机器	**bootstrap.<cluster_name>.<base_domain>.**	DNS A/AAAA 或 CNAME 记录，以及用于标识 bootstrap 机器的 DNS PTR 记录。这些记录必须由集群中的节点解析。
control plane 机器	**<control_plane><n>.<cluster_name>.<base_domain>.**	DNS A/AAAA 或 CNAME 记录，以识别 control plane 节点的每台机器。这些记录必须由集群中的节点解析。
计算机器	**<compute><n>.<cluster_name>.<base_domain>.**	DNS A/AAAA 或 CNAME 记录，用于识别 worker 节点的每台机器。这些记录必须由集群中的节点解析。



注意

在 OpenShift Container Platform 4.4 及更新的版本中，您不需要在 DNS 配置中指定 etcd 主机和 SRV 记录。

提示

您可以使用 dig 命令验证名称和反向名称解析。如需了解详细的 *验证步骤，请参阅为用户置备的基础架构验证 DNS 解析* 一节。

19.3.4.7.1. 用户置备的集群的 DNS 配置示例

本节提供 A 和 PTR 记录配置示例，它们满足了在用户置备的基础架构上部署 OpenShift Container Platform 的 DNS 要求。样本不是为选择一个 DNS 解决方案提供建议。

在这个示例中，集群名称为 ocp4，基域是 example.com。

用户置备的集群的 DNS A 记录配置示例

以下示例是 BIND 区域文件，其中显示了用户置备的集群中名称解析的 A 记录示例。

例 19.4. DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
  2019070700 ; serial
  3H  ; refresh (3 hours)
  30M  ; retry (30 minutes)
  2W  ; expiry (2 weeks)
  1W )  ; minimum (1 week)
 IN NS ns1.example.com.
 IN MX 10 smtp.example.com.
;
;
ns1.example.com.  IN A 192.168.1.5
smtp.example.com.  IN A 192.168.1.5
;
helper.example.com.  IN A 192.168.1.5
helper.ocp4.example.com. IN A 192.168.1.5
;
api.ocp4.example.com.  IN A 192.168.1.5 1
api-int.ocp4.example.com. IN A 192.168.1.5 2
;
*.apps.ocp4.example.com. IN A 192.168.1.5 3
;
bootstrap.ocp4.example.com. IN A 192.168.1.96 4
;
control-plane0.ocp4.example.com. IN A 192.168.1.97 5
control-plane1.ocp4.example.com. IN A 192.168.1.98 6
control-plane2.ocp4.example.com. IN A 192.168.1.99 7
;
compute0.ocp4.example.com. IN A 192.168.1.11 8
compute1.ocp4.example.com. IN A 192.168.1.7 9
;
;EOF
```

1 为 Kubernetes API 提供名称解析。记录引用 API 负载均衡器的 IP 地址。

2 为 Kubernetes API 提供名称解析。记录引用 API 负载均衡器的 IP 地址，用于内部集群通信。

3 为通配符路由提供名称解析。记录引用应用程序入口负载均衡器的 IP 地址。应用程序入口负载均衡器以运行 Ingress Controller Pod 的机器为目标。默认情况下，Ingress Controller Pod 在计算机器上运行。

注意

在这个示例中，将相同的负载均衡器用于 Kubernetes API 和应用入口流量。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。

4

为 bootstrap 机器提供名称解析。

5 6 7

为 control plane 机器提供名称解析。

8 9

为计算机器提供名称解析。

用户置备的集群的 DNS PTR 记录配置示例

以下示例 BIND 区域文件显示了用户置备的集群中反向名称解析的 PTR 记录示例。

例 19.5. 反向记录的 DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
  2019070700 ; serial
  3H  ; refresh (3 hours)
  30M  ; retry (30 minutes)
  2W  ; expiry (2 weeks)
  1W )  ; minimum (1 week)
 IN NS ns1.example.com.
;
5.1.168.192.in-addr.arpa. IN PTR api.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. IN PTR api-int.ocp4.example.com. 2
;
96.1.168.192.in-addr.arpa. IN PTR bootstrap.ocp4.example.com. 3
;
97.1.168.192.in-addr.arpa. IN PTR control-plane0.ocp4.example.com. 4
98.1.168.192.in-addr.arpa. IN PTR control-plane1.ocp4.example.com. 5
99.1.168.192.in-addr.arpa. IN PTR control-plane2.ocp4.example.com. 6
;
11.1.168.192.in-addr.arpa. IN PTR compute0.ocp4.example.com. 7
```

```
7.1.168.192.in-addr.arpa. IN PTR compute1.ocp4.example.com. 8
;
;EOF
```

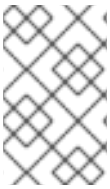
1 为 Kubernetes API 提供反向 DNS 解析。PTR 记录引用 API 负载均衡器的记录名称。

2 为 Kubernetes API 提供反向 DNS 解析。PTR 记录引用 API 负载均衡器的记录名称，用于内部集群通信。

3 为 bootstrap 机器提供反向 DNS 解析。

4 5 6 为 control plane 机器提供反向 DNS 解析。

7 8 为计算机器提供反向 DNS 解析。

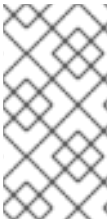


注意

OpenShift Container Platform 应用程序通配符不需要 PTR 记录。

19.3.4.8. 用户置备的基础架构的负载均衡要求

在安装 OpenShift Container Platform 前，您必须置备 API 和应用程序入口负载均衡基础架构。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。



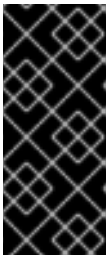
注意

如果要使用 Red Hat Enterprise Linux (RHEL) 实例部署 API 和应用程序入口负载均衡器，您必须单独购买 RHEL 订阅。

负载平衡基础架构必须满足以下要求：

1. **API** 负载均衡器 ：提供一个通用端点，供用户（包括人工和机器）与平台交互和配置。配置以下条件：

   - 仅第 4 层负载均衡.这可被称为 **Raw TCP** 或 **SSL Passthrough** 模式。
   - 无状态负载平衡算法。这些选项根据负载均衡器的实施而有所不同。



重要

不要为 **API** 负载均衡器配置会话持久性。为 **Kubernetes API** 服务器配置会话持久性可能会导致出现过量 **OpenShift Container Platform** 集群应用程序流量，以及过量的在集群中运行的 **Kubernetes API**。

在负载均衡器的前端和后端配置以下端口：

表 19.26. API 负载均衡器

port	后端机器（池成员）	internal	外部	描述
6443	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。您必须为 API 服务器健康检查探测配置 **/readyz** 端点。	X	X	Kubernetes API 服务器
22623	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。	X		机器配置服务器

注意

负载均衡器必须配置为，从 **API** 服务器关闭 **/readyz** 端点到从池中移除 **API** 服务器实例时最多需要 **30** 秒。在 **/readyz** 返回错误或健康后的时间范围内，端点必须被删除或添加。每 **5** 秒或 **10** 秒探测一次，有两个成功请求处于健康状态，三个成为不健康的请求是经过良好测试的值。

2.

应用程序入口负载均衡器 ：为应用程序流量从集群外部流提供入口点。OpenShift Container Platform 集群需要正确配置入口路由器。

配置以下条件：

- 仅第 4 层负载均衡.这可被称为 Raw TCP 或 SSL Passthrough 模式。
- 建议根据可用选项以及平台上托管的应用程序类型，使用基于连接的或基于会话的持久性。

提示

如果应用程序入口负载均衡器可以看到客户端的真实 IP 地址，启用基于 IP 的会话持久性可以提高使用端到端 TLS 加密的应用程序的性能。

在负载均衡器的前端和后端配置以下端口：

表 19.27. 应用程序入口负载均衡器

port	后端机器（池成员）	internal	外部	描述
443	默认情况下，运行 Ingress Controller Pod、计算或 worker 的机器。	X	X	HTTPS 流量
80	默认情况下，运行 Ingress Controller Pod、计算或 worker 的机器。	X	X	HTTP 流量

注意

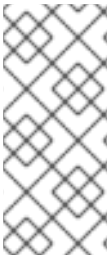
如果要部署一个带有零计算节点的三节点集群，Ingress Controller Pod 在 control plane 节点上运行。在三节点集群部署中，您必须配置应用程序入口负载均衡器，将 HTTP 和 HTTPS 流量路由到 control plane 节点。

19.3.4.8.1. 用户置备的集群的负载均衡器配置示例

本节提供了一个满足用户置备集群的负载均衡要求的 API 和应用程序入口负载均衡器配置示例。示例是 HAProxy 负载均衡器的 /etc/haproxy/haproxy.cfg 配置。这个示例不是为选择一个负载平衡解决方案

提供建议。

在这个示例中，将相同的负载均衡器用于 Kubernetes API 和应用入口流量。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。



注意

如果您使用 HAProxy 作为负载均衡器，并且 SELinux 设置为 enforcing，您必须通过运行 setsebool -P haproxy_connect_any=1 来确保 HAProxy 服务可以绑定到配置的 TCP 端口。

例 19.6. API 和应用程序入口负载均衡器配置示例

```
global
  log         127.0.0.1 local2
  pidfile     /var/run/haproxy.pid
  maxconn     4000
  daemon
defaults
  mode                    http
  log                     global
  option                  dontlognull
  option http-server-close
  option                  redispatch
  retries                 3
  timeout http-request    10s
  timeout queue           1m
  timeout connect         10s
  timeout client          1m
  timeout server          1m
  timeout http-keep-alive 10s
  timeout check           10s
  maxconn                 3000
listen api-server-6443 (1)
  bind *:6443
  mode tcp
  option  httpchk GET /readyz HTTP/1.0
  option  log-health-checks
  balance roundrobin
  server bootstrap bootstrap.ocp4.example.com:6443 verify none check check-ssl inter 10s
fall 2 rise 3 backup (2)
  server master0 master0.ocp4.example.com:6443 weight 1 verify none check check-ssl
inter 10s fall 2 rise 3
  server master1 master1.ocp4.example.com:6443 weight 1 verify none check check-ssl
inter 10s fall 2 rise 3
  server master2 master2.ocp4.example.com:6443 weight 1 verify none check check-ssl
inter 10s fall 2 rise 3
listen machine-config-server-22623 (3)
  bind *:22623
```

```
    mode tcp
    server bootstrap bootstrap.ocp4.example.com:22623 check inter 1s backup 4
    server master0 master0.ocp4.example.com:22623 check inter 1s
    server master1 master1.ocp4.example.com:22623 check inter 1s
    server master2 master2.ocp4.example.com:22623 check inter 1s
  listen ingress-router-443 5
    bind *:443
    mode tcp
    balance source
    server compute0 compute0.ocp4.example.com:443 check inter 1s
    server compute1 compute1.ocp4.example.com:443 check inter 1s
  listen ingress-router-80 6
    bind *:80
    mode tcp
    balance source
    server compute0 compute0.ocp4.example.com:80 check inter 1s
    server compute1 compute1.ocp4.example.com:80 check inter 1s
```

1

端口 6443 处理 Kubernetes API 流量并指向 control plane 机器。

2 4

bootstrap 条目必须在 OpenShift Container Platform 集群安装前就位，且必须在 bootstrap 过程完成后删除它们。

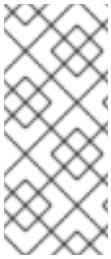
3

5

端口 443 处理 HTTPS 流量，并指向运行 Ingress Controller pod 的机器。默认情况下，Ingress Controller Pod 在计算机器上运行。

6

端口 80 处理 HTTP 流量，并指向运行 Ingress Controller pod 的机器。默认情况下，Ingress Controller Pod 在计算机器上运行。



注意

如果要部署一个带有零计算节点的三节点集群，Ingress Controller Pod 在 control plane 节点上运行。在三节点集群部署中，您必须配置应用程序入口负载均衡器，将 HTTP 和 HTTPS 流量路由到 control plane 节点。

提示

如果您使用 HAProxy 作为负载均衡器，您可以通过在 HAProxy 节点上运行 netstat -nltupe 来检查 haproxy 进程是否在侦听端口 6443、22623、443 和 80。

19.3.5. 准备用户置备的基础架构

在用户置备的基础架构上安装 OpenShift Container Platform 之前，您必须准备底层基础架构。

本节详细介绍了设置集群基础架构以准备 OpenShift Container Platform 安装所需的高级别步骤。这包括为您的集群节点配置 IP 网络和网络连接，通过防火墙启用所需的端口，以及设置所需的 DNS 和负载均衡基础架构。

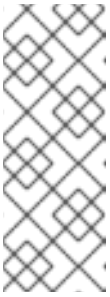
准备后，集群基础架构必须满足 *带有用户置备的基础架构部分的集群要求*。

先决条件

- 您已参阅 OpenShift Container Platform 4.x Tested Integrations 页面。
- 您已查看了 *具有用户置备基础架构的集群要求部分中详述的基础架构* 要求。

流程

1. 如果您使用 DHCP 向集群节点提供 IP 网络配置，请配置 DHCP 服务。
   a. 将节点的持久 IP 地址添加到您的 DHCP 服务器配置。在您的配置中，将相关网络接口的 MAC 地址与每个节点的预期 IP 地址匹配。
   b. 当您使用 DHCP 为集群机器配置 IP 寻址时，机器还通过 DHCP 获取 DNS 服务器信息。定义集群节点通过 DHCP 服务器配置使用的持久性 DNS 服务器地址。



注意

如果没有使用 DHCP 服务，则必须在 RHCOS 安装时为节点提供 IP 网络配置和 DNS 服务器地址。如果要从 ISO 镜像安装，这些参数可作为引导参数传递。如需有关静态 IP 置备和高级网络选项的更多信息，请参阅 *安装 RHCOS 并启动 OpenShift Container Platform bootstrap 过程* 部分。

c.

在 DHCP 服务器配置中定义集群节点的主机名。有关 *主机名注意事项的详情，请参阅通过 DHCP 设置集群节点* 主机名部分。

注意

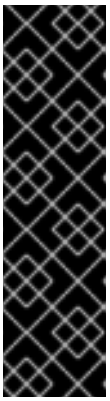
如果没有使用 DHCP 服务，集群节点可以通过反向 DNS 查找来获取其主机名。

2.

确保您的网络基础架构提供集群组件之间所需的网络连接。有关 *要求的详情，请参阅用户置备的基础架构* 的网络要求部分。

3.

将防火墙配置为启用 OpenShift Container Platform 集群组件进行通信所需的端口。*如需有关所需端口的详细信息，请参阅用户置备的基础架构* 部分的网络要求。



重要

默认情况下，OpenShift Container Platform 集群可以访问端口 1936，因为每个 control plane 节点都需要访问此端口。

避免使用 Ingress 负载均衡器公开此端口，因为这样做可能会导致公开敏感信息，如统计信息和指标（与 Ingress Controller 相关的统计信息和指标）。

4.

为集群设置所需的 DNS 基础架构。

a.

为 Kubernetes API、应用程序通配符、bootstrap 机器、control plane 机器和计算机器配置 DNS 名称解析。

b.

为 Kubernetes API、bootstrap 机器、control plane 机器和计算机器配置反向 DNS

解析。

如需有关 *OpenShift Container Platform DNS 要求的更多信息，请参阅用户置备* DNS 要求部分。

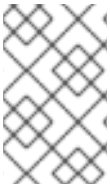
5. 验证您的 DNS 配置。

a. 从安装节点，针对 Kubernetes API 的记录名称、通配符路由和集群节点运行 DNS 查找。验证响应中的 IP 地址是否与正确的组件对应。

b. 从安装节点，针对负载均衡器和集群节点的 IP 地址运行反向 DNS 查找。验证响应中的记录名称是否与正确的组件对应。

有关详细的 *DNS 验证步骤，请参阅用户置备的基础架构*验证 DNS 解析部分。

6. 置备所需的 API 和应用程序入口负载平衡基础架构。有关 *要求的更多信息，请参阅用户置备的基础架构的负载平衡* 要求部分。



注意

某些负载平衡解决方案要求在初始化负载平衡之前，对群集节点进行 DNS 名称解析。

19.3.6. 验证用户置备的基础架构的 DNS 解析

您可以在在用户置备的基础架构上安装 OpenShift Container Platform 前验证 DNS 配置。

重要

本节中详述的验证步骤必须在安装集群前成功。

先决条件

- 已为您的用户置备的基础架构配置了所需的 DNS 记录。

流程

1. 从安装节点，针对 Kubernetes API 的记录名称、通配符路由和集群节点运行 DNS 查找。验证响应中包含的 IP 地址是否与正确的组件对应。

   a. 对 Kubernetes API 记录名称执行查询。检查结果是否指向 API 负载均衡器的 IP 地址：

   ```
   $ dig +noall +answer @<nameserver_ip> api.<cluster_name>.<base_domain> 1
   ```

   1 将 <nameserver_ip> 替换为 nameserver 的 IP 地址，<cluster_name> 替换为您的集群名称，<base_domain> 替换为您的基本域名。

   输出示例

   ```
   api.ocp4.example.com.  604800 IN A 192.168.1.5
   ```

   b. 对 Kubernetes 内部 API 记录名称执行查询。检查结果是否指向 API 负载均衡器的 IP 地址：

   ```
   $ dig +noall +answer @<nameserver_ip> api-int.<cluster_name>.<base_domain>
   ```

   输出示例

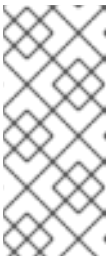
   ```
   api-int.ocp4.example.com.  604800 IN A 192.168.1.5
   ```

   c. 测试 *.apps.<cluster_name>.<base_domain> DNS 通配符查找示例。所有应用程序通配符查询都必须解析为应用程序入口负载均衡器的 IP 地址：

   ```
   $ dig +noall +answer @<nameserver_ip> random.apps.<cluster_name>.
   <base_domain>
   ```

输出示例

```
random.apps.ocp4.example.com.  604800 IN A 192.168.1.5
```



注意

在示例中，将相同的负载均衡器用于 Kubernetes API 和应用程序入口流量。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。

您可以使用另一个通配符值替换 random。例如，您可以查询到 OpenShift Container Platform 控制台的路由：

```
$ dig +noall +answer @<nameserver_ip> console-openshift-console.apps.
<cluster_name>.<base_domain>
```

输出示例

```
console-openshift-console.apps.ocp4.example.com. 604800 IN A 192.168.1.5
```

d. 针对 bootstrap DNS 记录名称运行查询。检查结果是否指向 bootstrap 节点的 IP 地址：

```
$ dig +noall +answer @<nameserver_ip> bootstrap.<cluster_name>.
<base_domain>
```

输出示例

```
bootstrap.ocp4.example.com.  604800 IN A 192.168.1.96
```

e. 使用此方法对 control plane 和计算节点的 DNS 记录名称执行查找。检查结果是否与每个节点的 IP 地址对应。

2. 从安装节点，针对负载均衡器和集群节点的 IP 地址运行反向 DNS 查找。验证响应中包含的记录名称是否与正确的组件对应。

a. 对 API 负载均衡器的 IP 地址执行反向查找。检查响应是否包含 Kubernetes API 和 Kubernetes 内部 API 的记录名称：

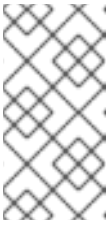
```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.5
```

输出示例

```
5.1.168.192.in-addr.arpa. 604800 IN PTR api-int.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. 604800 IN PTR api.ocp4.example.com. 2
```

1 为 Kubernetes 内部 API 提供记录名称。

2 为 Kubernetes API 提供记录名称。



注意

OpenShift Container Platform 应用程序通配符不需要 PTR 记录。针对应用程序入口负载均衡器的 IP 地址解析反向 DNS 解析不需要验证步骤。

b. 对 bootstrap 节点的 IP 地址执行反向查找。检查结果是否指向 bootstrap 节点的 DNS 记录名称：

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.96
```

输出示例

```
96.1.168.192.in-addr.arpa. 604800 IN PTR bootstrap.ocp4.example.com.
```

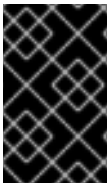
c. 使用此方法对 control plane 和计算节点的 IP 地址执行反向查找。检查结果是否与每个节点的 DNS 记录名称对应。

19.3.7. 为集群节点 SSH 访问生成密钥对

在 OpenShift Container Platform 安装过程中，您可以为安装程序提供 SSH 公钥。密钥通过它们的 Ignition 配置文件传递给 Red Hat Enterprise Linux CoreOS(RHCOS)节点，用于验证对节点的 SSH 访问。密钥添加到每个节点上 core 用户的 ~/.ssh/authorized_keys 列表中，这将启用免密码身份验证。

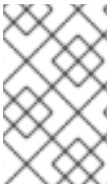
将密钥传递给节点后，您可以使用密钥对作为用户 核心 通过 SSH 连接到 RHCOS 节点。若要通过 SSH 访问节点，必须由 SSH 为您的本地用户管理私钥身份。

如果要通过 SSH 连接到集群节点来执行安装调试或灾难恢复，则必须在安装过程中提供 SSH 公钥。./openshift-install gather 命令还需要在集群节点上设置 SSH 公钥。



重要

不要在生产环境中跳过这个过程，在生产环境中需要灾难恢复和调试。



注意

您必须使用本地密钥，而不是使用特定平台方法配置 的密钥，如 AWS 密钥对。

流程

1. 如果您在本地计算机上没有可用于在集群节点上进行身份验证的现有 SSH 密钥对，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1

指定新 SSH 密钥的路径和文件名，如 ~/.ssh/id_ed25519。如果您已有密钥对，请确保您的公钥位于 ~/.ssh 目录中。

注意

如果您计划在 x86_64、ppc64le 和 s390x 架构上安装使用 RHEL 加密库（这些加密库已提交给 NIST 用于 FIPS 140-2/140-3 验证）的 OpenShift Container Platform 集群，则不要创建使用 ed25519 算法的密钥。相反，创建一个使用 rsa 或 ecdsa 算法的密钥。

2. 查看公共 SSH 密钥：

```
$ cat <path>/<file_name>.pub
```

例如，运行以下命令来查看 ~/.ssh/id_ed25519.pub 公钥：

```
$ cat ~/.ssh/id_ed25519.pub
```

3. 将 SSH 私钥身份添加到本地用户的 SSH 代理（如果尚未添加）。在集群节点上，或者要使用 ./openshift-install gather 命令，需要对该密钥进行 SSH 代理管理，才能在集群节点上进行免密码 SSH 身份验证。

注意

在某些发行版中，自动管理默认 SSH 私钥身份，如 ~/.ssh/id_rsa 和 ~/.ssh/id_dsa。

a. 如果 ssh-agent 进程尚未为您的本地用户运行，请将其作为后台任务启动：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果集群处于 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

4. 将 SSH 私钥添加到 ssh-agent ：

```
$ ssh-add <path>/<file_name> 1
```

1 指定 SSH 私钥的路径和文件名，如 ~/.ssh/id_ed25519.pub

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

后续步骤

- 安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

19.3.8. 手动创建安装配置文件

安装集群要求您手动创建安装配置文件。

先决条件

- 您在本地机器上有一个 SSH 公钥来提供给安装程序。该密钥将用于在集群节点上进行 SSH 身份验证，以进行调试和灾难恢复。

- 已获取 OpenShift Container Platform 安装程序和集群的 pull secret。

流程

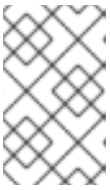
1. 创建一个安装目录来存储所需的安装资产：

```
$ mkdir <installation_directory>
```

重要

您必须创建一个目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

2. 自定义提供的 install-config.yaml 文件模板示例，并将其保存在 <installation_directory> 中。



注意

此配置文件必须命名为 install-config.yaml。

3. 备份 install-config.yaml 文件，以便您可以使用它安装多个集群。

重要

install-config.yaml 文件会在安装过程的下一步中使用。现在必须备份它。

其他资源

- IBM Power® 的安装配置参数。

19.3.8.1. IBM Power 的 install-config.yaml 文件示例

您可以自定义 install-config.yaml 文件，以指定有关 OpenShift Container Platform 集群平台的更多详情，或修改所需参数的值。

```
apiVersion: v1
baseDomain: example.com (1)
compute: (2)
- hyperthreading: Enabled (3)
  name: worker
  replicas: 0 (4)
  architecture: ppc64le
controlPlane: (5)
  hyperthreading: Enabled (6)
  name: master
  replicas: 3 (7)
  architecture: ppc64le
metadata:
  name: test (8)
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14 (9)
    hostPrefix: 23 (10)
  networkType: OVNKubernetes (11)
  serviceNetwork: (12)
  - 172.30.0.0/16
platform:
  none: {} (13)
fips: false (14)
pullSecret: '{"auths":{"<local_registry>": {"auth": "<credentials>","email":
"you@example.com"}}}' (15)
sshKey: 'ssh-ed25519 AAAA...' (16)
additionalTrustBundle: | (17)
  -----BEGIN CERTIFICATE-----
  ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
  -----END CERTIFICATE-----
imageContentSources: (18)
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-v4.0-art-dev
```

(1)

集群的基域。所有 DNS 记录都必须是这个基域的子域，并包含集群名称。

(2) (5)

controlPlane 部分是一个单个映射，但 compute 部分是一系列映射。为满足不同数据结构的要求，compute 部分的第一行必须以连字符 - 开头，controlPlane 部分 的第一行则不以连字符开

头。仅使用一个 control plane 池。

3 6

不支持并发多线程 (SMT)。

4

在用户置备的基础架构上安装 OpenShift Container Platform 时，必须将这个值设置为 0。在安装程序置备的安装中，参数控制集群为您创建和管理的计算机器数量。在用户置备的安装中，您必须在完成集群安装前手动部署计算机器。

注意

如果要安装一个三节点集群，在安装 Red Hat Enterprise Linux CoreOS(RHCOS)机器时不要部署任何计算机器。

7

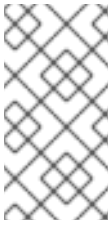
您添加到集群的 control plane 机器数量。由于集群使用这些值作为集群中的 etcd 端点数量，所以该值必须与您部署的 control plane 机器数量匹配。

8

您在 DNS 记录中指定的集群名称。

9

从中分配 Pod IP 地址的 IP 地址块。此块不得与现有物理网络重叠。这些 IP 地址用于 pod 网络。如果需要从外部网络访问 pod，您必须配置负载均衡器和路由器来管理流量。



注意

类 E CIDR 范围被保留以供以后使用。要使用 Class E CIDR 范围，您必须确保您的网络环境接受 Class E CIDR 范围内的 IP 地址。

10

分配给每个节点的子网前缀长度。例如，如果 hostPrefix 设为 23，则每个节点从 given cidr 中分配 a /23 子网，这样就能有 510（2^(32 - 23)- 2）个 pod IP 地址。如果需要从外部网络访问节点，请配置负载均衡器和路由器来管理流量。

11

12

用于服务 IP 地址的 IP 地址池。您只能输入一个 IP 地址池。此块不得与现有物理网络重叠。如果您需要从外部网络访问服务，请配置负载均衡器和路由器来管理流量。

13

您必须将平台设置为 none。您无法为 IBM Power® 基础架构提供额外的平台配置变量。

重要

使用平台类型 none 安装的集群无法使用一些功能，如使用 Machine API 管理计算机器。即使附加到集群的计算机器安装在通常支持该功能的平台上，也会应用这个限制。在安装后无法更改此参数。

14

是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 Red Hat Enterprise Linux (RHEL) 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅在 FIPS 模式中安装该系统。

当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS)时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。

15

对于 <local_registry>，请指定 registry 域名，以及您的镜像 registry 用来提供内容的可选端口。例如： registry.example.com 或 registry.example.com:5000。对于 <credentials>，请为您的镜像 registry 指定 base64 编码的用户名和密码。

16

Red Hat Enterprise Linux CoreOS(RHCOS)中 core 用户的 SSH 公钥。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 ssh-agent 进程使用的 SSH 密钥。

17

提供用于镜像 registry 的证书文件内容。

18

根据您用来镜像存储库的命令输出提供 imageContentSources 部分。

重要

- 使用 oc adm release mirror 命令时，请使用 imageContentSources 部分中的输出。
- 使用 oc mirror 命令时，请使用运行该命令结果的 ImageContentSourcePolicy 文件的 repositoryDigestMirrors 部分。
- ImageContentSourcePolicy 已被弃用。如需更多信息，请参阅*配置镜像 registry 存储库镜像*。

19.3.8.2. 在安装过程中配置集群范围的代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 install-config.yaml 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 install-config.yaml 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 Proxy 对象的 spec.noProxy 字段中添加站点来绕过代理。

注意

Proxy 对象 status.noProxy 字段使用安装配置中的 networking.machineNetwork[].cidr、networking.clusterNetwork[].cidr 和 networking.serviceNetwork[] 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，Proxy 对象 status.noProxy 字段也会使用实例元数据端点填充(169.254.169.254)。

流程

1. 编辑 install-config.yaml 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 http。

2. 用于创建集群外 HTTPS 连接的代理 URL。

3. 要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 . 以仅匹配子域。例如，.y.com 匹配 x.y.com，但不匹配 y.com。使用 * 绕过所有目的地的代理。

4. 如果提供，安装程序会在 openshift-config 命名空间中生成名为 user-ca-bundle 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 trusted-ca-bundle 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， Proxy 对象的 trustedCA 字段中

也会引用此配置映射。additionalTrustBundle 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

5 可选：决定 Proxy 对象的配置以引用 trustedCA 字段中 user-ca-bundle 配置映射的策略。允许的值是 Proxyonly 和 Always。仅在配置了 http/https 代理时，使用 Proxyonly 引用 user-ca-bundle 配置映射。使用 Always 始终引用 user-ca-bundle 配置映射。默认值为 Proxyonly。

注意

安装程序不支持代理的 readinessEndpoints 字段。

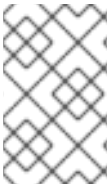
注意

如果安装程序超时，重启并使用安装程序的 wait-for 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. 保存该文件并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 cluster 的集群范围代理，该代理 使用 提供的 install-config.yaml 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 cluster Proxy 对象，但它会有一个空 spec。



注意

只支持名为 cluster 的 Proxy 对象，且无法创建额外的代理。

19.3.8.3. 配置三节点集群

另外，您可以在只由三台 control plane 机器组成的裸机集群中部署零台计算机器。这为集群管理员和开发人员提供了更小、效率更高的集群，用于测试、开发和生产。

在三节点 OpenShift Container Platform 环境中，三台 control plane 机器可以调度，这意味着应用程序工作负载被调度到它们上运行。

先决条件

- 您有一个现有的 install-config.yaml 文件。

流程

- 确保 install-config.yaml 文件中的计算副本数量设置为 0，如以下 计算 小节所示：

```
compute:
- name: worker
  platform: {}
  replicas: 0
```

注意

在用户置备的基础架构上安装 OpenShift Container Platform 时，无论您要部署的计算机器数量有多少，您必须将计算机器的 replicas 参数值设置为 0。在安装程序置备的安装中，参数控制集群为您创建和管理的计算机器数量。这不适用于手动部署计算机器的用户置备安装。

对于三节点集群安装，请按照以下步骤执行：

- 如果要部署一个带有零计算节点的三节点集群，Ingress Controller Pod 在 control plane 节点上运行。在三节点集群部署中，您必须配置应用程序入口负载均衡器，将 HTTP 和 HTTPS 流量路由到 control plane 节点。如需更多信息，*请参阅用户置备的基础架构的负载平衡要求* 部分。

- 在以下步骤中创建 Kubernetes 清单文件时，请确保 <installation_directory>/manifests/cluster-scheduler-02-config.yml 文件中的 mastersSchedulable 参数被设置为 true。这可让应用程序工作负载在 control plane 节点上运行。

- 在创建 Red Hat Enterprise Linux CoreOS(RHCOS)机器时，不要部署任何计算节点。

19.3.9. Cluster Network Operator 配置

集群网络的配置作为 Cluster Network Operator(CNO)配置的一部分指定，并存储在名为 cluster 的自定义资源(CR)对象中。CR 指定 operator.openshift.io API 组中的 Network API 的字段。

CNO 配置在集群安装过程中从 Network.config.openshift.io API 组中的 Network API 继承以下字段：

clusterNetwork

从中分配 Pod IP 地址的 IP 地址池。

serviceNetwork

服务的 IP 地址池.

defaultNetwork.type

集群网络插件。OVNKubernetes 是安装期间唯一支持的插件。

您可以通过在名为 cluster 的 CNO 对象中设置 defaultNetwork 对象的字段来为集群指定集群网络插件配置。

19.3.9.1. Cluster Network Operator 配置对象

下表中描述了 Cluster Network Operator(CNO)的字段：

表 19.28. Cluster Network Operator 配置对象

字段	类型	描述
metadata.name	**字符串**	CNO 对象的名称。这个名称始终是 **集群**。

spec.clusterNetwork	**array**	用于指定从哪些 IP 地址块分配 Pod IP 地址以及集群中每个节点的子网前缀长度的列表。例如：
		`spec:`
		`  clusterNetwork:`
		`  - cidr: 10.128.0.0/19`
		`  hostPrefix: 23`
		`  - cidr: 10.128.32.0/19`
		`  hostPrefix: 23`

字段	类型	描述
spec.serviceNetwork	**array**	服务的 IP 地址块。OpenShift SDN 和 OVN-Kubernetes 网络插件只支持服务网络的一个 IP 地址块。例如： spec: serviceNetwork: - 172.30.0.0/14 您只能在创建清单前在 **install-config.yaml** 文件中自定义此字段。该值在清单文件中是只读的。
spec.defaultNetwork	**object**	为集群网络配置网络插件。
spec.kubeProxyConfig	**object**	此对象的字段指定 kube-proxy 配置。如果使用 OVN-Kubernetes 集群网络供应商，则 kube-proxy 配置不会起作用。

defaultNetwork 对象配置

下表列出了 defaultNetwork 对象的值：

表 19.29. defaultNetwork 对象

字段	类型	描述
type	**字符串**	**OVNKubernetes**。Red Hat OpenShift Networking 网络插件在安装过程中被选择。此值在集群安装后无法更改。 **注意** OpenShift Container Platform 默认使用 OVN-Kubernetes 网络插件。OpenShift SDN 不再作为新集群的安装选择提供。
ovnKubernetesConfig	**object**	此对象仅对 OVN-Kubernetes 网络插件有效。

配置 OVN-Kubernetes 网络插件

下表描述了 OVN-Kubernetes 网络插件的配置字段：

表 19.30. ovnKubernetesConfig object

字段	类型	描述

字段	类型	描述
mtu	**integer**	Geneve（通用网络虚拟化封装）覆盖网络的最大传输单元 (MTU)。这根据主网络接口的 MTU 自动探测。您通常不需要覆盖检测到的 MTU。 如果自动探测的值不是您期望的值，请确认节点上主网络接口上的 MTU 是否正确。您不能使用这个选项更改节点上主网络接口的 MTU 值。 如果集群中不同节点需要不同的 MTU 值，则必须将此值设置为**比** 集群中的最低 MTU 值小 100。例如，如果集群中的某些节点的 MTU 为 **9001**，而某些节点的 MTU 为 **1500**，则必须将此值设置为 **1400**。
genevePort	**integer**	用于所有 Geneve 数据包的端口。默认值为 **6081**。此值在集群安装后无法更改。
ipsecConfig	**object**	指定用于自定义 IPsec 配置的配置对象。
ipv4	**object**	为 IPv4 设置指定配置对象。
ipv6	**object**	为 IPv6 设置指定配置对象。
policyAuditConfig	**object**	指定用于自定义网络策略审计日志的配置对象。如果未设置，则使用默认的审计日志设置。
gatewayConfig	**object**	可选：指定一个配置对象来自定义如何将出口流量发送到节点网关。 **注意** 在迁移出口流量时，工作负载和服务流量会受到一定影响，直到 Cluster Network Operator (CNO) 成功推出更改。

表 19.31. ovnKubernetesConfig.ipv4 object

字段	类型	描述
internalTransitSwitchSubnet	字符串	如果您的现有网络基础架构与 **100.88.0.0/16** IPv4 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。启用东西流量的分布式传输交换机的子网。此子网不能与 OVN-Kubernetes 或主机本身使用的任何其他子网重叠。必须足够大，以适应集群中的每个节点一个 IP 地址。 默认值为 **100.88.0.0/16**。

字段	类型	描述
internalJoinSubnet	字符串	如果您的现有网络基础架构与 **100.64.0.0/16** IPv4 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。您必须确保 IP 地址范围没有与 OpenShift Container Platform 安装使用的任何其他子网重叠。IP 地址范围必须大于可添加到集群的最大节点数。例如，如果 **clusterNetwork.cidr** 值为 **10.128.0.0/14**，并且 **clusterNetwork.hostPrefix** 值为 **/23**，则最大节点数量为 **2^(23-14)=512**。 默认值为 **100.64.0.0/16**。

表 19.32. ovnKubernetesConfig.ipv6 object

字段	类型	描述
internalTransitSwitchSubnet	字符串	如果您的现有网络基础架构与 **fd97::/64** IPv6 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。启用东西流量的分布式传输交换机的子网。此子网不能与 OVN-Kubernetes 或主机本身使用的任何其他子网重叠。必须足够大，以适应集群中的每个节点一个 IP 地址。 默认值为 **fd97::/64**。
internalJoinSubnet	字符串	如果您的现有网络基础架构与 **fd98::/64** IPv6 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。您必须确保 IP 地址范围没有与 OpenShift Container Platform 安装使用的任何其他子网重叠。IP 地址范围必须大于可添加到集群的最大节点数。 默认值为 **fd98::/64**。

表 19.33. policyAuditConfig object

字段	类型	描述
rateLimit	整数	每个节点每秒生成一次的消息数量上限。默认值为每秒 **20** 条消息。
maxFileSize	整数	审计日志的最大大小，以字节为单位。默认值为 **50000000** 或 50 MB。
maxLogFiles	整数	保留的日志文件的最大数量。

字段	类型	描述
目的地	字符串	以下附加审计日志目标之一： **libc** 主机上的 journald 进程的 libc **syslog（）** 函数。 **UDP:<host>:<port>** 一个 syslog 服务器。将 **<host>:<port>** 替换为 **syslog 服务器的主机** 和端口。 **Unix:<file>** 由 **<file>** 指定的 Unix 域套接字文件。 **null** 不要将审计日志发送到任何其他目标。
syslogFacility	字符串	syslog 工具，如 as **kern**，如 RFC5424 定义。默认值为 **local0。**

表 19.34. gatewayConfig object

字段	类型	描述
routingViaHost	**布尔值**	将此字段设置为 **true**，将来自 pod 的出口流量发送到主机网络堆栈。对于依赖于在内核路由表中手动配置路由的高级别安装和应用程序，您可能需要将出口流量路由到主机网络堆栈。默认情况下，出口流量在 OVN 中进行处理以退出集群，不受内核路由表中的特殊路由的影响。默认值为 **false**。 此字段与 Open vSwitch 硬件卸载功能有交互。如果将此字段设置为 **true**，则不会获得卸载的性能优势，因为主机网络堆栈会处理出口流量。
ipForwarding	**object**	您可以使用 **Network** 资源中的 **ipForwarding** 规格来控制 OVN-Kubernetes 管理接口上所有流量的 IP 转发。指定 **Restricted** 只允许 Kubernetes 相关流量的 IP 转发。指定 **Global** 以允许转发所有 IP 流量。对于新安装，默认值为 **Restricted**。对于到 OpenShift Container Platform 4.14 或更高版本的更新，默认值为 **Global**。
ipv4	**object**	可选：指定一个对象来为主机配置内部 OVN-Kubernetes 伪装地址，以服务 IPv4 地址的流量。
ipv6	**object**	可选：指定一个对象来为主机配置内部 OVN-Kubernetes 伪装地址，以服务 IPv6 地址的流量。

表 19.35. gatewayConfig.ipv4 对象

字段	类型	描述
internalMasqueradeSubnet	**字符串**	内部使用的伪装 IPv4 地址，以启用主机服务流量。主机配置了这些 IP 地址和共享网关网桥接口。默认值为 **169.254.169.0/29**。

表 19.36. gatewayConfig.ipv6 对象

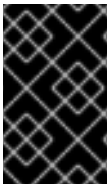
字段	类型	描述
internalMasqueradeSubnet	**字符串**	内部使用的伪装 IPv6 地址，以启用主机服务流量。主机配置了这些 IP 地址和共享网关网桥接口。默认值为 **fd69::/125**。

表 19.37. ipsecConfig 对象

字段	类型	描述
模式	**字符串**	指定 IPsec 实现的行为。必须是以下值之一： • **Disabled**: 在集群节点上不启用 IPsec。 • **External** ：对于带有外部主机的网络流量，启用 IPsec。 • **Full**: IPsec 为带有外部主机的 pod 流量和网络流量启用 IPsec。

启用 IPSec 的 OVN-Kubernetes 配置示例

```
defaultNetwork:
  type: OVNKubernetes
  ovnKubernetesConfig:
    mtu: 1400
    genevePort: 6081
      ipsecConfig:
        mode: Full
```



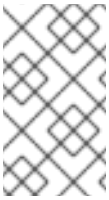
重要

使用 OVNKubernetes 可能会导致 IBM Power® 上的堆栈耗尽问题。

kubeProxyConfig 对象配置（仅限 OpenShiftSDN 容器网络接口）

kubeProxyConfig 对象的值在下表中定义：

表 19.38. kubeProxyConfig object

字段	类型	描述
iptablesSyncPeriod	**字符串**	**iptables** 规则的刷新周期。默认值为 **30s**。有效的后缀包括 **s**、**m** 和 **h**，具体参见 Go 时间 包 文档。  **注意** 由于 OpenShift Container Platform 4.3 及更高版本中引进了性能改进，不再需要调整 **iptablesSyncPeriod** 参数。

proxyArguments.iptables-min-sync-period	**array**	刷新 **iptables** 规则前的最短持续时间。此字段确保刷新的频率不会过于频繁。有效的后缀包括 **s**、**m** 和 **h**，具体参见 Go **time** 软件包。默认值为：
		`kubeProxyConfig:`
		`  proxyArguments:`
		`  iptables-min-sync-period:`
		`  - 0s`

19.3.10. 创建 Kubernetes 清单和 Ignition 配置文件

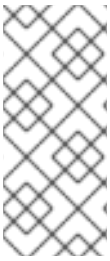
由于您必须修改一些集群定义文件并手动启动集群机器，因此您必须生成 Kubernetes 清单和 Ignition 配置文件来配置机器。

安装配置文件转换为 Kubernetes 清单。清单嵌套到 Ignition 配置文件中，稍后用于配置集群机器。

重要

- OpenShift Container Platform 安装程序生成的 Ignition 配置文件包含 24 小时后过期的证书，然后在该时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 node-bootstrapper 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。

- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。



注意

生成清单和 Ignition 文件的安装程序是特定的架构，可以从 客户端镜像镜像获取。安装程序的 Linux 版本（没有构架 postfix）仅在 ppc64le 上运行。此安装程序也可用作 Mac OS 版本。

先决条件

- 已获得 OpenShift Container Platform 安装程序。对于受限网络安装，这些文件位于您的镜像主机上。

- 已创建 install-config.yaml 安装配置文件。

流程

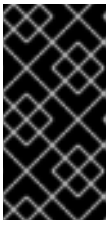
1. 进入包含 OpenShift Container Platform 安装程序的目录，并为集群生成 Kubernetes 清单：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 对于 <installation_directory>，请指定包含您创建的 install-config.yaml 文件的安装目录。

警告

如果您要安装一个三节点集群，请跳过以下步骤，以便可以调度 control plane 节点。



重要

当您将 control plane 节点从默认的不可调度配置为可以调度时，需要额外的订阅。这是因为 control plane 节点变为计算节点。

2. 检查 <installation_directory>/manifests/cluster-scheduler-02-config.yml Kubernetes 清单文件中的 mastersSchedulable 参数是否已设置为 false。此设置可防止在 control plane 机器上调度 pod：

   a. 打开 <installation_directory>/manifests/cluster-scheduler-02-config.yml 文件。

   b. 找到 mastersSchedulable 参数，并确保它被设置为 false。

   c. 保存并退出 文件。

3. 要创建 Ignition 配置文件，请从包含安装程序的目录运行以下命令：

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

1 对于 <installation_directory>，请指定相同的安装目录。

为安装目录中的 bootstrap、control plane 和计算节点创建 Ignition 配置文件。kubeadmin-password 和 kubeconfig 文件在 ./<installation_directory>/auth 目录中创建：

```
.
```

```
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

19.3.11. 安装 RHCOS 并启动 OpenShift Container Platform bootstrap 过程

要在您置备的 IBM Power® 基础架构上安装 OpenShift Container Platform，您必须在机器上安装 Red Hat Enterprise Linux CoreOS (RHCOS)。安装 RHCOS 时，您必须为您要安装的机器类型提供 OpenShift Container Platform 安装程序生成的 Ignition 配置文件。如果您配置了适当的网络、DNS 和负载均衡基础架构，OpenShift Container Platform bootstrap 过程会在 RHCOS 机器重启后自动启动。

按照以下步骤使用 ISO 镜像或网络 PXE 引导在机器上安装 RHCOS。

19.3.11.1. 使用 ISO 镜像安装 RHCOS

您可以使用 ISO 镜像在机器上安装 RHCOS。

先决条件

- 已为集群创建 Ignition 配置文件。
- 您已配置了适当的网络、DNS 和负载平衡基础架构。
- 您有一个可以从计算机以及您创建的机器访问的 HTTP 服务器。
- 您已参阅 *Advanced RHCOS 安装配置* 部分来了解配置功能的不同方法，如网络和磁盘分区。

流程

1. 获取每个 Ignition 配置文件的 SHA512 摘要。例如，您可以在运行 Linux 的系统上使用以下内容来获取 bootstrap.ign Ignition 配置文件的 SHA512 摘要：

```
$ sha512sum <installation_directory>/bootstrap.ign
```

-

后续步骤中会向 coreos-installer 提供摘要，以验证集群节点上 Ignition 配置文件的真实性。

2. 将安装程序创建的 bootstrap、control plane 和计算节点 Ignition 配置文件上传到 HTTP 服务器。注意这些文件的 URL。

重要

您可以在 Ignition 配置中添加或更改配置设置，然后将其保存到 HTTP 服务器。如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

3. 从安装主机上，验证 Ignition 配置文件是否在 URL 上可用。以下示例获取 bootstrap 节点的 Ignition 配置文件：

```
$ curl -k http://<HTTP_server>/bootstrap.ign 1
```

输出示例

```
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0{"ignition":
{"version":"3.2.0"},"passwd":{"users":[{"name":"core","sshAuthorizedKeys":["ssh-
rsa...
```

在命令中将 bootstrap.ign 替换为 master.ign 或 worker.ign，以验证 control plane 和计算节点的 Ignition 配置文件是否可用。

4. 虽然可以从 RHCOS 镜像镜像页面获取您选择的操作系统实例安装方法所需的 RHCOS 镜像，但推荐的方法是从 openshift-install 命令的输出获取 RHCOS 镜像的正确版本：

```
$ openshift-install coreos print-stream-json | grep '\.iso[^.]'
```

输出示例

```
"location": "<url>/art/storage/releases/rhcos-4.16-aarch64/<release>/aarch64/rhcos-
<release>-live.aarch64.iso",
"location": "<url>/art/storage/releases/rhcos-4.16-ppc64le/<release>/ppc64le/rhcos-
<release>-live.ppc64le.iso",
"location": "<url>/art/storage/releases/rhcos-4.16-s390x/<release>/s390x/rhcos-
<release>-live.s390x.iso",
"location": "<url>/art/storage/releases/rhcos-4.16/<release>/x86_64/rhcos-<release>-
live.x86_64.iso",
```

重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每个发行版本而改变。您必须下载最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。如果可用，请使用与 OpenShift Container Platform 版本匹配的镜像版本。这个过程只使用 ISO 镜像。此安装类型不支持 RHCOS qcow2 镜像。

ISO 文件名类似以下示例：

```
rhcos-<version>-live.<architecture>.iso
```

5. 使用 ISO 启动 RHCOS 安装。使用以下安装选项之一：

   - 将 ISO 映像刻录到磁盘并直接启动。
   - 使用 light-out 管理(LOM)接口使用 ISO 重定向。

6. 在不指定任何选项或中断实时引导序列的情况下引导 RHCOS ISO 镜像。等待安装程序在 RHCOS live 环境中引导进入 shell 提示符。

注意

可以中断 RHCOS 安装引导过程来添加内核参数。但是，在这个 ISO 过程中，您应该使用以下步骤中所述的 coreos-installer 命令，而不是添加内核参数。

7. 运行 coreos-installer 命令并指定满足您的安装要求的选项。您至少必须指定指向节点类型的 Ignition 配置文件的 URL，以及您要安装到的设备：

```
$ sudo coreos-installer install --ignition-url=http://<HTTP_server>/<node_type>.ign
<device> --ignition-hash=sha512-<digest> 1 2
```

1 1 您必须使用 sudo 运行 coreos-installer 命令，因为 core 用户没有执行安装所需的 root 权限。

2 当 Ignition 配置文件通过 HTTP URL 获取时，需要 --ignition-hash 选项来验证集群节点上 Ignition 配置文件的真实性。<digest> 是上一步中获取的 Ignition 配置文件 SHA512 摘要。

注意

如果要通过使用 TLS 的 HTTPS 服务器提供 Ignition 配置文件，您可以在运行 coreos-installer 前将内部证书颁发机构(CA)添加到系统信任存储中。

以下示例将引导节点安装初始化到 /dev/sda 设备。bootstrap 节点的 Ignition 配置文件从 IP 地址 192.168.1.2 的 HTTP Web 服务器获取：

```
$ sudo coreos-installer install --ignition-
url=http://192.168.1.2:80/installation_directory/bootstrap.ign /dev/sda --ignition-
hash=sha512-
a5a2d43879223273c9b60af66b44202a1d1248fc01cf156c46d4a79f552b6bad47bc8cc78dd
f0116e80c59d2ea9e32ba53bc807afbca581aa059311def2c3e3b
```

8. 在机器的控制台上监控 RHCOS 安装的进度。

重要

在开始安装 OpenShift Container Platform 之前，确保每个节点中安装成功。观察安装过程可以帮助确定可能会出现 RHCOS 安装问题的原因。

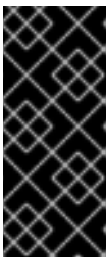
9. 安装 RHCOS 后，您必须重启系统。系统重启过程中，它会应用您指定的 Ignition 配置文件。

10. 检查控制台输出，以验证 Ignition 是否运行。

示例命令

```
Ignition: ran on 2022/03/14 14:48:33 UTC (this boot)
Ignition: user-provided config was applied
```

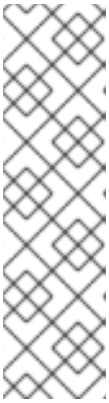
11. 继续为集群创建其他机器。



重要

此时您必须创建 bootstrap 和 control plane 机器。如果 control plane 机器不可调度，请在安装 OpenShift Container Platform 前至少创建两台计算机器。

如果存在所需的网络、DNS 和负载均衡器基础架构，OpenShift Container Platform bootstrap 过程会在 RHCOS 节点重启后自动启动。



注意

RHCOS 节点不包含 core 用户的默认密码。您可以使用可访问 SSH 私钥的用户的身份运行 ssh core@<node>.<cluster_name>.<base_domain > 来访问节点，该私钥与您在 install_config.yaml 文件中指定的公钥配对。运行 RHCOS 的 OpenShift Container Platform 4 集群节点不可变，它依赖于 Operator 来应用集群更改。不建议使用 SSH 访问集群节点。但是，当调查安装问题时，如果 OpenShift Container Platform API 不可用，或者 kubelet 在目标节点上无法正常工作，则调试或灾难恢复可能需要 SSH 访问。

19.3.11.1.1. 高级 RHCOS 安装参考

本节演示了网络配置和其他高级选项，允许您修改 Red Hat Enterprise Linux CoreOS(RHCOS)手动安装过程。下表描述了您可以用于 RHCOS live 安装程序和 coreos-installer 命令的内核参数和命令行选项。

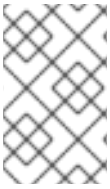
19.3.11.1.1.1. ISO 安装的网络和绑定选项

如果从 ISO 镜像安装 RHCOS，您可以在引导镜像时手动添加内核参数，以便为节点配置网络。如果没有指定网络参数，当 RHCOS 检测到需要网络来获取 Ignition 配置文件时，在 initramfs 中激活 DHCP。

重要

在手动添加网络参数时，还必须添加 rd.neednet=1 内核参数，以便在 initramfs 中启动网络。

以下信息提供了在 RHCOS 节点上为 ISO 安装配置网络和绑定的示例。示例描述了如何使用 ip=、name server = 和 bond= 内核参数。



注意

添加内核参数时顺序非常重要： ip=、name server=，然后 bond=。

网络选项在系统引导过程中传递给 dracut 工具。有关 dracut 支持的网络选项的更多信息，请参阅 dracut.cmdline 手册页。

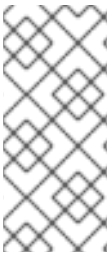
以下示例是 ISO 安装的网络选项。

配置 DHCP 或静态 IP 地址

要配置 IP 地址，可使用 DHCP(ip=dhcp)或设置单独的静态 IP 地址(ip=<host_ip>)。如果设置静态 IP，则必须在每个节点上识别 DNS 服务器 IP 地址（名称服务器=<dns_ip>）。以下示例集：

- 节点的 IP 地址为 10.10.10.2
- 网关地址为 10.10.10.254
- 子网掩码为 255.255.255.0
- 到 core0.example.com 的主机名
- DNS 服务器地址为 4.4.4.41
- 自动配置值为 none。当以静态方式配置 IP 网络时，不需要自动配置。

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
nameserver=4.4.4.41
```



注意

当您使用 DHCP 为 RHCOS 机器配置 IP 寻址时，机器还通过 DHCP 获取 DNS 服务器信息。对于基于 DHCP 的部署，您可以通过 DHCP 服务器配置定义 RHCOS 节点使用的 DNS 服务器地址。

配置没有静态主机名的 IP 地址

您可以在不分配静态主机名的情况下配置 IP 地址。如果用户没有设置静态主机名，则会提取并通过反向 DNS 查找自动设置。要在没有静态主机名的情况下配置 IP 地址，请参考以下示例：

- 节点的 IP 地址为 10.10.10.2
- 网关地址为 10.10.10.254

- 子网掩码为 255.255.255.0
- DNS 服务器地址为 4.4.4.41
- 自动配置值为 none。当以静态方式配置 IP 网络时，不需要自动配置。

```
ip=10.10.10.2::10.10.10.254:255.255.255.0::enp1s0:none
nameserver=4.4.4.41
```

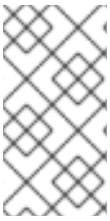
指定多个网络接口

您可以通过设置多个 ip= 条目来指定多个网络接口。

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
ip=10.10.10.3::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none
```

配置默认网关和路由

可选：您可以通过设置 a rd.route= 值来配置到额外网络的路由。



注意

当您配置一个或多个网络时，需要一个默认网关。如果额外网络网关与主要网络网关不同，则默认网关必须是主要网络网关。

- 运行以下命令来配置默认网关：

```
ip=::10.10.10.254::::
```

- 输入以下命令为额外网络配置路由：

```
rd.route=20.20.20.0/24:20.20.20.254:enp2s0
```

在单个接口中禁用 DHCP

您可以在单一接口中禁用 DHCP，例如当有两个或者多个网络接口时，且只有一个接口被使用。在示例中，enp1s0 接口具有一个静态网络配置，而 enp2s0 禁用了 DHCP，不使用它：

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
ip=::::core0.example.com:enp2s0:none
```

合并 DHCP 和静态 IP 配置

您可以将系统上的 DHCP 和静态 IP 配置与多个网络接口合并，例如：

```
ip=enp1s0:dhcp
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none
```

在独立接口上配置 VLAN

可选：您可以使用 vlan= 参数在单个接口上配置 VLAN。

- 要在网络接口中配置 VLAN 并使用静态 IP 地址，请运行以下命令：

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0.100:none
vlan=enp2s0.100:enp2s0
```

- 要在网络接口中配置 VLAN 并使用 DHCP，请运行以下命令：

```
ip=enp2s0.100:dhcp
vlan=enp2s0.100:enp2s0
```

提供多个 DNS 服务器

您可以通过为每个服务器添加一个 nameserver= 条目来提供多个 DNS 服务器，例如

```
nameserver=1.1.1.1
nameserver=8.8.8.8
```

将多个网络接口绑定到一个接口

可选：您可以使用 bond= 选项将多个网络接口绑定到一个接口。请参见以下示例：

- 配置绑定接口的语法为：bond=<name>[:<network_interfaces>][:options]

  <name> 是绑定设备名称 (bond0)、<network_interfaces> 代表以逗号分隔的物理（以太网）接口列表(em1,em2)，*options* 是用逗号分开的绑定选项列表。输入 modinfo bonding 查看可用选项。

- 当使用 bond= 创建绑定接口时，您必须指定如何分配 IP 地址以及绑定接口的其他信息。
  - 要将绑定接口配置为使用 DHCP，请将绑定的 IP 地址设置为 dhcp。例如：

```
bond=bond0:em1,em2:mode=active-backup
ip=bond0:dhcp
```

  - 要将绑定接口配置为使用静态 IP 地址，请输入您需要的特定 IP 地址和相关信息。例如：

```
bond=bond0:em1,em2:mode=active-backup
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0:none
```

将多个 SR-IOV 网络接口绑定到双端口 NIC 接口

重要

支持与为 SR-IOV 设备启用 NIC 分区关联的第 1 天操作只是一个技术预览功能。技术预览功能不受红帽产品服务等级协议（SLA）支持，且功能可能并不完整。红帽不推荐在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。

有关红帽技术预览功能支持范围的更多信息，请参阅技术预览功能支持范围。

可选：您可以使用 bond= 选项将多个 SR-IOV 网络接口绑定到双端口 NIC 接口。

在每个节点上，您必须执行以下任务：

1. 按照管理 SR-IOV 设备中的指导创建 SR-IOV 虚拟功能(VF)。按照"将 SR-IOV 网络设备附加到虚拟机"部分中的步骤操作。

2. 创建绑定，将所需的 VF 附加到绑定，并根据配置网络绑定的指导设置绑定链接状态。按照任何描述的步骤创建绑定。

以下示例演示了您必须使用的语法：

- 配置绑定接口的语法为：bond=<name>[:<network_interfaces>][:options]

  <name> 是绑定设备名称 (bond0)、<network_interfaces> 由内核中已知的名称来代表虚拟功能(VF)，并显示在 ip link 命令的输出中 (eno1f0,eno2f0)， *options* 是以逗号分隔的绑定选项列表。输入 modinfo bonding 查看可用选项。

- 当使用 bond= 创建绑定接口时，您必须指定如何分配 IP 地址以及绑定接口的其他信息。

  - 要将绑定接口配置为使用 DHCP，请将绑定的 IP 地址设置为 dhcp。例如：

    ```
    bond=bond0:eno1f0,eno2f0:mode=active-backup
    ip=bond0:dhcp
    ```

  - 要将绑定接口配置为使用静态 IP 地址，请输入您需要的特定 IP 地址和相关信息。例如：

    ```
    bond=bond0:eno1f0,eno2f0:mode=active-backup
    ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0:none
    ```

19.3.11.2. 使用 PXE 引导安装 RHCOS

您可以使用 PXE 引导在机器上安装 RHCOS。

先决条件

- 已为集群创建 Ignition 配置文件。
- 您已配置了适当的网络、DNS 和负载平衡基础架构。
- 您已配置了适当的 PXE 基础架构。
- 您有一个可以从计算机以及您创建的机器访问的 HTTP 服务器。
- 您已参阅 *Advanced RHCOS 安装配置* 部分来了解配置功能的不同方法，如网络和磁盘分区。

流程

1. 将安装程序创建的 bootstrap、control plane 和计算节点 Ignition 配置文件上传到 HTTP 服务器。注意这些文件的 URL。

重要

您可以在 Ignition 配置中添加或更改配置设置，然后将其保存到 HTTP 服务器。如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

2. 从安装主机上，验证 Ignition 配置文件是否在 URL 上可用。以下示例获取 bootstrap 节点的 Ignition 配置文件：

```
$ curl -k http://<HTTP_server>/bootstrap.ign 1
```

输出示例

```
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0{"ignition":
{"version":"3.2.0"},"passwd":{"users":[{"name":"core","sshAuthorizedKeys":["ssh-
rsa...
```

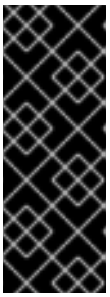
在命令中将 bootstrap.ign 替换为 master.ign 或 worker.ign，以验证 control plane 和计算节点的 Ignition 配置文件是否可用。

3. 虽然可以从 RHCOS image mirror 页面获取您选择的操作系统实例所需的 RHCOS kernel、initramfs 和 rootfs 文件，但推荐的方法是从 openshift-install 命令的输出中获取 RHCOS 文件的正确版本：

```
$ openshift-install coreos print-stream-json | grep -Eo '"https.*(kernel-
|initramfs.|rootfs.)\w+(\.img)?"'
```

输出示例

```
"<url>/art/storage/releases/rhcos-4.16-aarch64/<release>/aarch64/rhcos-<release>-live-kernel-aarch64"
"<url>/art/storage/releases/rhcos-4.16-aarch64/<release>/aarch64/rhcos-<release>-live-initramfs.aarch64.img"
"<url>/art/storage/releases/rhcos-4.16-aarch64/<release>/aarch64/rhcos-<release>-live-rootfs.aarch64.img"
"<url>/art/storage/releases/rhcos-4.16-ppc64le/49.84.202110081256-0/ppc64le/rhcos-<release>-live-kernel-ppc64le"
"<url>/art/storage/releases/rhcos-4.16-ppc64le/<release>/ppc64le/rhcos-<release>-live-initramfs.ppc64le.img"
"<url>/art/storage/releases/rhcos-4.16-ppc64le/<release>/ppc64le/rhcos-<release>-live-rootfs.ppc64le.img"
"<url>/art/storage/releases/rhcos-4.16-s390x/<release>/s390x/rhcos-<release>-live-kernel-s390x"
"<url>/art/storage/releases/rhcos-4.16-s390x/<release>/s390x/rhcos-<release>-live-initramfs.s390x.img"
"<url>/art/storage/releases/rhcos-4.16-s390x/<release>/s390x/rhcos-<release>-live-rootfs.s390x.img"
"<url>/art/storage/releases/rhcos-4.16/<release>/x86_64/rhcos-<release>-live-kernel-x86_64"
"<url>/art/storage/releases/rhcos-4.16/<release>/x86_64/rhcos-<release>-live-initramfs.x86_64.img"
"<url>/art/storage/releases/rhcos-4.16/<release>/x86_64/rhcos-<release>-live-rootfs.x86_64.img"
```



重要

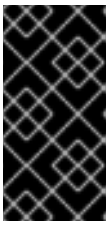
RHCOS 工件可能不会随着 OpenShift Container Platform 的每个发行版本而改变。您必须下载最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。这个过程只使用下面描述的适当 kernel、initram fs 和 rootfs 工件。此安装类型不支持 RHCOS QCOW2 镜像。

文件名包含 OpenShift Container Platform 版本号。它们类似以下示例：

- kernel:rhcos-<version>-live-kernel-<architecture>
- initramfs: rhcos-<version>-live-initramfs.<architecture>.img

- **rootfs: rhcos-<version>-live-rootfs.<architecture>.img**

4. **将 rootfs、kernel 和 initramfs 文件上传到 HTTP 服务器。**



重要

如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

5. **配置网络引导基础架构，以便在安装 RHCOS 后机器从本地磁盘启动。**

6. 为 **RHCOS 镜像配置 PXE 安装并开始安装。**

为您的环境修改以下示例菜单条目，并验证能否正确访问镜像和 Ignition 文件：

```
DEFAULT pxeboot
TIMEOUT 20
PROMPT 0
LABEL pxeboot
    KERNEL http://<HTTP_server>/rhcos-<version>-live-kernel-<architecture> (1)
    APPEND initrd=http://<HTTP_server>/rhcos-<version>-live-initramfs.<architecture>.img
coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-rootfs.<architecture>.img
coreos.inst.install_dev=/dev/sda coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign
(2) (3)
```

(1) (1)

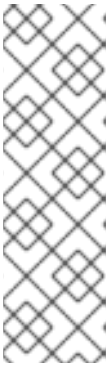
指定上传到 HTTP 服务器的 live kernel 文件位置。URL 必须是 HTTP、TFTP 或 FTP；不支持 HTTPS 和 NFS。

(2)

如果您使用多个 NIC，请在 ip 选项中指定一个接口。例如，要在名为 eno1 的 NIC 上使用 DHCP，请设置 ip=eno1:dhcp。

(3)

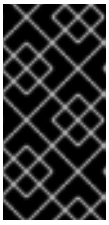
指定上传到 HTTP 服务器的 RHCOS 文件的位置。initrd 参数值是 initramfs 文件的位置，coreos.live.rootfs_url 参数值是 rootfs 文件的位置，coreos.inst.ignition_url 参数值则是 bootstrap Ignition 配置文件的位置。您还可以在 APPEND 行中添加更多内核参数来配置联网或其他引导选项。



注意

此配置不会在图形控制台的机器上启用串行控制台访问。要配置不同的控制台，请在 APPEND 行中添加一个或多个 console= 参数。例如，添加 console=tty0 console=ttyS0 以将第一个 PC 串口设置为主控制台，并将图形控制台设置为二级控制台。如需更多信息，请参阅如何在 Red Hat Enterprise Linux 中设置串行终端和/或控制台？ 和"启用 PXE 和 ISO 安装的串行控制台"部分。

7. 在机器的控制台上监控 RHCOS 安装的进度。



重要

在开始安装 OpenShift Container Platform 之前，确保每个节点中安装成功。观察安装过程可以帮助确定可能会出现 RHCOS 安装问题的原因。

8. 安装 RHCOS 后，系统会重启。在重启过程中，系统会应用您指定的 Ignition 配置文件。

9. 检查控制台输出，以验证 Ignition 是否运行。

示例命令

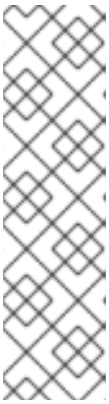
```
Ignition: ran on 2022/03/14 14:48:33 UTC (this boot)
Ignition: user-provided config was applied
```

10. 继续为集群创建机器。

重要

此时您必须创建 bootstrap 和 control plane 机器。如果 control plane 机器不可调度，请在安装集群前至少创建两台计算机器。

如果存在所需的网络、DNS 和负载均衡器基础架构，OpenShift Container Platform bootstrap 过程会在 RHCOS 节点重启后自动启动。



注意

RHCOS 节点不包含 core 用户的默认密码。您可以使用可访问 SSH 私钥的用户的身份运行 ssh core@<node>.<cluster_name>.<base_domain > 来访问节点，该私钥与您在 install_config.yaml 文件中指定的公钥配对。运行 RHCOS 的 OpenShift Container Platform 4 集群节点不可变，它依赖于 Operator 来应用集群更改。不建议使用 SSH 访问集群节点。但是，当调查安装问题时，如果 OpenShift Container Platform API 不可用，或者 kubelet 在目标节点上无法正常工作，则调试或灾难恢复可能需要 SSH 访问。

19.3.11.3. 在 RHCOS 上启用带有内核参数的多路径

在 OpenShift Container Platform 版本 4.16 中，您可以在安装过程中为置备的节点启用多路径。RHCOS 支持主磁盘上的多路径。多路径为硬件故障提供更强大的弹性，以实现更高的主机可用性。

在初始集群创建过程中，您可能需要在所有 master 节点或 worker 节点中添加内核参数。要在 master 节点或 worker 节点中添加内核参数，您可以创建一个 MachineConfig 对象，并将该对象注入 Ignition 在集群设置过程中使用的清单文件集合中。

流程

1. 进入包含安装程序的目录，并为集群生成 Kubernetes 清单：

```
$ ./openshift-install create manifests --dir <installation_directory>
```

2. 决定您要将内核参数添加到 worker 节点还是 control plane 节点。

- 创建机器配置文件。例如，创建一个 99-master-kargs-mpath.yaml 来指示集群添加 master 标签并识别多路径内核参数：

```
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: "master"
  name: 99-master-kargs-mpath
spec:
```

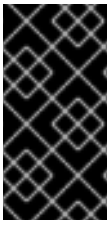
```
  kernelArguments:
    - 'rd.multipath=default'
    - 'root=/dev/disk/by-label/dm-mpath-root'
```

3. 在 worker 节点上启用多路径：

- 创建机器配置文件。例如，创建一个 99-worker-kargs-mpath.yaml 来指示集群添加 worker 标签并识别多路径内核参数：

```
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: "worker"
  name: 99-worker-kargs-mpath
spec:
  kernelArguments:
    - 'rd.multipath=default'
    - 'root=/dev/disk/by-label/dm-mpath-root'
```

现在，您可以继续创建集群。



重要

需要额外的安装后步骤才能完全启用多路径。如需更多信息，请参阅 *安装后机器配置任务* 中的"使用 RHCOS 上内核参数启用多路径"。

如果 MPIO 失败，请使用 bootlist 命令使用其他逻辑设备名称更新引导设备列表。命令显示引导列表，它会在系统以正常模式引导时指定可能的引导设备。

a. 要显示引导列表并在系统以正常模式引导时指定可能的引导设备，请输入以下命令：

```
$ bootlist -m normal -o
sda
```

b. 要为常规模式更新引导列表并添加备用设备名称，请输入以下命令：

```
$ bootlist -m normal -o /dev/sdc /dev/sdd /dev/sde
sdc
sdd
```

```
sde
```

如果原始引导磁盘路径停机，节点会从在普通引导设备列表中注册的替代设备重启。

19.3.12. 等待 bootstrap 过程完成

OpenShift Container Platform bootstrap 过程在集群节点首次引导到安装到磁盘的持久 RHCOS 环境后开始。通过 Ignition 配置文件提供的配置信息用于初始化 bootstrap 过程并在机器上安装 OpenShift Container Platform。您必须等待 bootstrap 过程完成。

先决条件

- 已为集群创建 Ignition 配置文件。
- 您已配置了适当的网络、DNS 和负载平衡基础架构。
- 已获得安装程序，并为集群生成 Ignition 配置文件。
- 已在集群机器上安装 RHCOS，并提供 OpenShift Container Platform 安装程序生成的 Ignition 配置文件。

流程

1. 监控 bootstrap 过程：

```
$ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \ 1
  --log-level=info 2
```

1. 对于 <installation_directory>，请指定安装文件保存到的目录的路径。
2. 要查看不同的安装详情，请指定 warn、debug 或 error，而不是 info。

输出示例

```
INFO Waiting up to 30m0s for the Kubernetes API at
https://api.test.example.com:6443...
INFO API v1.29.4 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
```

当 Kubernetes API 服务器提示已在 control plane 机器上引导它时，该命令会成功。

2. bootstrap 过程完成后，从负载均衡器中删除 bootstrap 机器。

重要

此时您必须从负载均衡器中删除 bootstrap 机器。您还可以删除或重新格式化 bootstrap 机器本身。

19.3.13. 使用 CLI 登录集群

您可以通过导出集群 kubeconfig 文件，以默认系统用户身份登录集群。kubeconfig 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。
- 已安装 oc CLI。

流程

1. 导出 kubeadmin 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 <installation_directory>，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 oc 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

19.3.14. 批准机器的证书签名请求

当您将机器添加到集群时，会为您添加的每台机器生成两个待处理证书签名请求(CSR)。您必须确认这些 CSR 已获得批准，或根据需要自行批准。必须首先批准客户端请求，然后批准服务器请求。

先决条件

- 您已将机器添加到集群中。

流程

1. 确认集群可以识别这些机器：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0  Ready    master  63m  v1.29.4
master-1  Ready    master  63m  v1.29.4
master-2  Ready    master  64m  v1.29.4
```

输出中列出了您创建的所有机器。

注意

在有些 CSR 被批准前，前面的输出可能不包括计算节点（也称为 worker 节点）。

2. 检查待处理的 CSR，并确保添加到集群中的每台机器都有 Pending 或 Approved 状态的客户端请求：

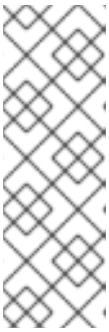
```
$ oc get csr
```

输出示例

```
NAME        AGE    REQUESTOR                                                   CONDITION
csr-8b2br   15m    system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
csr-8vnps   15m    system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
...
```

在本例中，两台机器加入集群。您可能会在列表中看到更多已批准的 CSR。

3. 如果 CSR 没有获得批准，在您添加的机器的所有待处理 CSR 都处于 Pending 状态后，请批准集群机器的 CSR：



注意

由于 CSR 会自动轮转，因此请在将机器添加到集群后一小时内批准您的 CSR。如果没有在一小时内批准它们，证书将会轮转，每个节点会存在多个证书。您必须批准所有这些证书。批准客户端 CSR 后，Kubelet 为服务证书创建一个二级 CSR，这需要手动批准。然后，如果 Kubelet 请求具有相同参数的新证书，则后续提供证书续订请求由 machine-approver 自动批准。

注意

对于在未启用机器 API 的平台上运行的集群，如裸机和其他用户置备的基础架构，您必须实施一种方法来自动批准 kubelet 提供证书请求(CSR)。如果没有批准请求，则 oc exec、ocrsh 和 oc logs 命令将无法成功，因为 API 服务器连接到 kubelet 时需要服务证书。与 Kubelet 端点联系的任何操作都需要此证书批准。该方法必须监视新的 CSR，确认 CSR 由 system: node 或 system:admin 组中的 node-bootstrapper 服务帐户提交，并确认节点的身份。

- 要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 <csr_name> 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}
{{"\n"}}{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```

注意

在有些 CSR 被批准前，一些 Operator 可能无法使用。

4. 现在，您的客户端请求已被批准，您必须查看添加到集群中的每台机器的服务器请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE     REQUESTOR                                                   CONDITION
csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal       Pending
csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal       Pending
...
```

5. 如果剩余的 CSR 没有被批准，且处于 Pending 状态，请批准集群机器的 CSR：

- 要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 <csr_name> 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

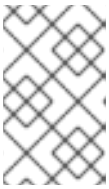
```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}' | xargs oc adm certificate approve
```

6. 批准所有客户端和服务器 CSR 后，机器将处于 Ready 状态。运行以下命令验证：

```
$ oc get nodes
```

输出示例

```
NAME       STATUS   ROLES    AGE  VERSION
master-0   Ready    master   73m  v1.29.4
master-1   Ready    master   73m  v1.29.4
master-2   Ready    master   74m  v1.29.4
worker-0   Ready    worker   11m  v1.29.4
worker-1   Ready    worker   11m  v1.29.4
```



注意

批准服务器 CSR 后可能需要几分钟时间让机器过渡到 Ready 状态。

其他信息

- 如需有关 CSR 的更多信息，请参阅 证书签名请求。

19.3.15. 初始 Operator 配置

在 control plane 初始化后，您必须立即配置一些 Operator，以便它们都可用。

先决条件

- 您的 control plane 已初始化。

流程

1. 观察集群组件上线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION   AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.16.0    True        False         False      19m
baremetal                                  4.16.0    True        False         False      37m
cloud-credential                           4.16.0    True        False         False      40m
cluster-autoscaler                         4.16.0    True        False         False      37m
config-operator                            4.16.0    True        False         False      38m
console                                    4.16.0    True        False         False      26m
csi-snapshot-controller                    4.16.0    True        False         False      37m
dns                                        4.16.0    True        False         False      37m
etcd                                       4.16.0    True        False         False      36m
image-registry                             4.16.0    True        False         False      31m
ingress                                    4.16.0    True        False         False      30m
```

```
insights                                   4.16.0    True        False         False      31m
kube-apiserver                             4.16.0    True        False         False      26m
kube-controller-manager                    4.16.0    True        False         False      36m
kube-scheduler                             4.16.0    True        False         False      36m
kube-storage-version-migrator              4.16.0    True        False         False      37m
machine-api                                4.16.0    True        False         False      29m
machine-approver                           4.16.0    True        False         False      37m
machine-config                             4.16.0    True        False         False      36m
marketplace                                4.16.0    True        False         False      37m
monitoring                                 4.16.0    True        False         False      29m
network                                    4.16.0    True        False         False      38m
node-tuning                                4.16.0    True        False         False      37m
openshift-apiserver                        4.16.0    True        False         False      32m
openshift-controller-manager               4.16.0    True        False         False      30m
openshift-samples                          4.16.0    True        False         False      32m
operator-lifecycle-manager                 4.16.0    True        False         False      37m
operator-lifecycle-manager-catalog         4.16.0    True        False         False      37m
operator-lifecycle-manager-packageserver   4.16.0    True        False         False      32m
service-ca                                 4.16.0    True        False         False      38m
storage                                    4.16.0    True        False         False      37m
```

2. 配置不可用的 Operator。

19.3.15.1. 禁用默认的 OperatorHub 目录源

在 OpenShift Container Platform 安装过程中，默认为 OperatorHub 配置由红帽和社区项目提供的源内容的 operator 目录。在受限网络环境中，必须以集群管理员身份禁用默认目录。

流程

- 通过在 OperatorHub 对象中添加 disableAllDefaultSources: true 来 禁用默认目录的源：

```
$ oc patch OperatorHub cluster --type json \
    -p '[{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]'
```

提示

或者，您可以使用 Web 控制台管理目录源。在 Administration → Cluster Settings → Configuration → OperatorHub 页面中，点 Sources 选项卡，您可以在其中创建、更新、删除、禁用和启用单独的源。

19.3.15.2. 镜像 registry 存储配置

对于不提供默认存储的平台，Image Registry Operator 最初不可用。安装后，您必须将 registry 配置为使用存储，以便 Registry Operator 可用。

显示配置生产集群所需的持久性卷的说明。如果适用，显示有关将空目录配置为存储位置的说明，这仅适用于非生产集群。

提供了在升级过程中使用 Recreate rollout 策略来允许镜像 registry 使用块存储类型的说明。

19.3.15.2.1. 更改镜像 registry 的管理状态

要启动镜像 registry，您必须将 Image Registry Operator 配置的 managementState 从 Removed 改为 Managed。

流程

- 将 managementState Image Registry Operator 配置从 Removed 改为 Managed。例如：

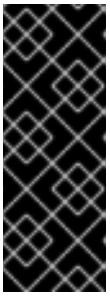
```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch
'{"spec":{"managementState":"Managed"}}'
```

19.3.15.2.2. 为 IBM Power 配置 registry 存储

作为集群管理员，在安装后需要配置 registry 来使用存储。

先决条件

- 您可以使用具有 cluster-admin 角色的用户访问集群。
- 在 IBM Power® 上有一个集群。
- 您已为集群置备持久性存储，如 Red Hat OpenShift Data Foundation。



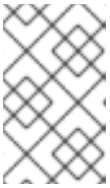
重要

当您只有一个副本时，OpenShift Container Platform 支持对镜像 registry 存储的 ReadWriteOnce 访问。ReadWriteOnce 访问还要求 registry 使用 Recreate rollout 策略。要部署支持高可用性的镜像 registry，需要两个或多个副本，ReadWriteMany 访问。

- 必须具有 100Gi 容量。

流程

1. 要将 registry 配置为使用存储，修改 configs.imageregistry/cluster 资源中的 spec.storage.pvc。



注意

使用共享存储时，请查看您的安全设置以防止外部访问。

2. 验证您没有 registry pod:

```
$ oc get pod -n openshift-image-registry -l docker-registry=default
```

输出示例

```
No resources found in openshift-image-registry namespace
```

注意

如果您的输出中有一个 registry pod，则不需要继续这个过程。

3. 检查 registry 配置：

```
$ oc edit configs.imageregistry.operator.openshift.io
```

输出示例

```
storage:
  pvc:
    claim:
```

将 claim 字段留空以允许自动创建 image-registry-storage PVC。

4. 检查 clusteroperator 状态：

```
$ oc get clusteroperator image-registry
```

输出示例

```
NAME           VERSION   AVAILABLE   PROGRESSING   DEGRADED   SINCE   MESSAGE
image-registry   4.16      True        False         False      6h50m
```

5. 确保 registry 设置为 managed，以启用镜像的构建和推送。

- 运行：

```
$ oc edit configs.imageregistry/cluster
```

然后，更改行

```
managementState: Removed
```

至

```
managementState: Managed
```

19.3.15.2.3. 在非生产集群中为镜像 registry 配置存储

您必须为 Image Registry Operator 配置存储。对于非生产集群，您可以将镜像 registry 设置为空目录。如果您这样做，重启 registry 时会丢失所有镜像。

流程

- 将镜像 registry 存储设置为空目录：

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch
'{"spec":{"storage":{"emptyDir":{}}}}'
```

警告

仅为非生产集群配置这个选项。

如果在 Image Registry Operator 初始化其组件前运行这个命令，oc patch 命令会失败并显示以下错误：

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster"
not found
```

等待几分钟，然后再次运行 命令。

19.3.16. 在用户置备的基础架构上完成安装

完成 Operator 配置后，可以在您提供的基础架构上完成集群安装。

先决条件

- 您的 control plane 已初始化。
- 已完成初始 Operator 配置。

流程

1. 使用以下命令确认所有集群组件都在线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION   AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.16.0    True        False         False      19m
baremetal                                  4.16.0    True        False         False      37m
cloud-credential                           4.16.0    True        False         False      40m
cluster-autoscaler                         4.16.0    True        False         False      37m
config-operator                            4.16.0    True        False         False      38m
console                                    4.16.0    True        False         False      26m
csi-snapshot-controller                    4.16.0    True        False         False      37m
dns                                        4.16.0    True        False         False      37m
etcd                                       4.16.0    True        False         False      36m
image-registry                             4.16.0    True        False         False      31m
ingress                                    4.16.0    True        False         False      30m
insights                                   4.16.0    True        False         False      31m
kube-apiserver                             4.16.0    True        False         False      26m
kube-controller-manager                    4.16.0    True        False         False      36m
kube-scheduler                             4.16.0    True        False         False      36m
kube-storage-version-migrator              4.16.0    True        False         False      37m
machine-api                                4.16.0    True        False         False      29m
machine-approver                           4.16.0    True        False         False      37m
machine-config                             4.16.0    True        False         False      36m
marketplace                                4.16.0    True        False         False      37m
monitoring                                 4.16.0    True        False         False      29m
network                                    4.16.0    True        False         False      38m
node-tuning                                4.16.0    True        False         False      37m
openshift-apiserver                        4.16.0    True        False         False      32m
openshift-controller-manager               4.16.0    True        False         False      30m
openshift-samples                          4.16.0    True        False         False      32m
operator-lifecycle-manager                 4.16.0    True        False         False      37m
operator-lifecycle-manager-catalog         4.16.0    True        False         False      37m
operator-lifecycle-manager-packageserver   4.16.0    True        False         False      32m
service-ca                                 4.16.0    True        False         False      38m
storage                                    4.16.0    True        False         False      37m
```

另外，当所有集群都可用时，以下命令会通知您。它还检索并显示凭证：

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

1

对于 <installation_directory>，请指定安装文件保存到的目录的路径。

输出示例

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

Cluster Version Operator 完成从 Kubernetes API 服务器部署 OpenShift Container Platform 集群时，该命令会成功。

重要

- 安装程序生成的 Ignition 配置文件包含 24 小时后过期的证书，然后在该时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 node-bootstrapper 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。

- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

2. 确认 Kubernetes API 服务器正在与 pod 通信。

   a. 要查看所有 pod 的列表，请使用以下命令：

```
$ oc get pods --all-namespaces
```

输出示例

```
NAMESPACE                       NAME                                        READY   STATUS
RESTARTS   AGE
openshift-apiserver-operator    openshift-apiserver-operator-85cb746d55-zqhs8
1/1     Running     1          9m
openshift-apiserver             apiserver-67b9g                             1/1     Running     0
3m
openshift-apiserver             apiserver-ljcmx                             1/1     Running     0
1m
openshift-apiserver             apiserver-z25h4                             1/1     Running     0
2m
openshift-authentication-operator authentication-operator-69d5d8bf84-vh2n8
1/1     Running     0          5m
...
```

b. 使用以下命令，查看上一命令的输出中所列 pod 的日志：

```
$ oc logs <pod_name> -n <namespace>
```

1

指定 pod 名称和命名空间，如上一命令的输出中所示。

如果 pod 日志显示，Kubernetes API 服务器可以与集群机器通信。

3. 启用多路径需要额外的步骤。不要在安装过程中启用多路径。

如需更多信息，请参阅 *安装后机器配置任务* 文档中的"使用 RHCOS 上使用内核参数启用多路径"。

4. 在 Cluster registration 页面注册 您的集群。

19.3.17. 后续步骤

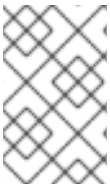
- 在 RHCOS 上启用带有内核参数的多路径。
- 自定义集群。
- 如果您用来安装集群的镜像 registry 具有可信任的 CA，请通过 配置额外的信任存储将其添加到集群中。
- 如果需要，您可以选择 不使用远程健康报告。
- 如果需要，请参阅 注册断开连接的集群

19.4. IBM POWER 的安装配置参数

在部署 OpenShift Container Platform 集群前，您可以提供一个自定义 的 install-config.yaml 安装配置文件，该文件描述了您的环境的详情。

19.4.1. IBM Power 可用的安装配置参数

下表指定您可以在安装过程中设置所需的、可选和 IBM Power 的安装配置参数。



注意

安装后，您无法在 install-config.yaml 文件中修改这些参数。

19.4.1.1. 所需的配置参数

下表描述了所需的安装配置参数：

表 19.39. 所需的参数

参数	描述	值

参数	描述	值
apiVersion:	**install-config.yaml** 内容的 API 版本。当前版本为 **v1**。安装程序可能还支持旧的 API 版本。	字符串
baseDomain:	云供应商的基域。基域用于创建到 OpenShift Container Platform 集群组件的路由。集群的完整 DNS 名称是 **baseDomain** 和 **metadata.name** 参数值的组合，其格式为 **<metadata.name>.<baseDomain>**。	完全限定域名或子域名，如 **example.com**。
metadata:	Kubernetes 资源 **ObjectMeta**，其中只消耗 **name** 参数。	对象
metadata: name:	集群的名称。集群的 DNS 记录是 **{{.metadata.name}}.{{.baseDomain}}** 的子域。	小写字母、连字符(**-**)和句点(**.**)字符串，如 **dev**。
platform:	对于特定平台的配置取决于执行安装的环境： **aws**, **baremetal**, **azure**, **gcp**, **ibmcloud**, **nutanix**, **openstack**, **powervs**, **vsphere**, 或 **{}**.有关 **platform.<platform>** 参数的更多信息，请参考下表中您的特定平台。	对象

pullSecret:	从 Red Hat OpenShift Cluster Manager 获取 pull secret，验证从 Quay.io 等服务中下载 OpenShift Container Platform 组件的容器镜像。	<pre>{
		  "auths":{
		  "cloud.openshift.com":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  },
		  "quay.io":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  }
		  }
		}</pre>

19.4.1.2. 网络配置参数

您可以根据现有网络基础架构的要求自定义安装配置。例如，您可以扩展集群网络的 IP 地址块，或者提供不同于默认值的不同 IP 地址块。

- 如果使用 Red Hat OpenShift Networking OVN-Kubernetes 网络插件，则支持 IPv4 和 IPv6 地址系列。

如果将集群配置为使用两个 IP 地址系列，请查看以下要求：

- 两个 IP 系列都必须将相同的网络接口用于默认网关。
- 两个 IP 系列都必须具有默认网关。
- 您必须为所有网络配置参数指定 IPv4 和 IPv6 地址。例如，以下配置 IPv4 地址列在 IPv6 地址的前面。

```
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  - cidr: fd00:10:128::/56
    hostPrefix: 64
  serviceNetwork:
  - 172.30.0.0/16
  - fd00:172:16::/112
```

注意

Red Hat OpenShift Data Foundation 灾难恢复解决方案不支持 Globalnet。对于区域灾难恢复场景，请确保为每个集群中的集群和服务网络使用非重叠的专用 IP 地址。

表 19.40. 网络参数

参数	描述	值

参数	描述	值
networking:	集群网络的配置。	对象  **注意** 您无法在安装后修改 **网络** 对象指定的参数。
networking: networkType:	要安装的 Red Hat OpenShift Networking 网络插件。	**OVNKubernetes**。**OVNKubernetes** 是 Linux 网络和包含 Linux 和 Windows 服务器的混合网络的 CNI 插件。默认值为 **OVNKubernetes**。
networking: clusterNetwork:	pod 的 IP 地址块。 默认值为 **10.128.0.0/14**，主机前缀为 **/23**。 如果您指定了多个 IP 地址块，块不得重叠。	对象数组。例如： networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23
networking: clusterNetwork: cidr:	使用 **networking.clusterNetwork** 时需要此项。IP 地址块。 IPv4 网络。	无类别域间路由(CIDR)表示法中的 IP 地址块。IPv4 块的前缀长度介于 **0 到 32** 之间。
networking: clusterNetwork: hostPrefix:	分配给每个节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从 given **cidr** 中分配 a **/23** 子网。**hostPrefix** 值 **23** 提供 510（2^(32 - 23)- 2）pod IP 地址。	子网前缀。 默认值为 **23**。
networking: serviceNetwork:	服务的 IP 地址块。默认值为 **172.30.0.0/16**。 OVN-Kubernetes 网络插件只支持服务网络的一个 IP 地址块。	CIDR 格式具有 IP 地址块的数组。例如： networking: serviceNetwork: - 172.30.0.0/16

参数	描述	值
networking: machineNetwork:	机器的 IP 地址块。 如果您指定了多个 IP 地址块，块不得重叠。 如果您指定了多个 IP 内核参数，**machineNetwork.cidr** 值必须是主网络的 CIDR。	对象数组。例如： networking: machineNetwork: - cidr: 10.0.0.0/16
networking: machineNetwork: cidr:	使用 **networking.machineNetwork** 时需要此项。IP 地址块。对于 libvirt 和 IBM Power® Virtual Server 以外的所有平台，默认值为 **10.0.0.0/16**。对于 libvirt，默认值 为 **192.168.126.0/24**。对于 IBM Power® Virtual Server，默认值为 **192.168.0.0/24**。	CIDR 表示法中的 IP 网络块。 例如： **10.0.0.0/16**。 **注意** 将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

19.4.1.3. 可选的配置参数

下表描述了可选的安装配置参数：

表 19.41. 可选参数

参数	描述	值
additionalTrustBundle:	添加到节点可信证书存储中的 PEM 编码 X.509 证书捆绑包。配置了代理时，也可以使用此信任捆绑包。	字符串
capabilities:	控制可选核心组件的安装。您可以通过禁用可选组件来减少 OpenShift Container Platform 集群的空间。如需更多信息，请参阅 *安装*中的"集群功能"页面。	字符串数组
capabilities: baselineCapabilitySet:	选择要启用的一组初始可选功能。有效值为 **None**、**v4.11**、**v4.12** 和 **vCurrent**。默认值为 **vCurrent**。	字符串

参数	描述	值
capabilities: additionalEnabledCapabilities:	将可选功能集合扩展到您在 **baselineCapabilitySet** 中指定的范围。您可以在此参数中指定多个功能。	字符串数组
cpuPartitioningMode:	启用工作负载分区，它会隔离 OpenShift Container Platform 服务、集群管理工作负载和基础架构 pod，以便在保留的一组 CPU 上运行。工作负载分区只能在安装过程中启用，且在安装后无法禁用。虽然此字段启用工作负载分区，但它不会将工作负载配置为使用特定的 CPU。如需更多信息，请参阅 *Scalability and Performance* 部分中的 *Workload partitioning* 页面。	**None** 或 **AllNodes**。**None** 是默认值。
compute:	组成计算节点的机器的配置。	**MachinePool** 对象的数组。
compute: architecture:	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **ppc64le** （默认值）。	字符串
compute: hyperthreading:	是否在计算机器上启用或禁用并发多 **线程或超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。	**enabled** 或 **Disabled**
compute: name:	使用 **compute** 时需要此项。机器池的名称。	**worker**
compute: platform:	使用 **compute** 时需要此项。使用此参数指定托管 worker 机器的云供应商。此参数值必须与 **controlPlane.platform** 参数值匹配。	**aws**, **azure**, **gcp**, **ibmcloud**, **nutanix**, **openstack**, **powervs**, **vsphere**, 或 **{}**

参数	描述	值
compute: replicas:	要置备的计算机器数量，也称为 worker 机器。	大于或等于 **2** 的正整数。默认值为 **3**。
featureSet:	为功能集启用集群。功能集是 OpenShift Container Platform 功能的集合，默认情况下不启用。有关在安装过程中启用功能集的更多信息，请参阅"使用功能门启用功能"。	字符串.要启用的功能集的名称，如 **TechPreviewNoUpgrade**。
controlPlane:	组成 control plane 的机器的配置。	**MachinePool** 对象的数组。
controlPlane: architecture:	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **ppc64le** （默认值）。	字符串
controlPlane: hyperthreading:	是否在 control plane 机器上启用或禁用并发多 **线程或超线程**。默认情况下，启用并发多线程以提高机器内核的性能。 **重要** 如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。	**enabled** 或 **Disabled**
controlPlane: name:	使用 **controlPlane** 时需要此项。机器池的名称。	**master**
controlPlane: platform:	使用 **controlPlane** 时需要此项。使用此参数指定托管 control plane 机器的云供应商。此参数值必须与 **compute.platform** 参数值匹配。	**aws**, **azure**, **gcp**, **ibmcloud**, **nutanix**, **openstack**, **powervs**, **vsphere**, 或 **{}**
controlPlane: replicas:	要置备的 control plane 机器数量。	部署单节点 OpenShift 时支持的值为 **3** 或 **1**。

参数	描述	值
credentialsMode:	Cloud Credential Operator(CCO)模式。如果没有指定模式，CCO 会动态尝试决定提供的凭证的功能，在支持多个模式的平台上首选 mint 模式。	**Mint**、**Passthrough**、**Manual** 或空字符串(**""**)。[1]

参数	描述	值
fips:	启用或禁用 FIPS 模式。默认值为 **false**（禁用）。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。  **重要** 要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 Red Hat Enterprise Linux (RHEL) 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅在 FIPS 模式中安装该系统。 当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS) 时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。  **注意** 如果使用 Azure File 存储，则无法启用 FIPS 模式。	**false** 或 **true**

参数	描述	值
imageContentSources:	release-image 内容的源和存储库。	对象数组。包括一个 **source** 以及可选的 **mirrors**，如本表的以下行所述。
imageContentSources: source:	使用 **imageContentSources** 时需要此项。指定用户在镜像拉取规格中引用的存储库。	字符串
imageContentSources: mirrors:	指定可能还包含同一镜像的一个或多个仓库。	字符串数组
publish:	如何发布或公开集群的面向用户的端点，如 Kubernetes API、OpenShift 路由。	**内部或外部 .**默认值为 **External**。 在非云平台上不支持将此字段设置为 **Internal**。 **重要** 如果将字段的值设为 **Internal**，集群将无法运行。如需更多信息，请参阅 BZ#1953035。
sshKey:	用于验证对集群机器的访问的 SSH 密钥。 **注意** 对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。	例如，**sshKey: ssh-ed25519 AAAA..**。

1. **不是所有 CCO 模式都支持所有云供应商。有关 CCO 模式的更多信息，请参阅*身份验证和授权*内容中的"管理云供应商凭证"条目。**

第 20 章 在 IBM POWER VIRTUAL SERVER 上安装

20.1. 准备在 IBM POWER VIRTUAL SERVER 上安装

本节中介绍的安装工作流用于 IBM Power® Virtual Server 基础架构环境。

20.1.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读选择集群安装方法并为用户准备它的文档。

20.1.2. 在 IBM Power Virtual Server 上安装 OpenShift Container Platform 的要求

在 IBM Power® Virtual Server 上安装 OpenShift Container Platform 前，您必须创建一个服务帐户并配置 IBM Cloud® 帐户。有关创建帐户、配置 DNS 和支持的 IBM Power® Virtual Server 区域的详情，请参阅配置 IBM Cloud® 帐户。

在将集群安装到 IBM Power® Virtual Server 时，您必须手动管理云凭证。在安装集群前，请为手动模式配置 Cloud Credential Operator (CCO)

20.1.3. 选择在 IBM Power Virtual Server 上安装 OpenShift Container Platform 的方法

您可以使用安装程序置备的基础架构在 IBM Power® Virtual Server 上安装 OpenShift Container Platform。此过程涉及使用安装程序为集群置备底层基础架构。目前，不支持使用用户置备的基础架构在 IBM Power® Virtual Server 上安装 OpenShift Container Platform。

有关安装程序置备安装过程的更多信息，请参阅安装过程。

20.1.3.1. 在安装程序置备的基础架构上安装集群

您可以使用以下方法之一在 OpenShift Container Platform 安装程序置备的 IBM Power® Virtual Server 基础架构上安装集群：

- 在 IBM Power® Virtual Server 上安装自定义集群：您可以在安装程序置备的 IBM Power® Virtual Server 基础架构上安装自定义集群。安装程序允许在安装阶段应用一些自定义。其它自

定义选项可在安装后使用。

- 在 IBM Power® Virtual Server 上将集群安装到现有的 VPC 中：您可以在 IBM Power® Virtual Server 上安装 OpenShift Container Platform 到现有的 Virtual Private Cloud (VPC) 中。如果您按照公司的说明设置了限制，可以使用这个安装方法，例如在创建新帐户或基础架构时的限制。

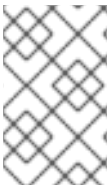
- 在 IBM Power® Virtual Server 上安装私有集群 ：您可以在 IBM Power® Virtual Server 上安装私有集群。您可以使用此方法在互联网不可见的内部网络中部署 OpenShift Container Platform。

- 在受限网络中的 IBM Power® Virtual Server 上安装集群 ：您可以使用安装发行内容的内部镜像在 IBM Power® Virtual Server 上安装 OpenShift Container Platform。您可以使用此方法安装不需要活跃互联网连接的集群来获取软件组件。

20.1.4. 配置 Cloud Credential Operator 工具

Cloud Credential Operator(CCO)将云供应商凭证作为 Kubernetes 自定义资源定义(CRD)进行管理。要在 IBM Power® Virtual Server 上安装集群，您必须将 CCO 设置为手动模式，作为安装过程的一部分。

当 Cloud Credential Operator(CCO)以手动模式运行时，要从集群外部创建和管理云凭证，提取并准备 CCO 实用程序(ccoctl)二进制文件。



注意

ccoctl 工具是在 Linux 环境中运行的 Linux 二进制文件。

先决条件

- 您可以访问具有集群管理员权限的 OpenShift Container Platform 帐户。

- 已安装 OpenShift CLI(oc)。

流程

1. 运行以下命令，为 OpenShift Container Platform 发行镜像设置变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

2. 运行以下命令，从 OpenShift Container Platform 发行镜像获取 CCO 容器镜像：

```
$ CCO_IMAGE=$(oc adm release info --image-for='cloud-credential-operator'
$RELEASE_IMAGE -a ~/.pull-secret)
```

注意

确保 $RELEASE_IMAGE 的架构与将使用 ccoctl 工具的环境架构相匹配。

3. 运行以下命令，将 CCO 容器镜像中的 ccoctl 二进制文件提取到 OpenShift Container Platform 发行镜像中：

```
$ oc image extract $CCO_IMAGE \
  --file="/usr/bin/ccoctl.<rhel_version>" \ 1
  -a ~/.pull-secret
```

1 对于 <rhel_version>，请指定与主机使用的 Red Hat Enterprise Linux (RHEL) 版本对应的值。如果没有指定值，则默认使用 ccoctl.rhel8。以下值有效：

- rhel8: 为使用 RHEL 8 的主机指定这个值。
- rhel9 ：为使用 RHEL 9 的主机指定这个值。

4. 运行以下命令更改权限以使 ccoctl 可执行：

```
$ chmod 775 ccoctl.<rhel_version>
```

验证

- 要验证 ccoctl 是否可使用，请运行以下命令来显示帮助文件：

```
$ ccoctl --help
```

ccoctl --help 的输出

```
OpenShift credentials provisioning tool

Usage:
  ccoctl [command]

Available Commands:
  aws         Manage credentials objects for AWS cloud
  azure       Manage credentials objects for Azure
  gcp         Manage credentials objects for Google cloud
  help        Help about any command
  ibmcloud    Manage credentials objects for IBM Cloud
  nutanix     Manage credentials objects for Nutanix

Flags:
  -h, --help   help for ccoctl

Use "ccoctl [command] --help" for more information about a command.
```

其他资源

- 轮转 API 密钥

20.1.5. 后续步骤

- 配置 IBM Cloud® 帐户

20.2. 配置 IBM CLOUD 帐户

在安装 OpenShift Container Platform 之前，您必须配置 IBM Cloud® 帐户。

20.2.1. 先决条件

- 您有一个带有订阅的 IBM Cloud® 帐户。您不能在免费或试用 IBM Cloud® 帐户上安装 OpenShift Container Platform。

20.2.2. IBM Power Virtual Server 的配额和限制

OpenShift Container Platform 集群使用多个 IBM Cloud® 和 IBM Power® Virtual Server 组件，默认配额和限值会影响您安装 OpenShift Container Platform 集群的能力。如果您使用特定的集群配置，在某些区域部署集群，或者从您的帐户运行多个集群，您可能需要为 IBM Cloud® 帐户请求其他资源。

有关默认 IBM Cloud® 配额和服务限制的完整列表，请参阅 IBM Cloud® 的配额和服务限制 文档。

虚拟私有云

每个 OpenShift Container Platform 集群创建自己的虚拟私有云(VPC)。每个区域的 VPC 的默认配额为 10。如果您创建了 10 个 VPC，则需要在尝试安装前提高配额。

应用程序负载均衡器

默认情况下，每个集群创建两个应用程序负载均衡器 (ALBs)：

- control plane API 服务器的内部负载均衡器
- control plane API 服务器的外部负载均衡器

您可以创建额外的 LoadBalancer 服务对象创建额外的 ALBs。VPC 的默认配额是每个区域 50 个。要获得超过 50 个 ALB，您必须提高此配额。

支持 VPC ALB。IBM Power® Virtual Server 不支持经典的 ALB。

传输网关

每个 OpenShift Container Platform 集群都会创建自己的 Transit 网关，以启用与 VPC 的通信。每个帐户传输网关的默认配额是 10。如果您创建了 10 个传输网关，则需要在尝试安装前提高配额。

动态主机配置协议服务

每个 IBM Power® Virtual Server 实例有一个动态主机配置协议(DHCP)服务的限制。

网络

由于网络限制，每个帐户通过 IPI 安装了一个 OpenShift 集群的限制。这不是可配置的。

虚拟服务器实例

默认情况下，集群使用以下资源创建服务器实例：

- 0.5 CPU
- 32 GB RAM
- 系统要求：s922
- 处理器类型：uncapped, shared
- 存储级：Tier-3

创建以下节点：

- 一台 bootstrap 机器，它会在安装完成后删除
- 三个 control plane 节点
- 三个计算节点

如需更多信息，请参阅 IBM Cloud® 文档中的创建电源系统虚拟服务器。

20.2.3. 配置 DNS 解析

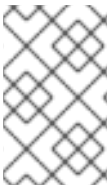
如何配置 DNS 解析取决于您安装的 OpenShift Container Platform 集群的类型：

- 如果要安装公共集群，请使用 IBM Cloud® Internet Services (CIS)。

- 如果要安装私有集群，请使用 IBM Cloud® DNS Services (DNS Services)。

20.2.4. 使用 IBM Cloud Internet 服务进行 DNS 解析

安装程序使用 IBM Cloud® Internet Services (CIS) 来配置集群 DNS 解析，并为公共集群提供名称查找。



注意

此产品不支持 IPv6，因此无法实现双堆栈或 IPv6 环境。

您必须在与集群相同的帐户的 CIS 中创建域区。您还必须确保该区域对域具有权威。您可以使用根域或子域进行此操作。

先决条件

- 已安装 IBM Cloud® CLI。
- 您有一个现有的域和注册商。如需更多信息，请参阅 IBM® 文档。

流程

1. 创建用于集群的 CIS 实例：
   a. 安装 CIS 插件：

```
$ ibmcloud plugin install cis
```

   b. 使用 CLI 登录 IBM Cloud®：

```
$ ibmcloud login
```

   c. 创建 CIS 实例：

```
$ ibmcloud cis instance-create <instance_name> standard 1
```

1

CIS 至少需要一个 标准 计划来管理集群子域及其 DNS 记录。

2. 将现有域连接到您的 CIS 实例：

a. 为 CIS 设置上下文实例：

```
$ ibmcloud cis instance-set <instance_CRN> 1
```

1

实例 CRN （云资源名称）。例如： ibmcloud cis instance-set crn:v1:bluemix:public:power-iaas:osa21:a/65b64c1f1c29460d8c2e4bbfbd893c2c:c09233ac-48a5-4ccb-a051-d1cfb3fc7eb5::

b. 为 CIS 添加域：

```
$ ibmcloud cis domain-add <domain_name> 1
```

1

完全限定域名。您可以根据计划配置，使用根域或子域值作为域名。

注意

根域使用格式 openshiftcorp.com。子域使用格式为 cluster.openshiftcorp.com。

3. 打开 CIS Web 控制台，进入 Overview 页面，并记录您的 CIS 名称服务器。这些名称服务器将在下一步中使用。

4. 在域的注册商或 DNS 供应商中为您的域或子域配置名称服务器。如需更多信息，请参阅 IBM Cloud® 文档。

20.2.5. IBM Cloud IAM 策略和 API 密钥

要将 OpenShift Container Platform 安装到 IBM Cloud® 帐户中，安装程序需要一个 IAM API 密钥，它提供访问 IBM Cloud® 服务 API 的身份验证和授权。您可以使用包含所需策略的现有 IAM API 密钥或创建新策略。

有关 IBM Cloud® IAM 概述，请参阅 IBM Cloud® 文档。

20.2.5.1. 先决条件权限

表 20.1. 先决条件权限

角色	权限
Viewer, Operator, Editor, Administrator, Reader, Writer, Manager	<resource_group> 资源组中的互联网服务
Viewer, Operator, Editor, Administrator, User API key creator, Service ID creator	IAM Identity Service 服务
Viewer, Operator, Administrator, Editor, Reader, Writer, Manager, Console Administrator	<resource_group> 资源组中的 VPC Infrastructure Services 服务
Viewer	资源组 ：查看资源组本身的访问权限。资源类型应等于 **Resource group**，值为 <your_resource_group_name>。

20.2.5.2. cluster-creation 权限

表 20.2. cluster-creation 权限

角色	权限
Viewer	<resource_group> （为您的团队创建资源组）
Viewer, Operator, Editor, Reader, Writer, Manager	Default 资源组中启用了所有 Identity 和 IAM 服务
Viewer, Reader	互联网服务

角色	权限
Viewer, Operator, Reader, Writer, Manager, Content Reader, Object Reader, Object Writer, Editor	云对象存储服务
Viewer	默认资源组： 资源类型应等于 **Resource group**，值为 **Default**。如果您的帐户管理员将帐户的默认资源组改为 Default 以外的名称，请使用该值。
Viewer, Operator, Editor, Reader, Manager	<resource_group> 资源组中的 IBM Power® Virtual Server 服务的工作区
Viewer, Operator, Editor, Reader, Writer, Manager, Administrator	<resource_group> 资源组中的 Internet Services 服务：CIS 功能范围字符串等于可靠性
Viewer, Operator, Editor	传输网关服务
Viewer, Operator, Editor, Administrator, Reader, Writer, Manager, Console Administrator	VPC Infrastructure Services 服务 <resource_group> 资源组

20.2.5.3. 访问策略分配

在 IBM Cloud® IAM 中，可以将访问策略附加到不同的主题：

- 访问组（**推荐**）
- **服务 ID**
- **用户**

建议的方法是在访问组中定义 IAM 访问策略。这有助于组织 OpenShift Container Platform 所需的所有访问权限，并可让您向这个组注册用户和服务 ID。如果需要，您还可以为 用户和服务 ID 分配访问权限。

20.2.5.4. 创建 API 密钥

您必须为 IBM Cloud® 帐户创建用户 API 密钥或服务 ID API 密钥。

先决条件

- 您已为 IBM Cloud® 帐户分配了所需的访问策略。
- 您已将 IAM 访问策略附加到访问组或其他适当的资源。

流程

- 根据您定义的 IAM 访问策略，创建一个 API 密钥。

  例如，如果您为用户分配了访问策略，您必须创建一个 用户 API 密钥。如果您将访问策略分配给服务 ID，您必须创建一个服务 ID API 密钥。如果您的访问策略分配给一个访问组，您可以使用任一 API 密钥类型。有关 IBM Cloud® API 密钥的更多信息，请参阅了解 API 密钥。

20.2.6. 支持的 IBM Power Virtual Server 区域和区域

您可以将 OpenShift Container Platform 集群部署到以下区域：

- dal (Dallas, USA)
  - dal10
  - dal12
- eu-de (Frankfurt, Germany)
  - eu-de-1
  - eu-de-2
- lon (London, UK)
  - lon04

- mad (Madrid, Spain)
  - mad02
  - mad04
- osa (Osaka, Japan)
  - osa21
- sao (Sao Paulo, Brazil)
  - sao01
  - sao04
- syd (Sydney, Australia)
  - syd04
- wdc (Washington DC, USA)
  - wdc06
  - wdc07

您可以选择指定安装程序在其中创建任何 VPC 组件的 IBM Cloud® 区域。IBM Cloud® 支持的区域有：

- us-south
- eu-de
- eu-es
- eu-gb
- jp-osa
- au-syd
- br-sao
- ca-tor
- jp-tok

20.2.7. 后续步骤

- 创建 IBM Power® Virtual Server 工作区

20.3. 创建 IBM POWER VIRTUAL SERVER 工作区

20.3.1. 创建 IBM Power Virtual Server 工作区

使用以下步骤创建 IBM Power® Virtual Server 工作区。

流程

1. 要创建 IBM Power® Virtual Server 工作区，请完成步骤 1 到 创建 IBM Power® Virtual Server 的 IBM Cloud® 文档中的第 5 步。

2. 置备完成后，输入以下命令检索新工作区的 32 个字符字母数字字符(GUID)：

```
$ ibmcloud resource service-instance <workspace name>
```

20.3.2. 后续步骤

- 使用自定义在 IBM Power® Virtual Server 上安装集群

20.4. 使用自定义在 IBM POWER VIRTUAL SERVER 上安装集群

在 OpenShift Container Platform 版本 4.16 中，您可以在安装程序在 IBM Power Virtual Server 上置备的基础架构上安装自定义的集群。要自定义安装，请在安装集群前修改 install-config.yaml 文件中的参数。

20.4.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读有关 选择集群安装方法的文档，并为用户准备它。
- 已将 IBM Cloud® 帐户配置为托管集群。
- 如果使用防火墙，则会 将其配置为允许集群需要访问的站点。
- 在安装集群前已经配置了 ccoctl 工具。如需更多信息，请参阅配置 Cloud Credential Operator 工具程序。

20.4.2. OpenShift Container Platform 互联网访问

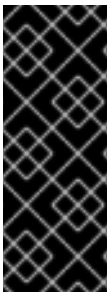
在 OpenShift Container Platform 4.16 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。

- 访问 Quay.io，以获取安装集群所需的软件包。

- 获取执行集群更新所需的软件包。



重要

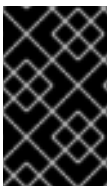
如果您的集群无法直接访问互联网，则可以在置备的某些类型的基础架构上执行受限网络安装。在此过程中，您可以下载所需的内容，并使用它为镜像 registry 填充安装软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群前，您要更新镜像 registry 的内容。

20.4.3. 为集群节点 SSH 访问生成密钥对

在 OpenShift Container Platform 安装过程中，您可以为安装程序提供 SSH 公钥。密钥通过它们的 Ignition 配置文件传递给 Red Hat Enterprise Linux CoreOS(RHCOS)节点，用于验证对节点的 SSH 访问。密钥添加到每个节点上 core 用户的 ~/.ssh/authorized_keys 列表中，这将启用免密码身份验证。

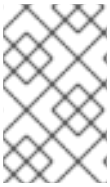
将密钥传递给节点后，您可以使用密钥对作为用户 核心 通过 SSH 连接到 RHCOS 节点。若要通过 SSH 访问节点，必须由 SSH 为您的本地用户管理私钥身份。

如果要通过 SSH 连接到集群节点来执行安装调试或灾难恢复，则必须在安装过程中提供 SSH 公钥。./openshift-install gather 命令还需要在集群节点上设置 SSH 公钥。



重要

不要在生产环境中跳过这个过程，在生产环境中需要灾难恢复和调试。



注意

您必须使用本地密钥，而不是使用特定平台方法配置 的密钥，如 AWS 密钥对。

流程

1. 如果您在本地计算机上没有可用于在集群节点上进行身份验证的现有 SSH 密钥对，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 ~/.ssh/id_ed25519。如果您已有密钥对，请确保您的公钥位于 ~/.ssh 目录中。

2. 查看公共 SSH 密钥：

```
$ cat <path>/<file_name>.pub
```

例如，运行以下命令来查看 ~/.ssh/id_ed25519.pub 公钥：

```
$ cat ~/.ssh/id_ed25519.pub
```

3. 将 SSH 私钥身份添加到本地用户的 SSH 代理（如果尚未添加）。在集群节点上，或者要使用 ./openshift-install gather 命令，需要对该密钥进行 SSH 代理管理，才能在集群节点上进行免密码 SSH 身份验证。

注意

在某些发行版中，自动管理默认 SSH 私钥身份，如 ~/.ssh/id_rsa 和 ~/.ssh/id_dsa。

a. 如果 ssh-agent 进程尚未为您的本地用户运行，请将其作为后台任务启动：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

4. 将 SSH 私钥添加到 ssh-agent ：

```
$ ssh-add <path>/<file_name> 1
```

1 指定 SSH 私钥的路径和文件名，如 ~/.ssh/id_ed25519.pub

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

后续步骤

- 安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

20.4.4. 获取安装程序

在安装 OpenShift Container Platform 前，将安装文件下载到您用于安装的主机上。

先决条件

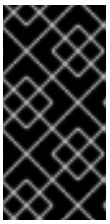
- 您有一台运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB。

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐户，请使用您的凭证登录。如果没有，请创建一个帐户。

2. 选择您的基础架构供应商。

3. 进入到安装类型的页面，下载与您的主机操作系统和架构对应的安装程序，并将该文件放在您要存储安装配置文件的目录中。



重要

安装程序会在用来安装集群的计算机上创建几个文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。

重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，请为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret。此 pull secret 允许您与所含授权机构提供的服务进行身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

20.4.5. 导出 API 密钥

您必须将您创建的 API 密钥设置为全局变量；安装程序会在启动期间设置 API 密钥。

先决条件

- 您已为 IBM Cloud® 帐户创建了用户 API 密钥或服务 ID API 密钥。

流程

- 将帐户的 API 密钥导出为全局变量：

```
$ export IBMCLOUD_API_KEY=<api_key>
```

重要

您必须按照指定方式设置变量名称，安装程序需要在启动期间存在变量名称。

20.4.6. 创建安装配置文件

您可以自定义 OpenShift Container Platform 集群。

先决条件

- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。

流程

1. 创建 install-config.yaml 文件。

   a. 进入包含安装程序的目录并运行以下命令：

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

   1 对于 <installation_directory>，请指定要存储安装程序创建的文件的目录名称。

   在指定目录时：

   - 验证该目录是否具有执行权限。在安装目录中运行 Terraform 二进制文件需要这个权限。
   - 使用空目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

注意

始终删除 ~/.powervs 目录，以避免重复使用过时的配置。运行以下命令：

```
$ rm -rf ~/.powervs
```

b. 在提示符处，提供云的配置详情：

i. 可选：选择用于访问集群机器的 SSH 密钥。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 ssh-agent 进程使用的 SSH 密钥。

ii. 选择 powervs 作为目标平台。

iii. 选择要将集群部署到的区域。

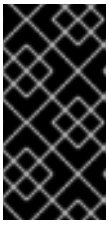
iv. 选择要将集群部署到的区域。

v. 选择集群要部署到的基域。基域与您为集群创建的公共 DNS 区对应。

vi. 为集群输入描述性名称。

2. 修改 install-config.yaml 文件。您可以在"安装配置参数"部分找到有关可用参数的更多信息。

3. 备份 install-config.yaml 文件，以便您可以使用它安装多个集群。



重要

install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用此文件，必须现在备份。

其他资源

- IBM Power® Virtual Server 的安装配置参数

20.4.6.1. IBM Power Virtual Server 的自定义 install-config.yaml 文件示例

您可以自定义 install-config.yaml 文件，以指定有关 OpenShift Container Platform 集群平台的更多详情，或修改所需参数的值。

重要

此示例 YAML 文件仅供参考。您必须使用安装程序来获取 install-config.yaml 文件，并进行修改。

```
apiVersion: v1
baseDomain: example.com
compute: 1 2
- architecture: ppc64le
  hyperthreading: Enabled 3
  name: worker
  platform:
    powervs:
      smtLevel: 8 4
  replicas: 3
controlPlane: 5 6
  architecture: ppc64le
  hyperthreading: Enabled 7
  name: master
  platform:
    powervs:
      smtLevel: 8 8
  replicas: 3
metadata:
  creationTimestamp: null
  name: example-cluster-name
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 192.168.0.0/24
```

```
  networkType: OVNKubernetes 9
  serviceNetwork:
  - 172.30.0.0/16
platform:
  powervs:
    userID: ibm-user-id
    region: powervs-region
    zone: powervs-zone
    powervsResourceGroup: "ibmcloud-resource-group" 10
    serviceInstanceGUID: "powervs-region-service-instance-guid"
    vpcRegion : vpc-region
publish: External
pullSecret: '{"auths": ...}' 11
sshKey: ssh-ed25519 AAAA... 12
```

1 5

如果没有提供这些参数和值，安装程序会提供默认值。

2 6

controlPlane 部分是一个单个映射，但 compute 部分是一系列映射。为满足不同数据结构的要求，compute 部分的第一行必须以连字符 - 开头，controlPlane 部分 的第一行则不以连字符开头。虽然这两个部分目前都定义了单个机器池，但 OpenShift Container Platform 可能会在安装过程中支持多个计算池。仅使用一个 control plane 池。

3 7

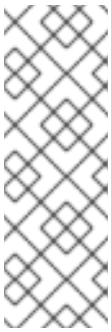
是否要启用或禁用并发多线程或 超线程。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将 参数值设置为 Disabled 来禁用它。如果在某些集群机器中禁用并发多线程，则必须在所有集群机器中禁用它。

重要

如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。

4 8

smtLevel 指定要设置为 control plane 和计算机器的 SMT 级别。支持的值有 1、2、4、8、'off' 和 'on'。默认值为 8。smtLevel 'off' 将 SMT 设置为 off，smtlevel 'on' 将 SMT 设置为集群节点上的默认值 8。



注意

如果没有启用并发多线程(SMT)或超线程，一个 vCPU 相当于一个物理内核。启用后，总 vCPU 被计算为：(Thread (s) per core * Core (s) per socket)* Socket (s))。smtLevel 控制每个内核的线程。在部署集群节点时，较低 SMT 级别可能需要额外分配的内核。您可以将 install-config.yaml 文件中的 'processors' 参数设置为适当的值，以满足成功部署 OpenShift Container Platform 的要求。

9

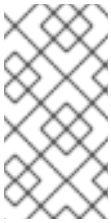
要安装的集群网络插件。默认值 OVNKubernetes 是唯一支持的值。

10

现有资源组的名称。

11

必需。安装程序会提示您输入这个值。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 ssh-agent 进程使用的 SSH 密钥。

20.4.6.2. 在安装过程中配置集群范围的代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 install-config.yaml 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 install-config.yaml 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 Proxy 对象的 spec.noProxy 字段中添加站点来绕过代理。

注意

Proxy 对象 status.noProxy 字段使用安装配置中的 networking.machineNetwork[].cidr、networking.clusterNetwork[].cidr 和 networking.serviceNetwork[] 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，Proxy 对象 status.noProxy 字段也会使用实例元数据端点填充(169.254.169.254)。

流程

1. 编辑 install-config.yaml 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 http。

2. 用于创建集群外 HTTPS 连接的代理 URL。

3. 要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 . 以仅匹配子域。例如，.y.com 匹配 x.y.com，但不匹配 y.com。使用 * 绕过所有目的地的代理。

4. 如果提供，安装程序会在 openshift-config 命名空间中生成名为 user-ca-bundle 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 trusted-ca-bundle 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， Proxy 对象的 trustedCA 字段中

也会引用此配置映射。additionalTrustBundle 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

5 可选：决定 Proxy 对象的配置以引用 trustedCA 字段中 user-ca-bundle 配置映射的策略。允许的值是 Proxyonly 和 Always。仅在配置了 http/https 代理时，使用 Proxyonly 引用 user-ca-bundle 配置映射。使用 Always 始终引用 user-ca-bundle 配置映射。默认值为 Proxyonly。

注意

安装程序不支持代理的 readinessEndpoints 字段。

注意

如果安装程序超时，重启并使用安装程序的 wait-for 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. 保存该文件并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 cluster 的集群范围代理，该代理 使用 提供的 install-config.yaml 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 cluster Proxy 对象，但它会有一个空 spec。

注意

只支持名为 cluster 的 Proxy 对象，且无法创建额外的代理。

20.4.7. 手动创建 IAM

安装集群需要 Cloud Credential Operator(CCO)以手动模式运行。虽然安装程序为手动模式配置 CCO，但您必须为云供应商指定身份和访问管理 secret。

您可以使用 Cloud Credential Operator (CCO)实用程序(ccoctl)创建所需的 IBM Cloud® 资源。

先决条件

先决条件

- 您已配置了 ccoctl 二进制文件。
- 您有一个现有的 install-config.yaml 文件。

流程

1. 编辑 install-config.yaml 配置文件，使其包含将 credentialsMode 参数设置为 Manual。

install-config.yaml 配置文件示例

```
apiVersion: v1
baseDomain: cluster1.example.com
credentialsMode: Manual 1
compute:
- architecture: ppc64le
  hyperthreading: Enabled
```

1 添加这一行将 credentialsMode 参数设置为 Manual。

2. 要生成清单，请在包含安装程序的目录中运行以下命令：

```
$ openshift-install create manifests --dir <installation_directory>
```

3. 在包含安装程序的目录中，运行以下命令，使用安装文件中的发行镜像设置 $RELEASE_IMAGE 变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

4. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 CredentialsRequest 自定义资源 (CR) 列表：

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
  --included \ 1
  --install-config=<path_to_directory_with_installation_configuration>/install-
config.yaml \ 2
  --to=<path_to_directory_for_credentials_requests> 3
```

1 --included 参数仅包含特定集群配置所需的清单。

2 指定 install-config.yaml 文件的位置。

3 指定要存储 CredentialsRequest 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

此命令为每个 CredentialsRequest 对象创建一个 YAML 文件。

CredentialsRequest 对象示例

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  labels:
    controller-tools.k8s.io: "1.0"
  name: openshift-image-registry-ibmcos
  namespace: openshift-cloud-credential-operator
spec:
  secretRef:
    name: installer-cloud-credentials
    namespace: openshift-image-registry
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: IBMCloudProviderSpec
    policies:
    - attributes:
      - name: serviceName
        value: cloud-object-storage
      roles:
      - crn:v1:bluemix:public:iam::::role:Viewer
      - crn:v1:bluemix:public:iam::::role:Operator
      - crn:v1:bluemix:public:iam::::role:Editor
```

```
    - crn:v1:bluemix:public:iam::::serviceRole:Reader
    - crn:v1:bluemix:public:iam::::serviceRole:Writer
   - attributes:
    - name: resourceType
      value: resource-group
    roles:
    - crn:v1:bluemix:public:iam::::role:Viewer
```

5. 为每个凭证请求创建服务 ID，分配定义的策略、创建 API 密钥并生成 secret：

```
$ ccoctl ibmcloud create-service-id \
  --credentials-requests-dir=<path_to_credential_requests_directory> \ 1
  --name=<cluster_name> \ 2
  --output-dir=<installation_directory> \ 3
  --resource-group-name=<resource_group_name> 4
```

1 指定包含组件 CredentialsRequest 对象文件的目录。

2 指定 OpenShift Container Platform 集群的名称。

3 可选：指定您希望 ccoctl 实用程序在其中创建对象的目录。默认情况下，实用程序在运行命令的目录中创建对象。

4 可选：指定用于限制访问策略的资源组的名称。

注意

如果您的集群使用 TechPreviewNoUpgrade 功能集启用的技术预览功能，则必须包含 --enable-tech-preview 参数。

如果提供了不正确的资源组名称，安装会在 bootstrap 阶段失败。要查找正确的资源组名称，请运行以下命令：

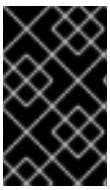
```
$ grep resourceGroup <installation_directory>/manifests/cluster-
infrastructure-02-config.yml
```

验证

- **确保在集群的 manifests 目录中生成了适当的 secret。**

20.4.8. 部署集群

您可以在兼容云平台上安装 OpenShift Container Platform。



重要

在初始安装过程中，您只能运行安装程序的 create cluster 命令一次。

先决条件

- **您已使用托管集群的云平台配置了帐户。**
- **您有 OpenShift Container Platform 安装程序和集群的 pull secret。**
- **已确认主机上的云供应商帐户具有部署集群的正确权限。权限不正确的帐户会导致安装过程失败，并显示包括缺失权限的错误消息。**

流程

- **进入包含安装程序的目录并初始化集群部署：**

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1 对于 <installation_directory>，请指定自定义 ./install-config.yaml 文件的位置。

2 要查看不同的安装详情，请指定 warn、debug 或 error，而不是 info。

验证

当集群部署成功完成时：

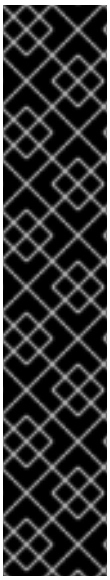
- 终端会显示用于访问集群的说明，包括指向 Web 控制台和 kubeadmin 用户的凭证的链接。
- 凭证信息还会输出到 <installation_directory>/.openshift_install.log.

重要

不要删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```



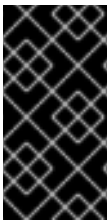
重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 node-bootstrapper 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。

- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

20.4.9. 安装 OpenShift CLI

您可以安装 OpenShift CLI(oc)来使用命令行界面与 OpenShift Container Platform 进行交互。您可以在 Linux、Windows 或 macOS 上安装 oc。



重要

如果安装了旧版本的 oc，则可能无法使用 OpenShift Container Platform 4.16 中的所有命令。下载并安装新版本的 oc。

在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI(oc)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 产品变体 下拉列表中选择架构。
3. 从 版本 下拉列表中选择适当的版本。
4. 点 OpenShift v4.16 Linux Client 条目旁的 Download Now 来保存文件。

5. 解包存档：

```
$ tar xvf <file>
```

6. 将 oc 二进制文件放到 PATH 中的目录中。

   要查看您的 PATH，请执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 oc 命令：

```
$ oc <command>
```

在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI(oc)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 版本 下拉列表中选择适当的版本。
3. 点 OpenShift v4.16 Windows Client 条目旁的 Download Now 来保存文件。
4. 使用 ZIP 程序解压存档。
5. 将 oc 二进制文件移到 PATH 中的目录中。

   要查看您的 PATH，请打开命令提示并执行以下命令：

```
C:\> path
```

验证

- 安装 OpenShift CLI 后，可以使用 oc 命令：

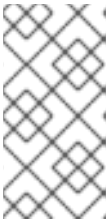
```
C:\> oc <command>
```

在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI(oc)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 版本 下拉列表中选择适当的版本。
3. 点 OpenShift v4.16 macOS Client 条目旁的 Download Now 来保存文件。



注意

对于 macOS arm64，请选择 OpenShift v4.16 macOS arm64 Client 条目。

4. 解包和解压存档。
5. 将 oc 二进制文件移到 PATH 的目录中。

要查看您的 PATH，请打开终端并执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 oc 命令：

```
$ oc <command>
```

20.4.10. 使用 CLI 登录集群

您可以通过导出集群 kubeconfig 文件，以默认系统用户身份登录集群。kubeconfig 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。
- 已安装 oc CLI。

流程

1. 导出 kubeadmin 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 <installation_directory>，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 oc 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

其他资源

- 访问Web控制台

20.4.11. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 关于远程健康监控

20.4.12. 后续步骤

- 自定义集群
- 如果需要，您可以 选择不使用远程健康报告

20.5. 在 IBM POWER VIRTUAL SERVER 上将集群安装到现有的 VPC 中

在 OpenShift Container Platform 版本 4.16 中，您可以在 IBM Cloud® 上将集群安装到现有的 Virtual Private Cloud (VPC)中。安装程序会置备所需基础架构的其余部分，您可以进一步自定义这些基础架构。要自定义安装，请在安装集群前修改 install-config.yaml 文件中的参数。

20.5.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读有关 选择集群安装方法的文档，并为用户准备它。
- 已将 IBM Cloud® 帐户配置为托管集群。

- 如果使用防火墙，则会 将其配置为允许集群需要访问的站点。

- 在安装集群前已经配置了 ccoctl 工具。如需更多信息，请参阅配置 Cloud Credential Operator 工具程序。

20.5.2. 关于使用自定义 VPC

在 OpenShift Container Platform 4.16 中，您可以使用现有的 IBM® Virtual Private Cloud (VPC) 部署集群。

因为安装程序不知道现有子网中的其他组件，所以无法选择子网 CIDR 等。您必须为安装集群的子网配置网络。

20.5.2.1. 使用 VPC 的要求

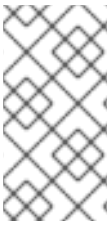
您必须在安装集群前正确配置现有的 VPC 及其子网。在这种情况下，安装程序不会创建 VPC 或 VPC 子网。

安装程序无法：

- 从属网络范围供集群使用

- 为子网设置路由表

- 设置 VPC 选项，如 DHCP



注意

安装程序要求您使用由云提供的 DNS 服务器。不支持使用自定义 DNS 服务器，并导致安装失败。

20.5.2.2. VPC 验证

VPC 和所有子网都必须位于现有资源组中。集群部署到此资源组。

作为安装的一部分，在 install-config.yaml 文件中指定以下内容：

- 资源组的名称
- VPC 的名称
- VPC 子网的名称

要确保您提供的子网适合，安装程序会确认您指定的所有子网都存在。

注意

不支持子网 ID。

20.5.2.3. 集群间隔离

如果您将 OpenShift Container Platform 部署到现有网络中，集群服务的隔离将在以下方面减少：

- 整个网络允许 ICMP 入站流量。
- 整个网络都允许 TCP 端口 22 入站流量 (SSH)。
- 整个网络都允许 control plane TCP 6443 Ingress (Kubernetes API)。
- 整个网络都允许 control plane TCP 22623 Ingress (MCS)。

20.5.3. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类型的基础架构上执行受限网络安装。在此过程中，您可以下载所需的内容，并使用它为镜像 registry 填充安装软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群前，您要更新镜像 registry 的内容。

20.5.4. 为集群节点 SSH 访问生成密钥对

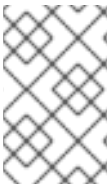
在 OpenShift Container Platform 安装过程中，您可以为安装程序提供 SSH 公钥。密钥通过它们的 Ignition 配置文件传递给 Red Hat Enterprise Linux CoreOS(RHCOS)节点，用于验证对节点的 SSH 访问。密钥添加到每个节点上 core 用户的 ~/.ssh/authorized_keys 列表中，这将启用免密码身份验证。

将密钥传递给节点后，您可以使用密钥对作为用户 核心 通过 SSH 连接到 RHCOS 节点。若要通过 SSH 访问节点，必须由 SSH 为您的本地用户管理私钥身份。

如果要通过 SSH 连接到集群节点来执行安装调试或灾难恢复，则必须在安装过程中提供 SSH 公钥。./openshift-install gather 命令还需要在集群节点上设置 SSH 公钥。

重要

不要在生产环境中跳过这个过程，在生产环境中需要灾难恢复和调试。



注意

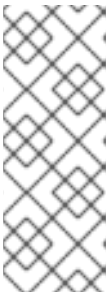
您必须使用本地密钥，而不是使用特定平台方法配置 的密钥，如 AWS 密钥对。

流程

1. 如果您在本地计算机上没有可用于在集群节点上进行身份验证的现有 SSH 密钥对，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 ~/.ssh/id_ed25519。如果您已有密钥对，请确保您的公钥位于 ~/.ssh 目录中。



注意

如果您计划在 x86_64、ppc64le 和 s390x 架构上安装使用 RHEL 加密库（这些加密库已提交给 NIST 用于 FIPS 140-2/140-3 验证）的 OpenShift Container Platform 集群，则不要创建使用 ed25519 算法的密钥。相反，创建一个使用 rsa 或 ecdsa 算法的密钥。

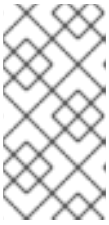
2. 查看公共 SSH 密钥：

```
$ cat <path>/<file_name>.pub
```

例如，运行以下命令来查看 ~/.ssh/id_ed25519.pub 公钥：

```
$ cat ~/.ssh/id_ed25519.pub
```

3. 将 SSH 私钥身份添加到本地用户的 SSH 代理（如果尚未添加）。在集群节点上，或者要使用 ./openshift-install gather 命令，需要对该密钥进行 SSH 代理管理，才能在集群节点上进行免密码 SSH 身份验证。



注意

在某些发行版中，自动管理默认 SSH 私钥身份，如 ~/.ssh/id_rsa 和 ~/.ssh/id_dsa。

a. 如果 ssh-agent 进程尚未为您的本地用户运行，请将其作为后台任务启动：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果集群处于 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

4. 将 SSH 私钥添加到 ssh-agent ：

```
$ ssh-add <path>/<file_name> 1
```

1 指定 SSH 私钥的路径和文件名，如 ~/.ssh/id_ed25519.pub

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

后续步骤

- 安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

20.5.5. 获取安装程序

在安装 OpenShift Container Platform 前，将安装文件下载到您用于安装的主机上。

先决条件

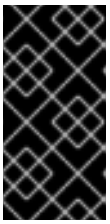
- 您有一台运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB。

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐户，请使用您的凭证登录。如果没有，请创建一个帐户。

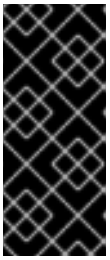
2. 选择您的基础架构供应商。

3. 进入到安装类型的页面，下载与您的主机操作系统和架构对应的安装程序，并将该文件放在您要存储安装配置文件的目录中。



重要

安装程序会在用来安装集群的计算机上创建几个文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。



重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，请为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret。此 pull secret 允许您与所含授权机构提供的服务进行身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

20.5.6. 导出 API 密钥

您必须将您创建的 API 密钥设置为全局变量；安装程序会在启动期间设置 API 密钥。

先决条件

- 您已为 IBM Cloud® 帐户创建了用户 API 密钥或服务 ID API 密钥。

流程

- 将帐户的 API 密钥导出为全局变量：

```
$ export IBMCLOUD_API_KEY=<api_key>
```

重要

您必须按照指定方式设置变量名称，安装程序需要在启动期间存在变量名称。

20.5.7. 创建安装配置文件

您可以自定义 OpenShift Container Platform 集群。

先决条件

- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。

流程

1. 创建 install-config.yaml 文件。
   a. 进入包含安装程序的目录并运行以下命令：
      -

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

1

对于 <installation_directory>，请指定要存储安装程序创建的文件的目录名称。

在指定目录时：

- 验证该目录是否具有执行权限。在安装目录中运行 Terraform 二进制文件需要这个权限。
- 使用空目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

b. 在提示符处，提供云的配置详情：

i. 可选：选择用于访问集群机器的 SSH 密钥。

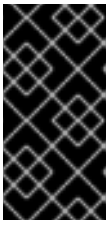
注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 ssh-agent 进程使用的 SSH 密钥。

ii. 为集群输入描述性名称。

2. 修改 install-config.yaml 文件。您可以在"安装配置参数"部分找到有关可用参数的更多信息。

3. 备份 install-config.yaml 文件，以便您可以使用它安装多个集群。



重要

install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用此文件，必须现在备份。

其他资源

- IBM Power® Virtual Server 的安装配置参数

20.5.7.1. 集群安装的最低资源要求

每台集群机器都必须满足以下最低要求：

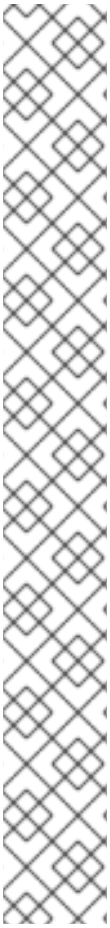
表 20.3. 最低资源要求

机器	操作系统	vCPU [1]	虚拟内存	Storage	每秒输入/输出 (IOPS) [2]
bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane（控制平面）	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS、RHEL 8.6 及更新版本 [3]	2	8 GB	100 GB	300

1. 当未启用并发多线程(SMT)或超线程时，一个 vCPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比例：（每个内核数的线程）× sockets = vCPU。

2. OpenShift Container Platform 和 Kubernetes 对磁盘性能非常敏感，建议使用更快的存储速度，特别是 control plane 节点上需要 10 ms p99 fsync 持续时间的 etcd。请注意，在许多云平台上，存储大小和 IOPS 可一起扩展，因此您可能需要过度分配存储卷来获取足够的性能。

3. 与所有用户置备的安装一样，如果您选择在集群中使用 RHEL 计算机器，则负责所有操作系统生命周期管理和维护，包括执行系统更新、应用补丁和完成所有其他必要的任务。RHEL 7 计算机器的使用已弃用，并已在 OpenShift Container Platform 4.10 及更新的版本中删除。



注意

从 OpenShift Container Platform 版本 4.13 开始，RHCOS 基于 RHEL 版本 9.2，它更新了微架构要求。以下列表包含每个架构需要的最小指令集架构 (ISA)：

- x86-64 体系结构需要 x86-64-v2 ISA
- ARM64 架构需要 ARMv8.0-A ISA
- IBM Power 架构需要 Power 9 ISA
- s390x 架构需要 z14 ISA

如需更多信息，请参阅 RHEL 架构。

如果平台的实例类型满足集群机器的最低要求，则 OpenShift Container Platform 支持使用它。

其他资源

- 优化存储

20.5.7.2. IBM Power Virtual Server 的自定义 install-config.yaml 文件示例

您可以自定义 install-config.yaml 文件，以指定有关 OpenShift Container Platform 集群平台的更多详情，或修改所需参数的值。

重要

此示例 YAML 文件仅供参考。您必须使用安装程序来获取 install-config.yaml 文件，并进行修改。

```
apiVersion: v1
baseDomain: example.com
compute: 1 2
```

```
- architecture: ppc64le
  hyperthreading: Enabled (3)
  name: worker
  platform:
    powervs:
      smtLevel: 8 (4)
  replicas: 3
controlPlane: (5) (6)
  architecture: ppc64le
  hyperthreading: Enabled (7)
  name: master
  platform:
    powervs:
      smtLevel: 8 (8)
  replicas: 3
metadata:
  creationTimestamp: null
  name: example-cluster-existing-vpc
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14 (9)
    hostPrefix: 23
  machineNetwork:
  - cidr: 192.168.0.0/24
  networkType: OVNKubernetes (10)
  serviceNetwork:
  - 172.30.0.0/16
platform:
  powervs:
    userID: ibm-user-id
    powervsResourceGroup: "ibmcloud-resource-group"
    region: powervs-region
    vpcRegion : vpc-region
    vpcName: name-of-existing-vpc (11)
    vpcSubnets: (12)
    - powervs-region-example-subnet-1
    zone: powervs-zone
    serviceInstanceGUID: "powervs-region-service-instance-guid"
credentialsMode: Manual
publish: External (13)
pullSecret: '{"auths": ...}' (14)
fips: false
sshKey: ssh-ed25519 AAAA... (15)
```

(1) (5)

如果没有提供这些参数和值，安装程序会提供默认值。

(2) (6)

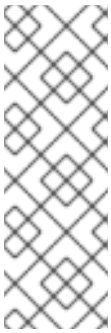
controlPlane 部分是一个单个映射，但 compute 部分是一系列映射。为满足不同数据结构的要求，compute 部分的第一行必须以连字符 - 开头，controlPlane 部分 的第一行则不以连字符开头。两个部分目前都定义一个机器池。仅使用一个 control plane 池。

3 7

是否要启用或禁用并发多线程或 超线程。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将 参数值设置为 Disabled 来禁用它。如果在某些集群机器中禁用并发多线程，则必须在所有集群机器中禁用它。

4 8

smtLevel 指定要设置为 control plane 和计算机器的 SMT 级别。支持的值有 1、2、4、8、'off' 和 'on'。默认值为 8。smtLevel 'off' 将 SMT 设置为 off，smtlevel 'on' 将 SMT 设置为集群节点上的默认值 8。



注意

如果没有启用并发多线程(SMT)或超线程，一个 vCPU 相当于一个物理内核。启用后，每个内核为(Thread (s)的总 vCPU 计算为（Thread (s) per socket) " Socket (s)）。smtLevel 控制每个内核的线程。在部署集群节点时，较低 SMT 级别可能需要额外分配的内核。您可以将 install-config.yaml 文件中的 'processors' 参数设置为适当的值，以满足成功部署 OpenShift Container Platform 的要求。

9

机器 CIDR 必须包含计算机器和 control plane 机器的子网。

10

用于安装的集群网络插件。支持的值是 OVNKubernetes。

11

指定现有 VPC 的名称。

12

指定现有 VPC 子网的名称。子网必须属于您指定的 VPC。为区域中的每个可用区指定一个子网。

13

指定如何发布集群的面向用户的端点。

14

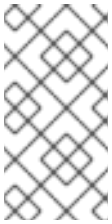
必需。安装程序会提示您输入这个值。

15

提供用于访问集群中机器的 sshKey 值。

重要

如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 ssh-agent 进程使用的 SSH 密钥。

20.5.7.3. 在安装过程中配置集群范围的代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 install-config.yaml 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 install-config.yaml 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 Proxy 对象的 spec.noProxy 字段中添加站点来绕过代理。

注意

Proxy 对象 status.noProxy 字段使用安装配置中的 networking.machineNetwork[].cidr、networking.clusterNetwork[].cidr 和 networking.serviceNetwork[] 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，Proxy 对象 status.noProxy 字段也会使用实例元数据端点填充(169.254.169.254)。

流程

1. 编辑 install-config.yaml 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 http。

2 用于创建集群外 HTTPS 连接的代理 URL。

3 要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 . 以仅匹配子域。例如，.y.com 匹配 x.y.com，但不匹配 y.com。使用 * 绕过所有目的地的代理。

4 如果提供，安装程序会在 openshift-config 命名空间中生成名为 user-ca-bundle 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 trusted-ca-bundle 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， Proxy 对象的 trustedCA 字段中也会引用此配置映射。additionalTrustBundle 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

5 可选：决定 Proxy 对象的配置以引用 trustedCA 字段中 user-ca-bundle 配置映射的策略。允许的值是 Proxyonly 和 Always。仅在配置了 http/https 代理时，使用 Proxyonly 引用 user-ca-bundle 配置映射。使用 Always 始终引用 user-ca-bundle 配置映射。默认值为 Proxyonly。

注意

安装程序不支持代理的 readinessEndpoints 字段。

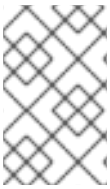
注意

如果安装程序超时，重启并使用安装程序的 wait-for 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. **保存该文件并在安装 OpenShift Container Platform 时引用。**

安装程序会创建一个名为 cluster 的集群范围代理，该代理 使用 提供的 install-config.yaml 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 cluster Proxy 对象，但它会有一个空 spec。



注意

只支持名为 cluster 的 Proxy 对象，且无法创建额外的代理。

20.5.8. 手动创建 IAM

安装集群需要 Cloud Credential Operator(CCO)以手动模式运行。虽然安装程序为手动模式配置 CCO，但您必须为云供应商指定身份和访问管理 secret。

您可以使用 Cloud Credential Operator (CCO)实用程序(ccoctl)创建所需的 IBM Cloud® 资源。

先决条件

- **您已配置了 ccoctl 二进制文件。**
- **您有一个现有的 install-config.yaml 文件。**

流程

1. 编辑 install-config.yaml 配置文件，使其包含将 credentialsMode 参数设置为 Manual。

install-config.yaml 配置文件示例

```
apiVersion: v1
baseDomain: cluster1.example.com
credentialsMode: Manual 1
compute:
- architecture: ppc64le
  hyperthreading: Enabled
```

1 添加这一行将 credentialsMode 参数设置为 Manual。

2. 要生成清单，请在包含安装程序的目录中运行以下命令：

```
$ openshift-install create manifests --dir <installation_directory>
```

3. 在包含安装程序的目录中，运行以下命令，使用安装文件中的发行镜像设置 $RELEASE_IMAGE 变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

4. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 CredentialsRequest 自定义资源 (CR) 列表：

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
  --included \ 1
  --install-config=<path_to_directory_with_installation_configuration>/install-config.yaml \ 2
  --to=<path_to_directory_for_credentials_requests> 3
```

1

2

3

指定要存储 CredentialsRequest 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

此命令为每个 CredentialsRequest 对象创建一个 YAML 文件。

CredentialsRequest 对象示例

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  labels:
    controller-tools.k8s.io: "1.0"
  name: openshift-image-registry-ibmcos
  namespace: openshift-cloud-credential-operator
spec:
  secretRef:
    name: installer-cloud-credentials
    namespace: openshift-image-registry
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: IBMCloudProviderSpec
    policies:
    - attributes:
      - name: serviceName
        value: cloud-object-storage
      roles:
      - crn:v1:bluemix:public:iam::::role:Viewer
      - crn:v1:bluemix:public:iam::::role:Operator
      - crn:v1:bluemix:public:iam::::role:Editor
      - crn:v1:bluemix:public:iam::::serviceRole:Reader
      - crn:v1:bluemix:public:iam::::serviceRole:Writer
    - attributes:
      - name: resourceType
        value: resource-group
      roles:
      - crn:v1:bluemix:public:iam::::role:Viewer
```

5. 为每个凭证请求创建服务 ID，分配定义的策略、创建 API 密钥并生成 secret：

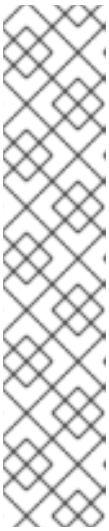
```
$ ccoctl ibmcloud create-service-id \
  --credentials-requests-dir=<path_to_credential_requests_directory> \1
  --name=<cluster_name> \2
  --output-dir=<installation_directory> \3
  --resource-group-name=<resource_group_name> 4
```

1. 指定包含组件 CredentialsRequest 对象文件的目录。

2. 指定 OpenShift Container Platform 集群的名称。

3. 可选：指定您希望 ccoctl 实用程序在其中创建对象的目录。默认情况下，实用程序在运行命令的目录中创建对象。

4. 可选：指定用于限制访问策略的资源组的名称。



注意

如果您的集群使用 TechPreviewNoUpgrade 功能集启用的技术预览功能，则必须包含 --enable-tech-preview 参数。

如果提供了不正确的资源组名称，安装会在 bootstrap 阶段失败。要查找正确的资源组名称，请运行以下命令：

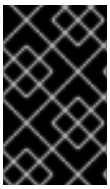
```
$ grep resourceGroup <installation_directory>/manifests/cluster-infrastructure-02-config.yml
```

验证

- 确保在集群的 manifests 目录中生成了适当的 secret。

20.5.9. 部署集群

您可以在兼容云平台上安装 OpenShift Container Platform。



重要

在初始安装过程中，您只能运行安装程序的 create cluster 命令一次。

先决条件

- 您已使用托管集群的云平台配置了帐户。
- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。
- 已确认主机上的云供应商帐户具有部署集群的正确权限。权限不正确的帐户会导致安装过程失败，并显示包括缺失权限的错误消息。

流程

- 进入包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

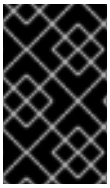
1. 对于 <installation_directory>，请指定自定义 ./install-config.yaml 文件的位置。
2. 要查看不同的安装详情，请指定 warn、debug 或 error，而不是 info。

验证

当集群部署成功完成时：

- 终端会显示用于访问集群的说明，包括指向 Web 控制台和 kubeadmin 用户的凭证的链接。

- 凭证信息还会输出到 <installation_directory>/.openshift_install.log.

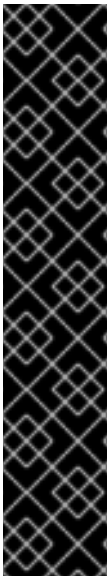


重要

不要删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 node-bootstrapper 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。

- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

20.5.10. 安装 OpenShift CLI

您可以安装 OpenShift CLI(oc)来使用命令行界面与 OpenShift Container Platform 进行交互。您可以在 Linux、Windows 或 macOS 上安装 oc。

重要

如果安装了旧版本的 oc，则可能无法使用 OpenShift Container Platform 4.16 中的所有命令。下载并安装新版本的 oc。

在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI(oc)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 产品变体 下拉列表中选择架构。
3. 从 版本 下拉列表中选择适当的版本。
4. 点 OpenShift v4.16 Linux Client 条目旁的 Download Now 来保存文件。
5. 解包存档：

```
$ tar xvf <file>
```

6. 将 oc 二进制文件放到 PATH 中的目录中。

要查看您的 PATH，请执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 oc 命令：

```
$ oc <command>
```

在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI(oc)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 版本 下拉列表中选择适当的版本。
3. 点 OpenShift v4.16 Windows Client 条目旁的 Download Now 来保存文件。
4. 使用 ZIP 程序解压存档。
5. 将 oc 二进制文件移到 PATH 中的目录中。

   要查看您的 PATH，请打开命令提示并执行以下命令：

   ```
   C:\> path
   ```

验证

- 安装 OpenShift CLI 后，可以使用 oc 命令：

  ```
  C:\> oc <command>
  ```

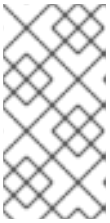
在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI(oc)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 版本 下拉列表中选择适当的版本。

3. 点 OpenShift v4.16 macOS Client 条目旁的 Download Now 来保存文件。



注意

对于 macOS arm64，请选择 OpenShift v4.16 macOS arm64 Client 条目。

4. 解包和解压存档。

5. 将 oc 二进制文件移到 PATH 的目录中。

要查看您的 PATH，请打开终端并执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 oc 命令：

```
$ oc <command>
```

20.5.11. 使用 CLI 登录集群

您可以通过导出集群 kubeconfig 文件，以默认系统用户身份登录集群。kubeconfig 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。
- 已安装 oc CLI。

流程

1. 导出 kubeadmin 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 <installation_directory>，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 oc 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

其他资源

- 访问Web控制台

20.5.12. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 关于远程健康监控

20.5.13. 后续步骤

- 自定义集群
- 可选：不使用远程健康报告

20.6. 在 IBM POWER VIRTUAL SERVER 上安装私有集群

在 OpenShift Container Platform 版本 4.16 中，您可以将私有集群安装到现有的 VPC 和 IBM Power® Virtual Server Workspace 中。安装程序会置备所需基础架构的其余部分，您可以进一步自定义这些基础架构。要自定义安装，请在安装集群前修改 install-config.yaml 文件中的参数。

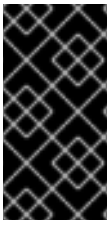
20.6.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读有关选择集群安装方法的文档，并为用户准备它。
- 已将 IBM Cloud® 帐户配置为托管集群。
- 如果使用防火墙，则会将其配置为允许集群需要访问的站点。
- 在安装集群前已经配置了 ccoctl 工具。如需更多信息，请参阅配置 Cloud Credential Operator 工具程序。

20.6.2. 私有集群

您可以部署不公开外部端点的私有 OpenShift Container Platform 集群。私有集群只能从内部网络访问，且无法在互联网中看到。

默认情况下，OpenShift Container Platform 被置备为使用可公开访问的 DNS 和端点。在部署集群时，私有集群会将 DNS、Ingress Controller 和 API 服务器设置为私有。这意味着集群资源只能从您的内部网络访问，且不能在互联网中看到。



重要

如果集群有任何公共子网，管理员创建的负载均衡器服务可能会公开访问。为确保集群安全性，请验证这些服务是否已明确标注为私有。

要部署私有集群，您必须：

- 使用满足您的要求的现有网络。
- 使用 IBM Cloud® DNS Services 创建 DNS 区域，并将其指定为集群的基域。如需更多信息，请参阅"使用 IBM Cloud® DNS 服务来配置 DNS 解析"。
- 从有权访问的机器中部署：
  - 您置备的云的 API 服务。
  - 您调配的网络上的主机。
  - 用于获取安装介质的互联网。

您可以使用符合这些访问要求的机器，并按照您的公司规定进行操作。例如，此机器可以是云网络上的堡垒主机，也可以是可通过 VPN 访问网络的机器。

20.6.3. IBM Power Virtual Server 中的私有集群

要在 IBM Power® Virtual Server 上创建私有集群，您必须提供一个现有的私有虚拟私有云 (VPC) 和子网来托管集群。安装程序还必须能够解析集群所需的 DNS 记录。安装程序只为内部流量配置 Ingress Operator 和 API 服务器。

集群仍然需要访问互联网来访问 IBM Cloud® API。

安装私有集群时不需要或创建以下项目：

- 公共子网
- 支持公共入口的公共网络负载均衡器
- 与集群的 baseDomain 匹配的公共 DNS 区域

您还需要创建一个 IBM® DNS 服务，其中包含与您的 baseDomain 匹配的 DNS 区域。与使用 IBM® CIS 用于 DNS 的 Power VS 上的标准部署不同，您必须为 DNS 服务使用 IBM® DNS。

20.6.3.1. 限制

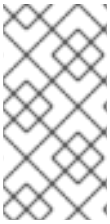
IBM Power® Virtual Server 上的私有集群只受到与集群部署的现有 VPC 相关的限制。

20.6.4. 使用 VPC 的要求

您必须在安装集群前正确配置现有的 VPC 及其子网。在这种情况下，安装程序不会创建 VPC 或 VPC 子网。

安装程序无法：

- 从属网络范围供集群使用
- 为子网设置路由表
- 设置 VPC 选项，如 DHCP



注意

安装程序要求您使用由云提供的 DNS 服务器。不支持使用自定义 DNS 服务器，并导致安装失败。

20.6.4.1. VPC 验证

VPC 和所有子网都必须位于现有资源组中。集群部署到此资源组。

作为安装的一部分，在 install-config.yaml 文件中指定以下内容：

- 资源组的名称
- VPC 的名称
- VPC 子网的名称

要确保您提供的子网适合，安装程序会确认您指定的所有子网都存在。

注意

不支持子网 ID。

20.6.4.2. 集群间隔离

如果您将 OpenShift Container Platform 部署到现有网络中，集群服务的隔离将在以下方面减少：

- 整个网络允许 ICMP 入站流量。
- 整个网络都允许 TCP 端口 22 入站流量 (SSH)。
- 整个网络都允许 control plane TCP 6443 Ingress (Kubernetes API)。
- 整个网络都允许 control plane TCP 22623 Ingress (MCS)。

20.6.5. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

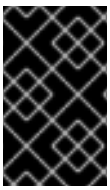
如果您的集群无法直接访问互联网，则可以在置备的某些类型的基础架构上执行受限网络安装。在此过程中，您可以下载所需的内容，并使用它为镜像 registry 填充安装软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群前，您要更新镜像 registry 的内容。

20.6.6. 为集群节点 SSH 访问生成密钥对

在 OpenShift Container Platform 安装过程中，您可以为安装程序提供 SSH 公钥。密钥通过它们的 Ignition 配置文件传递给 Red Hat Enterprise Linux CoreOS(RHCOS)节点，用于验证对节点的 SSH 访问。密钥添加到每个节点上 core 用户的 ~/.ssh/authorized_keys 列表中，这将启用免密码身份验证。

将密钥传递给节点后，您可以使用密钥对作为用户 核心 通过 SSH 连接到 RHCOS 节点。若要通过 SSH 访问节点，必须由 SSH 为您的本地用户管理私钥身份。

如果要通过 SSH 连接到集群节点来执行安装调试或灾难恢复，则必须在安装过程中提供 SSH 公钥。./openshift-install gather 命令还需要在集群节点上设置 SSH 公钥。



重要

不要在生产环境中跳过这个过程，在生产环境中需要灾难恢复和调试。

注意

您必须使用本地密钥，而不是使用特定平台方法配置 的密钥，如 AWS 密钥对。

流程

1. 如果您在本地计算机上没有可用于在集群节点上进行身份验证的现有 SSH 密钥对，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 ~/.ssh/id_ed25519。如果您已有密钥对，请确保您的公钥位于 ~/.ssh 目录中。

2. 查看公共 SSH 密钥：

```
$ cat <path>/<file_name>.pub
```

例如，运行以下命令来查看 ~/.ssh/id_ed25519.pub 公钥：

```
$ cat ~/.ssh/id_ed25519.pub
```

3. 将 SSH 私钥身份添加到本地用户的 SSH 代理（如果尚未添加）。在集群节点上，或者要使用 ./openshift-install gather 命令，需要对该密钥进行 SSH 代理管理，才能在集群节点上进行免密码 SSH 身份验证。

注意

在某些发行版中，自动管理默认 SSH 私钥身份，如 ~/.ssh/id_rsa 和 ~/.ssh/id_dsa。

a. 如果 ssh-agent 进程尚未为您的本地用户运行，请将其作为后台任务启动：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

4. 将 SSH 私钥添加到 ssh-agent ：

```
$ ssh-add <path>/<file_name> 1
```

1 指定 SSH 私钥的路径和文件名，如 ~/.ssh/id_ed25519.pub

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

后续步骤

- 安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

20.6.7. 获取安装程序

在安装 OpenShift Container Platform 前，将安装文件下载到您用于安装的主机上。

先决条件

- 您有一台运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB。

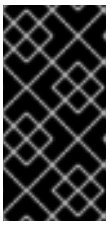
流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐

户，请使用您的凭证登录。如果没有，请创建一个帐户。

2. 选择您的基础架构供应商。

3. 进入到安装类型的页面，下载与您的主机操作系统和架构对应的安装程序，并将该文件放在您要存储安装配置文件的目录中。



重要

安装程序会在用来安装集群的计算机上创建几个文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。

重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，请为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret。此 pull secret 允许您与所含授权机构提供的服务进行身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

20.6.8. 导出 API 密钥

您必须将您创建的 API 密钥设置为全局变量；安装程序会在启动期间设置 API 密钥。

先决条件

- 您已为 IBM Cloud® 帐户创建了用户 API 密钥或服务 ID API 密钥。

流程

- 将帐户的 API 密钥导出为全局变量：

```
$ export IBMCLOUD_API_KEY=<api_key>
```

重要

您必须按照指定方式设置变量名称，安装程序需要在启动期间存在变量名称。

20.6.9. 手动创建安装配置文件

安装集群要求您手动创建安装配置文件。

先决条件

- 您在本地机器上有一个 SSH 公钥来提供给安装程序。该密钥将用于在集群节点上进行 SSH 身份验证，以进行调试和灾难恢复。
- 已获取 OpenShift Container Platform 安装程序和集群的 pull secret。

流程

1. 创建一个安装目录来存储所需的安装资产：

```
$ mkdir <installation_directory>
```

重要

您必须创建一个目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

2. 自定义提供的 install-config.yaml 文件模板示例，并将其保存在 <installation_directory> 中。

注意

此配置文件必须命名为 install-config.yaml。

3. **备份 install-config.yaml 文件，以便您可以使用它安装多个集群。**

重要

install-config.yaml 文件会在安装过程的下一步中使用。现在必须备份它。

其他资源

- **IBM Power® Virtual Server 的安装配置参数**

20.6.9.1. 集群安装的最低资源要求

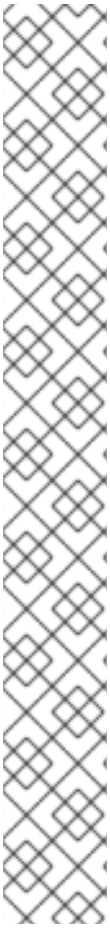
每台集群机器都必须满足以下最低要求：

表 20.4. 最低资源要求

机器	操作系统	vCPU [1]	虚拟内存	Storage	每秒输入/输出 (IOPS) [2]
bootstrap	RHCOS	2	16 GB	100 GB	300
Control plane（控制平面）	RHCOS	2	16 GB	100 GB	300
Compute	RHCOS	2	8 GB	100 GB	300

1. **当未启用并发多线程(SMT)或超线程时，一个 vCPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比例：（每个内核数的线程）× sockets = vCPU。**

2. **OpenShift Container Platform 和 Kubernetes 对磁盘性能敏感，建议使用更快的存储，特别是 control plane 节点上的 etcd。请注意，在许多云平台上，存储大小和 IOPS 可一起扩展，因此您可能需要过度分配存储卷来获取足够的性能。**



注意

从 OpenShift Container Platform 版本 4.13 开始，RHCOS 基于 RHEL 版本 9.2，它更新了微架构要求。以下列表包含每个架构需要的最小指令集架构 (ISA)：

- x86-64 体系结构需要 x86-64-v2 ISA
- ARM64 架构需要 ARMv8.0-A ISA
- IBM Power 架构需要 Power 9 ISA
- s390x 架构需要 z14 ISA

如需更多信息，请参阅 RHEL 架构。

如果平台的实例类型满足集群机器的最低要求，则 OpenShift Container Platform 支持使用它。

其他资源

- 优化存储

20.6.9.2. IBM Power Virtual Server 的自定义 install-config.yaml 文件示例

您可以自定义 install-config.yaml 文件，以指定有关 OpenShift Container Platform 集群平台的更多详情，或修改所需参数的值。

重要

此示例 YAML 文件仅供参考。您必须使用安装程序来获取 install-config.yaml 文件，并进行修改。

```
apiVersion: v1
baseDomain: example.com
compute: 1 2
```

```
- architecture: ppc64le
  hyperthreading: Enabled (3)
  name: worker
  platform:
    powervs:
      smtLevel: 8 (4)
  replicas: 3
controlPlane: (5) (6)
  architecture: ppc64le
  hyperthreading: Enabled (7)
  name: master
  platform:
    powervs:
      smtLevel: 8 (8)
  replicas: 3
metadata:
  creationTimestamp: null
  name: example-private-cluster-name
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14 (9)
    hostPrefix: 23
  machineNetwork:
  - cidr: 192.168.0.0/24
  networkType: OVNKubernetes (10)
  serviceNetwork:
  - 172.30.0.0/16
platform:
  powervs:
    userID: ibm-user-id
    powervsResourceGroup: "ibmcloud-resource-group"
    region: powervs-region
    vpcName: name-of-existing-vpc (11)
    vpcSubnets:
    - powervs-region-example-subnet-1
    vpcRegion : vpc-region
    zone: powervs-zone
    serviceInstanceGUID: "powervs-region-service-instance-guid"
publish: Internal (12)
pullSecret: '{"auths": ...}' (13)
sshKey: ssh-ed25519 AAAA... (14)
```

(1) (5)

如果没有提供这些参数和值，安装程序会提供默认值。

(2) (6)

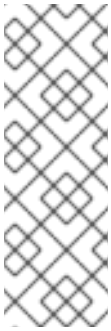
controlPlane 部分是一个单个映射，但 compute 部分是一系列映射。为满足不同数据结构的要求，compute 部分的第一行必须以连字符 - 开头，controlPlane 部分 的第一行则不以连字符开头。两个部分目前都定义一个机器池。仅使用一个 control plane 池。

3 7

是否要启用或禁用并发多线程或 超线程。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将 参数值设置为 Disabled 来禁用它。如果在某些集群机器中禁用并发多线程，则必须在所有集群机器中禁用它。

4 8

smtLevel 指定要设置为 control plane 和计算机器的 SMT 级别。支持的值有 1、2、4、8、'off' 和 'on'。默认值为 8。smtLevel 'off' 将 SMT 设置为 off，smtlevel 'on' 将 SMT 设置为集群节点上的默认值 8。



注意

如果没有启用并发多线程(SMT)或超线程，一个 vCPU 相当于一个物理内核。启用后，每个内核为(Thread (s)的总 vCPU 计算为（Thread (s) per socket) " Socket (s)）。smtLevel 控制每个内核的线程。在部署集群节点时，较低 SMT 级别可能需要额外分配的内核。您可以将 install-config.yaml 文件中的 'processors' 参数设置为适当的值，以满足成功部署 OpenShift Container Platform 的要求。

9

机器 CIDR 必须包含计算机器和 control plane 机器的子网。

10

要安装的集群网络插件。默认值 OVNKubernetes 是唯一支持的值。

11

指定现有 VPC 的名称。

12

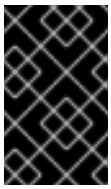
指定如何发布集群的面向用户的端点。将 publish 设置为 Internal 以部署私有集群。

13

必需。安装程序会提示您输入这个值。

14

提供用于访问集群中机器的 sshKey 值。



重要

如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 ssh-agent 进程使用的 SSH 密钥。

20.6.9.3. 在安装过程中配置集群范围的代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 install-config.yaml 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 install-config.yaml 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 Proxy 对象的 spec.noProxy 字段中添加站点来绕过代理。

注意

Proxy 对象 status.noProxy 字段使用安装配置中的 networking.machineNetwork[].cidr、networking.clusterNetwork[].cidr 和 networking.serviceNetwork[] 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，Proxy 对象 status.noProxy 字段也会使用实例元数据端点填充(169.254.169.254)。

流程

1. 编辑 install-config.yaml 文件并添加代理设置。例如：

```
apiVersion: v1
```

```
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1

用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 http。

2

用于创建集群外 HTTPS 连接的代理 URL。

3

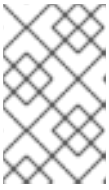
要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 . 以仅匹配子域。例如，.y.com 匹配 x.y.com，但不匹配 y.com。使用 * 绕过所有目的地的代理。

4

如果提供，安装程序会在 openshift-config 命名空间中生成名为 user-ca-bundle 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 trusted-ca-bundle 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， Proxy 对象的 trustedCA 字段中也会引用此配置映射。additionalTrustBundle 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

5

可选：决定 Proxy 对象的配置以引用 trustedCA 字段中 user-ca-bundle 配置映射的策略。允许的值是 Proxyonly 和 Always。仅在配置了 http/https 代理时，使用 Proxyonly 引用 user-ca-bundle 配置映射。使用 Always 始终引用 user-ca-bundle 配置映射。默认值为 Proxyonly。



注意

安装程序不支持代理的 readinessEndpoints 字段。

注意

如果安装程序超时，重启并使用安装程序的 wait-for 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. 保存该文件并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 cluster 的集群范围代理，该代理 使用 提供的 install-config.yaml 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 cluster Proxy 对象，但它会有一个空 spec。

注意

只支持名为 cluster 的 Proxy 对象，且无法创建额外的代理。

20.6.10. 手动创建 IAM

安装集群需要 Cloud Credential Operator(CCO)以手动模式运行。虽然安装程序为手动模式配置 CCO，但您必须为云供应商指定身份和访问管理 secret。

您可以使用 Cloud Credential Operator (CCO)实用程序(ccoctl)创建所需的 IBM Cloud® 资源。

先决条件

- 您已配置了 ccoctl 二进制文件。
- 您有一个现有的 install-config.yaml 文件。

流程

1. 编辑 install-config.yaml 配置文件，使其包含将 credentialsMode 参数设置为 Manual。

install-config.yaml 配置文件示例

```
apiVersion: v1
baseDomain: cluster1.example.com
credentialsMode: Manual (1)
compute:
- architecture: ppc64le
  hyperthreading: Enabled
```

(1) 添加这一行将 credentialsMode 参数设置为 Manual。

2. 要生成清单，请在包含安装程序的目录中运行以下命令：

```
$ openshift-install create manifests --dir <installation_directory>
```

3. 在包含安装程序的目录中，运行以下命令，使用安装文件中的发行镜像设置 $RELEASE_IMAGE 变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

4. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 CredentialsRequest 自定义资源 (CR) 列表：

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
  --included \(1)
  --install-config=<path_to_directory_with_installation_configuration>/install-
config.yaml \(2)
  --to=<path_to_directory_for_credentials_requests> (3)
```

(1) --included 参数仅包含特定集群配置所需的清单。

(2) 指定 install-config.yaml 文件的位置。

(3)

此命令为每个 CredentialsRequest 对象创建一个 YAML 文件。

CredentialsRequest 对象示例

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  labels:
    controller-tools.k8s.io: "1.0"
  name: openshift-image-registry-ibmcos
  namespace: openshift-cloud-credential-operator
spec:
  secretRef:
    name: installer-cloud-credentials
    namespace: openshift-image-registry
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: IBMCloudProviderSpec
    policies:
    - attributes:
      - name: serviceName
        value: cloud-object-storage
      roles:
      - crn:v1:bluemix:public:iam::::role:Viewer
      - crn:v1:bluemix:public:iam::::role:Operator
      - crn:v1:bluemix:public:iam::::role:Editor
      - crn:v1:bluemix:public:iam::::serviceRole:Reader
      - crn:v1:bluemix:public:iam::::serviceRole:Writer
    - attributes:
      - name: resourceType
        value: resource-group
      roles:
      - crn:v1:bluemix:public:iam::::role:Viewer
```

5. 为每个凭证请求创建服务 ID，分配定义的策略、创建 API 密钥并生成 secret：

```
$ ccoctl ibmcloud create-service-id \
  --credentials-requests-dir=<path_to_credential_requests_directory> \ 1
  --name=<cluster_name> \ 2
  --output-dir=<installation_directory> \ 3
  --resource-group-name=<resource_group_name> 4
```

1

指定包含组件 CredentialsRequest 对象文件的目录。

2

指定 OpenShift Container Platform 集群的名称。

3

可选：指定您希望 ccoctl 实用程序在其中创建对象的目录。默认情况下，实用程序在运行命令的目录中创建对象。

4

可选：指定用于限制访问策略的资源组的名称。

注意

如果您的集群使用 TechPreviewNoUpgrade 功能集启用的技术预览功能，则必须包含 --enable-tech-preview 参数。

如果提供了不正确的资源组名称，安装会在 bootstrap 阶段失败。要查找正确的资源组名称，请运行以下命令：

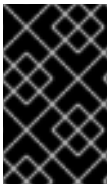
```
$ grep resourceGroup <installation_directory>/manifests/cluster-
infrastructure-02-config.yml
```

验证

- 确保在集群的 manifests 目录中生成了适当的 secret。

20.6.11. 部署集群

您可以在兼容云平台上安装 OpenShift Container Platform。



重要

在初始安装过程中，您只能运行安装程序的 create cluster 命令一次。

先决条件

- 您已使用托管集群的云平台配置了帐户。
- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。
- 已确认主机上的云供应商帐户具有部署集群的正确权限。权限不正确的帐户会导致安装过程失败，并显示包括缺失权限的错误消息。

流程

- 进入包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
  --log-level=info 2
```

1. 对于 <installation_directory>，请指定自定义 ./install-config.yaml 文件的位置。
2. 要查看不同的安装详情，请指定 warn、debug 或 error，而不是 info。

验证

当集群部署成功完成时：

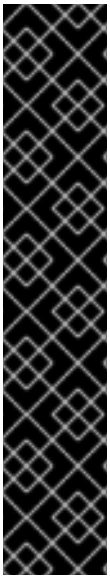
- 终端会显示用于访问集群的说明，包括指向 Web 控制台和 kubeadmin 用户的凭证的链接。
- 凭证信息还会输出到 <installation_directory>/.openshift_install.log.

重要

不要删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 node-bootstrapper 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。

- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

20.6.12. 安装 OpenShift CLI

您可以安装 OpenShift CLI(oc)来使用命令行界面与 OpenShift Container Platform 进行交互。您可以在 Linux、Windows 或 macOS 上安装 oc。

重要

如果安装了旧版本的 oc，则可能无法使用 OpenShift Container Platform 4.16 中的所有命令。下载并安装新版本的 oc。

在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI(oc)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 产品变体 下拉列表中选择架构。
3. 从 版本 下拉列表中选择适当的版本。
4. 点 OpenShift v4.16 Linux Client 条目旁的 Download Now 来保存文件。
5. 解包存档：

```
$ tar xvf <file>
```

6. 将 oc 二进制文件放到 PATH 中的目录中。

   要查看您的 PATH，请执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 oc 命令：

```
$ oc <command>
```

在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI(oc)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 版本 下拉列表中选择适当的版本。

3. 点 OpenShift v4.16 Windows Client 条目旁的 Download Now 来保存文件。

4. 使用 ZIP 程序解压存档。

5. 将 oc 二进制文件移到 PATH 中的目录中。

   要查看您的 PATH，请打开命令提示并执行以下命令：

```
C:\> path
```

验证

- 安装 OpenShift CLI 后，可以使用 oc 命令：

```
C:\> oc <command>
```

在 macOS 上安装 OpenShift CLI

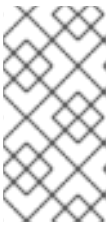
您可以按照以下流程在 macOS 上安装 OpenShift CLI(oc)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。

2. 从 版本 下拉列表中选择适当的版本。

3. 点 OpenShift v4.16 macOS Client 条目旁的 Download Now 来保存文件。



注意

对于 macOS arm64，请选择 OpenShift v4.16 macOS arm64 Client 条目。

4. 解包和解压存档。

5. 将 oc 二进制文件移到 PATH 的目录中。

   要查看您的 PATH，请打开终端并执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 oc 命令：

```
$ oc <command>
```

20.6.13. 使用 CLI 登录集群

您可以通过导出集群 kubeconfig 文件，以默认系统用户身份登录集群。kubeconfig 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。
- 已安装 oc CLI。

流程

1. 导出 kubeadmin 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

   1. 对于 <installation_directory>，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 oc 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

其他资源

- 访问Web控制台

20.6.14. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 关于远程健康监控

20.6.15. 后续步骤

- 自定义集群
- 可选： 不使用远程健康报告

20.7. 在受限网络中的 IBM POWER VIRTUAL SERVER 上安装集群

在 OpenShift Container Platform 4.16 中，您可以通过在 IBM Cloud® 的现有 Virtual Private Cloud

(VPC)上创建安装发行内容的内部镜像在受限网络中的 IBM Cloud® 上安装集群。

20.7.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读有关 选择集群安装方法的文档，并为用户准备它。
- 已将 IBM Cloud® 帐户配置为托管集群。
- 您已 将断开连接的安装的镜像镜像 到 registry，并获取了 OpenShift Container Platform 版本的 imageContentSources 数据。

  重要

  由于安装介质位于镜像主机上，因此您可以使用该计算机完成所有安装步骤。
- 在 IBM Cloud® 中有一个现有的 VPC。在受限网络中安装集群时，无法使用安装程序置备的 VPC。您必须使用用户置备的 VPC 来满足以下要求之一：
  - 包含镜像 registry
  - 具有防火墙规则或对等连接，以访问在其他位置托管的镜像 registry
- 如果使用防火墙，将其配置为允许集群需要访问的站点。
- 在安装集群前已经配置了 ccoctl 工具。如需更多信息，请参阅配置 Cloud Credential Operator 工具程序。

20.7.2. 关于在受限网络中安装

在 OpenShift Container Platform 4.16 中，可以执行不需要有效的互联网连接来获取软件组件的安

装。受限网络安装可以使用安装程序置备的基础架构或用户置备的基础架构完成，具体取决于您要安装集群的云平台。

如果您选择在云平台中执行受限网络安装，您仍需要访问其云 API。有些云功能，比如 Amazon Web Service 的 Route 53 DNS 和 IAM 服务，需要访问互联网。根据您的网络，在裸机硬件、Nutanix 或 VMware vSphere 上安装可能需要较少的互联网访问。

要完成受限网络安装，您必须创建一个 registry，以镜像 OpenShift 镜像 registry 的内容并包含安装介质。您可以在镜像主机上创建此 registry，该主机可同时访问互联网和您的封闭网络，也可以使用满足您的限制条件的其他方法。

20.7.2.1. 其他限制

受限网络中的集群有以下额外限制和限制：

- ClusterVersion 状态包含一个 Unable to retrieve available updates 错误。
- 默认情况下，您无法使用 Developer Catalog 的内容，因为您无法访问所需的镜像流标签。

20.7.3. 关于使用自定义 VPC

在 OpenShift Container Platform 4.16 中，您可以将集群部署到现有 IBM® Virtual Private Cloud (VPC)的子网。

20.7.3.1. 使用 VPC 的要求

您必须在安装集群前正确配置现有的 VPC 及其子网。在这种情况下，安装程序不会创建 VPC 或 VPC 子网。

安装程序无法：

- 从属网络范围供集群使用
- 为子网设置路由表

- 设置 VPC 选项，如 DHCP

注意

安装程序要求您使用由云提供的 DNS 服务器。不支持使用自定义 DNS 服务器，并导致安装失败。

20.7.3.2. VPC 验证

VPC 和所有子网都必须位于现有资源组中。集群部署到此资源组。

作为安装的一部分，在 install-config.yaml 文件中指定以下内容：

- 资源组的名称
- VPC 的名称
- VPC 子网的名称

要确保您提供的子网适合，安装程序会确认您指定的所有子网都存在。

注意

不支持子网 ID。

20.7.3.3. 集群间隔离

如果您将 OpenShift Container Platform 部署到现有网络中，集群服务的隔离将在以下方面减少：

- 整个网络允许 ICMP 入站流量。

- 整个网络都允许 TCP 端口 22 入站流量 (SSH)。
- 整个网络都允许 control plane TCP 6443 Ingress (Kubernetes API)。
- 整个网络都允许 control plane TCP 22623 Ingress (MCS)。

20.7.4. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来获得用来安装集群的镜像。

您必须具有以下互联网访问权限：

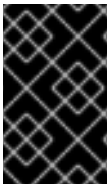
- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

20.7.5. 为集群节点 SSH 访问生成密钥对

在 OpenShift Container Platform 安装过程中，您可以为安装程序提供 SSH 公钥。密钥通过它们的 Ignition 配置文件传递给 Red Hat Enterprise Linux CoreOS(RHCOS)节点，用于验证对节点的 SSH 访问。密钥添加到每个节点上 core 用户的 ~/.ssh/authorized_keys 列表中，这将启用免密码身份验证。

将密钥传递给节点后，您可以使用密钥对作为用户 核心 通过 SSH 连接到 RHCOS 节点。若要通过 SSH 访问节点，必须由 SSH 为您的本地用户管理私钥身份。

如果要通过 SSH 连接到集群节点来执行安装调试或灾难恢复，则必须在安装过程中提供 SSH 公钥。./openshift-install gather 命令还需要在集群节点上设置 SSH 公钥。



重要

不要在生产环境中跳过这个过程，在生产环境中需要灾难恢复和调试。

注意

您必须使用本地密钥，而不是使用特定平台方法配置 的密钥，如 AWS 密钥对。

流程

1. 如果您在本地计算机上没有可用于在集群节点上进行身份验证的现有 SSH 密钥对，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 ~/.ssh/id_ed25519。如果您已有密钥对，请确保您的公钥位于 ~/.ssh 目录中。

2. 查看公共 SSH 密钥：

```
$ cat <path>/<file_name>.pub
```

例如，运行以下命令来查看 ~/.ssh/id_ed25519.pub 公钥：

```
$ cat ~/.ssh/id_ed25519.pub
```

3. 将 SSH 私钥身份添加到本地用户的 SSH 代理（如果尚未添加）。在集群节点上，或者要使用 ./openshift-install gather 命令，需要对该密钥进行 SSH 代理管理，才能在集群节点上进行免密码 SSH 身份验证。

注意

在某些发行版中，自动管理默认 SSH 私钥身份，如 ~/.ssh/id_rsa 和 ~/.ssh/id_dsa。

a. 如果 ssh-agent 进程尚未为您的本地用户运行，请将其作为后台任务启动：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

4. 将 SSH 私钥添加到 ssh-agent ：

```
$ ssh-add <path>/<file_name> 1
```

1 指定 SSH 私钥的路径和文件名，如 ~/.ssh/id_ed25519.pub

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

后续步骤

- 安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

20.7.6. 导出 API 密钥

您必须将您创建的 API 密钥设置为全局变量；安装程序会在启动期间设置 API 密钥。

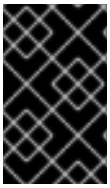
先决条件

- 您已为 IBM Cloud® 帐户创建了用户 API 密钥或服务 ID API 密钥。

流程

- 将帐户的 API 密钥导出为全局变量：

```
$ export IBMCLOUD_API_KEY=<api_key>
```



重要

您必须按照指定方式设置变量名称，安装程序需要在启动期间存在变量名称。

20.7.7. 创建安装配置文件

您可以自定义 OpenShift Container Platform 集群。

先决条件

- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。对于受限网络安装，这些文件位于您的镜像主机上。
- 您有创建镜像 registry 期间生成的 imageContentSources 值。
- 您已获取了镜像 registry 的证书内容。
- 您已检索了 Red Hat Enterprise Linux CoreOS (RHCOS) 镜像，并将其上传到可访问的位置。

流程

1. 创建 install-config.yaml 文件。
   a. 进入包含安装程序的目录并运行以下命令：

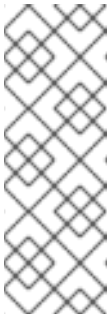
```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

1

对于 <installation_directory>，请指定要存储安装程序创建的文件的目录名称。

在指定目录时：

- 验证该目录是否具有执行权限。在安装目录中运行 Terraform 二进制文件需要这个权限。
- 使用空目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。



注意

始终删除 ~/.powervs 目录，以避免重复使用过时的配置。运行以下命令：

```
$ rm -rf ~/.powervs
```

b. 在提示符处，提供云的配置详情：

i. 可选：选择用于访问集群机器的 SSH 密钥。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 ssh-agent 进程使用的 SSH 密钥。

ii. 选择 powervs 作为目标平台。

iii. 选择要将集群部署到的区域。

iv. 选择要将集群部署到的区域。

v. 选择集群要部署到的基域。基域与您为集群创建的公共 DNS 区对应。

vi. 为集群输入描述性名称。

2. 编辑 install-config.yaml 文件，以提供在受限网络中安装所需的额外信息。

a. 更新 pullSecret 值，使其包含 registry 的身份验证信息：

```
pullSecret: '{"auths":{"<mirror_host_name>:5000": {"auth": "
<credentials>","email": "you@example.com"}}}'
```

对于 <mirror_host_name>，请指定 您在镜像 registry 证书中指定的 registry 域名 ；对于 <credentials>， 请指定您的镜像 registry 的 base64 编码用户名和密码。

b. 添加 additionalTrustBundle 参数和值。

```
additionalTrustBundle: |
  -----BEGIN CERTIFICATE-----

ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
  -----END CERTIFICATE-----
```

该值必须是您用于镜像 registry 的证书文件内容。证书文件可以是现有的可信证书颁发机构，也可以是您为镜像 registry 生成的自签名证书。

c. 定义 VPC 的网络和子网，以便在父 platform.ibmcloud 字段下安装集群：

```
vpcName: <existing_vpc>
vpcSubnets: <vpcSubnet>
```

对于 platform.powervs.vpcName，请指定现有 IBM Cloud® 的名称。对于

platform.powervs.vpcSubnets，请指定现有子网。

d. 添加镜像内容资源，类似于以下 YAML 摘录：

```
imageContentSources:
- mirrors:
  - <mirror_host_name>:5000/<repo_name>/release
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <mirror_host_name>:5000/<repo_name>/release
  source: registry.redhat.io/ocp/release
```

对于这些值，请使用您在创建镜像 registry 时记录的 imageContentSources。

e. 可选：将发布策略设置为 Internal：

```
publish: Internal
```

通过设置这个选项，您可以创建一个内部 Ingress Controller 和一个私有负载均衡器。

3. 对您需要的 install-config.yaml 文件进行任何其他修改。

有关参数的更多信息，请参阅"安装配置参数"。

4. 备份 install-config.yaml 文件，以便您可以使用它安装多个集群。

重要

install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用此文件，必须现在备份。

其他资源

- IBM Power® Virtual Server 的安装配置参数

20.7.7.1. 集群安装的最低资源要求

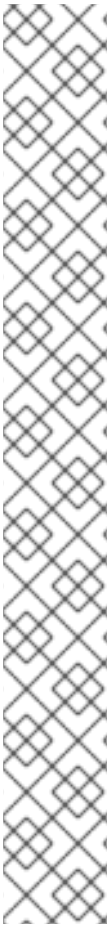
每台集群机器都必须满足以下最低要求：

表 20.5. 最低资源要求

机器	操作系统	vCPU [1]	虚拟内存	Storage	每秒输入/输出 (IOPS) [2]
bootstrap	RHCOS	2	16 GB	100 GB	300
Control plane（控制平面）	RHCOS	2	16 GB	100 GB	300
Compute	RHCOS	2	8 GB	100 GB	300

1. 当未启用并发多线程(SMT)或超线程时，一个 vCPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比例：（每个内核数的线程）× sockets = vCPU。

2. OpenShift Container Platform 和 Kubernetes 对磁盘性能敏感，建议使用更快的存储，特别是 control plane 节点上的 etcd。请注意，在许多云平台上，存储大小和 IOPS 可一起扩展，因此您可能需要过度分配存储卷来获取足够的性能。



注意

从 OpenShift Container Platform 版本 4.13 开始，RHCOS 基于 RHEL 版本 9.2，它更新了微架构要求。以下列表包含每个架构需要的最小指令集架构 (ISA)：

- x86-64 体系结构需要 x86-64-v2 ISA
- ARM64 架构需要 ARMv8.0-A ISA
- IBM Power 架构需要 Power 9 ISA
- s390x 架构需要 z14 ISA

如需更多信息，请参阅 RHEL 架构。

如果平台的实例类型满足集群机器的最低要求，则 OpenShift Container Platform 支持使用它。

其他资源

- 优化存储

20.7.7.2. IBM Power Virtual Server 的自定义 install-config.yaml 文件示例

您可以自定义 install-config.yaml 文件，以指定有关 OpenShift Container Platform 集群平台的更多详情，或修改所需参数的值。

重要

此示例 YAML 文件仅供参考。您必须使用安装程序来获取 install-config.yaml 文件，并进行修改。

```
apiVersion: v1
baseDomain: example.com (1)
controlPlane: (2) (3)
  hyperthreading: Enabled (4)
  name: master
  platform:
    powervs:
      smtLevel: 8 (5)
  replicas: 3
compute: (6) (7)
- hyperthreading: Enabled (8)
  name: worker
  platform:
    powervs:
      smtLevel: 8 (9)
    ibmcloud: {}
  replicas: 3
metadata:
  name: example-restricted-cluster-name (10)
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14 (11)
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16 (12)
  networkType: OVNKubernetes (13)
  serviceNetwork:
  - 192.168.0.0/24
platform:
```

```
  powervs:
    userid: ibm-user-id
    powervsResourceGroup: "ibmcloud-resource-group" 14
    region: "powervs-region"
    vpcRegion: "vpc-region"
    vpcName: name-of-existing-vpc 15
    vpcSubnets: 16
      - name-of-existing-vpc-subnet
    zone: "powervs-zone"
    serviceInstanceID: "service-instance-id"
publish: Internal
credentialsMode: Manual
pullSecret: '{"auths":{"<local_registry>": {"auth": "<credentials>","email":
"you@example.com"}}}' 17
sshKey: ssh-ed25519 AAAA... 18
additionalTrustBundle: | 19
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
imageContentSources: 20
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-v4.0-art-dev
```

1 10

必需。

2 6

如果没有提供这些参数和值，安装程序会提供默认值。

3 7

controlPlane 部分是一个单个映射，但 compute 部分是一系列映射。为满足不同数据结构的要求，compute 部分的第一行必须以连字符 - 开头，controlPlane 部分 的第一行则不以连字符开头。仅使用一个 control plane 池。

4 8

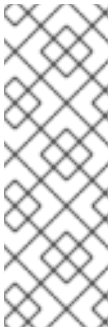
启用或禁用并发多线程，也称为 Hyper-Threading。默认情况下，启用并发多线程以提高机器内核的性能。您可以通过将 参数值设置为 Disabled 来禁用它。如果在某些集群机器中禁用并发多线程，则必须在所有集群机器中禁用它。

重要

如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。如果您禁用并发多线程，请为您的机器使用较大的类型，如 n1-standard-8。

5 9

smtLevel 指定要设置为 control plane 和计算机器的 SMT 级别。支持的值有 1、2、4、8、'off' 和 'on'。默认值为 8。smtLevel 'off' 将 SMT 设置为 off，smtlevel 'on' 将 SMT 设置为集群节点上的默认值 8。



注意

如果没有启用并发多线程(SMT)或超线程，一个 vCPU 相当于一个物理内核。启用后，每个内核为(Thread (s)的总 vCPU 计算为（Thread (s) per socket) " Socket (s)）。smtLevel 控制每个内核的线程。在部署集群节点时，较低 SMT 级别可能需要额外分配的内核。您可以将 install-config.yaml 文件中的 'processors' 参数设置为适当的值，以满足成功部署 OpenShift Container Platform 的要求。

11

机器 CIDR 必须包含计算机器和 control plane 机器的子网。

12

CIDR 必须包含 platform.ibmcloud.controlPlaneSubnets 和 platform.ibmcloud.computeSubnets 中定义的子网。

13

要安装的集群网络插件。默认值 OVNKubernetes 是唯一支持的值。

14

现有资源组的名称。现有的 VPC 和子网应该位于此资源组中。集群部署到此资源组。

15

指定现有 VPC 的名称。

16

指定现有 VPC 子网的名称。子网必须属于您指定的 VPC。为区域中的每个可用区指定一个子网。

17

对于 <local_registry>，请指定 registry 域名，以及您的镜像 registry 用来提供内容的可选端口。例如： registry.example.com 或 registry.example.com:5000。对于 <credentials>，请为您的镜像 registry 指定 base64 编码的用户名和密码。

18

您可选择提供用于访问集群中机器的 sshKey 值。

19

提供用于镜像 registry 的证书文件内容。

20

提供命令输出中的 imageContentSources 部分来 镜像存储库。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 ssh-agent 进程使用的 SSH 密钥。

20.7.7.3. 在安装过程中配置集群范围的代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 install-config.yaml 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 install-config.yaml 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 Proxy 对象的 spec.noProxy 字段中添加站点来绕过代理。

注意

Proxy 对象 status.noProxy 字段使用安装配置中的 networking.machineNetwork[].cidr、networking.clusterNetwork[].cidr 和 networking.serviceNetwork[] 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，Proxy 对象 status.noProxy 字段也会使用实例元数据端点填充(169.254.169.254)。

流程

1. 编辑 install-config.yaml 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 http。

2. 用于创建集群外 HTTPS 连接的代理 URL。

3. 要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 . 以仅匹配子域。例如，.y.com 匹配 x.y.com，但不匹配 y.com。使用 * 绕过所有目的地的代理。

4. 如果提供，安装程序会在 openshift-config 命名空间中生成名为 user-ca-bundle 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 trusted-ca-bundle 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， Proxy 对象的 trustedCA 字段中

也会引用此配置映射。additionalTrustBundle 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

5 可选：决定 Proxy 对象的配置以引用 trustedCA 字段中 user-ca-bundle 配置映射的策略。允许的值是 Proxyonly 和 Always。仅在配置了 http/https 代理时，使用 Proxyonly 引用 user-ca-bundle 配置映射。使用 Always 始终引用 user-ca-bundle 配置映射。默认值为 Proxyonly。

注意

安装程序不支持代理的 readinessEndpoints 字段。

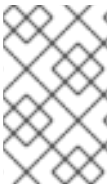
注意

如果安装程序超时，重启并使用安装程序的 wait-for 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. 保存该文件并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 cluster 的集群范围代理，该代理 使用 提供的 install-config.yaml 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 cluster Proxy 对象，但它会有一个空 spec。



注意

只支持名为 cluster 的 Proxy 对象，且无法创建额外的代理。

20.7.8. 手动创建 IAM

安装集群需要 Cloud Credential Operator(CCO)以手动模式运行。虽然安装程序为手动模式配置 CCO，但您必须为云供应商指定身份和访问管理 secret。

您可以使用 Cloud Credential Operator (CCO)实用程序(ccoctl)创建所需的 IBM Cloud® 资源。

先决条件

先决条件

- 您已配置了 ccoctl 二进制文件。
- 您有一个现有的 install-config.yaml 文件。

流程

1. 编辑 install-config.yaml 配置文件，使其包含将 credentialsMode 参数设置为 Manual。

install-config.yaml 配置文件示例

```
apiVersion: v1
baseDomain: cluster1.example.com
credentialsMode: Manual 1
compute:
- architecture: ppc64le
  hyperthreading: Enabled
```

1 添加这一行将 credentialsMode 参数设置为 Manual。

2. 要生成清单，请在包含安装程序的目录中运行以下命令：

```
$ openshift-install create manifests --dir <installation_directory>
```

3. 在包含安装程序的目录中，运行以下命令，使用安装文件中的发行镜像设置 $RELEASE_IMAGE 变量：

```
$ RELEASE_IMAGE=$(./openshift-install version | awk '/release image/ {print $3}')
```

4. 运行以下命令，从 OpenShift Container Platform 发行镜像中提取 CredentialsRequest 自定义资源 (CR) 列表：

```
$ oc adm release extract \
  --from=$RELEASE_IMAGE \
  --credentials-requests \
  --included \ 1
  --install-config=<path_to_directory_with_installation_configuration>/install-
config.yaml \ 2
  --to=<path_to_directory_for_credentials_requests> 3
```

1 --included 参数仅包含特定集群配置所需的清单。

2 指定 install-config.yaml 文件的位置。

3 指定要存储 CredentialsRequest 对象的目录的路径。如果指定的目录不存在，这个命令会创建它。

此命令为每个 CredentialsRequest 对象创建一个 YAML 文件。

CredentialsRequest 对象示例

```
apiVersion: cloudcredential.openshift.io/v1
kind: CredentialsRequest
metadata:
  labels:
    controller-tools.k8s.io: "1.0"
  name: openshift-image-registry-ibmcos
  namespace: openshift-cloud-credential-operator
spec:
  secretRef:
    name: installer-cloud-credentials
    namespace: openshift-image-registry
  providerSpec:
    apiVersion: cloudcredential.openshift.io/v1
    kind: IBMCloudProviderSpec
    policies:
    - attributes:
      - name: serviceName
        value: cloud-object-storage
      roles:
      - crn:v1:bluemix:public:iam::::role:Viewer
      - crn:v1:bluemix:public:iam::::role:Operator
      - crn:v1:bluemix:public:iam::::role:Editor
```

```
    - crn:v1:bluemix:public:iam::::serviceRole:Reader
    - crn:v1:bluemix:public:iam::::serviceRole:Writer
   - attributes:
    - name: resourceType
      value: resource-group
    roles:
    - crn:v1:bluemix:public:iam::::role:Viewer
```

5. 为每个凭证请求创建服务 ID，分配定义的策略、创建 API 密钥并生成 secret：

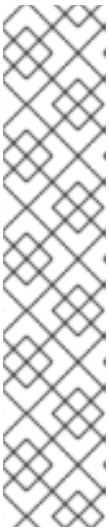
```
$ ccoctl ibmcloud create-service-id \
  --credentials-requests-dir=<path_to_credential_requests_directory> \ 1
  --name=<cluster_name> \ 2
  --output-dir=<installation_directory> \ 3
  --resource-group-name=<resource_group_name> 4
```

1. 指定包含组件 CredentialsRequest 对象文件的目录。

2. 指定 OpenShift Container Platform 集群的名称。

3. 可选：指定您希望 ccoctl 实用程序在其中创建对象的目录。默认情况下，实用程序在运行命令的目录中创建对象。

4. 可选：指定用于限制访问策略的资源组的名称。



注意

如果您的集群使用 TechPreviewNoUpgrade 功能集启用的技术预览功能，则必须包含 --enable-tech-preview 参数。

如果提供了不正确的资源组名称，安装会在 bootstrap 阶段失败。要查找正确的资源组名称，请运行以下命令：

```
$ grep resourceGroup <installation_directory>/manifests/cluster-
infrastructure-02-config.yml
```

验证

- **确保在集群的 manifests 目录中生成了适当的 secret。**

20.7.9. 部署集群

您可以在兼容云平台上安装 OpenShift Container Platform。

重要

在初始安装过程中，您只能运行安装程序的 create cluster 命令一次。

先决条件

- **您已使用托管集群的云平台配置了帐户。**
- **您有 OpenShift Container Platform 安装程序和集群的 pull secret。**
- **已确认主机上的云供应商帐户具有部署集群的正确权限。权限不正确的帐户会导致安装过程失败，并显示包括缺失权限的错误消息。**

流程

- 进入包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

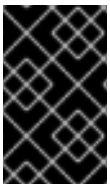
1 对于 <installation_directory>，请指定自定义 ./install-config.yaml 文件的位置。

2 要查看不同的安装详情，请指定 warn、debug 或 error，而不是 info。

验证

当集群部署成功完成时：

- 终端会显示用于访问集群的说明，包括指向 Web 控制台和 kubeadmin 用户的凭证的链接。
- 凭证信息还会输出到 <installation_directory>/.openshift_install.log.



重要

不要删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 node-bootstrapper 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。

- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

20.7.10. 安装 OpenShift CLI

您可以安装 OpenShift CLI(oc)来使用命令行界面与 OpenShift Container Platform 进行交互。您可以在 Linux、Windows 或 macOS 上安装 oc。

重要

如果安装了旧版本的 oc，则可能无法使用 OpenShift Container Platform 4.16 中的所有命令。下载并安装新版本的 oc。

在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI(oc)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 产品变体 下拉列表中选择架构。
3. 从 版本 下拉列表中选择适当的版本。
4. 点 OpenShift v4.16 Linux Client 条目旁的 Download Now 来保存文件。

5. 解包存档：

```
$ tar xvf <file>
```

6. 将 oc 二进制文件放到 PATH 中的目录中。

   要查看您的 PATH，请执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 oc 命令：

```
$ oc <command>
```

在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI(oc)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 版本 下拉列表中选择适当的版本。
3. 点 OpenShift v4.16 Windows Client 条目旁的 Download Now 来保存文件。
4. 使用 ZIP 程序解压存档。
5. 将 oc 二进制文件移到 PATH 中的目录中。

   要查看您的 PATH，请打开命令提示并执行以下命令：

```
C:\> path
```

验证

- 安装 OpenShift CLI 后，可以使用 oc 命令：

```
C:\> oc <command>
```

在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI(oc)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 版本 下拉列表中选择适当的版本。
3. 点 OpenShift v4.16 macOS Client 条目旁的 Download Now 来保存文件。

注意

对于 macOS arm64，请选择 OpenShift v4.16 macOS arm64 Client 条目。

4. 解包和解压存档。
5. 将 oc 二进制文件移到 PATH 的目录中。

要查看您的 PATH，请打开终端并执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 oc 命令：

```
$ oc <command>
```

20.7.11. 使用 CLI 登录集群

您可以通过导出集群 kubeconfig 文件，以默认系统用户身份登录集群。kubeconfig 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。
- 已安装 oc CLI。

流程

1. 导出 kubeadmin 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 <installation_directory>，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 oc 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

其他资源

- 访问Web控制台

20.7.12. 禁用默认的 OperatorHub 目录源

在 OpenShift Container Platform 安装过程中，默认为 OperatorHub 配置由红帽和社区项目提供的源内容的 operator 目录。在受限网络环境中，必须以集群管理员身份禁用默认目录。

流程

- 通过在 OperatorHub 对象中添加 disableAllDefaultSources: true 来 禁用默认目录的源：

```
$ oc patch OperatorHub cluster --type json \
    -p '[{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]'
```

提示

或者，您可以使用 Web 控制台管理目录源。在 Administration → Cluster Settings → Configuration → OperatorHub 页面中，点 Sources 选项卡，您可以在其中创建、更新、删除、禁用和启用单独的源。

20.7.13. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 关于远程健康监控

20.7.14. 后续步骤

- 自定义集群
- 可选：不使用远程健康报告

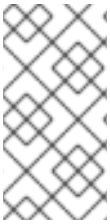
- 可选：注册断开连接的集群

20.8. 在 IBM POWER VIRTUAL SERVER 上卸载集群

您可以删除部署到 IBM Power® Virtual Server 的集群。

20.8.1. 删除使用安装程序置备的基础架构的集群

您可以从云中删除使用安装程序置备的基础架构的集群。



注意

卸载后，检查云供应商是否有未正确删除的资源，特别是在用户置备基础架构(UPI)集群中。可能存在安装程序未创建或安装程序无法访问的资源。

先决条件

- 有用于部署集群的安装程序副本。
- 有创建集群时安装程序生成的文件。
- 您已配置了 ccoctl 二进制文件。
- 已安装 IBM Cloud® CLI 并安装或更新 VPC 基础架构服务插件。如需更多信息，请参阅 IBM Cloud® CLI 文档中的 "先决条件"。

流程

1. 如果满足以下条件，则需要执行此步骤：
   - 安装程序作为安装过程的一部分创建了资源组。
   - 部署集群后，或您的应用程序创建了持久性卷声明(PVC)。

在这种情况下，在卸载集群时不会删除 PVC，这可以防止资源组被成功删除。要防止失败：

a. 使用 CLI 登录 IBM Cloud®。

b. 要列出 PVC，请运行以下命令：

```
$ ibmcloud is volumes --resource-group-name <infrastructure_id>
```

有关列出卷的更多信息，请参阅 IBM Cloud® CLI 文档。

c. 要删除 PVC，请运行以下命令：

```
$ ibmcloud is volume-delete --force <volume_id>
```

有关删除卷的更多信息，请参阅 IBM Cloud® CLI 文档。

2. 导出在安装过程中创建的 API 密钥。

```
$ export IBMCLOUD_API_KEY=<api_key>
```

注意

您必须按照指定方式设置变量名称。安装程序需要存在变量名称来删除安装集群时所创建的服务 ID。

3. 在用来安装集群的计算机中包含安装程序的目录中，运行以下命令：

```
$ ./openshift-install destroy cluster \
--dir <installation_directory> --log-level info 1 2
```

1. 对于 <installation_directory>，请指定安装文件保存到的目录的路径。

2.

注意

- 您必须为集群指定包含集群定义文件的目录。安装程序需要此目录中的 **metadata.json** 文件来删除集群。
- 您可能需要运行 **openshift-install destroy** 命令最多三次，以确保正确清理。

4. 删除为集群创建的手动 **CCO** 凭证：

```
$ ccoctl ibmcloud delete-service-id \
  --credentials-requests-dir <path_to_credential_requests_directory> \
  --name <cluster_name>
```

注意

如果您的集群使用 **TechPreviewNoUpgrade** 功能集启用的技术预览功能，则必须包含 **--enable-tech-preview** 参数。

5. 可选：删除 **<installation_directory>** 目录和 **OpenShift Container Platform** 安装程序。

20.9. IBM POWER VIRTUAL SERVER 的安装配置参数

在 **IBM Power® Virtual Server** 上部署 **OpenShift Container Platform** 前，您可以提供参数来自定义集群以及托管它的平台。在创建 **install-config.yaml** 文件时，您可以通过命令行为所需参数提供值。然后，您可以修改 **install-config.yaml** 文件以进一步自定义集群。

20.9.1. IBM Power Virtual Server 可用的安装配置参数

下表指定可设置为安装过程的一部分所需的、可选和 **IBM Power Virtual Server** 的安装配置参数。

注意

安装后，您无法在 install-config.yaml 文件中修改这些参数。

20.9.1.1. 所需的配置参数

下表描述了所需的安装配置参数：

表 20.6. 所需的参数

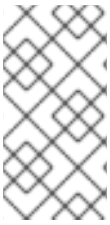
参数	描述	值
apiVersion:	**install-config.yaml** 内容的 API 版本。当前版本为 **v1**。安装程序可能还支持旧的 API 版本。	字符串
baseDomain:	云供应商的基域。基域用于创建到 OpenShift Container Platform 集群组件的路由。集群的完整 DNS 名称是 **baseDomain** 和 **metadata.name** 参数值的组合，其格式为 **<metadata.name>.<baseDomain>**。	完全限定域名或子域名，如 **example.com**。
metadata:	Kubernetes 资源 **ObjectMeta**，其中只消耗 **name** 参数。	对象
metadata: name:	集群的名称。集群的 DNS 记录是 **{{.metadata.name}}.{{.baseDomain}}** 的子域。	小写字母、连字符(**-**)和句点(**.**)字符串，如 **dev**。
platform:	对于特定平台的配置取决于执行安装的环境： **aws**, **baremetal**, **azure**, **gcp**, **ibmcloud**, **nutanix**, **openstack**, **powervs**, **vsphere**, 或 **{}**.有关 **platform.<platform>** 参数的更多信息，请参考下表中您的特定平台。	对象

参数	描述	值
pullSecret:	从 Red Hat OpenShift Cluster Manager 获取 pull secret，验证从 Quay.io 等服务中下载 OpenShift Container Platform 组件的容器镜像。	{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }
platform: powervs: userID:	UserID 是用户的 IBM Cloud® 帐户的登录。	字符串.例如，**existing_user_id**。
platform: powervs: powervsResourceGroup:	PowerVSResourceGroup 是创建 IBM Power® Virtual Server 资源的资源组。如果使用现有的 VPC，现有的 VPC 和子网应位于此资源组中。	字符串.例如： **existing_resource_group**。
platform: powervs: region:	指定创建集群的 IBM Cloud® 共存区域。	字符串.例如，**existing_region**。
platform: powervs: zone:	指定创建集群的 IBM Cloud® 共存区域。	字符串.例如： **existing_zone**。

20.9.1.2. 网络配置参数

您可以根据现有网络基础架构的要求自定义安装配置。例如，您可以扩展集群网络的 IP 地址块，或者提供不同于默认值的不同 IP 地址块。

仅支持 IPv4 地址。



注意

Red Hat OpenShift Data Foundation 灾难恢复解决方案不支持 Globalnet。对于区域灾难恢复场景，请确保为每个集群中的集群和服务网络使用非重叠的专用 IP 地址。

表 20.7. 网络参数

参数	描述	值
networking:	集群网络的配置。	对象 **注意** 您无法在安装后修改 **网络** 对象指定的参数。
networking: networkType:	要安装的 Red Hat OpenShift Networking 网络插件。	默认值为 **OVNKubernetes**。
networking: clusterNetwork:	pod 的 IP 地址块。 默认值为 **10.128.0.0/14**，主机前缀为 **/23**。 如果您指定了多个 IP 地址块，块不得重叠。	对象数组。例如： networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23
networking: clusterNetwork: cidr:	使用 **networking.clusterNetwork** 时需要此项。IP 地址块。 IPv4 网络。	无类别域间路由(CIDR)表示法中的 IP 地址块。IPv4 块的前缀长度介于 **0 到 32** 之间 **。**
networking: clusterNetwork: hostPrefix:	分配给每个节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从 given **cidr** 中分配 a/**23** 子网。**hostPrefix** 值 **23** 提供 510（2^(32 - 23)- 2）pod IP 地址。	子网前缀。 默认值为 **23**。
networking: serviceNetwork:	服务的 IP 地址块。默认值为 **172.30.0.0/16**。 OVN-Kubernetes 网络插件只支持服务网络的一个 IP 地址块。	CIDR 格式具有 IP 地址块的数组。例如： networking: serviceNetwork: - 172.30.0.0/16

参数	描述	值
networking: machineNetwork:	机器的 IP 地址块。	对象数组。例如： networking: machineNetwork: - cidr: 10.0.0.0/16
networking: machineNetwork: cidr:	使用 **networking.machineNetwork** 时需要此项。IP 地址块。对于 libvirt 和 IBM Power® Virtual Server 以外的所有平台，默认值为 **10.0.0.0/16**。对于 libvirt，默认值 为 **192.168.126.0/24**。对于 IBM Power® Virtual Server，默认值为 **192.168.0.0/24**。	CIDR 表示法中的 IP 网络块。 例如，**192.168.0.0/24**。 **注意** 将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

20.9.1.3. 可选的配置参数

下表描述了可选的安装配置参数：

表 20.8. 可选参数

参数	描述	值
additionalTrustBundle:	添加到节点可信证书存储中的 PEM 编码 X.509 证书捆绑包。配置了代理时，也可以使用此信任捆绑包。	字符串
capabilities:	控制可选核心组件的安装。您可以通过禁用可选组件来减少 OpenShift Container Platform 集群的空间。如需更多信息，请参阅 *安装*中的"集群功能"页面。	字符串数组
capabilities: baselineCapabilitySet:	选择要启用的一组初始可选功能。有效值为 **None**、**v4.11**、**v4.12** 和 **vCurrent**。默认值为 **vCurrent**。	字符串

参数	描述	值
capabilities: additionalEnabledCapabilities:	将可选功能集合扩展到您在 **baselineCapabilitySet** 中指定的范围。您可以在此参数中指定多个功能。	字符串数组
cpuPartitioningMode:	启用工作负载分区，它会隔离 OpenShift Container Platform 服务、集群管理工作负载和基础架构 pod，以便在保留的一组 CPU 上运行。工作负载分区只能在安装过程中启用，且在安装后无法禁用。虽然此字段启用工作负载分区，但它不会将工作负载配置为使用特定的 CPU。如需更多信息，请参阅 *Scalability and Performance* 部分中的 *Workload partitioning* 页面。	**None** 或 **AllNodes**.**None** 是默认值。
compute:	组成计算节点的机器的配置。	**MachinePool** 对象的数组。
compute: architecture:	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **ppc64le** （默认值）。	字符串
compute: hyperthreading:	是否在计算机器上启用或禁用并发多 **线程或超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。	**enabled** 或 **Disabled**
compute: smtLevel:	SMTLevel 指定设置为 control plane 和计算机器的 SMT 级别。有效值为 1、2、3、4、5、6、8、**off** 和 **。**	字符串
compute: name:	使用 **compute** 时需要此项。机器池的名称。	**worker**

参数	描述	值
compute: platform:	使用 **compute** 时需要此项。使用此参数指定托管 worker 机器的云供应商。此参数值必须与 **controlPlane.platform** 参数值匹配。示例用法，**compute.platform.powervs.sysType**.	**aws**, **azure**, **gcp**, **ibmcloud**, **nutanix**, **openstack**, **powervs**, **vsphere**, 或 **{}**
compute: replicas:	要置备的计算机器数量，也称为 worker 机器。	大于或等于 **2** 的正整数。默认值为 **3**。
featureSet:	为功能集启用集群。功能集是 OpenShift Container Platform 功能的集合，默认情况下不启用。有关在安装过程中启用功能集的更多信息，请参阅"使用功能门启用功能"。	字符串.要启用的功能集的名称，如 **TechPreviewNoUpgrade**。
controlPlane:	组成 control plane 的机器的配置。	**MachinePool** 对象的数组。
controlPlane: architecture:	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **ppc64le** （默认值）。	字符串
controlPlane: hyperthreading:	是否在 control plane 机器上启用或禁用并发多 **线程或超线程**。默认情况下，启用并发多线程以提高机器内核的性能。 **重要** 如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。	**enabled** 或 **Disabled**
controlPlane: name:	使用 **controlPlane** 时需要此项。机器池的名称。	**master**
controlPlane: platform:	使用 **controlPlane** 时需要此项。使用此参数指定托管 control plane 机器的云供应商。此参数值必须与 **compute.platform** 参数值匹配。示例用法，**controlPlane.platform.powervs.processors**.	**aws**, **azure**, **gcp**, **ibmcloud**, **nutanix**, **openstack**, **powervs**, **vsphere**, 或 **{}**

参数	描述	值
controlPlane: replicas:	要置备的 control plane 机器数量。	部署单节点 OpenShift 时支持的值为 **3** 或 **1**。
credentialsMode:	Cloud Credential Operator(CCO)模式。如果没有指定模式，CCO 会动态尝试决定提供的凭证的功能，在支持多个模式的平台上首选 mint 模式。	**Mint**、**Passthrough**、**Manual** 或空字符串(**""**)。[1]
imageContentSources:	release-image 内容的源和存储库。	对象数组。包括一个 **source** 以及可选的 **mirrors**，如本表的以下行所述。
imageContentSources: source:	使用 **imageContentSources** 时需要此项。指定用户在镜像拉取规格中引用的存储库。	字符串
imageContentSources: mirrors:	指定可能还包含同一镜像的一个或多个仓库。	字符串数组
publish:	如何发布或公开集群的面向用户的端点，如 Kubernetes API、OpenShift 路由。	**内部或外部 .**默认值为 **External**。 非云平台不支持将此字段设置为 **Internal**。
sshKey:	用于验证对集群机器的访问的 SSH 密钥。 **注意** 对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。	例如，**sshKey: ssh-ed25519 AAAA..**。

参数	描述	值
platform: powervs: vpcRegion:	指定在其中创建 VPC 资源的 IBM Cloud® 区域。	字符串.例如：**existing_vpc_region**。
platform: powervs: vpcSubnets:	指定创建集群资源的现有子网（按名称）。	字符串.例如，**powervs_region_example_subnet**。
platform: powervs: vpcName:	指定 IBM Cloud® 名称。	字符串.例如： **existing_vpcName**。
platform: powervs: serviceInstanceGUID:	ServiceInstanceGUID 是从 IBM Cloud® Catalog 创建的 Power IAAS 实例的 ID。	字符串.例如，**existing_service_instance_GUID**。
platform: powervs: clusterOSImage:	ClusterOSImage 是一个预先创建的 IBM Power® Virtual Server 引导镜像，用于覆盖集群节点的默认镜像。	字符串.例如：**existing_cluster_os_image**。
platform: powervs: defaultMachinePlatform:	DefaultMachinePlatform 是在 IBM Power® Virtual Server 上安装用于没有定义自身平台配置的机器池时使用的默认配置。	字符串.例如，**existing_machine_platform**。
platform: powervs: memoryGiB:	虚拟机内存大小（以 GB 为单位）。	有效的整数必须是整数，长度至少为 2，没有超过 64 的整数，具体取决于机器类型。

参数	描述	值
platform: powervs: procType:	ProcType 定义实例的处理器共享模型。	有效值为 Capped、Dedicated 和 Shared。
platform: powervs: processors:	Processors 定义实例的处理单元。	处理器数量必须从 .5 到 32 个内核。处理器必须递增 .25。
platform: powervs: sysType:	SysType 为实例定义系统类型。	系统类型必须是 **e980** 或 **s922**。

1. **不是所有 CCO 模式都支持所有云供应商。有关 CCO 模式的更多信息，请参阅*身份验证和授权*内容中的"管理云供应商凭证"条目。**

第 21 章 在 OPENSTACK 上安装

21.1. 准备在 OPENSTACK 上安装

您可以在 Red Hat OpenStack Platform (RHOSP) 上安装 OpenShift Container Platform。

21.1.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读有关 选择集群安装方法的文档，并为用户准备它。

21.1.2. 选择在 OpenStack 上安装 OpenShift Container Platform 的方法

您可以在安装程序置备或用户置备的基础架构上安装 OpenShift Container Platform。默认安装类型使用安装程序置备的基础架构，安装程序会在其中为集群置备底层基础架构。您还可以在您置备的基础架构上安装 OpenShift Container Platform。如果不使用安装程序置备的基础架构，您必须自己管理和维护集群资源。

如需有关安装程序置备和用户置备的安装过程的更多信息，请参阅 安装过程。

21.1.2.1. 在安装程序置备的基础架构上安装集群

您可以使用以下方法之一在 OpenShift Container Platform 安装程序置备的 Red Hat OpenStack Platform(RHOSP)基础架构上安装集群：

- 使用自定义在 OpenStack 上安装集群 ：您可以在 RHOSP 上安装自定义集群。安装程序允许在安装阶段应用一些自定义。其它自定义选项可在安装后使用。
- 在受限网络中的 OpenStack 上安装集群： 您可以通过创建安装发行内容的内部镜像在受限或断开连接的网络中在 RHOSP 上安装 OpenShift Container Platform。您可以使用此方法安装不需要活跃互联网连接的集群来获取软件组件。您还可以使用此安装方法来确保集群只使用满足您组织对外部内容控制的容器镜像。

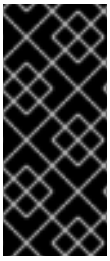
21.1.2.2. 在用户置备的基础架构上安装集群

您可以使用以下方法之一在您置备的 RHOSP 基础架构上安装集群：

- 在您自己的基础架构的 OpenStack 上安装集群 ：您可以在用户置备的 RHOSP 基础架构上安装 OpenShift Container Platform。通过使用这个安装方法，您可以将集群与现有的基础架构和修改进行集成。对于在用户置备的基础架构上安装，您必须创建所有 RHOSP 资源，如 Nova 服务器、Neutron 端口和安全组。您可以使用提供的 Ansible playbook 来协助部署过程。

21.1.3. 为旧 HTTPS 证书扫描 RHOSP 端点

从 OpenShift Container Platform 4.10 开始，HTTPS 证书必须包含主题替代名称(SAN)字段。运行以下脚本，为仅包含 CommonName 字段的传统证书在 Red Hat OpenStack Platform(RHOSP)目录中扫描每个 HTTPS 端点。



重要

OpenShift Container Platform 在安装或升级前不会检查底层 RHOSP 基础架构是否有旧证书。使用提供的脚本来自行检查这些证书。在安装或升级集群前无法更新旧证书将导致集群无法正常工作。

先决条件

- 在运行脚本的机器中，有以下软件：
  - Bash 版本 4.0 或更高版本
  - grep
  - OpenStack 客户端
  - jq
  - openssl 版本 1.1.1l 或更高版本
- 使用目标云的 RHOSP 凭证填充机器。

流程

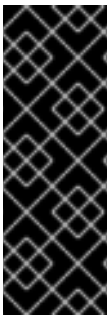
1. 将以下脚本保存到机器中：

```
#!/usr/bin/env bash

set -Eeuo pipefail

declare catalog san
catalog="$(mktemp)"
san="$(mktemp)"
readonly catalog san

declare invalid=0

openstack catalog list --format json --column Name --column Endpoints \
 | jq -r '.[] | .Name as $name | .Endpoints[] | select(.interface=="public") | [$name,
.interface, .url] | join(" ")' \
 | sort \
 > "$catalog"

while read -r name interface url; do
 # Ignore HTTP
 if [[ ${url#"http://"} != "$url" ]]; then
  continue
 fi

 # Remove the schema from the URL
 noschema=${url#"https://"}

 # If the schema was not HTTPS, error
 if [[ "$noschema" == "$url" ]]; then
  echo "ERROR (unknown schema): $name $interface $url"
  exit 2
 fi

 # Remove the path and only keep host and port
 noschema="${noschema%%/*}"
 host="${noschema%%:*}"
 port="${noschema##*:}"

 # Add the port if was implicit
 if [[ "$port" == "$host" ]]; then
  port='443'
 fi

 # Get the SAN fields
 openssl s_client -showcerts -servername "$host" -connect "$host:$port" </dev/null
2>/dev/null \
  | openssl x509 -noout -ext subjectAltName \
  > "$san"

 # openssl returns the empty string if no SAN is found.
 # If a SAN is found, openssl is expected to return something like:
```

```
#
#    X509v3 Subject Alternative Name:
#        DNS:standalone, DNS:osp1, IP Address:192.168.2.1, IP Address:10.254.1.2
if [[ "$(grep -c "Subject Alternative Name" "$san" || true)" -gt 0 ]]; then
 echo "PASS: $name $interface $url"
else
 invalid=$((invalid+1))
 echo "INVALID: $name $interface $url"
fi
done < "$catalog"

# clean up temporary files
rm "$catalog" "$san"

if [[ $invalid -gt 0 ]]; then
 echo "${invalid} legacy certificates were detected. Update your certificates to include
a SAN field."
 exit 1
else
 echo "All HTTPS certificates for this cloud are valid."
fi
```

2. 运行脚本。

3. 将脚本报告为 INVALID 的任何证书替换为包含 SAN 字段的证书。



重要

在安装 OpenShift Container Platform 4.10 之前，您必须替换所有旧的 HTTPS 证书，或将集群更新至该版本。旧证书将被拒绝，并显示以下信息：

```
x509: certificate relies on legacy Common Name field, use SANs instead
```

21.1.3.1. 手动为旧 HTTPS 证书扫描 RHOSP 端点

从 OpenShift Container Platform 4.10 开始，HTTPS 证书必须包含主题替代名称(SAN)字段。如果您无法访问 "Scanning RHOSP 端点用于旧 HTTPS 证书"中列出的预备工具，请执行以下步骤为仅包含 CommonName 字段的传统证书扫描 Red Hat OpenStack Platform (RHOSP)目录中的每个 HTTPS 端点。

重要

OpenShift Container Platform 在安装或升级前不会检查底层 RHOSP 基础架构是否有旧证书。使用以下步骤自己检查这些证书。在安装或升级集群前无法更新旧证书将导致集群无法正常工作。

流程

1. **在命令行中运行以下命令查看 RHOSP 公共端点的 URL：**

```
$ openstack catalog list
```

记录命令返回的每个 HTTPS 端点的 URL。

2. **对于每个公共端点，请注意主机和端口。**

提示

通过删除方案、端口和路径来确定端点的主机。

3. **对于每个端点，运行以下命令来提取证书的 SAN 字段：**

a. **设置 host 变量：**

```
$ host=<host_name>
```

b. **设置 port 变量：**

```
$ port=<port_number>
```

如果端点的 URL 中没有包括端口，使用值 443。

c. **检索证书的 SAN 字段：**

```
$ openssl s_client -showcerts -servername "$host" -connect "$host:$port"
</dev/null 2>/dev/null \
   | openssl x509 -noout -ext subjectAltName
```

输出示例

```
X509v3 Subject Alternative Name:
   DNS:your.host.example.net
```

对于每个端点，查找类似于上例的输出。如果没有端点的输出，则该端点的证书无效，必须重新发布。

重要

在安装 OpenShift Container Platform 4.10 之前，您必须替换所有旧的 HTTPS 证书，或将集群更新至该版本。旧证书被拒绝，并显示以下信息：

```
x509: certificate relies on legacy Common Name field, use SANs instead
```

21.2. 准备在 OPENSTACK 上安装使用 SR-IOV 或 OVS-DPDK 的集群

在 Red Hat OpenStack Platform (RHOSP) 上安装使用单根 I/O 虚拟化 (SR-IOV) 或 Open vSwitch 的 OpenShift Container Platform 集群前，您必须了解每个技术的要求，然后执行准备任务。

21.2.1. 使用 SR-IOV 或 OVS-DPDK 的 RHOSP 上的集群要求

如果您的部署使用了 SR-IOV 或 OVS-DPDK，您必须满足以下条件：

- RHOSP 计算节点必须使用支持巨页的类别。

21.2.1.1. 使用 SR-IOV 的 RHOSP 上的集群的要求

要使用带有部署的单根 I/O 虚拟化 (SR-IOV)，您必须满足以下要求：

- 规划 Red Hat OpenStack Platform(RHOSP) SR-IOV 部署。

- OpenShift Container Platform 必须支持您使用的 NIC。有关支持的 NIC 列表，请参阅"网络"文档中的"单一根 I/O 虚拟化 (SR-IOV) 硬件网络"子网络"部分。

- 对于具有附加 SR-IOV NIC 的每个节点，您的 RHOSP 集群必须具有：

  - 来自 RHOSP 配额的一个实例

  - 附加到机器子网的端口

  - 每个 SR-IOV 虚拟功能有一个端口

  - 类别至少有 16 GB 内存、4 个 vCPU 和 25 GB 存储空间

- SR-IOV 部署通常采用性能优化，如专用或隔离的 CPU。为获得最佳性能，请将您的底层 RHOSP 部署配置为使用这些优化，然后在优化的基础架构上运行 OpenShift Container Platform 计算机器。

  - 有关配置高性能 RHOSP 计算节点的更多信息，请参阅 为性能配置计算节点。

21.2.1.2. 使用 OVS-DPDK 的 RHOSP 上的集群的要求

要将 Open vSwitch 与具有部署的 Data Plane Development Kit (OVS-DPDK) 搭配使用，您必须满足以下要求：

- 请参阅在网络功能虚拟化规划和配置指南中的规划您的 OVS-DPDK 部署部分，以规划您的 Red Hat OpenStack Platform (RHOSP) OVS-DPDK 部署。

- 根据在网络功能虚拟化规划和配置指南中的配置 OVS-DPDK 部署部分来配置 RHOSP OVS-DPDK 部署。

21.2.2. 准备安装使用 SR-IOV 的集群

在安装使用 SR-IOV 的集群前，您必须配置 RHOSP。

使用 SR-IOV 安装集群时，必须使用 cgroup v1 部署集群。如需更多信息，启用 Linux 控制组群版本 1 (cgroup v1)。

重要

cgroup v1 是一个已弃用的功能。弃用的功能仍然包含在 OpenShift Container Platform 中，并将继续被支持。但是，这个功能会在以后的发行版本中被删除，且不建议在新的部署中使用。

有关 OpenShift Container Platform 中已弃用或删除的主要功能的最新列表，请参阅 OpenShift Container Platform 发行注记中*已弃用和删除的功能*部分。

21.2.2.1. 为计算机器创建 SR-IOV 网络

如果您的 Red Hat OpenStack Platform(RHOSP)部署支持 单根 I/O 虚拟化(SR-IOV)，您可以置备计算机器在其上运行的 SR-IOV 网络。

注意

以下说明将创建外部扁平网络和一个可附加到计算机器的外部基于 VLAN 的网络。根据您的 RHOSP 部署，可能需要其他网络类型。

先决条件

- 集群支持 SR-IOV。

  注意

  如果您不确定集群支持什么，请参阅 OpenShift Container Platform SR-IOV 硬件网络文档。

- 作为 RHOSP 部署的一部分，您创建了 Radio 和 uplink 提供商网络。所有示例命令都使用名称 Radio 和 uplink 来代表这些网络。

流程

1. 在命令行中创建一个单体 RHOSP 网络：

```
$ openstack network create radio --provider-physical-network radio --provider-network-type flat --external
```

2. 创建 uplink RHOSP 网络：

```
$ openstack network create uplink --provider-physical-network uplink --provider-network-type vlan --external
```

3. 为无线网络创建子网：

```
$ openstack subnet create --network radio --subnet-range <radio_network_subnet_range> radio
```

4. 为 uplink 网络创建一个子网：

```
$ openstack subnet create --network uplink --subnet-range <uplink_network_subnet_range> uplink
```

21.2.3. 准备安装使用 OVS-DPDK 的集群

在安装使用 SR-IOV 的集群前，您必须配置 RHOSP。

- 在 RHOSP 上安装集群前，为 OVS-DPDK 创建类别和部署实例。

执行预安装任务后，请按照在 RHOSP 安装说明中相关的 OpenShift Container Platform 安装集群。然后，在此页面中执行"下一步"下的任务。

21.2.4. 后续步骤

- 对于任何类型的部署：

- 使用支持巨页的 Node Tuning Operator 配置。

- 部署集群后完成 SR-IOV 配置：
  - 安装 SR-IOV Operator。
  - 配置 SR-IOV 网络设备。
  - 创建 SR-IOV 计算机器。

- 在部署集群后，参考以下内容以改进其性能：
  - 在 OpenStack 上使用 OVS-DPDK 的集群测试 pod 模板。
  - 在 OpenStack 上使用 SR-IOV 的集群测试 pod 模板。
  - 在 OpenStack 上使用 OVS-DPDK 的集群的性能配置集模板。

21.3. 使用自定义在 OPENSTACK 上安装集群

在 OpenShift Container Platform 版本 4.16 中，您可以在 Red Hat OpenStack Platform (RHOSP) 上安装自定义集群。要自定义安装，请在安装集群前修改 install-config.yaml 中的参数。

21.3.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读有关 选择集群安装方法的文档，并为用户准备它。
- 使用OpenShift 集群支持的平台部分验证 OpenShift Container Platform 4.16 是否与您的 RHOSP 版本兼

容。https://docs.redhat.com/en/documentation/openshift_container_platform/4.16/html-single/architecture/#supported-platforms-for-openshift-clusters_architecture-installation您还可以通过查看 RHOSP 上的 OpenShift Container Platform 支持来比较不同版本的平台支持。

- 您已在 RHOSP 中安装了存储服务，如块存储(Cinder)或对象存储(Swift)。对象存储是 OpenShift Container Platform registry 集群部署的推荐存储技术。如需更多信息，请参阅优化存储。

- 您可以了解集群扩展、control plane 大小和 etcd 的性能和可扩展性实践。如需更多信息，请参阅 扩展集群的建议实践。

- 您已在 RHOSP 中启用了元数据服务。

21.3.2. 在 RHOSP 上安装 OpenShift Container Platform 的资源指南

为支持 OpenShift Container Platform 安装，您的 Red Hat OpenStack Platform(RHOSP)配额必须满足以下要求：

表 21.1. RHOSP 上默认 OpenShift Container Platform 集群的建议资源

resource	value
浮动 IP 地址	3
端口	15
路由器	1
子网	1
RAM	88 GB
VCPU	22
卷存储	275 GB
实例	7
安全组	3
安全组规则	60

resource	value
服务器组	2 - 每个机器池中每个额外可用区加 1

集群或许能以少于推荐资源运行，但其性能无法保证。

重要

如果 RHOSP 对象存储(Swift)可用，并由具有 swiftoperator 角色的用户帐户执行，它将用作 OpenShift Container Platform 镜像 registry 的默认后端。在这种情况下，卷存储需要 175 GB。根据镜像 registry 的大小，Swift 空间要求会有所不同。

注意

默认情况下，您的安全组和安全组规则配额可能较低。如果遇到问题，请以管理员 身份运行 openstack quota set --secgroups 3 --secgroup-rules 60 <project> 来提高配额。

OpenShift Container Platform 部署包含 control plane 机器、计算机器和 bootstrap 机器。

21.3.2.1. control plane 机器

默认情况下，OpenShift Container Platform 安装过程会创建三台 control plane 机器。

每台机器都需要：

- 来自 RHOSP 配额的实例
- 来自 RHOSP 配额的端口
- 至少有 16 GB 内存和 4 个 vCPU 的类别

- RHOSP 配额中至少有 100 GB 存储空间

21.3.2.2. 计算机器

默认情况下，OpenShift Container Platform 安装过程会创建三台计算机器。

每台机器都需要：

- 来自 RHOSP 配额的实例
- 来自 RHOSP 配额的端口
- 至少有 8 GB 内存和 2 个 vCPU 的类别
- RHOSP 配额中至少有 100 GB 存储空间

提示

计算机器托管您在 OpenShift Container Platform 上运行的应用程序；运行数量应尽可能多。

21.3.2.3. bootstrap 机器

在安装过程中，会临时置备 bootstrap 机器来支持 control plane。生产 control plane 就绪后，bootstrap 机器会被取消置备。

bootstrap 机器需要：

- 来自 RHOSP 配额的实例
- 来自 RHOSP 配额的端口

- 至少有 16 GB 内存和 4 个 vCPU 的类别
- RHOSP 配额中至少有 100 GB 存储空间

21.3.2.4. 用户置备的基础架构的负载均衡要求

在安装 OpenShift Container Platform 前，您可以置备自己的 API 和应用程序入口负载平衡基础架构，以代替默认的内部负载平衡解决方案。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。

注意

如果要使用 Red Hat Enterprise Linux (RHEL) 实例部署 API 和应用程序入口负载均衡器，您必须单独购买 RHEL 订阅。

负载平衡基础架构必须满足以下要求：

1. API 负载均衡器 ：提供一个通用端点，供用户（包括人工和机器）与平台交互和配置。配置以下条件：
   - 仅第 4 层负载均衡.这可被称为 Raw TCP 或 SSL Passthrough 模式。
   - 无状态负载平衡算法。这些选项根据负载均衡器的实施而有所不同。

重要

不要为 API 负载均衡器配置会话持久性。为 Kubernetes API 服务器配置会话持久性可能会导致出现过量 OpenShift Container Platform 集群应用程序流量，以及过量的在集群中运行的 Kubernetes API。

在负载均衡器的前端和后端配置以下端口：

表 21.2. API 负载均衡器

port	后端机器（池成员）	internal	外部	描述
6443	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。您必须为 API 服务器健康检查探测配置 **/readyz** 端点。	X	X	Kubernetes API 服务器
22623	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。	X		机器配置服务器

注意

负载均衡器必须配置为，从 API 服务器关闭 /readyz 端点到从池中移除 API 服务器实例时最多需要 30 秒。在 /readyz 返回错误或健康后的时间范围内，端点必须被删除或添加。每 5 秒或 10 秒探测一次，有两个成功请求处于健康状态，三个成为不健康的请求是经过良好测试的值。

2. **应用程序入口负载均衡器 ：为应用程序流量从集群外部流提供入口点。OpenShift Container Platform 集群需要正确配置入口路由器。**

配置以下条件：

- **仅第 4 层负载均衡.这可被称为 Raw TCP 或 SSL Passthrough 模式。**
- **建议根据可用选项以及平台上托管的应用程序类型，使用基于连接的或基于会话的持久性。**

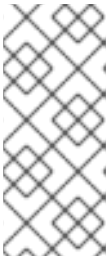
提示

如果应用程序入口负载均衡器可以看到客户端的真实 IP 地址，启用基于 IP 的会话持久性可以提高使用端到端 TLS 加密的应用程序的性能。

在负载均衡器的前端和后端配置以下端口：

表 21.3. 应用程序入口负载均衡器

port	后端机器（池成员）	internal	外部	描述
443	默认情况下，运行 Ingress Controller Pod、计算或 worker 的机器。	X	X	HTTPS 流量
80	默认情况下，运行 Ingress Controller Pod、计算或 worker 的机器。	X	X	HTTP 流量



注意

如果要部署一个带有零计算节点的三节点集群，Ingress Controller Pod 在 control plane 节点上运行。在三节点集群部署中，您必须配置应用程序入口负载均衡器，将 HTTP 和 HTTPS 流量路由到 control plane 节点。

21.3.2.4.1. 使用用户管理的负载均衡器部署的集群的负载均衡器配置示例

本节提供了一个满足用户管理的负载均衡器部署的集群的负载均衡要求的 API 和应用程序入口负载均衡器配置示例。示例是 HAProxy 负载均衡器的 /etc/haproxy/haproxy.cfg 配置。这个示例不是为选择一个负载平衡解决方案提供建议。

在这个示例中，将相同的负载均衡器用于 Kubernetes API 和应用入口流量。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。

注意

如果您使用 HAProxy 作为负载均衡器，并且 SELinux 设置为 enforcing，您必须通过运行 setsebool -P haproxy_connect_any=1 来确保 HAProxy 服务可以绑定到配置的 TCP 端口。

例 21.1. API 和应用程序入口负载均衡器配置示例

```
global
  log         127.0.0.1 local2
  pidfile     /var/run/haproxy.pid
  maxconn     4000
  daemon
defaults
  mode                    http
  log                     global
  option                  dontlognull
  option http-server-close
  option                  redispatch
  retries                 3
  timeout http-request    10s
```

```
  timeout queue           1m
  timeout connect         10s
  timeout client          1m
  timeout server          1m
  timeout http-keep-alive 10s
  timeout check           10s
  maxconn                 3000
listen api-server-6443 (1)
  bind *:6443
  mode tcp
  option  httpchk GET /readyz HTTP/1.0
  option  log-health-checks
  balance roundrobin
  server bootstrap bootstrap.ocp4.example.com:6443 verify none check check-ssl inter 10s
fall 2 rise 3 backup (2)
  server master0 master0.ocp4.example.com:6443 weight 1 verify none check check-ssl
inter 10s fall 2 rise 3
  server master1 master1.ocp4.example.com:6443 weight 1 verify none check check-ssl
inter 10s fall 2 rise 3
  server master2 master2.ocp4.example.com:6443 weight 1 verify none check check-ssl
inter 10s fall 2 rise 3
listen machine-config-server-22623 (3)
  bind *:22623
  mode tcp
  server bootstrap bootstrap.ocp4.example.com:22623 check inter 1s backup (4)
  server master0 master0.ocp4.example.com:22623 check inter 1s
  server master1 master1.ocp4.example.com:22623 check inter 1s
  server master2 master2.ocp4.example.com:22623 check inter 1s
listen ingress-router-443 (5)
  bind *:443
  mode tcp
  balance source
  server compute0 compute0.ocp4.example.com:443 check inter 1s
  server compute1 compute1.ocp4.example.com:443 check inter 1s
listen ingress-router-80 (6)
  bind *:80
  mode tcp
  balance source
  server compute0 compute0.ocp4.example.com:80 check inter 1s
  server compute1 compute1.ocp4.example.com:80 check inter 1s
```

(1)

端口 6443 处理 Kubernetes API 流量并指向 control plane 机器。

(2) (4)

bootstrap 条目必须在 OpenShift Container Platform 集群安装前就位，且必须在 bootstrap 过程完成后删除它们。

(3)

端口 22623 处理机器配置服务器流量并指向 control plane 机器。

5

端口 443 处理 HTTPS 流量，并指向运行 Ingress Controller pod 的机器。默认情况下，Ingress Controller Pod 在计算机器上运行。

6

端口 80 处理 HTTP 流量，并指向运行 Ingress Controller pod 的机器。默认情况下，Ingress Controller Pod 在计算机器上运行。

注意

如果要部署一个带有零计算节点的三节点集群，Ingress Controller Pod 在 control plane 节点上运行。在三节点集群部署中，您必须配置应用程序入口负载均衡器，将 HTTP 和 HTTPS 流量路由到 control plane 节点。

提示

如果您使用 HAProxy 作为负载均衡器，您可以通过在 HAProxy 节点上运行 netstat -nltupe 来检查 haproxy 进程是否在侦听端口 6443、22623、443 和 80。

21.3.3. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

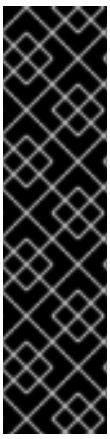
- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类型的基础架构上执行受限网络安装。在此过程中，您可以下载所需的内容，并使用它为镜像 registry 填充安装软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群前，您要更新镜像 registry 的内容。

21.3.4. 在 RHOSP 上启用 Swift

Swift 由具有 swiftoperator 角色的用户帐户操控。在运行安装程序前，将该角色添加到帐户。



重要

如果 Red Hat OpenStack Platform(RHOSP)对象存储服务（通常称为 Swift）可用，OpenShift Container Platform 会使用它作为镜像 registry 存储。如果无法使用，安装程序会依赖于 RHOSP 块存储服务，通常称为 Cinder。

如果 Swift 存在并且您想要使用 Swift，您必须启用对其的访问。如果不存在，或者您不想使用它，请跳过这个部分。

重要

RHOSP 17 将 Ceph RGW 的 rgw_max_attr_size 参数设置为 256 个字符。此设置会导致将容器镜像上传到 OpenShift Container Platform registry 的问题。您必须将 rgw_max_attr_size 的值设置为至少 1024 个字符。

在安装前，检查您的 RHOSP 部署是否会受到此问题的影响。如果是，请重新配置 Ceph RGW。

先决条件

- 在目标环境中有一个 RHOSP 管理员帐户。
- 已安装 Swift 服务。
- 在 Ceph RGW 上，启用了 account in url 选项。

流程

在 RHOSP 上启用 Swift：

1. 在 RHOSP CLI 中以管理员身份，将 swiftoperator 角色添加到将要访问 Swift 的帐户中：

```
$ openstack role add --user <user> --project <project> swiftoperator
```

您的 RHOSP 部署现在可以将 Swift 用于镜像 registry。

21.3.5. 在 RHOSP 上运行的集群中使用自定义存储配置镜像 registry

在 Red Hat OpenStack Platform（RHOSP）上安装集群后，您可以使用位于 registry 存储的特定可用区的 Cinder 卷。

流程

1. 创建一个 YAML 文件，用于指定要使用的存储类和可用性区域。例如：

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: custom-csi-storageclass
provisioner: cinder.csi.openstack.org
volumeBindingMode: WaitForFirstConsumer
allowVolumeExpansion: true
parameters:
  availability: <availability_zone_name>
```

注意

OpenShift Container Platform 不验证您选择的可用区是否存在。应用配置前，请验证可用性区域的名称。

2. 在命令行中应用配置：

```
$ oc apply -f <storage_class_file_name>
```

输出示例

```
storageclass.storage.k8s.io/custom-csi-storageclass created
```

3. 创建一个 YAML 文件，用于指定使用存储类和 openshift-image-registry 命名空间的持久性卷声明（PVC）。例如：

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: csi-pvc-imageregistry
  namespace: openshift-image-registry (1)
  annotations:
    imageregistry.openshift.io: "true"
spec:
  accessModes:
  - ReadWriteOnce
  volumeMode: Filesystem
  resources:
    requests:
      storage: 100Gi (2)
  storageClassName: <your_custom_storage_class> (3)
```

(1) 输入命名空间 openshift-image-registry。此命名空间允许 Cluster Image Registry Operator 使用 PVC。

(2) 可选：调整卷大小。

(3) 输入您创建的存储类的名称。

4. 在命令行中应用配置：

```
$ oc apply -f <pvc_file_name>
```

输出示例

```
persistentvolumeclaim/csi-pvc-imageregistry created
```

5. 将镜像 registry 配置中的原始持久性卷声明替换为新声明：

```
$ oc patch configs.imageregistry.operator.openshift.io/cluster --type 'json' -p='[{"op":
"replace", "path": "/spec/storage/pvc/claim", "value": "csi-pvc-imageregistry"}]'
```

输出示例

```
config.imageregistry.operator.openshift.io/cluster patched
```

接下来的几分钟内，配置将更新。

验证

确认 registry 正在使用您定义的资源：

1. 验证 PVC 声明值是否与您在 PVC 定义中提供的名称相同：

```
$ oc get configs.imageregistry.operator.openshift.io/cluster -o yaml
```

输出示例

```
...
status:
  ...
  managementState: Managed
  pvc:
    claim: csi-pvc-imageregistry
...
```

2. 验证 PVC 的状态是否为 Bound：

```
$ oc get pvc -n openshift-image-registry csi-pvc-imageregistry
```

输出示例

```
NAME                     STATUS   VOLUME                                     CAPACITY   ACCESS
MODES   STORAGECLASS          AGE
csi-pvc-imageregistry  Bound    pvc-72a8f9c9-f462-11e8-b6b6-fa163e18b7b5   100Gi
RWO           custom-csi-storageclass  11m
```

21.3.6. 验证外部网络访问

OpenShift Container Platform 安装过程需要外部网络访问权限。您必须为其提供外部网络值，否则部署会失败。在开始这个过程前，请验证 Red Hat OpenStack Platform(RHOSP)中是否存在具有外部路由器类型的网络。

先决条件

- 配置 OpenStack 的网络服务，使其让 DHCP 代理转发实例 DNS 查询

流程

1. 使用 RHOSP CLI 验证"外部"网络的名称和 ID：

```
$ openstack network list --long -c ID -c Name -c "Router Type"
```

输出示例

```
+--------------------------------------+---------------+-------------+
| ID                                   | Name          | Router Type |
+--------------------------------------+---------------+-------------+
```

```
| 148a8023-62a7-4672-b018-003462f8d7dc | public_network | External    |
+--------------------------------------+----------------+-------------+
```

网络列表中会显示具有外部路由器类型的网络。如果至少有一个没有，请参阅创建默认浮动 IP 网络和创建默认供应商网络。

重要

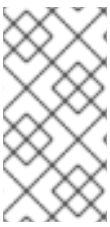
如果外部网络的 CIDR 范围与其中一个默认网络范围重叠，您必须在开始安装过程前更改 install-config.yaml 文件中的匹配网络范围。

默认的网络范围为：

网络	范围
machineNetwork	10.0.0.0/16
serviceNetwork	172.30.0.0/16
clusterNetwork	10.128.0.0/14

警告

如果安装程序找到多个同名的网络，它会随机设置其中之一。为避免这种行为，请在 RHOSP 中为资源创建唯一名称。



注意

如果启用了 Neutron 中继服务插件，则默认创建一个中继端口。如需更多信息，请参阅 Neutron 中继端口。

21.3.7. 为安装程序定义参数

OpenShift Container Platform 安装程序依赖于一个名为 clouds.yaml 的文件。该文件描述了 Red Hat OpenStack Platform(RHOSP)配置参数，包括项目名称、登录信息和授权服务 URL。

流程

1. 创建 clouds.yaml 文件：

   - 如果您的 RHOSP 发行版包含 Horizon Web UI，请在该 UI 中生成 clouds.yaml 文件。

     重要

     记得在 auth 字段中添加密码。您还可以将 secret 保存在 clouds.yaml 以外的 一个独立的文件中。

   - 如果您的 RHOSP 发行版不包含 Horizon Web UI，或者您不想使用 Horizon，请自行创建该文件。如需有关 clouds.yaml 的详细信息，请参阅 RHOSP 文档中的 配置文件。

     ```
     clouds:
       shiftstack:
         auth:
           auth_url: http://10.10.14.42:5000/v3
           project_name: shiftstack
           username: <username>
           password: <password>
           user_domain_name: Default
           project_domain_name: Default
       dev-env:
         region_name: RegionOne
         auth:
           username: <username>
           password: <password>
           project_name: 'devonly'
           auth_url: 'https://10.10.14.22:5001/v2.0'
     ```

2. 如果您的 RHOSP 安装使用自签名证书颁发机构(CA)证书进行端点身份验证：

   a. 将证书颁发机构文件复制到您的机器中。

   b. 将 cacerts 键添加到 clouds.yaml 文件。该值必须是到 CA 证书的绝对、不可 root 访

问的路径：

```
clouds:
  shiftstack:
    ...
    cacert: "/etc/pki/ca-trust/source/anchors/ca.crt.pem"
```

提示

使用自定义 CA 证书运行安装程序后，您可以通过编辑 cloud-provider-config keymap 中的 ca-cert.pem 键的值来更新证书。在命令行中运行：

```
$ oc edit configmap -n openshift-config cloud-provider-config
```

3. 将 clouds.yaml 文件放在以下位置之一：

   a. OS_CLIENT_CONFIG_FILE 环境变量的值

   b. 当前目录

   c. 特定于 Unix 的用户配置目录，如 ~/.config/openstack/clouds.yaml

   d. 特定于 Unix 的站点配置目录，如 /etc/openstack/clouds.yaml

   安装程序会按顺序搜索 clouds.yaml。

21.3.8. 设置 OpenStack Cloud Controller Manager 选项

另外，您可以编辑集群的 OpenStack Cloud Controller Manager (CCM) 配置。此配置控制 OpenShift Container Platform 与 Red Hat OpenStack Platform (RHOSP) 的交互方式。

有关配置参数的完整列表，请参阅"安装 OpenStack"文档中的"OpenStack Cloud Controller Manager 参考指南"页面。

流程

1. 如果您还没有为集群生成清单文件，请运行以下命令生成这些文件：

```
$ openshift-install --dir <destination_directory> create manifests
```

2. 在文本编辑器中，打开 cloud-provider 配置清单文件。例如：

```
$ vi openshift/manifests/cloud-provider-config.yaml
```

3. 根据 CCM 参考指南修改选项。

针对负载均衡配置 Octavia 的情况比较常见。例如：

```
#...
[LoadBalancer]
lb-provider = "amphora" (1)
floating-network-id="d3deb660-4190-40a3-91f1-37326fe6ec4a" (2)
create-monitor = True (3)
monitor-delay = 10s (4)
monitor-timeout = 10s (5)
monitor-max-retries = 1 (6)
#...
```

(1) 此属性设置负载均衡器使用的 Octavia 供应商。它接受 "ovn" 或 "amphora" 作为值。如果您选择使用 OVN，还必须将 lb-method 设置为 SOURCE_IP_PORT。

(2) 如果要将多个外部网络用于集群，则需要此属性。云提供商在网络上创建此处指定的浮动 IP 地址。

(3) 此属性控制云供应商是否为 Octavia 负载均衡器创建运行状况监控器。将值设为 True 来创建运行状况监视器。从 RHOSP 16.2 开始，这个功能仅适用于 Amphora 供应商。

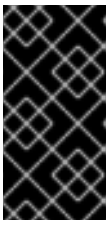
(4) 此属性设定监控端点的频率。该值必须采用 time.ParseDuration() 格式。如果 create-monitor 属性的值为 True，则需要此属性。

(5)

此属性设定监控请求在超时前打开的时间。该值必须采用 time.ParseDuration() 格式。如果 create-monitor 属性的值为 True，则需要此属性。

6

此属性定义在负载均衡器被标记为在线前需要成功完成监控请求。该值必须是整数。如果 create-monitor 属性的值为 True，则需要此属性。



重要

在保存更改之前，请验证该文件的结构是否正确。如果属性没有放入相应的部分，集群可能会失败。

重要

如果使用将 .spec.externalTrafficPolicy 属性的值设置为 Local 的服务，则必须将 create-monitor 属性的值设置为 True。RHOSP 16.2 中的 OVN Octavia 供应商不支持健康监控器。因此，当 lb-provider 值设为 "ovn" 时，将 ETP 参数值设置为 Local 的服务可能无法响应。

4. 保存对文件的更改并开始安装。

提示

您可以在运行安装程序后更新云供应商配置。在命令行中运行：

```
$ oc edit configmap -n openshift-config cloud-provider-config
```

保存更改后，您的集群将需要一些时间重新配置其自身。如果您的任一节点都没有 SchedulingDisabled 状态，则此过程已完成。

21.3.9. 获取安装程序

在安装 OpenShift Container Platform 前，将安装文件下载到您用于安装的主机上。

先决条件

- 您有一台运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB。

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐户，请使用您的凭证登录。如果没有，请创建一个帐户。

2. 选择您的基础架构供应商。

3. 进入到安装类型的页面，下载与您的主机操作系统和架构对应的安装程序，并将该文件放在您要存储安装配置文件的目录中。

重要

安装程序会在用来安装集群的计算机上创建几个文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。

重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，请为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret。此 pull secret 允许您与所含授权机构提供的服务进行身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

21.3.10. 创建安装配置文件

您可以自定义在 Red Hat OpenStack Platform(RHOSP)上安装的 OpenShift Container Platform 集群。

先决条件

- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。

流程

1. 创建 install-config.yaml 文件。

   a. 进入包含安装程序的目录并运行以下命令：

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

   1 对于 <installation_directory>，请指定要存储安装程序创建的文件的目录名称。

   在指定目录时：

   - 验证该目录是否具有执行权限。在安装目录中运行 Terraform 二进制文件需要这个权限。
   - 使用空目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

   b. 在提示符处，提供云的配置详情：

      i. 可选：选择用于访问集群机器的 SSH 密钥。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 ssh-agent 进程使用的 SSH 密钥。

..

ii. 选择 **openstack** 作为目标平台。

iii. 指定用于安装集群的 **Red Hat OpenStack Platform(RHOSP)**外部网络名称。

iv. 指定用于从外部访问 **OpenShift API** 的浮动 **IP** 地址。

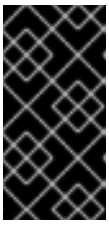
v. 指定至少有 **16 GB RAM** 用于 **control plane** 节点，以及计算节点的 **8 GB RAM**。

vi. 选择集群要部署到的基域。所有 **DNS** 记录都将是这个基域的子域，并且还包括集群名称。

vii. 为集群输入一个名称。名称长度必须为 **14** 个或更少字符。

2. 修改 **install-config.yaml** 文件。您可以在"安装配置参数"部分找到有关可用参数的更多信息。

3. 备份 **install-config.yaml** 文件，以便您可以使用它安装多个集群。



重要

install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用此文件，必须现在备份。

其他资源

- OpenStack 的安装配置参数

21.3.10.1. 在安装过程中配置集群范围的代理

生产环境可能会拒绝直接访问互联网，而是提供 **HTTP** 或 **HTTPS** 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 **OpenShift Container Platform** 集群配置为使用代理。

先决条件

- 您有一个现有的 install-config.yaml 文件。

- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 Proxy 对象的 spec.noProxy 字段中添加站点来绕过代理。

注意

Proxy 对象 status.noProxy 字段使用安装配置中的 networking.machineNetwork[].cidr、networking.clusterNetwork[].cidr 和 networking.serviceNetwork[] 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，Proxy 对象 status.noProxy 字段也会使用实例元数据端点填充(169.254.169.254)。

流程

1. 编辑 install-config.yaml 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 http。

2. 用于创建集群外 HTTPS 连接的代理 URL。

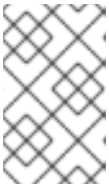
3.

4

如果提供，安装程序会在 openshift-config 命名空间中生成名为 user-ca-bundle 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 trusted-ca-bundle 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， Proxy 对象的 trustedCA 字段中也会引用此配置映射。additionalTrustBundle 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

5

可选：决定 Proxy 对象的配置以引用 trustedCA 字段中 user-ca-bundle 配置映射的策略。允许的值是 Proxyonly 和 Always。仅在配置了 http/https 代理时，使用 Proxyonly 引用 user-ca-bundle 配置映射。使用 Always 始终引用 user-ca-bundle 配置映射。默认值为 Proxyonly。



注意

安装程序不支持代理的 readinessEndpoints 字段。

注意

如果安装程序超时，重启并使用安装程序的 wait-for 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. 保存该文件并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 cluster 的集群范围代理，该代理 使用 提供的 install-config.yaml 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 cluster Proxy 对象，但它会有一个空 spec。

注意

只支持名为 cluster 的 Proxy 对象，且无法创建额外的代理。

21.3.10.2. RHOSP 部署中的自定义子网

另外，您还可以在您选择的 Red Hat OpenStack Platform(RHOSP)子网中部署集群。子网的 GUID

作为 install-config.yaml 文件中的 platform.openstack.machinesSubnet 的值传递。

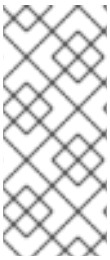
此子网被用作集群的主子网。默认情况下，其上会创建节点和端口。您可以通过将 platform.openstack.machinesSubnet 属性的值设置为子网的 UUID，在不同的 RHOSP 子网中创建节点和端口。

在使用自定义子网运行 OpenShift Container Platform 安装程序前，请验证您的配置是否满足以下要求：

- platform.openstack.machinesSubnet 使用的子网启用了 DHCP。
- platform.openstack.machinesSubnet 的 CIDR 与 networking.machineNetwork 的 CIDR 匹配。
- 安装程序用户有在此网络上创建端口的权限，包括带有固定 IP 地址的端口。

使用自定义子网的集群有以下限制：

- 如果您计划安装使用浮动 IP 地址的集群，则必须将 platform.openstack.machinesSubnet 子网附加到连接到 externalNetwork 网络的路由器。
- 如果在 install-config.yaml 文件中设置了 platform.openstack.machinesSubnet 值，安装程序不会为您的 RHOSP 机器创建专用网络或子网。
- 您不能与自定义子网同时使用 platform.openstack.externalDNS 属性。要将 DNS 添加到使用自定义子网的集群，请在 RHOSP 网络上配置 DNS。



注意

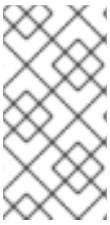
默认情况下，API VIP 使用 x.x.x.5，Ingress VIP 从网络 CIDR 块获取 x.x.x.7。要覆盖这些默认值，请为 DHCP 分配池之外的 platform.openstack.apiVIPs 和 platform.openstack.ingressVIPs 设置值。

重要

集群安装后无法调整网络的 CIDR 范围。红帽不提供有关在集群安装过程中确定范围的直接指导，因为它需要仔细考虑每个命名空间创建的 pod 数量。

21.3.10.3. 使用裸机部署集群

如果您希望集群使用裸机，请修改 install-config.yaml 文件。集群可以同时在裸机上运行 control plane 和计算机器，或者只在计算机器上运行。



注意

确保 install-config.yaml 文件反映了您用于裸机 worker 的 RHOSP 网络是否支持浮动 IP 地址。

先决条件

- RHOSP Bare Metal 服务(Ironic) 通过 RHOSP Compute API 启用并访问。
- 裸机 可作为 RHOSP 类别 提供。
- 如果您的集群在一个大于 16.1.6 且小于 16.2.4 的 RHOSP 版本上运行，则裸机 worker 无法正常工作，因为存在一个已知问题会导致元数据服务对 OpenShift Container Platform 节点上的服务不可用。
- RHOSP 网络支持 VM 和裸机服务器附加。
- 您的网络配置不依赖于供应商网络。不支持提供商网络。
- 如果要将机器部署到预先存在的网络中，则会置备 RHOSP 子网。
- 如果要在安装程序置备的网络中部署机器，RHOSP Bare Metal 服务(Ironic)可以侦听在租户网络上运行的 Preboot eXecution Environment(PXE)引导机器并与之交互。
- 作为 OpenShift Container Platform 安装过程的一部分，创建了 install-config.yaml 文

件。

流程

1. 在 install-config.yaml 文件中编辑机器的类别：

   a. 如果要使用裸机 control plane 机器，将 controlPlane.platform.openstack.type 的值改为裸机类型。

   b. 将 compute.platform.openstack.type 的值改为一个裸机类型。

   c. 如果要将机器部署到预先存在的网络中，请将 platform.openstack.machinesSubnet 的值改为网络的 RHOSP 子网 UUID。control plane 和计算机器必须使用相同的子网。

   裸机 install-config.yaml 文件示例

```
controlPlane:
  platform:
    openstack:
      type: <bare_metal_control_plane_flavor> 1
...

compute:
 - architecture: amd64
   hyperthreading: Enabled
   name: worker
   platform:
     openstack:
       type: <bare_metal_compute_flavor> 2
   replicas: 3
...

platform:
  openstack:
    machinesSubnet: <subnet_UUID> 3
...
```

1

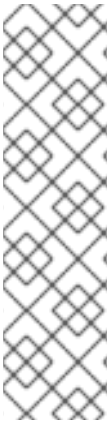
2

将此值更改为用于计算机器的裸机类别。

3

如果要使用预先存在的网络，将此值改为 RHOSP 子网的 UUID。

使用更新的 install-config.yaml 文件完成安装过程。部署期间创建的计算机器使用添加到 文件的类别。



注意

在等待裸机引导时，安装程序可能会超时。

如果安装程序超时，重启并使用安装程序的 wait-for 命令完成部署。例如：

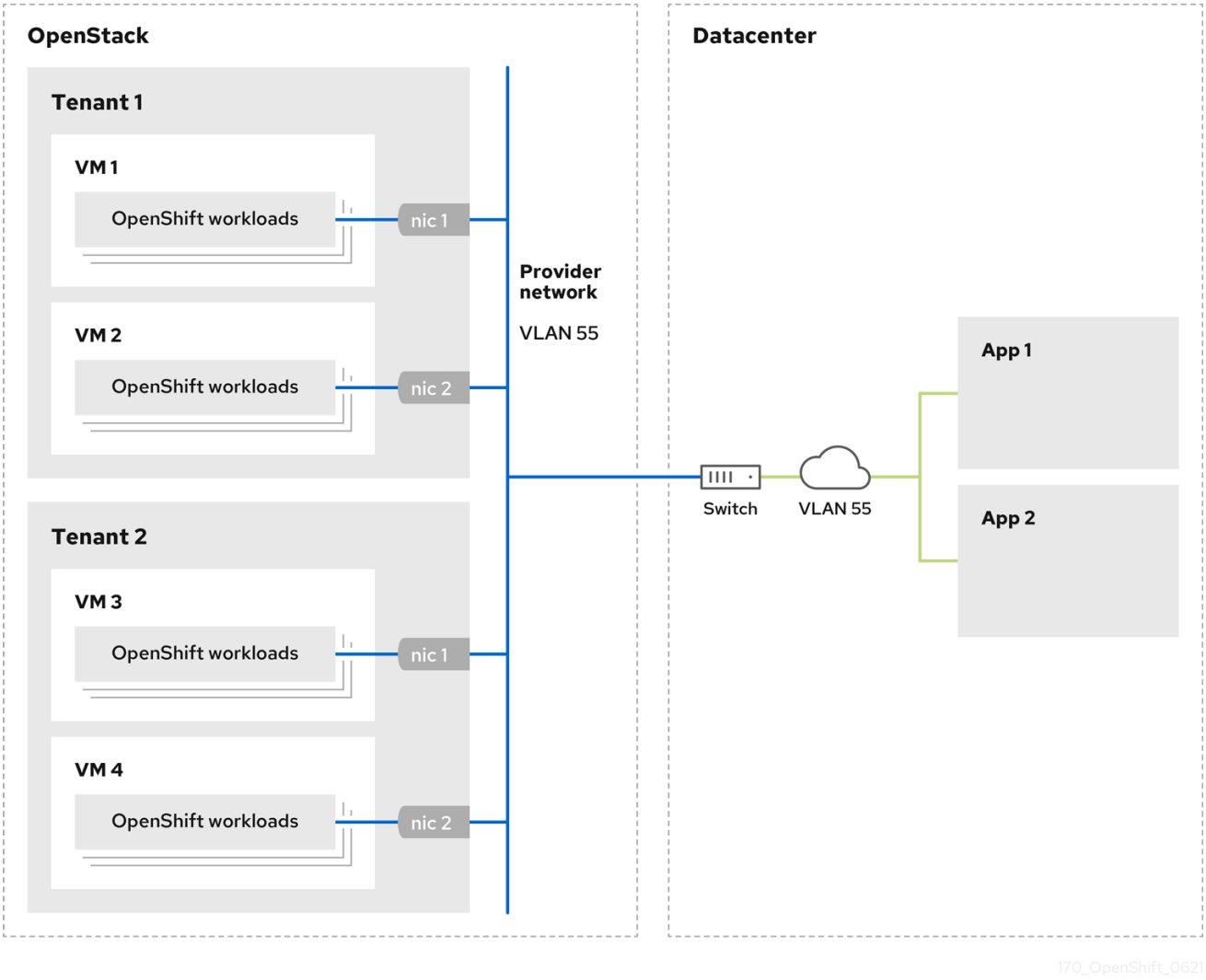
```
$ ./openshift-install wait-for install-complete --log-level debug
```

21.3.10.4. RHOSP 提供商网络上的集群部署

您可以使用供应商网络上的主网络接口在 Red Hat OpenStack Platform(RHOSP)上部署 OpenShift Container Platform 集群。提供商网络通常用于为项目提供可用于访问互联网的公共网络的直接访问权限。您还可以在项目间共享提供商网络，作为网络创建流程的一部分。

RHOSP 提供商网络直接映射到数据中心内的现有物理网络。RHOSP 管理员必须创建它们。

在以下示例中，OpenShift Container Platform 工作负载使用提供商网络连接到数据中心：



在提供商网络上安装的 OpenShift Container Platform 集群不需要租户网络或浮动 IP 地址。安装程序不会在安装过程中创建这些资源。

提供商网络类型示例包括 flat（未标记）和 VLAN（802.1Q 标记）。

注意

集群可以在网络类型允许的情况下支持任意数量的提供商网络连接。例如，VLAN 网络通常支持多达 4096 个连接。

您可以在 RHOSP 文档中的 了解更多有关供应商和租户网络的信息。

21.3.10.4.1. 集群安装的 RHOSP 提供商网络要求

在安装 OpenShift Container Platform 集群前，您的 Red Hat OpenStack Platform(RHOSP)部署和提供商网络必须满足以下多个条件：

- RHOSP 网络服务(Neutron)通过 RHOSP 网络 API 启用 并访问。
- RHOSP 网络服务 启用了端口安全性并允许地址对扩展。
- 提供商网络可以与其他租户共享。

提示

使用 openstack network create 命令和 --share 标志来创建可共享的网络。

- 用于安装集群的 RHOSP 项目必须拥有提供商网络以及适当的子网。

提示

要为名为"openshift"的项目创建网络，请输入以下命令

```
$ openstack network create --project openshift
```

要为名为"openshift"的项目创建子网，请输入以下命令

```
$ openstack subnet create --project openshift
```

要了解更多有关在 RHOSP 上创建网络的信息，请阅读 提供商网络文档。

如果集群归 admin 用户所有，则必须以该用户身份运行安装程序，以便在网络上创建端口。

重要

提供商网络必须由用于创建集群的 RHOSP 项目所有。如果没有，则 RHOSP Compute 服务(Nova)无法从该网络请求端口。

- 验证提供商网络可以访问 RHOSP 元数据服务 IP 地址，默认为 169.254.169.254。

  根据 RHOSP SDN 和网络服务配置，您可能需要在创建子网时提供路由。例如：

  ```
  $ openstack subnet create --dhcp --host-route
  destination=169.254.169.254/32,gateway=192.0.2.2 ...
  ```

- 可选：要保护网络，请创建 基于角色的访问控制(RBAC) 规则，以限制对单个项目的网络访问。

21.3.10.4.2. 在提供商网络上部署具有主接口的集群

您可以在 Red Hat OpenStack Platform(RHOSP) 提供商网络上部署具有主网络接口的 OpenShift Container Platform 集群。

先决条件

- 您的 Red Hat OpenStack Platform(RHOSP)部署被配置为"RHOSP 供应商网络要求用于集群安装"。

流程

1. 在文本编辑器中，打开 install-config.yaml 文件。
2. 将 platform.openstack.apiVIPs 属性的值设置为 API VIP 的 IP 地址。
3. 将 platform.openstack.ingressVIPs 属性的值设置为 Ingress VIP 的 IP 地址。
4. 将 platform.openstack.machinesSubnet 属性的值设置为提供商网络子网的 UUID。
5. 将 networking.machineNetwork.cidr 属性的值设置为提供商网络子网的 CIDR 块。

重要

platform.openstack.apiVIPs 和 platform.openstack.ingressVIPs 属性必须从 networking.machineNetwork.cidr 块中取消分配 IP 地址。

依赖于 RHOSP 提供商网络的集群的安装配置文件部分

```
...
platform:
  openstack:
    apiVIPs: 1
      - 192.0.2.13
    ingressVIPs: 2
      - 192.0.2.23
    machinesSubnet: fa806b2f-ac49-4bce-b9db-124bc64209bf
    # ...
networking:
  machineNetwork:
  - cidr: 192.0.2.0/24
```

1 2

在 OpenShift Container Platform 4.12 及更新的版本中，apiVIP 和 ingressVIP 配置设置已弃用。反之，使用列表格式在 apiVIPs 和 ingressVIPs 配置设置中输入值。

警告

您不能在将提供商网络用于主网络接口时设置 platform.openstack.externalNetwork 或 platform.openstack.externalDNS 参数。

在部署集群时，安装程序使用 install-config.yaml 文件在提供商网络上部署集群。

提示

您可以将额外的网络（包括提供商网络）添加到 platform.openstack.additionalNetworkIDs 列表中。

部署集群后，您可以将 pod 附加到额外网络。如需更多信息，请参阅 了解多个网络。

21.3.10.5. RHOSP 的自定义 install-config.yaml 文件示例

以下示例 install-config.yaml 文件演示了所有可能的 Red Hat OpenStack Platform (RHOSP) 自定义选项。

重要

此示例文件仅供参考。您必须使用安装程序来获取 install-config.yaml 文件。

例 21.2. 单个堆栈 install-config.yaml 文件示例

```
apiVersion: v1
baseDomain: example.com
controlPlane:
  name: master
  platform: {}
  replicas: 3
compute:
- name: worker
  platform:
    openstack:
      type: ml.large
  replicas: 3
metadata:
  name: example
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  serviceNetwork:
  - 172.30.0.0/16
  networkType: OVNKubernetes
platform:
  openstack:
    cloud: mycloud
    externalNetwork: external
    computeFlavor: m1.xlarge
    apiFloatingIP: 128.0.0.1
```

```
fips: false
pullSecret: '{"auths": ...}'
sshKey: ssh-ed25519 AAAA...
```

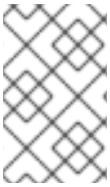
例 21.3. 双堆栈 install-config.yaml 文件示例

```
apiVersion: v1
baseDomain: example.com
controlPlane:
  name: master
  platform: {}
  replicas: 3
compute:
- name: worker
  platform:
    openstack:
      type: ml.large
  replicas: 3
metadata:
  name: example
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  - cidr: fd01::/48
    hostPrefix: 64
  machineNetwork:
  - cidr: 192.168.25.0/24
  - cidr: fd2e:6f44:5dd8:c956::/64
  serviceNetwork:
  - 172.30.0.0/16
  - fd02::/112
  networkType: OVNKubernetes
platform:
  openstack:
    cloud: mycloud
    externalNetwork: external
    computeFlavor: m1.xlarge
    apiVIPs:
    - 192.168.25.10
    - fd2e:6f44:5dd8:c956:f816:3eff:fec3:5955
    ingressVIPs:
    - 192.168.25.132
    - fd2e:6f44:5dd8:c956:f816:3eff:fe40:aecb
    controlPlanePort:
      fixedIPs:
      - subnet:
          name: openshift-dual4
      - subnet:
          name: openshift-dual6
      network:
        name: openshift-dual
```

```
fips: false
pullSecret: '{"auths": ...}'
sshKey: ssh-ed25519 AAAA...
```

21.3.10.6. 可选：使用双栈网络配置集群

您可以在 RHOSP 上创建双栈集群。但是，只有在您使用带有 IPv4 和 IPv6 子网的 RHOSP 网络时，才会启用双栈配置。



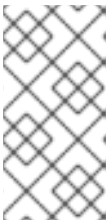
注意

RHOSP 不支持将 IPv4 单堆栈集群转换为双栈集群网络。

21.3.10.6.1. 部署双栈集群

流程

1. 创建具有 IPv4 和 IPv6 子网的网络。ipv6-ra-mode 和 ipv6-address-mode 字段的可用地址模式包括：dhcpv6-stateful,dhcpv6-stateless, 和 slaac。



注意

双栈网络 MTU 必须同时容纳 IPv6 的最小 MTU，即 1280，OVN-Kubernetes 封装开销为 100。

注意

必须在子网上启用 DHCP。

2. 创建 API 和 Ingress VIP 端口。

3. 将 IPv6 子网添加到路由器，以启用路由器公告。如果使用提供商网络，您可以通过将网络添加为外部网关来启用路由器广告，该网关也可以启用外部连接。

4. 要为集群节点配置 IPv4 和 IPv6 地址端点，请编辑 install-config.yaml 文件。以下是 install-config.yaml 文件示例：

示例 install-config.yaml

```
apiVersion: v1
baseDomain: mydomain.test
compute:
- name: worker
  platform:
    openstack:
      type: m1.xlarge
  replicas: 3
controlPlane:
  name: master
  platform:
    openstack:
      type: m1.xlarge
  replicas: 3
metadata:
  name: mycluster
networking:
  machineNetwork: 1
  - cidr: "192.168.25.0/24"
  - cidr: "fd2e:6f44:5dd8:c956::/64"
  clusterNetwork: 2
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  - cidr: fd01::/48
    hostPrefix: 64
  serviceNetwork: 3
  - 172.30.0.0/16
  - fd02::/112
platform:
  openstack:
    ingressVIPs: ['192.168.25.79', 'fd2e:6f44:5dd8:c956:f816:3eff:fef1:1bad'] 4
    apiVIPs: ['192.168.25.199', 'fd2e:6f44:5dd8:c956:f816:3eff:fe78:cf36'] 5
    controlPlanePort: 6
      fixedIPs: 7
      - subnet: 8
          name: subnet-v4
          id: subnet-v4-id
      - subnet: 9
          name: subnet-v6
          id: subnet-v6-id
      network: 10
        name: dualstack
        id: network-id
```

1 2 3

您必须为 IPv4 和 IPv6 地址系列指定一个 IP 地址范围。

4

指定 Ingress VIP 服务的虚拟 IP (VIP)地址端点，以为集群提供接口。

5

指定 API VIP 服务的虚拟 IP (VIP)地址端点，以为集群提供接口。

6

指定集群中所有节点使用的双栈网络详情。

7

此字段中指定的任何子网的 CIDR 必须与 networks.machineNetwork 中列出的 CIDR 匹配。

8 9

您可以为 name 或 id 指定值，或同时指定两者的值。

10

在 ControlPlanePort 字段中指定 network 是可选的。

或者，如果您需要一个 IPv6 主双栈集群，请按照以下示例编辑 install-config.yaml 文件：

示例 install-config.yaml

```
apiVersion: v1
baseDomain: mydomain.test
compute:
- name: worker
  platform:
    openstack:
      type: m1.xlarge
  replicas: 3
controlPlane:
  name: master
  platform:
    openstack:
```

```
      type: m1.xlarge
  replicas: 3
metadata:
  name: mycluster
networking:
  machineNetwork: (1)
  - cidr: "fd2e:6f44:5dd8:c956::/64"
  - cidr: "192.168.25.0/24"
  clusterNetwork: (2)
  - cidr: fd01::/48
    hostPrefix: 64
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  serviceNetwork: (3)
  - fd02::/112
  - 172.30.0.0/16
platform:
  openstack:
    ingressVIPs: ['fd2e:6f44:5dd8:c956:f816:3eff:fef1:1bad', '192.168.25.79'] (4)
    apiVIPs: ['fd2e:6f44:5dd8:c956:f816:3eff:fe78:cf36', '192.168.25.199'] (5)
    controlPlanePort: (6)
      fixedIPs: (7)
      - subnet: (8)
          name: subnet-v6
          id: subnet-v6-id
      - subnet: (9)
          name: subnet-v4
          id: subnet-v4-id
      network: (10)
        name: dualstack
        id: network-id
```

(1) (2) (3)

您必须为 IPv4 和 IPv6 地址系列指定一个 IP 地址范围。

(4)

指定 Ingress VIP 服务的虚拟 IP (VIP)地址端点，以为集群提供接口。

(5)

指定 API VIP 服务的虚拟 IP (VIP)地址端点，以为集群提供接口。

(6)

指定集群中所有节点使用的双栈网络详情。

7

此字段中指定的任何子网的 CIDR 必须与 networks.machineNetwork 中列出的 CIDR 匹配。

8 9

您可以为 name 或 id 指定值，或同时指定两者的值。

10

在 ControlPlanePort 字段中指定 network 是可选的。

21.3.10.7. 使用用户管理的负载均衡器在 OpenStack 上安装集群配置

以下示例 install-config.yaml 文件演示了如何配置使用外部用户管理的负载均衡器而不是默认的内部负载均衡器的集群。

```
apiVersion: v1
baseDomain: mydomain.test
compute:
- name: worker
  platform:
    openstack:
      type: m1.xlarge
  replicas: 3
controlPlane:
  name: master
  platform:
    openstack:
      type: m1.xlarge
  replicas: 3
metadata:
  name: mycluster
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 192.168.10.0/24
platform:
  openstack:
    cloud: mycloud
    machinesSubnet: 8586bf1a-cc3c-4d40-bdf6-c243decc603a 1
    apiVIPs:
    - 192.168.10.5
    ingressVIPs:
```

```
- 192.168.10.7
loadBalancer:
  type: UserManaged 2
```

1

无论您使用哪个负载均衡器，负载均衡器都会部署到这个子网中。

2

UserManaged 值表示您使用用户管理的负载均衡器。

21.3.11. 为集群节点 SSH 访问生成密钥对

在 OpenShift Container Platform 安装过程中，您可以为安装程序提供 SSH 公钥。密钥通过它们的 Ignition 配置文件传递给 Red Hat Enterprise Linux CoreOS(RHCOS)节点，用于验证对节点的 SSH 访问。密钥添加到每个节点上 core 用户的 ~/.ssh/authorized_keys 列表中，这将启用免密码身份验证。

将密钥传递给节点后，您可以使用密钥对作为用户 核心 通过 SSH 连接到 RHCOS 节点。若要通过 SSH 访问节点，必须由 SSH 为您的本地用户管理私钥身份。

如果要通过 SSH 连接到集群节点来执行安装调试或灾难恢复，则必须在安装过程中提供 SSH 公钥。./openshift-install gather 命令还需要在集群节点上设置 SSH 公钥。

重要

不要在生产环境中跳过这个过程，在生产环境中需要灾难恢复和调试。

流程

1. 如果您在本地计算机上没有可用于在集群节点上进行身份验证的现有 SSH 密钥对，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1

指定新 SSH 密钥的路径和文件名，如 ~/.ssh/id_ed25519。如果您已有密钥对，请确保您的公钥位于 ~/.ssh 目录中。

注意

如果您计划在 x86_64、ppc64le 和 s390x 架构上安装使用 RHEL 加密库（这些加密库已提交给 NIST 用于 FIPS 140-2/140-3 验证）的 OpenShift Container Platform 集群，则不要创建使用 ed25519 算法的密钥。相反，创建一个使用 rsa 或 ecdsa 算法的密钥。

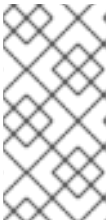
2. 查看公共 SSH 密钥：

```
$ cat <path>/<file_name>.pub
```

例如，运行以下命令来查看 ~/.ssh/id_ed25519.pub 公钥：

```
$ cat ~/.ssh/id_ed25519.pub
```

3. 将 SSH 私钥身份添加到本地用户的 SSH 代理（如果尚未添加）。在集群节点上，或者要使用 ./openshift-install gather 命令，需要对该密钥进行 SSH 代理管理，才能在集群节点上进行免密码 SSH 身份验证。



注意

在某些发行版中，自动管理默认 SSH 私钥身份，如 ~/.ssh/id_rsa 和 ~/.ssh/id_dsa。

a. 如果 ssh-agent 进程尚未为您的本地用户运行，请将其作为后台任务启动：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果集群处于 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

4. 将 SSH 私钥添加到 ssh-agent ：

```
$ ssh-add <path>/<file_name> 1
```

1 指定 SSH 私钥的路径和文件名，如 ~/.ssh/id_ed25519.pub

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

后续步骤

- 安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

21.3.12. 启用对环境的访问

在部署时，所有 OpenShift Container Platform 机器都是在 Red Hat OpenStack Platform(RHOSP) 租户网络中创建的。因此，大多数 RHOSP 部署中都无法直接访问它们。

您可以在安装过程中使用浮动 IP 地址(FIP)来配置 OpenShift Container Platform API 和应用程序访问。您还可以在没有配置 FIP 的情况下完成安装，但安装程序不会配置一种从外部访问 API 或应用程序的方法。

21.3.12.1. 启用通过浮动 IP 地址进行访问

创建浮动 IP(FIP)地址，以便从外部访问 OpenShift Container Platform API 和集群应用程序。

流程

1. 使用 Red Hat OpenStack Platform(RHOSP)CLI 创建 API FIP：

```
$ openstack floating ip create --description "API <cluster_name>.<base_domain>"
<external_network>
```

2. 使用 Red Hat OpenStack Platform(RHOSP)CLI，创建应用程序或 Ingress，FIP：

```
$ openstack floating ip create --description "Ingress <cluster_name>.<base_domain>"
<external_network>
```

3. 在 API 和 Ingress FIP 的 DNS 服务器中添加符合这些模式的记录：

```
api.<cluster_name>.<base_domain>.  IN  A  <API_FIP>
*.apps.<cluster_name>.<base_domain>. IN  A <apps_FIP>
```

注意

如果您不控制 DNS 服务器，可以通过将集群域名（如以下内容）添加到 /etc/hosts 文件中来访问集群：

- **<api_floating_ip> api.<cluster_name>.<base_domain>**
- **<application_floating_ip> grafana-openshift-monitoring.apps.<cluster_name>.<base_domain>**
- **<application_floating_ip> prometheus-k8s-openshift-monitoring.apps.<cluster_name>.<base_domain>**
- **<application_floating_ip> oauth-openshift.apps.<cluster_name>.<base_domain>**
- **<application_floating_ip> console-openshift-console.apps.<cluster_name>.<base_domain>**
- **application_floating_ip integrated-oauth-server-openshift-authentication.apps.<cluster_name>.<base_domain>**

/etc/hosts 文件中的集群域名授予对本地集群的 Web 控制台和监控界面的访问权限。您还可以使用 kubectl 或 oc。您可以使用指向 <application_floating_ip> 的额外条目来访问用户应用程序。此操作使 API 和应用程序可供您访问，不适合于生产部署，但允许对开发和测试进行安装。

4. **将 FIP 添加到 install-config.yaml 文件中，作为以下参数的值：**

   - **platform.openstack.ingressFloatingIP**
   - **platform.openstack.apiFloatingIP**

如果使用这些值，还必须在 install-config.yaml 文件中输入一个外部网络作为

platform.openstack.externalNetwork 参数的值。

提示

您可以通过分配浮动 IP 地址并更新防火墙配置，使 OpenShift Container Platform 资源在集群外可用。

21.3.12.2. 完成没有浮动 IP 地址的安装

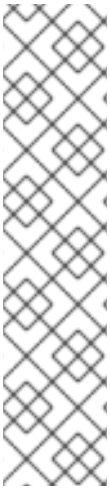
您可以在 Red Hat OpenStack Platform(RHOSP)上安装 OpenShift Container Platform，而无需提供浮动 IP 地址。

在 install-config.yaml 文件中，不要定义以下参数：

- platform.openstack.ingressFloatingIP
- platform.openstack.apiFloatingIP

如果无法提供外部网络，也可以将 platform.openstack.externalNetwork 留 空。如果没有为 platform.openstack.externalNetwork 提供值，则不会为您创建路由器，如果没有额外的操作，安装程序将无法从 Glance 检索镜像。您必须自行配置外部连接。

如果您在因为缺少浮动 IP 地址或名称解析而无法访问集群 API 的系统中运行安装程序，安装会失败。在这些情况下，您可以使用代理网络或者从与机器位于相同网络的系统中运行安装程序。



注意

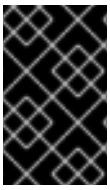
您可以通过为 API 和 Ingress 端口创建 DNS 记录来启用名称解析。例如：

```
api.<cluster_name>.<base_domain>.  IN  A  <api_port_IP>
*.apps.<cluster_name>.<base_domain>. IN  A <ingress_port_IP>
```

如果您不控制 DNS 服务器，您可以将记录添加到 /etc/hosts 文件中。此操作使 API 可供您自己访问，不适合于生产部署，而是允许安装以进行开发和测试。

21.3.13. 部署集群

您可以在兼容云平台上安装 OpenShift Container Platform。



重要

在初始安装过程中，您只能运行安装程序的 create cluster 命令一次。

先决条件

- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。
- 已确认主机上的云供应商帐户具有部署集群的正确权限。权限不正确的帐户会导致安装过程失败，并显示包括缺失权限的错误消息。

流程

- 进入包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1. 对于 <installation_directory>，请指定自定义 ./install-config.yaml 文件的位置。
2. 要查看不同的安装详情，请指定 warn、debug 或 error，而不是 info。

验证

当集群部署成功完成时：

- 终端会显示用于访问集群的说明，包括指向 Web 控制台和 kubeadmin 用户的凭证的链接。

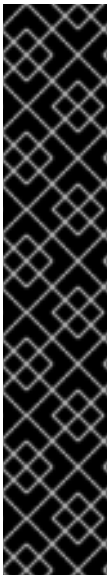
- 凭证信息还会输出到 <installation_directory>/.openshift_install.log.

重要

不要删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 node-bootstrapper 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。

- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

21.3.14. 验证集群状态

您可以在安装过程中或安装后验证 OpenShift Container Platform 集群的状态。

流程

1. 在集群环境中，导出管理员的 kubeconfig 文件：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 <installation_directory>，请指定安装文件保存到的目录的路径。

kubeconfig 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。

2. 查看部署后创建的 control plane 和计算机器：

```
$ oc get nodes
```

3. 查看集群的版本：

```
$ oc get clusterversion
```

4. 查看 Operator 的状态：

```
$ oc get clusteroperator
```

5. 查看集群中的所有正在运行的 pod:

```
$ oc get pods -A
```

21.3.15. 使用 CLI 登录集群

您可以通过导出集群 kubeconfig 文件，以默认系统用户身份登录集群。kubeconfig 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。

- 已安装 oc CLI。

流程

1. 导出 kubeadmin 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 <installation_directory>，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 oc 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

其他资源

- 如需有关 访问和了解 OpenShift Container Platform Web 控制台的更多详情，请参阅 访问 Web 控制台。

21.3.16. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform

订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于 远程健康监控

21.3.17. 后续步骤

- 自定义集群。
- 如果需要，您可以选择 不使用远程健康报告。
- 如果您需要启用对节点端口的外部访问，请使用节点端口配置集群流量。
- 如果您没有将 RHOSP 配置为接受通过浮动 IP 地址的应用程序流量，请使用 浮动 IP 地址配置 RHOSP 访问。

21.4. 在您自己的基础架构的 OPENSTACK 上安装集群

在 OpenShift Container Platform 版本 4.16 中，您可以在运行于用户置备的基础架构上的 Red Hat OpenStack Platform（RHOSP）上安装集群。

通过利用您自己的基础架构，您可以将集群与现有的基础架构集成。与安装程序置备的安装相比，这个过程需要进行更多的操作，因为您必须创建所有 RHOSP 资源，如 Nova 服务器、Neutron 端口和安全组。但是，红帽提供了 Ansible playbook 来帮助您完成部署过程。

21.4.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读有关 选择集群安装方法的文档，并为用户准备它。
- 使用OpenShift 集群支持的平台部分验证 OpenShift Container Platform 4.16 是否与您的 RHOSP 版本兼容。https://docs.redhat.com/en/documentation/openshift_container_platform/4.16/html-

single/architecture/#supported-platforms-for-openshift-clusters_architecture-installation您还可以通过查看 RHOSP 上的 OpenShift Container Platform 支持来比较不同版本的平台支持。

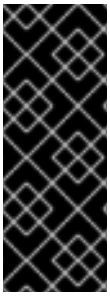
- 您有一个 RHOSP 帐户，您要安装 OpenShift Container Platform。
- 您可以了解集群扩展、control plane 大小和 etcd 的性能和可扩展性实践。如需更多信息，请参阅 扩展集群的建议实践。
- 在运行安装程序的机器上，您有：
  - 用来保存在安装过程中创建的文件的一个单一目录
  - Python 3

21.4.2. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。



重要

如果您的集群无法直接访问互联网，则可以在置备的某些类型的基础架构上执行受限网络安装。在此过程中，您可以下载所需的内容，并使用它为镜像 registry 填充安装软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群前，您要更新镜像 registry 的内容。

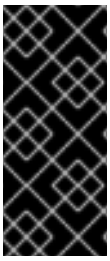
21.4.3. 在 RHOSP 上安装 OpenShift Container Platform 的资源指南

为支持 OpenShift Container Platform 安装，您的 Red Hat OpenStack Platform(RHOSP)配额必须满足以下要求：

表 21.4. RHOSP 上默认 OpenShift Container Platform 集群的建议资源

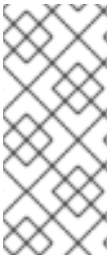
resource	value
浮动 IP 地址	3
端口	15
路由器	1
子网	1
RAM	88 GB
VCPU	22
卷存储	275 GB
实例	7
安全组	3
安全组规则	60
服务器组	2 - 每个机器池中每个额外可用区加 1

集群或许能以少于推荐资源运行，但其性能无法保证。



重要

如果 RHOSP 对象存储(Swift)可用，并由具有 swiftoperator 角色的用户帐户执行，它将用作 OpenShift Container Platform 镜像 registry 的默认后端。在这种情况下，卷存储需要 175 GB。根据镜像 registry 的大小，Swift 空间要求会有所不同。



注意

默认情况下，您的安全组和安全组规则配额可能较低。如果遇到问题，请以管理员 身份运行 openstack quota set --secgroups 3 --secgroup-rules 60 <project> 来提高配额。

OpenShift Container Platform 部署包含 control plane 机器、计算机器和 bootstrap 机器。

21.4.3.1. control plane 机器

默认情况下，OpenShift Container Platform 安装过程会创建三台 control plane 机器。

每台机器都需要：

- 来自 RHOSP 配额的实例
- 来自 RHOSP 配额的端口
- 至少有 16 GB 内存和 4 个 vCPU 的类别
- RHOSP 配额中至少有 100 GB 存储空间

21.4.3.2. 计算机器

默认情况下，OpenShift Container Platform 安装过程会创建三台计算机器。

每台机器都需要：

- 来自 RHOSP 配额的实例
- 来自 RHOSP 配额的端口
- 至少有 8 GB 内存和 2 个 vCPU 的类别
- RHOSP 配额中至少有 100 GB 存储空间

提示

计算机器托管您在 OpenShift Container Platform 上运行的应用程序；运行数量应尽可能多。

21.4.3.3. bootstrap 机器

在安装过程中，会临时置备 bootstrap 机器来支持 control plane。生产 control plane 就绪后，bootstrap 机器会被取消置备。

bootstrap 机器需要：

- 来自 RHOSP 配额的实例
- 来自 RHOSP 配额的端口
- 至少有 16 GB 内存和 4 个 vCPU 的类别
- RHOSP 配额中至少有 100 GB 存储空间

21.4.4. 下载 playbook 依赖项

用于简化用户置备的基础架构安装过程的 Ansible playbook 需要几个 Python 模块。在您要运行安装程序的机器上，添加模块的存储库，然后下载它们。

注意

这些说明假设您使用 Red Hat Enterprise Linux(RHEL)8。

先决条件

- Python 3 已安装在您的机器上。

流程

1. 在命令行中添加软件仓库：

   a. 使用 Red Hat Subscription Manager 注册：

   ```
   $ sudo subscription-manager register # If not done already
   ```

   b. 获取最新的订阅数据：

   ```
   $ sudo subscription-manager attach --pool=$YOUR_POOLID # If not done already
   ```

   c. 禁用当前的软件仓库：

   ```
   $ sudo subscription-manager repos --disable=* # If not done already
   ```

   d. 添加所需的软件仓库：

   ```
   $ sudo subscription-manager repos \
     --enable=rhel-8-for-x86_64-baseos-rpms \
     --enable=openstack-16-tools-for-rhel-8-x86_64-rpms \
     --enable=ansible-2.9-for-rhel-8-x86_64-rpms \
     --enable=rhel-8-for-x86_64-appstream-rpms
   ```

2. 安装模块：

   ```
   $ sudo yum install python3-openstackclient ansible python3-openstacksdk python3-
   netaddr ansible-collections-openstack
   ```

3. 确保 python 命令指向 python 3:

```
$ sudo alternatives --set python /usr/bin/python3
```

21.4.5. 下载安装 playbook

下载 Ansible playbook，可用于在您自己的 Red Hat OpenStack Platform(RHOSP)基础架构上安装 OpenShift Container Platform。

先决条件

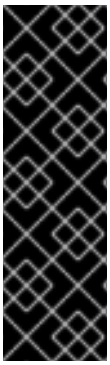
- curl 命令行工具可在您的 计算机上使用。

流程

- 要将 playbook 下载到您的工作目录中，请从命令行运行以下脚本：

```
$ xargs -n 1 curl -O <<< '
    https://raw.githubusercontent.com/openshift/installer/release-
4.16/upi/openstack/bootstrap.yaml
    https://raw.githubusercontent.com/openshift/installer/release-
4.16/upi/openstack/common.yaml
    https://raw.githubusercontent.com/openshift/installer/release-
4.16/upi/openstack/compute-nodes.yaml
    https://raw.githubusercontent.com/openshift/installer/release-
4.16/upi/openstack/control-plane.yaml
    https://raw.githubusercontent.com/openshift/installer/release-
4.16/upi/openstack/down-bootstrap.yaml
    https://raw.githubusercontent.com/openshift/installer/release-
4.16/upi/openstack/down-compute-nodes.yaml
    https://raw.githubusercontent.com/openshift/installer/release-
4.16/upi/openstack/down-control-plane.yaml
    https://raw.githubusercontent.com/openshift/installer/release-
4.16/upi/openstack/down-network.yaml
    https://raw.githubusercontent.com/openshift/installer/release-
4.16/upi/openstack/down-security-groups.yaml
    https://raw.githubusercontent.com/openshift/installer/release-
4.16/upi/openstack/down-containers.yaml
    https://raw.githubusercontent.com/openshift/installer/release-
4.16/upi/openstack/inventory.yaml
    https://raw.githubusercontent.com/openshift/installer/release-
4.16/upi/openstack/network.yaml
    https://raw.githubusercontent.com/openshift/installer/release-
4.16/upi/openstack/security-groups.yaml
    https://raw.githubusercontent.com/openshift/installer/release-
4.16/upi/openstack/update-network-resources.yaml'
```

Playbook 下载到您的计算机。



重要

在安装过程中，您可以修改 playbook 来配置部署。

在集群生命周期中保留所有 playbook。您必须具有 playbook，才能从 RHOSP 中删除 OpenShift Container Platform 集群。

重要

您必须将对 bootstrap.yaml, compute-nodes.yaml, control-plane.yaml, network.yaml, 和 security-groups.yaml 文件的编辑与相应的前缀为 down- 的 playbook 匹配。例如，对 bootstrap.yaml 文件的编辑也必须反映在 down-bootstrap.yaml 文件中。如果没有编辑这两个文件，则支持的删除集群过程将失败。

21.4.6. 获取安装程序

在安装 OpenShift Container Platform 前，将安装文件下载到您用于安装的主机上。

先决条件

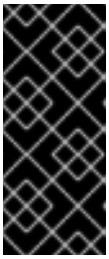
- 您有一台运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB。

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐户，请使用您的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入到安装类型的页面，下载与您的主机操作系统和架构对应的安装程序，并将该文件放在您要存储安装配置文件的目录中。

重要

安装程序会在用来安装集群的计算机上创建几个文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。



重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，请为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret。此 pull secret 允许您与所含授权机构提供的服务进行身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

21.4.7. 为集群节点 SSH 访问生成密钥对

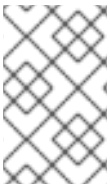
在 OpenShift Container Platform 安装过程中，您可以为安装程序提供 SSH 公钥。密钥通过它们的 Ignition 配置文件传递给 Red Hat Enterprise Linux CoreOS(RHCOS)节点，用于验证对节点的 SSH 访问。密钥添加到每个节点上 core 用户的 ~/.ssh/authorized_keys 列表中，这将启用免密码身份验证。

将密钥传递给节点后，您可以使用密钥对作为用户 核心 通过 SSH 连接到 RHCOS 节点。若要通过 SSH 访问节点，必须由 SSH 为您的本地用户管理私钥身份。

如果要通过 SSH 连接到集群节点来执行安装调试或灾难恢复，则必须在安装过程中提供 SSH 公钥。./openshift-install gather 命令还需要在集群节点上设置 SSH 公钥。

重要

不要在生产环境中跳过这个过程，在生产环境中需要灾难恢复和调试。



注意

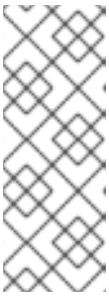
您必须使用本地密钥，而不是使用特定平台方法配置 的密钥，如 AWS 密钥对。

流程

1. 如果您在本地计算机上没有可用于在集群节点上进行身份验证的现有 SSH 密钥对，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 ~/.ssh/id_ed25519。如果您已有密钥对，请确保您的公钥位于 ~/.ssh 目录中。



注意

如果您计划在 x86_64、ppc64le 和 s390x 架构上安装使用 RHEL 加密库（这些加密库已提交给 NIST 用于 FIPS 140-2/140-3 验证）的 OpenShift Container Platform 集群，则不要创建使用 ed25519 算法的密钥。相反，创建一个使用 rsa 或 ecdsa 算法的密钥。

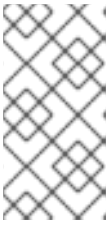
2. 查看公共 SSH 密钥：

```
$ cat <path>/<file_name>.pub
```

例如，运行以下命令来查看 ~/.ssh/id_ed25519.pub 公钥：

```
$ cat ~/.ssh/id_ed25519.pub
```

3. 将 SSH 私钥身份添加到本地用户的 SSH 代理（如果尚未添加）。在集群节点上，或者要使用 ./openshift-install gather 命令，需要对该密钥进行 SSH 代理管理，才能在集群节点上进行免密码 SSH 身份验证。



注意

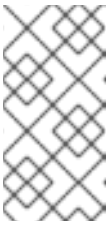
在某些发行版中，自动管理默认 SSH 私钥身份，如 ~/.ssh/id_rsa 和 ~/.ssh/id_dsa。

a. 如果 ssh-agent 进程尚未为您的本地用户运行，请将其作为后台任务启动：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```



注意

如果集群处于 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

4. 将 SSH 私钥添加到 ssh-agent ：

```
$ ssh-add <path>/<file_name> 1
```

1 指定 SSH 私钥的路径和文件名，如 ~/.ssh/id_ed25519.pub

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

后续步骤

- 安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

21.4.8. 创建 Red Hat Enterprise Linux CoreOS(RHCOS)镜像

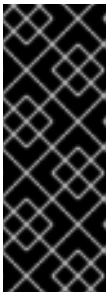
OpenShift Container Platform 安装程序要求 Red Hat OpenStack Platform(RHOSP)集群中存在 Red Hat Enterprise Linux CoreOS(RHCOS)镜像。检索最新的 RHCOS 镜像，然后使用 RHOSP CLI 上传该镜像。

先决条件

- 已安装 RHOSP CLI。

流程

1. 登录到红帽客户门户网站的产品 下载页面。
2. 在 Version 下，为 Red Hat Enterprise Linux（RHEL）8 选择 OpenShift Container Platform 4.16 的最新发行版本。



重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每个发行版本而改变。您必须下载最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。如果可用，请使用与 OpenShift Container Platform 版本匹配的镜像版本。

3. 下载 *Red Hat Enterprise Linux CoreOS(RHCOS)- OpenStack Image(QCOW)*。
4. 解压缩镜像。

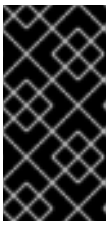
注意

您必须解压 RHOSP 镜像，然后集群才能使用它。下载的文件的名称可能不包含压缩扩展名，如 .gz 或. tgz。要查找是否或者如何压缩文件，请在命令行中输入：

```
$ file <name_of_downloaded_file>
```

5. 从您下载的镜像，使用 RHOSP CLI 在集群中创建名为 rhcos 的镜像：

```
$ openstack image create --container-format=bare --disk-format=qcow2 --file rhcos-${RHCOS_VERSION}-openstack.qcow2 rhcos
```



重要

根据您的 RHOSP 环境，您可能能够以. raw 或 .qcow2 格式 上传镜像。如果使用 Ceph，则必须使用 .raw 格式。

警告

如果安装程序发现多个同名的镜像，它会随机选择其中之一。为避免这种行为，请在 RHOSP 中为资源创建唯一名称。

将镜像上传到 RHOSP 后，就可以在安装过程中使用它。

21.4.9. 验证外部网络访问

OpenShift Container Platform 安装过程需要外部网络访问权限。您必须为其提供外部网络值，否则部署会失败。在开始这个过程前，请验证 Red Hat OpenStack Platform(RHOSP)中是否存在具有外部路由器类型的网络。

先决条件

- 配置 OpenStack 的网络服务，使其让 DHCP 代理转发实例 DNS 查询

流程

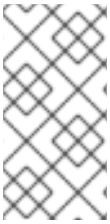
1. 使用 RHOSP CLI 验证"外部"网络的名称和 ID：

```
$ openstack network list --long -c ID -c Name -c "Router Type"
```

输出示例

```
+--------------------------------------+----------------+-------------+
| ID                                   | Name           | Router Type |
+--------------------------------------+----------------+-------------+
| 148a8023-62a7-4672-b018-003462f8d7dc | public_network | External    |
+--------------------------------------+----------------+-------------+
```

网络列表中会显示具有外部路由器类型的网络。如果至少有一个没有，请参阅创建默认浮动 IP 网络和创建默认供应商网络。



注意

如果启用了 Neutron 中继服务插件，则默认创建一个中继端口。如需更多信息，请参阅 Neutron 中继端口。

21.4.10. 启用对环境的访问

在部署时，所有 OpenShift Container Platform 机器都是在 Red Hat OpenStack Platform(RHOSP)租户网络中创建的。因此，大多数 RHOSP 部署中都无法直接访问它们。

您可以在安装过程中使用浮动 IP 地址(FIP)来配置 OpenShift Container Platform API 和应用程序访问。您还可以在没有配置 FIP 的情况下完成安装，但安装程序不会配置一种从外部访问 API 或应用程序的方法。

21.4.10.1. 启用通过浮动 IP 地址进行访问

创建浮动 IP(FIP)地址，以便外部访问 OpenShift Container Platform API、集群应用程序和 bootstrap 过程。

流程

1. 使用 Red Hat OpenStack Platform(RHOSP)CLI 创建 API FIP：

```
$ openstack floating ip create --description "API <cluster_name>.<base_domain>"
<external_network>
```

2. 使用 Red Hat OpenStack Platform(RHOSP)CLI，创建应用程序或 Ingress，FIP：

```
$ openstack floating ip create --description "Ingress <cluster_name>.<base_domain>"
<external_network>
```

3. 使用 Red Hat OpenStack Platform(RHOSP)CLI 创建 bootstrap FIP:

```
$ openstack floating ip create --description "bootstrap machine" <external_network>
```

4. 在 API 和 Ingress FIP 的 DNS 服务器中添加符合这些模式的记录：

```
api.<cluster_name>.<base_domain>.  IN  A  <API_FIP>
*.apps.<cluster_name>.<base_domain>. IN  A <apps_FIP>
```

注意

如果您不控制 DNS 服务器，可以通过将集群域名（如以下内容）添加到 /etc/hosts 文件中来访问集群：

- **<api_floating_ip> api.<cluster_name>.<base_domain>**
- **<application_floating_ip> grafana-openshift-monitoring.apps.<cluster_name>.<base_domain>**
- **<application_floating_ip> prometheus-k8s-openshift-monitoring.apps.<cluster_name>.<base_domain>**
- **<application_floating_ip> oauth-openshift.apps.<cluster_name>.<base_domain>**
- **<application_floating_ip> console-openshift-console.apps.<cluster_name>.<base_domain>**
- **application_floating_ip integrated-oauth-server-openshift-authentication.apps.<cluster_name>.<base_domain>**

/etc/hosts 文件中的集群域名授予对本地集群的 Web 控制台和监控界面的访问权限。您还可以使用 kubectl 或 oc。您可以使用指向 <application_floating_ip> 的额外条目来访问用户应用程序。此操作使 API 和应用程序可供您访问，不适合于生产部署，但允许对开发和测试进行安装。

5. **将 FIP 添加到 inventory.yaml 文件中，作为以下变量的值：**

- **os_api_fip**
- **os_bootstrap_fip**
-

- os_ingress_fip

如果使用这些值，还必须在 inventory.yaml 文件中输入外部网络作为 os_external_network 变量的值。

提示

您可以通过分配浮动 IP 地址并更新防火墙配置，使 OpenShift Container Platform 资源在集群外可用。

21.4.10.2. 完成没有浮动 IP 地址的安装

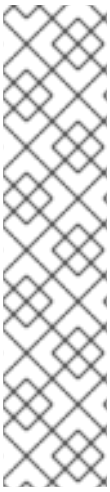
您可以在 Red Hat OpenStack Platform(RHOSP)上安装 OpenShift Container Platform，而无需提供浮动 IP 地址。

在 inventory.yaml 文件中，不要定义以下变量：

- os_api_fip
- os_bootstrap_fip
- os_ingress_fip

如果无法提供外部网络，也可以将 os_external_network 留空。如果没有提供 for os_external_network 值，则不会为您创建路由器，如果没有额外的操作，安装程序将无法从 Glance 检索镜像。之后在安装过程中，当您创建网络资源时，您必须自行配置外部连接。

如果您使用 wait-for 命令从因为缺少浮动 IP 地址或名称解析而无法访问集群 API 的系统中运行安装程序，安装会失败。在这些情况下，您可以使用代理网络或者从与机器位于相同网络的系统中运行安装程序。



注意

您可以通过为 API 和 Ingress 端口创建 DNS 记录来启用名称解析。例如：

```
api.<cluster_name>.<base_domain>. IN A <api_port_IP>
*.apps.<cluster_name>.<base_domain>. IN A <ingress_port_IP>
```

如果您不控制 DNS 服务器，您可以将记录添加到 /etc/hosts 文件中。此操作使 API 可供您自己访问，不适合于生产部署，而是允许安装以进行开发和测试。

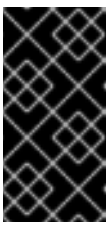
21.4.11. 为安装程序定义参数

OpenShift Container Platform 安装程序依赖于一个名为 clouds.yaml 的文件。该文件描述了 Red Hat OpenStack Platform(RHOSP)配置参数，包括项目名称、登录信息和授权服务 URL。

流程

1. 创建 clouds.yaml 文件：

- 如果您的 RHOSP 发行版包含 Horizon Web UI，请在该 UI 中生成 clouds.yaml 文件。



重要

记得在 auth 字段中添加密码。您还可以将 secret 保存在 clouds.yaml 以外的 一个独立的文件中。

- 如果您的 RHOSP 发行版不包含 Horizon Web UI，或者您不想使用 Horizon，请自行创建该文件。如需有关 clouds.yaml 的详细信息，请参阅 RHOSP 文档中的 配置文件。

```
clouds:
  shiftstack:
    auth:
      auth_url: http://10.10.14.42:5000/v3
      project_name: shiftstack
      username: <username>
      password: <password>
      user_domain_name: Default
      project_domain_name: Default
  dev-env:
```

```
    region_name: RegionOne
    auth:
      username: <username>
      password: <password>
      project_name: 'devonly'
      auth_url: 'https://10.10.14.22:5001/v2.0'
```

2. 如果您的 RHOSP 安装使用自签名证书颁发机构(CA)证书进行端点身份验证：

   a. 将证书颁发机构文件复制到您的机器中。

   b. 将 cacerts 键添加到 clouds.yaml 文件。该值必须是到 CA 证书的绝对、不可 root 访问的路径：

```
clouds:
  shiftstack:
    ...
    cacert: "/etc/pki/ca-trust/source/anchors/ca.crt.pem"
```

   提示

   使用自定义 CA 证书运行安装程序后，您可以通过编辑 cloud-provider-config keymap 中的 ca-cert.pem 键的值来更新证书。在命令行中运行：

```
$ oc edit configmap -n openshift-config cloud-provider-config
```

3. 将 clouds.yaml 文件放在以下位置之一：

   a. OS_CLIENT_CONFIG_FILE 环境变量的值

   b. 当前目录

   c. 特定于 Unix 的用户配置目录，如 ~/.config/openstack/clouds.yaml

   d. 特定于 Unix 的站点配置目录，如 /etc/openstack/clouds.yaml

安装程序会按顺序搜索 clouds.yaml。

21.4.12. 在 RHOSP 上创建网络资源

在您自己的基础架构的 Red Hat OpenStack Platform(RHOSP)安装上创建 OpenShift Container Platform 所需的网络资源。为节省时间，请运行提供的 Ansible playbook，以生成安全组、网络、子网、路由器和端口。

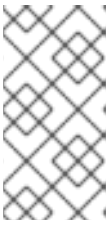
先决条件

- 下载了"下载 playbook 依赖项"中的模块。
- 下载了"下载安装 playbook"中的 playbook。

流程

1. 对于双堆栈集群部署，编辑 inventory.yaml 文件并取消注释 os_subnet6 属性。
2. 在命令行中运行以下命令创建网络资源：

```
$ ansible-playbook -i inventory.yaml network.yaml
```



注意

API 和 Ingress VIP 字段将在 inventory.yaml playbook 中覆盖，IP 地址分配给网络端口。

注意

network.yaml playbook 创建的资源由 down-network.yaml playbook 删除。

21.4.13. 创建安装配置文件

您可以自定义在 Red Hat OpenStack Platform(RHOSP)上安装的 OpenShift Container Platform 集群。

先决条件

- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。

流程

1. 创建 install-config.yaml 文件。

   a. 进入包含安装程序的目录并运行以下命令：

   ```
   $ ./openshift-install create install-config --dir <installation_directory> 1
   ```

   1 对于 <installation_directory>，请指定要存储安装程序创建的文件的目录名称。

   在指定目录时：

   - 验证该目录是否具有执行权限。在安装目录中运行 Terraform 二进制文件需要这个权限。
   - 使用空目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

   b. 在提示符处，提供云的配置详情：

      i. 可选：选择用于访问集群机器的 SSH 密钥。

      注意

      对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 ssh-agent 进程使用的 SSH 密钥。

      ..

ii. 选择 **openstack** 作为目标平台。

iii. 指定用于安装集群的 **Red Hat OpenStack Platform(RHOSP)**外部网络名称。

iv. 指定用于从外部访问 **OpenShift API** 的浮动 **IP** 地址。

v. 指定至少有 **16 GB RAM** 用于 **control plane** 节点，以及计算节点的 **8 GB RAM**。

vi. 选择集群要部署到的基域。所有 **DNS** 记录都将是这个基域的子域，并且还包括集群名称。

vii. 为集群输入一个名称。名称长度必须为 **14** 个或更少字符。

2. 修改 **install-config.yaml** 文件。您可以在"安装配置参数"部分找到有关可用参数的更多信息。

3. 备份 **install-config.yaml** 文件，以便您可以使用它安装多个集群。

重要

install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用该文件，您必须立即备份该文件。

现在，文件 **install-config.yaml** 位于您指定的目录中。

其他资源

- OpenStack 的安装配置参数

21.4.13.1. RHOSP 部署中的自定义子网

另外，您还可以在您选择的 **Red Hat OpenStack Platform(RHOSP)**子网中部署集群。子网的 **GUID** 作为 **install-config.yaml** 文件中的 **platform.openstack.machinesSubnet** 的值传递。

此子网被用作集群的主子网。默认情况下，其上会创建节点和端口。您可以通过将 platform.openstack.machinesSubnet 属性的值设置为子网的 UUID，在不同的 RHOSP 子网中创建节点和端口。

在使用自定义子网运行 OpenShift Container Platform 安装程序前，请验证您的配置是否满足以下要求：

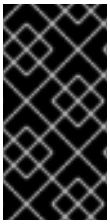
- platform.openstack.machinesSubnet 使用的子网启用了 DHCP。
- platform.openstack.machinesSubnet 的 CIDR 与 networking.machineNetwork 的 CIDR 匹配。
- 安装程序用户有在此网络上创建端口的权限，包括带有固定 IP 地址的端口。

使用自定义子网的集群有以下限制：

- 如果您计划安装使用浮动 IP 地址的集群，则必须将 platform.openstack.machinesSubnet 子网附加到连接到 externalNetwork 网络的路由器。
- 如果在 install-config.yaml 文件中设置了 platform.openstack.machinesSubnet 值，安装程序不会为您的 RHOSP 机器创建专用网络或子网。
- 您不能与自定义子网同时使用 platform.openstack.externalDNS 属性。要将 DNS 添加到使用自定义子网的集群，请在 RHOSP 网络上配置 DNS。

注意

默认情况下，API VIP 使用 x.x.x.5，Ingress VIP 从网络 CIDR 块获取 x.x.x.7。要覆盖这些默认值，请为 DHCP 分配池之外的 platform.openstack.apiVIPs 和 platform.openstack.ingressVIPs 设置值。

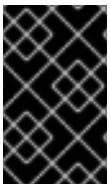


重要

集群安装后无法调整网络的 CIDR 范围。红帽不提供有关在集群安装过程中确定范围的直接指导，因为它需要仔细考虑每个命名空间创建的 pod 数量。

21.4.13.2. RHOSP 的自定义 install-config.yaml 文件示例

以下示例 install-config.yaml 文件演示了所有可能的 Red Hat OpenStack Platform (RHOSP) 自定义选项。



重要

此示例文件仅供参考。您必须使用安装程序来获取 install-config.yaml 文件。

例 21.4. 单个堆栈 install-config.yaml 文件示例

```
apiVersion: v1
baseDomain: example.com
controlPlane:
  name: master
  platform: {}
  replicas: 3
compute:
- name: worker
  platform:
    openstack:
      type: ml.large
  replicas: 3
metadata:
  name: example
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  serviceNetwork:
  - 172.30.0.0/16
  networkType: OVNKubernetes
platform:
  openstack:
    cloud: mycloud
    externalNetwork: external
    computeFlavor: m1.xlarge
    apiFloatingIP: 128.0.0.1
fips: false
pullSecret: '{"auths": ...}'
sshKey: ssh-ed25519 AAAA...
```

例 21.5. 双堆栈 install-config.yaml 文件示例

```
apiVersion: v1
baseDomain: example.com
controlPlane:
  name: master
  platform: {}
  replicas: 3
compute:
- name: worker
  platform:
    openstack:
      type: ml.large
  replicas: 3
metadata:
  name: example
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  - cidr: fd01::/48
    hostPrefix: 64
  machineNetwork:
  - cidr: 192.168.25.0/24
  - cidr: fd2e:6f44:5dd8:c956::/64
  serviceNetwork:
  - 172.30.0.0/16
  - fd02::/112
  networkType: OVNKubernetes
platform:
  openstack:
    cloud: mycloud
    externalNetwork: external
    computeFlavor: m1.xlarge
    apiVIPs:
    - 192.168.25.10
    - fd2e:6f44:5dd8:c956:f816:3eff:fec3:5955
    ingressVIPs:
    - 192.168.25.132
    - fd2e:6f44:5dd8:c956:f816:3eff:fe40:aecb
    controlPlanePort:
      fixedIPs:
      - subnet:
          name: openshift-dual4
      - subnet:
          name: openshift-dual6
      network:
        name: openshift-dual
fips: false
pullSecret: '{"auths": ...}'
sshKey: ssh-ed25519 AAAA...
```

21.4.13.3. 为机器设置自定义子网

安装程序默认使用的 IP 范围可能与您在安装 OpenShift Container Platform 时创建的 Neutron 子网不匹配。如有必要，通过编辑安装配置文件来更新新机器的 CIDR 值。

先决条件

- 有 OpenShift Container Platform 安装程序生成的 install-config.yaml 文件。
- 已安装 Python 3。

流程

1. 在命令行中，浏览包含 install-config.yaml 和 inventory.yaml 文件的目录。
2. 在该目录中，运行脚本来编辑 install-config.yaml 文件或手动更新该文件：
   - 要使用脚本设置值，请运行以下命令：

```
$ python -c 'import yaml
path = "install-config.yaml"
data = yaml.safe_load(open(path))
inventory = yaml.safe_load(open("inventory.yaml"))["all"]["hosts"]["localhost"]
machine_net = [{"cidr": inventory["os_subnet_range"]}]
api_vips = [inventory["os_apiVIP"]]
ingress_vips = [inventory["os_ingressVIP"]]
ctrl_plane_port = {"network": {"name": inventory["os_network"]}, "fixedIPs":
[{"subnet": {"name": inventory["os_subnet"]}}]}
if inventory.get("os_subnet6"): (1)
    machine_net.append({"cidr": inventory["os_subnet6_range"]})
    api_vips.append(inventory["os_apiVIP6"])
    ingress_vips.append(inventory["os_ingressVIP6"])
    data["networking"]["networkType"] = "OVNKubernetes"
    data["networking"]["clusterNetwork"].append({"cidr":
inventory["cluster_network6_cidr"], "hostPrefix":
inventory["cluster_network6_prefix"]})
    data["networking"]
["serviceNetwork"].append(inventory["service_subnet6_range"])
    ctrl_plane_port["fixedIPs"].append({"subnet": {"name":
inventory["os_subnet6"]}})
data["networking"]["machineNetwork"] = machine_net
data["platform"]["openstack"]["apiVIPs"] = api_vips
data["platform"]["openstack"]["ingressVIPs"] = ingress_vips
```

```
data["platform"]["openstack"]["controlPlanePort"] = ctrl_plane_port
del data["platform"]["openstack"]["externalDNS"]
open(path, "w").write(yaml.dump(data, default_flow_style=False))'
```

1 适用于双栈 (IPv4/IPv6) 环境。

21.4.13.4. 清空计算机器池

要继续使用您自己的基础架构的安装，请将安装配置文件中的计算机器数量设置为零。之后，您可以手动创建这些机器。

先决条件

- 有 OpenShift Container Platform 安装程序生成的 install-config.yaml 文件。

流程

1. 在命令行中，浏览包含 install-config.yaml 的目录。
2. 在该目录中，运行脚本来编辑 install-config.yaml 文件或手动更新该文件：
   - 要使用脚本设置值，请运行：

```
$ python -c '
import yaml;
path = "install-config.yaml";
data = yaml.safe_load(open(path));
data["compute"][0]["replicas"] = 0;
open(path, "w").write(yaml.dump(data, default_flow_style=False))'
```

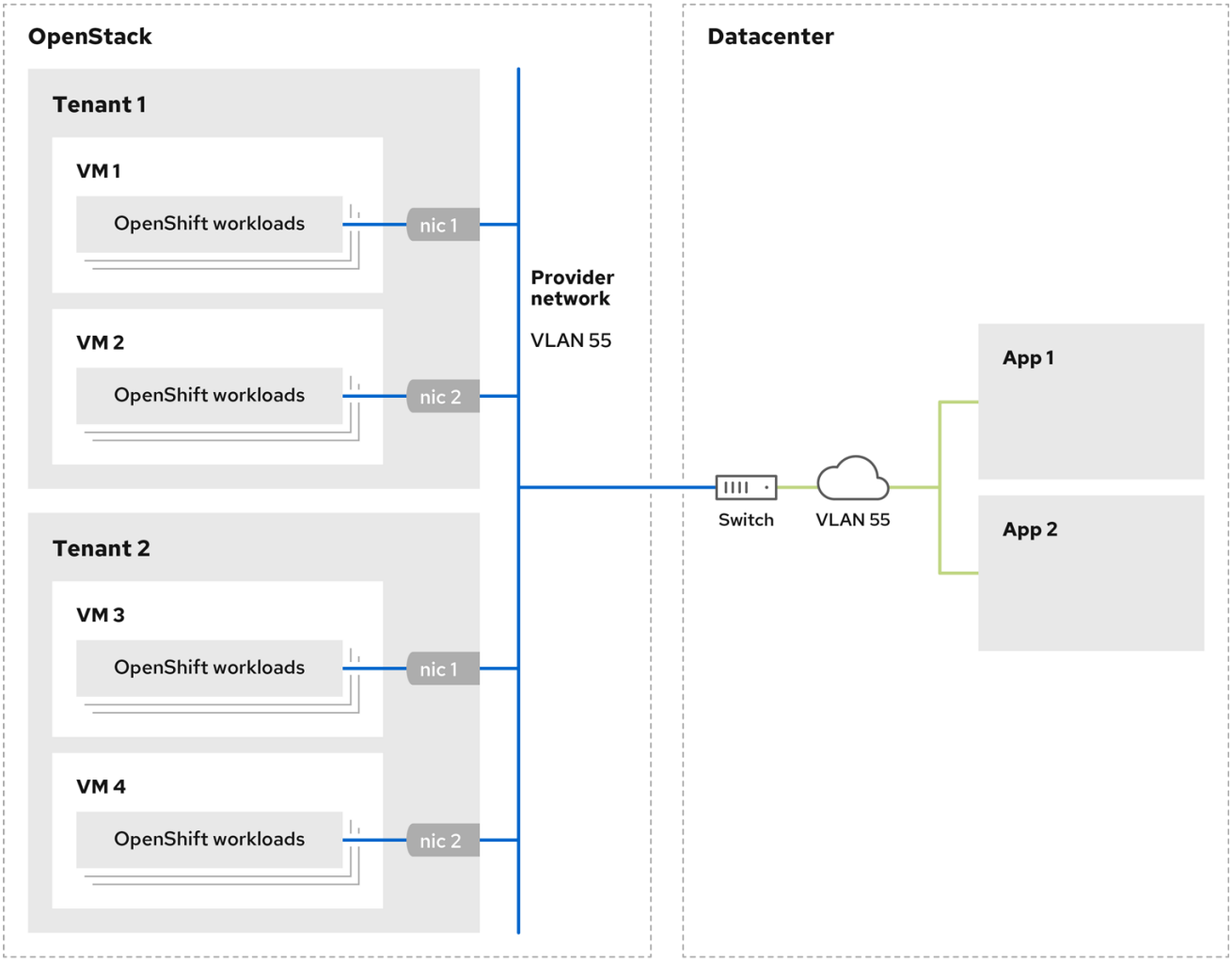
   - 要手动设置值，打开该文件并将 compute.<first entry>.replicas 的值设置为 0。

21.4.13.5. RHOSP 提供商网络上的集群部署

您可以使用供应商网络上的主网络接口在 Red Hat OpenStack Platform(RHOSP)上部署 OpenShift Container Platform 集群。提供商网络通常用于为项目提供可用于访问互联网的公共网络的直接访问权限。您还可以在项目间共享提供商网络，作为网络创建流程的一部分。

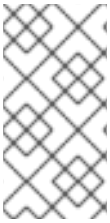
RHOSP 提供商网络直接映射到数据中心内的现有物理网络。RHOSP 管理员必须创建它们。

在以下示例中，OpenShift Container Platform 工作负载使用提供商网络连接到数据中心：



在提供商网络上安装的 OpenShift Container Platform 集群不需要租户网络或浮动 IP 地址。安装程序不会在安装过程中创建这些资源。

提供商网络类型示例包括 flat（未标记）和 VLAN（802.1Q 标记）。



注意

集群可以在网络类型允许的情况下支持任意数量的提供商网络连接。例如，VLAN 网络通常支持多达 4096 个连接。

您可以在 RHOSP 文档中的 了解更多有关供应商和租户网络的信息。

21.4.13.5.1. 集群安装的 RHOSP 提供商网络要求

在安装 OpenShift Container Platform 集群前，您的 Red Hat OpenStack Platform(RHOSP)部署和提供商网络必须满足以下多个条件：

- RHOSP 网络服务(Neutron)通过 RHOSP 网络 API 启用 并访问。
- RHOSP 网络服务 启用了端口安全性并允许地址对扩展。
- 提供商网络可以与其他租户共享。

提示

使用 openstack network create 命令和 --share 标志来创建可共享的网络。

- 用于安装集群的 RHOSP 项目必须拥有提供商网络以及适当的子网。

提示

要为名为"openshift"的项目创建网络，请输入以下命令

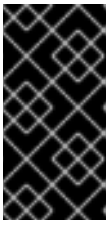
```
$ openstack network create --project openshift
```

要为名为"openshift"的项目创建子网，请输入以下命令

```
$ openstack subnet create --project openshift
```

要了解更多有关在 RHOSP 上创建网络的信息，请阅读 提供商网络文档。

如果集群归 admin 用户所有，则必须以该用户身份运行安装程序，以便在网络上创建端口。



重要

提供商网络必须由用于创建集群的 RHOSP 项目所有。如果没有，则 RHOSP Compute 服务(Nova)无法从该网络请求端口。

- 验证提供商网络可以访问 RHOSP 元数据服务 IP 地址，默认为 169.254.169.254。

  根据 RHOSP SDN 和网络服务配置，您可能需要在创建子网时提供路由。例如：

```
$ openstack subnet create --dhcp --host-route
destination=169.254.169.254/32,gateway=192.0.2.2 ...
```

- 可选：要保护网络，请创建 基于角色的访问控制(RBAC) 规则，以限制对单个项目的网络访问。

21.4.13.5.2. 在提供商网络上部署具有主接口的集群

您可以在 Red Hat OpenStack Platform(RHOSP) 提供商网络上部署具有主网络接口的 OpenShift Container Platform 集群。

先决条件

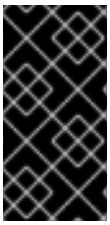
- 您的 Red Hat OpenStack Platform(RHOSP)部署被配置为"RHOSP 供应商网络要求用于集群安装"。

流程

1. 在文本编辑器中，打开 install-config.yaml 文件。
2. 将 platform.openstack.apiVIPs 属性的值设置为 API VIP 的 IP 地址。
3. 将 platform.openstack.ingressVIPs 属性的值设置为 Ingress VIP 的 IP 地址。
4. 将 platform.openstack.machinesSubnet 属性的值设置为提供商网络子网的 UUID。

-

5. 将 networking.machineNetwork.cidr 属性的值设置为提供商网络子网的 CIDR 块。



重要

platform.openstack.apiVIPs 和 platform.openstack.ingressVIPs 属性必须从 networking.machineNetwork.cidr 块中取消分配 IP 地址。

依赖于 RHOSP 提供商网络的集群的安装配置文件部分

```
...
platform:
  openstack:
    apiVIPs: 1
      - 192.0.2.13
    ingressVIPs: 2
      - 192.0.2.23
    machinesSubnet: fa806b2f-ac49-4bce-b9db-124bc64209bf
    # ...
networking:
  machineNetwork:
  - cidr: 192.0.2.0/24
```

1 2 在 OpenShift Container Platform 4.12 及更新的版本中，apiVIP 和 ingressVIP 配置设置已弃用。反之，使用列表格式在 apiVIPs 和 ingressVIPs 配置设置中输入值。

警告

您不能在将提供商网络用于主网络接口时设置 platform.openstack.externalNetwork 或 platform.openstack.externalDNS 参数。

在部署集群时，安装程序使用 install-config.yaml 文件在提供商网络上部署集群。

提示

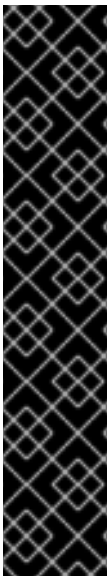
您可以将额外的网络（包括提供商网络）添加到 platform.openstack.additionalNetworkIDs 列表中。

部署集群后，您可以将 pod 附加到额外网络。如需更多信息，请参阅 了解多个网络。

21.4.14. 创建 Kubernetes 清单和 Ignition 配置文件

由于您必须修改一些集群定义文件并手动启动集群机器，因此您必须生成 Kubernetes 清单和 Ignition 配置文件来配置机器。

安装配置文件转换为 Kubernetes 清单。清单嵌套到 Ignition 配置文件中，稍后用于配置集群机器。



重要

- OpenShift Container Platform 安装程序生成的 Ignition 配置文件包含 24 小时后过期的证书，然后在该时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 node-bootstrapper 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。
- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

先决条件

- 已获得 OpenShift Container Platform 安装程序。
- 已创建 install-config.yaml 安装配置文件。

流程

1. 进入包含 OpenShift Container Platform 安装程序的目录，并为集群生成 Kubernetes 清单：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 对于 <installation_directory>，请指定包含您创建的 install-config.yaml 文件的安装目录。

2. 删除定义 control plane 机器的 Kubernetes 清单文件、计算机器集和 control plane 机器集：

```
$ rm -f openshift/99_openshift-cluster-api_master-machines-*.yaml
openshift/99_openshift-cluster-api_worker-machineset-*.yaml openshift/99_openshift-
machine-api_master-control-plane-machine-set.yaml
```

由于您要自行创建和管理这些资源，因此不必初始化这些资源。

- 您可以使用机器 API 来保留计算机器集文件来创建计算机器，但您必须更新对它们的引用以匹配您的环境。

3. 检查 <installation_directory>/manifests/cluster-scheduler-02-config.yml Kubernetes 清单文件中的 mastersSchedulable 参数是否已设置为 false。此设置可防止在 control plane 机器上调度 pod：

   a. 打开 <installation_directory>/manifests/cluster-scheduler-02-config.yml 文件。

   b. 找到 mastersSchedulable 参数，并确保它被设置为 false。

   c. 保存并退出 文件。

4. 要创建 Ignition 配置文件，请从包含安装程序的目录运行以下命令：

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

1

对于 <installation_directory>，请指定相同的安装目录。

为安装目录中的 bootstrap、control plane 和计算节点创建 Ignition 配置文件。kubeadmin-password 和 kubeconfig 文件在 ./<installation_directory>/auth 目录中创建：

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

5. 将元数据文件的 infraID 密钥导出为环境变量：

```
$ export INFRA_ID=$(jq -r .infraID metadata.json)
```

提示

从 metadata.json 中提取 infraID 密钥，并将它用作您创建的所有 RHOSP 资源的前缀。通过这样做，您可以避免在同一项目中进行多个部署时的名称冲突。

21.4.15. 准备 bootstrap Ignition 文件

OpenShift Container Platform 安装过程依赖于从 bootstrap Ignition 配置文件创建的 bootstrap 机器。

编辑 文件并上传该文件。然后，创建一个 Red Hat OpenStack Platform(RHOSP)用来下载主文件的辅助 bootstrap Ignition 配置文件。

先决条件

- 您有安装程序生成的 bootstrap Ignition 文件，即 bootstrap.ign。

- 安装程序元数据文件中的基础架构 ID 被设置为环境变量($INFRA_ID)。
  - 如果没有设置变量，请参阅 创建 Kubernetes 清单和 Ignition 配置文件。
- 您可以使用 HTTP(S)来存储 bootstrap Ignition 文件。
  - 所记录的步骤使用 RHOSP 镜像服务(Glance)，但也可以使用 RHOSP 存储服务(Swift)、Amazon S3、内部 HTTP 服务器或临时 Nova 服务器。

流程

1. 运行以下 Python 脚本：该脚本修改 bootstrap Ignition 文件，以设置主机名，并在运行时设置 CA 证书文件：

```
import base64
import json
import os

with open('bootstrap.ign', 'r') as f:
    ignition = json.load(f)

files = ignition['storage'].get('files', [])

infra_id = os.environ.get('INFRA_ID', 'openshift').encode()
hostname_b64 = base64.standard_b64encode(infra_id + b'-
bootstrap\n').decode().strip()
files.append(
{
    'path': '/etc/hostname',
    'mode': 420,
    'contents': {
        'source': 'data:text/plain;charset=utf-8;base64,' + hostname_b64
    }
})

ca_cert_path = os.environ.get('OS_CACERT', '')
if ca_cert_path:
    with open(ca_cert_path, 'r') as f:
        ca_cert = f.read().encode()
        ca_cert_b64 = base64.standard_b64encode(ca_cert).decode().strip()

    files.append(
    {
        'path': '/opt/openshift/tls/cloud-ca-cert.pem',
        'mode': 420,
        'contents': {
```

```
            'source': 'data:text/plain;charset=utf-8;base64,' + ca_cert_b64
        }
    })

ignition['storage']['files'] = files;

with open('bootstrap.ign', 'w') as f:
    json.dump(ignition, f)
```

2. 使用 RHOSP CLI，创建使用 bootstrap Ignition 文件的镜像：

```
$ openstack image create --disk-format=raw --container-format=bare --file
bootstrap.ign <image_name>
```

3. 获取镜像的详情：

```
$ openstack image show <image_name>
```

记录 file 值；它遵循 v2/images/<image_ID>/file 模式。

注意

验证您创建的镜像是否活跃。

4. 检索镜像服务的公共地址：

```
$ openstack catalog show image
```

5. 将公共地址与 镜像文件 值组合，并在存储位置保存结果。位置遵循 <image_service_public_URL>/v2/images/<image_ID>/file 模式。

6. 生成身份验证令牌并保存令牌 ID：

```
$ openstack token issue -c id -f value
```

7. 将以下内容插入到名为 $INFRA_ID-bootstrap-ignition.json 的文件中，并编辑占位符以匹配您自己的值：

```
{
  "ignition": {
    "config": {
      "merge": [{
        "source": "<storage_url>", 1
        "httpHeaders": [{
          "name": "X-Auth-Token", 2
          "value": "<token_ID>" 3
        }]
      }]
    },
    "security": {
      "tls": {
        "certificateAuthorities": [{
          "source": "data:text/plain;charset=utf-8;base64,<base64_encoded_certificate>"
4
        }]
      }
    },
    "version": "3.2.0"
  }
}
```

1 将 ignition.config.merge.source 值替换为 bootstrap Ignition 文件存储 URL。

2 在 httpHeaders 中将 name 设置为 "X-Auth-Token"。

3 在 httpHeaders 中将 value 设置为您的令牌 ID。

4 如果 bootstrap Ignition 文件服务器使用自签名证书，请包含 base64 编码的证书。

8. 保存辅助 Ignition 配置文件。

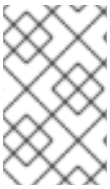
bootstrap Ignition 数据将在安装过程中传递给 RHOSP。

警告

bootstrap Ignition 文件包含敏感信息，如 clouds.yaml 凭证。确保将其保存在安全的地方，并在完成安装后将其删除。

21.4.16. 在 RHOSP 上创建 control plane Ignition 配置文件

在您自己的基础架构的 Red Hat OpenStack Platform(RHOSP)上安装 OpenShift Container Platform 需要 control plane Ignition 配置文件。您必须创建多个配置文件。



注意

与 bootstrap Ignition 配置一样，您必须明确为每个 control plane 机器定义主机名。

先决条件

- 来自安装程序元数据文件中的基础架构 ID 被设置为环境变量($INFRA_ID)。
  - 如果没有设置变量，请参阅"创建 Kubernetes 清单和 Ignition 配置文件"。

流程

- 在命令行中运行以下 Python 脚本：

```
$ for index in $(seq 0 2); do
    MASTER_HOSTNAME="$INFRA_ID-master-$index\n"
    python -c "import base64, json, sys;
ignition = json.load(sys.stdin);
storage = ignition.get('storage', {});
files = storage.get('files', []);
files.append({'path': '/etc/hostname', 'mode': 420, 'contents': {'source':
'data:text/plain;charset=utf-8;base64,' +
base64.standard_b64encode(b'$MASTER_HOSTNAME').decode().strip(), 'verification':
{}}, 'filesystem': 'root'});
storage['files'] = files;
ignition['storage'] = storage
```

```
json.dump(ignition, sys.stdout)" <master.ign >"$INFRA_ID-master-$index-
ignition.json"
done
```

您现在有三个 control plane Ignition 文件：& lt;INFRA_ID>-master-0-ignition.json、<INFRA_ID>-master-1-ignition.json 和 <INFRA_ID>-master-2-ignition.json。

21.4.17. 更新 RHOSP 上的网络资源

更新您自己的基础架构的 Red Hat OpenStack Platform (RHOSP) 安装的 OpenShift Container Platform 所需的网络资源。

先决条件

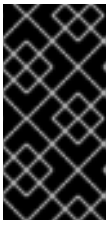
- Python 3 已安装在您的机器上。
- 下载了"下载 playbook 依赖项"中的模块。
- 下载了"下载安装 playbook"中的 playbook。

流程

1. 可选：在 inventory.yaml playbook 中添加外部网络值：

inventory.yaml Ansible Playbook 中的外部网络值示例

```
...
    # The public network providing connectivity to the cluster. If not
    # provided, the cluster external connectivity must be provided in another
    # way.

    # Required for os_api_fip, os_ingress_fip, os_bootstrap_fip.
    os_external_network: 'external'
...
```



重要

如果您没有在 inventory.yaml 文件中提供 for os_external_network 值，您必须确保虚拟机可以自行访问 Glance 和外部连接。

2. 可选：将外部网络和浮动 IP(FIP)地址值添加到 inventory.yaml playbook 中：

inventory.yaml Ansible Playbook 中的 FIP 值示例

```
...
# OpenShift API floating IP address. If this value is non-empty, the
# corresponding floating IP will be attached to the Control Plane to
# serve the OpenShift API.
os_api_fip: '203.0.113.23'

# OpenShift Ingress floating IP address. If this value is non-empty, the
# corresponding floating IP will be attached to the worker nodes to serve
# the applications.
os_ingress_fip: '203.0.113.19'

# If this value is non-empty, the corresponding floating IP will be
# attached to the bootstrap machine. This is needed for collecting logs
# in case of install failure.
os_bootstrap_fip: '203.0.113.20'
```

重要

如果没有为 os_api_fip 和 os_ingress_fip 定义值，则必须执行安装后网络配置。

如果没有为 os_bootstrap_fip 定义值，安装程序无法从失败的安装中下载调试信息。

如需更多信息，请参阅"启用对环境的访问"。

3. 在命令行中，通过运行 security-groups.yaml playbook 来创建安全组：

```
$ ansible-playbook -i inventory.yaml security-groups.yaml
```

4. 在命令行中，运行 update-network-resources.yaml playbook 更新网络资源：

```
$ ansible-playbook -i inventory.yaml update-network-resources.yaml 1
```

1 此 playbook 将添加标签到网络、子网、端口和路由器。它还将浮动 IP 地址附加到 API 和 Ingress 端口，并为这些端口设置安全组。

5. 可选：如果要控制 Nova 服务器使用的默认解析程序，请运行 RHOSP CLI 命令：

```
$ openstack subnet set --dns-nameserver <server_1> --dns-nameserver <server_2>
"$INFRA_ID-nodes"
```

6. 可选： 您可以使用您创建的 inventory.yaml 文件来自定义安装。例如，您可以部署使用裸机的集群。

21.4.17.1. 使用裸机部署集群

如果您希望集群使用裸机，请修改 inventory.yaml 文件。集群可以同时在裸机上运行 control plane 和计算机器，或者只在计算机器上运行。

注意

确保 install-config.yaml 文件反映了您用于裸机 worker 的 RHOSP 网络是否支持浮动 IP 地址。

先决条件

- RHOSP Bare Metal 服务(Ironic) 通过 RHOSP Compute API 启用并访问。
- 裸机 可作为 RHOSP 类别 提供。
- 如果您的集群在一个大于 16.1.6 且小于 16.2.4 的 RHOSP 版本上运行，则裸机 worker 无法正常工作，因为存在一个已知问题会导致元数据服务对 OpenShift Container Platform 节点上

的服务不可用。

- RHOSP 网络支持 VM 和裸机服务器附加。
- 您的网络配置不依赖于供应商网络。不支持提供商网络。
- 如果要将机器部署到预先存在的网络中，则会置备 RHOSP 子网。
- 如果要在安装程序置备的网络中部署机器，RHOSP Bare Metal 服务(Ironic)可以侦听在租户网络上运行的 Preboot eXecution Environment(PXE)引导机器并与之交互。
- 作为 OpenShift Container Platform 安装过程的一部分，创建了 inventory.yaml 文件。

流程

1. 在 inventory.yaml 文件中，编辑机器的类别：
   a. 如果要使用裸机 control plane 机器，请将 os_flavor_master 的值改为裸机类型。
   b. 将值 of os_flavor_worker 更改为裸机类型。

裸机 inventory.yaml 文件示例

```
all:
  hosts:
    localhost:
      ansible_connection: local
      ansible_python_interpreter: "{{ansible_playbook_python}}"

      # User-provided values
      os_subnet_range: '10.0.0.0/16'
      os_flavor_master: 'my-bare-metal-flavor' 1
      os_flavor_worker: 'my-bare-metal-flavor' 2
      os_image_rhcos: 'rhcos'
      os_external_network: 'external'
...
```

1 如果要使用裸机 control plane 机器，请将这个值改为一个裸机类型。

2 将此值更改为用于计算机器的裸机类别。

使用更新的 inventory.yaml 文件完成安装过程。部署期间创建的机器使用添加到该文件中的类别。

注意

在等待裸机引导时，安装程序可能会超时。

如果安装程序超时，重启并使用安装程序的 wait-for 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

21.4.18. 在 RHOSP 上创建 bootstrap 机器

创建 bootstrap 机器，为其提供在 Red Hat OpenStack Platform(RHOSP)上运行所需的网络访问权限。红帽提供了一个 Ansible playbook，您可以运行它来简化此过程。

先决条件

- 下载了"下载 playbook 依赖项"中的模块。
- 下载了"下载安装 playbook"中的 playbook。
- inventory.yaml、common.yaml 和 bootstrap.yaml Ansible playbook 位于一个通用目录中。

- 安装程序创建的 metadata.json 文件与 Ansible playbook 位于同一个目录中。

流程

1. 在命令行中，将工作目录更改为 playbook 的位置。
2. 在命令行中运行 bootstrap.yaml playbook：

```
$ ansible-playbook -i inventory.yaml bootstrap.yaml
```

3. bootstrap 服务器活跃后，查看日志以验证是否收到 Ignition 文件：

```
$ openstack console log show "$INFRA_ID-bootstrap"
```

21.4.19. 在 RHOSP 上创建 control plane 机器

使用您生成的 Ignition 配置文件创建三台 control plane 机器。红帽提供了一个 Ansible playbook，您可以运行它来简化此过程。

先决条件

- 下载了"下载 playbook 依赖项"中的模块。
- 下载了"下载安装 playbook"中的 playbook。
- 来自安装程序元数据文件中的基础架构 ID 被设置为环境变量($INFRA_ID)。
- inventory.yaml、common.yaml 和 control-plane.yaml Ansible playbook 位于一个通用目录中。
- 您有三个在"Creating control plane Ignition 配置文件"中创建的 Ignition 文件。

流程

1. 在命令行中，将工作目录更改为 playbook 的位置。

2. 如果 control plane Ignition 配置文件尚未位于工作目录中，请将它们复制到其中。

3. 在命令行中运行 control-plane.yaml playbook：

```
$ ansible-playbook -i inventory.yaml control-plane.yaml
```

4. 运行以下命令来监控 bootstrap 过程：

```
$ openshift-install wait-for bootstrap-complete
```

您将看到确认 control plane 机器正在运行并加入集群的信息：

```
INFO API v1.29.4 up
INFO Waiting up to 30m0s for bootstrapping to complete...
...
INFO It is now safe to remove the bootstrap resources
```

21.4.20. 使用 CLI 登录集群

您可以通过导出集群 kubeconfig 文件，以默认系统用户身份登录集群。kubeconfig 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。
- 已安装 oc CLI。

流程

1. 导出 kubeadmin 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 <installation_directory>，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 oc 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

21.4.21. 从 RHOSP 删除 bootstrap 资源

删除您不再需要的 bootstrap 资源。

先决条件

- 下载了"下载 playbook 依赖项"中的模块。
- 下载了"下载安装 playbook"中的 playbook。
- inventory.yaml、common.yaml 和 down-bootstrap.yaml Ansible playbook 位于一个通用目录中。
- control plane 机器正在运行。
  - 如果您不知道机器的状态，请参阅"验证集群状态"。

流程

1. 在命令行中，将工作目录更改为 playbook 的位置。

2. 在命令行中运行 down-bootstrap.yaml playbook：

```
$ ansible-playbook -i inventory.yaml down-bootstrap.yaml
```

bootstrap 端口、服务器和浮动 IP 地址会被删除。

警告

如果您之前没有禁用 bootstrap Ignition 文件 URL，现在需要禁用。

21.4.22. 在 RHOSP 上创建计算机器

启动 control plane 后，创建计算机器。红帽提供了一个 Ansible playbook，您可以运行它来简化此过程。

先决条件

- 下载了"下载 playbook 依赖项"中的模块。
- 下载了"下载安装 playbook"中的 playbook。
- inventory.yaml、common.yaml 和 compute-nodes.yaml Ansible playbook 位于一个通用目录中。
- 安装程序创建的 metadata.json 文件与 Ansible playbook 位于同一个目录中。
- control plane 处于活跃状态。

流程

流程

1. 在命令行中，将工作目录更改为 playbook 的位置。

2. 在命令行中运行 playbook:

```
$ ansible-playbook -i inventory.yaml compute-nodes.yaml
```

后续步骤

- 批准机器的证书签名请求。

21.4.23. 批准机器的证书签名请求

当您将机器添加到集群时，会为您添加的每台机器生成两个待处理证书签名请求(CSR)。您必须确认这些 CSR 已获得批准，或根据需要自行批准。必须首先批准客户端请求，然后批准服务器请求。

先决条件

- 您已将机器添加到集群中。

流程

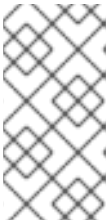
1. 确认集群可以识别这些机器：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0  Ready    master  63m  v1.29.4
master-1  Ready    master  63m  v1.29.4
master-2  Ready    master  64m  v1.29.4
```

输出中列出了您创建的所有机器。



注意

在有些 CSR 被批准前，前面的输出可能不包括计算节点（也称为 worker 节点）。

2. 检查待处理的 CSR，并确保添加到集群中的每台机器都有 Pending 或 Approved 状态的客户端请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE    REQUESTOR                                                   CONDITION
csr-8b2br   15m    system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
csr-8vnps   15m    system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
...
```

在本例中，两台机器加入集群。您可能会在列表中看到更多已批准的 CSR。

3. 如果 CSR 没有获得批准，在您添加的机器的所有待处理 CSR 都处于 Pending 状态后，请批准集群机器的 CSR：

注意

由于 CSR 会自动轮转，因此请在将机器添加到集群后一小时内批准您的 CSR。如果没有在一小时内批准它们，证书将会轮转，每个节点会存在多个证书。您必须批准所有这些证书。批准客户端 CSR 后，Kubelet 为服务证书创建一个二级 CSR，这需要手动批准。然后，如果 Kubelet 请求具有相同参数的新证书，则后续提供证书续订请求由 machine-approver 自动批准。

注意

对于在未启用机器 API 的平台上运行的集群，如裸机和其他用户置备的基础架构，您必须实施一种方法来自动批准 kubelet 提供证书请求(CSR)。如果没有批准请求，则 oc exec、ocrsh 和 oc logs 命令将无法成功，因为 API 服务器连接到 kubelet 时需要服务证书。与 Kubelet 端点联系的任何操作都需要此证书批准。该方法必须监视新的 CSR，确认 CSR 由 system: node 或 system:admin 组中的 node-bootstrapper 服务帐户提交，并确认节点的身份。

- 要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 <csr_name> 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}
{{"\n"}}{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```

注意

在有些 CSR 被批准前，一些 Operator 可能无法使用。

4. 现在，您的客户端请求已被批准，您必须查看添加到集群中的每台机器的服务器请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE    REQUESTOR                                                   CONDITION
csr-bfd72   5m26s  system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv   5m26s  system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

5. 如果剩余的 CSR 没有被批准，且处于 Pending 状态，请批准集群机器的 CSR：

- 要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 <csr_name> 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}
{{"\n"}}{{end}}{{end}}' | xargs oc adm certificate approve
```

6. 批准所有客户端和服务器 CSR 后，机器将处于 Ready 状态。运行以下命令验证：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0  Ready    master  73m  v1.29.4
master-1  Ready    master  73m  v1.29.4
master-2  Ready    master  74m  v1.29.4
worker-0  Ready    worker  11m  v1.29.4
worker-1  Ready    worker  11m  v1.29.4
```

注意

批准服务器 CSR 后可能需要几分钟时间让机器过渡到 Ready 状态。

其他信息

- 如需有关 CSR 的更多信息，请参阅 证书签名请求。

21.4.24. 验证安装是否成功

验证 OpenShift Container Platform 安装已经完成。

先决条件

- 有安装程序(openshift-install)

流程

- 在命令行中输入：

```
$ openshift-install --log-level debug wait-for install-complete
```

程序输出控制台 URL，以及管理员的登录信息。

21.4.25. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于 远程健康监控

21.4.26. 后续步骤

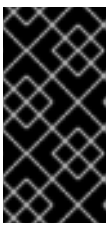
- 自定义集群。
- 如果需要，您可以选择 不使用远程健康报告。
- 如果您需要启用对节点端口的外部访问，请使用节点端口配置集群流量。
- 如果您没有将 RHOSP 配置为接受通过浮动 IP 地址的应用程序流量，请使用 浮动 IP 地址配置 RHOSP 访问。

21.5. 在受限网络中的 OPENSTACK 上安装集群

在 OpenShift Container Platform 4.16 中，您可以通过创建安装发行内容的内部镜像在受限网络中的 Red Hat OpenStack Platform（RHOSP）上安装集群。

21.5.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读有关 选择集群安装方法的文档，并为用户准备它。
- 使用OpenShift 集群支持的平台部分验证 OpenShift Container Platform 4.16 是否与您的 RHOSP 版本兼容。https://docs.redhat.com/en/documentation/openshift_container_platform/4.16/html-single/architecture/#supported-platforms-for-openshift-clusters_architecture-installation您还可以通过查看 RHOSP 上的 OpenShift Container Platform 支持来比较不同版本的平台支持。
- 您在镜像主机上创建 registry，并获取您的 OpenShift Container Platform 版本的 imageContentSources 数据。



重要

由于安装介质位于镜像主机上，因此您可以使用该计算机完成所有安装步骤。

- 您可以了解集群扩展、control plane 大小和 etcd 的性能和可扩展性实践。如需更多信息，请参阅 扩展集群的建议实践。

- 您已在 RHOSP 中启用了元数据服务。

21.5.2. 关于在受限网络中安装

在 OpenShift Container Platform 4.16 中，可以执行不需要有效的互联网连接来获取软件组件的安装。受限网络安装可以使用安装程序置备的基础架构或用户置备的基础架构完成，具体取决于您要安装集群的云平台。

如果您选择在云平台中执行受限网络安装，您仍需要访问其云 API。有些云功能，比如 Amazon Web Service 的 Route 53 DNS 和 IAM 服务，需要访问互联网。根据您的网络，在裸机硬件、Nutanix 或 VMware vSphere 上安装可能需要较少的互联网访问。

要完成受限网络安装，您必须创建一个 registry，以镜像 OpenShift 镜像 registry 的内容并包含安装介质。您可以在镜像主机上创建此 registry，该主机可同时访问互联网和您的封闭网络，也可以使用满足您的限制条件的其他方法。

21.5.2.1. 其他限制

受限网络中的集群有以下额外限制和限制：

- ClusterVersion 状态包含一个 Unable to retrieve available updates 错误。

- 默认情况下，您无法使用 Developer Catalog 的内容，因为您无法访问所需的镜像流标签。

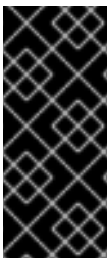
21.5.3. 在 RHOSP 上安装 OpenShift Container Platform 的资源指南

为支持 OpenShift Container Platform 安装，您的 Red Hat OpenStack Platform(RHOSP)配额必须满足以下要求：

表 21.5. RHOSP 上默认 OpenShift Container Platform 集群的建议资源

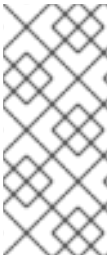
resource	value
浮动 IP 地址	3
端口	15
路由器	1
子网	1
RAM	88 GB
VCPU	22
卷存储	275 GB
实例	7
安全组	3
安全组规则	60
服务器组	2 - 每个机器池中每个额外可用区加 1

集群或许能以少于推荐资源运行，但其性能无法保证。



重要

如果 RHOSP 对象存储(Swift)可用，并由具有 swiftoperator 角色的用户帐户执行，它将用作 OpenShift Container Platform 镜像 registry 的默认后端。在这种情况下，卷存储需要 175 GB。根据镜像 registry 的大小，Swift 空间要求会有所不同。



注意

默认情况下，您的安全组和安全组规则配额可能较低。如果遇到问题，请以管理员 身份运行 openstack quota set --secgroups 3 --secgroup-rules 60 <project> 来提高配额。

OpenShift Container Platform 部署包含 control plane 机器、计算机器和 bootstrap 机器。

21.5.3.1. control plane 机器

默认情况下，OpenShift Container Platform 安装过程会创建三台 control plane 机器。

每台机器都需要：

- 来自 RHOSP 配额的实例
- 来自 RHOSP 配额的端口
- 至少有 16 GB 内存和 4 个 vCPU 的类别
- RHOSP 配额中至少有 100 GB 存储空间

21.5.3.2. 计算机器

默认情况下，OpenShift Container Platform 安装过程会创建三台计算机器。

每台机器都需要：

- 来自 RHOSP 配额的实例
- 来自 RHOSP 配额的端口
- 至少有 8 GB 内存和 2 个 vCPU 的类别
- RHOSP 配额中至少有 100 GB 存储空间

提示

计算机器托管您在 OpenShift Container Platform 上运行的应用程序；运行数量应尽可能多。

21.5.3.3. bootstrap 机器

在安装过程中，会临时置备 bootstrap 机器来支持 control plane。生产 control plane 就绪后，bootstrap 机器会被取消置备。

bootstrap 机器需要：

- 来自 RHOSP 配额的实例
- 来自 RHOSP 配额的端口
- 至少有 16 GB 内存和 4 个 vCPU 的类别
- RHOSP 配额中至少有 100 GB 存储空间

21.5.4. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来获得用来安装集群的镜像。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

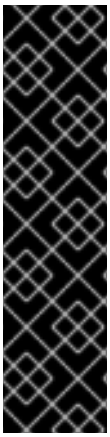
21.5.5. 在 RHOSP 上启用 Swift

Swift 由具有 swiftoperator 角色的用户帐户操控。在运行安装程序前，将该角色添加到帐户。

重要

如果 Red Hat OpenStack Platform(RHOSP)对象存储服务（通常称为 Swift）可用，OpenShift Container Platform 会使用它作为镜像 registry 存储。如果无法使用，安装程序会依赖于 RHOSP 块存储服务，通常称为 Cinder。

如果 Swift 存在并且您想要使用 Swift，您必须启用对其的访问。如果不存在，或者您不想使用它，请跳过这个部分。



重要

RHOSP 17 将 Ceph RGW 的 rgw_max_attr_size 参数设置为 256 个字符。此设置会导致将容器镜像上传到 OpenShift Container Platform registry 的问题。您必须将 rgw_max_attr_size 的值设置为至少 1024 个字符。

在安装前，检查您的 RHOSP 部署是否会受到此问题的影响。如果是，请重新配置 Ceph RGW。

先决条件

- 在目标环境中有一个 RHOSP 管理员帐户。
- 已安装 Swift 服务。
- 在 Ceph RGW 上，启用了 account in url 选项。

流程

在 RHOSP 上启用 Swift：

1. 在 RHOSP CLI 中以管理员身份，将 swiftoperator 角色添加到将要访问 Swift 的帐户中：

```
$ openstack role add --user <user> --project <project> swiftoperator
```

您的 RHOSP 部署现在可以将 Swift 用于镜像 registry。

21.5.6. 为安装程序定义参数

OpenShift Container Platform 安装程序依赖于一个名为 clouds.yaml 的文件。该文件描述了 Red Hat OpenStack Platform(RHOSP)配置参数，包括项目名称、登录信息和授权服务 URL。

流程

1. 创建 clouds.yaml 文件：

   - 如果您的 RHOSP 发行版包含 Horizon Web UI，请在该 UI 中生成 clouds.yaml 文件。

     重要

     记得在 auth 字段中添加密码。您还可以将 secret 保存在 clouds.yaml 以外的 一个独立的文件中。

   - 如果您的 RHOSP 发行版不包含 Horizon Web UI，或者您不想使用 Horizon，请自行创建该文件。如需有关 clouds.yaml 的详细信息，请参阅 RHOSP 文档中的 配置文件。

     ```
     clouds:
       shiftstack:
         auth:
           auth_url: http://10.10.14.42:5000/v3
           project_name: shiftstack
           username: <username>
           password: <password>
           user_domain_name: Default
           project_domain_name: Default
       dev-env:
         region_name: RegionOne
         auth:
           username: <username>
           password: <password>
           project_name: 'devonly'
           auth_url: 'https://10.10.14.22:5001/v2.0'
     ```

2. 如果您的 RHOSP 安装使用自签名证书颁发机构(CA)证书进行端点身份验证：

   a. 将证书颁发机构文件复制到您的机器中。

b. 将 cacerts 键添加到 clouds.yaml 文件。该值必须是到 CA 证书的绝对、不可 root 访问的路径：

```
clouds:
  shiftstack:
    ...
    cacert: "/etc/pki/ca-trust/source/anchors/ca.crt.pem"
```

提示

使用自定义 CA 证书运行安装程序后，您可以通过编辑 cloud-provider-config keymap 中的 ca-cert.pem 键的值来更新证书。在命令行中运行：

```
$ oc edit configmap -n openshift-config cloud-provider-config
```

3. 将 clouds.yaml 文件放在以下位置之一：

a. OS_CLIENT_CONFIG_FILE 环境变量的值

b. 当前目录

c. 特定于 Unix 的用户配置目录，如 ~/.config/openstack/clouds.yaml

d. 特定于 Unix 的站点配置目录，如 /etc/openstack/clouds.yaml

安装程序会按顺序搜索 clouds.yaml。

21.5.7. 设置 OpenStack Cloud Controller Manager 选项

另外，您可以编辑集群的 OpenStack Cloud Controller Manager (CCM) 配置。此配置控制 OpenShift Container Platform 与 Red Hat OpenStack Platform (RHOSP) 的交互方式。

有关配置参数的完整列表，请参阅"安装 OpenStack"文档中的"OpenStack Cloud Controller Manager 参考指南"页面。

流程

流程

1. 如果您还没有为集群生成清单文件，请运行以下命令生成这些文件：

```
$ openshift-install --dir <destination_directory> create manifests
```

2. 在文本编辑器中，打开 cloud-provider 配置清单文件。例如：

```
$ vi openshift/manifests/cloud-provider-config.yaml
```

3. 根据 CCM 参考指南修改选项。

针对负载均衡配置 Octavia 的情况比较常见。例如：

```
#...
[LoadBalancer]
lb-provider = "amphora" 1
floating-network-id="d3deb660-4190-40a3-91f1-37326fe6ec4a" 2
create-monitor = True 3
monitor-delay = 10s 4
monitor-timeout = 10s 5
monitor-max-retries = 1 6
#...
```

1. 此属性设置负载均衡器使用的 Octavia 供应商。它接受 "ovn" 或 "amphora" 作为值。如果您选择使用 OVN，还必须将 lb-method 设置为 SOURCE_IP_PORT。

2. 如果要将多个外部网络用于集群，则需要此属性。云提供商在网络上创建此处指定的浮动 IP 地址。

3. 此属性控制云供应商是否为 Octavia 负载均衡器创建运行状况监控器。将值设为 True 来创建运行状况监视器。从 RHOSP 16.2 开始，这个功能仅适用于 Amphora 供应商。

4. 此属性设定监控端点的频率。该值必须采用 time.ParseDuration() 格式。如果 create-monitor 属性的值为 True，则需要此属性。

5

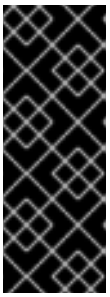
此属性设定监控请求在超时前打开的时间。该值必须采用 time.ParseDuration() 格式。如果 create-monitor 属性的值为 True，则需要此属性。

6

此属性定义在负载均衡器被标记为在线前需要成功完成监控请求。该值必须是整数。如果 create-monitor 属性的值为 True，则需要此属性。

重要

在保存更改之前，请验证该文件的结构是否正确。如果属性没有放入相应的部分，集群可能会失败。



重要

如果使用将 .spec.externalTrafficPolicy 属性的值设置为 Local 的服务，则必须将 create-monitor 属性的值设置为 True。RHOSP 16.2 中的 OVN Octavia 供应商不支持健康监控器。因此，当 lb-provider 值设为 "ovn" 时，将 ETP 参数值设置为 Local 的服务可能无法响应。

4. 保存对文件的更改并开始安装。

提示

您可以在运行安装程序后更新云供应商配置。在命令行中运行：

```
$ oc edit configmap -n openshift-config cloud-provider-config
```

保存更改后，您的集群将需要一些时间重新配置其自身。如果您的任一节点都没有 SchedulingDisabled 状态，则此过程已完成。

21.5.8. 为受限网络安装创建 RHCOS 镜像

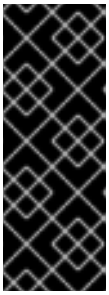
下载 Red Hat Enterprise Linux CoreOS(RHCOS)镜像，以在受限网络 Red Hat OpenStack Platform(RHOSP)环境中安装 OpenShift Container Platform。

先决条件

- 获取 OpenShift Container Platform 安装程序。对于受限网络安装，该程序位于您的镜像 registry 主机上。

流程

1. 登录到红帽客户门户网站的产品 下载页面。
2. 在 Version 下，为 RHEL 8 选择 OpenShift Container Platform 4.16 的最新发行版本。



重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每个发行版本而改变。您必须下载最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。如果可用，请使用与 OpenShift Container Platform 版本匹配的镜像版本。

3. 下载 Red Hat Enterprise Linux CoreOS(RHCOS)- OpenStack Image(QCOW) 镜像。
4. 解压缩镜像。

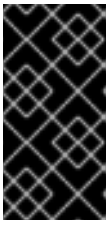
注意

您必须解压镜像，然后集群才能使用它。下载的文件的名称可能不包含压缩扩展名，如 .gz 或. tgz。要查找是否或者如何压缩文件，请在命令行中输入：

```
$ file <name_of_downloaded_file>
```

5. 将您解压缩的镜像上传到堡垒服务器可访问的位置，如 Glance。例如：

```
$ openstack image create --file rhcos-44.81.202003110027-0-openstack.x86_64.qcow2 --
disk-format qcow2 rhcos-${RHCOS_VERSION}
```



重要

根据您的 RHOSP 环境，您可能能够以. raw 或 .qcow2 格式 上传镜像。如果使用 Ceph，则必须使用 .raw 格式。

警告

如果安装程序发现多个同名的镜像，它会随机选择其中之一。为避免这种行为，请在 RHOSP 中为资源创建唯一名称。

该镜像现在可用于受限安装。记录 OpenShift Container Platform 部署中使用的镜像名称或位置。

21.5.9. 创建安装配置文件

您可以自定义在 Red Hat OpenStack Platform(RHOSP)上安装的 OpenShift Container Platform 集群。

先决条件

- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。对于受限网络安装，这些文件位于您的镜像主机上。
- 您有创建镜像 registry 期间生成的 imageContentSources 值。
- 您已获取了镜像 registry 的证书内容。
- 您已检索了 Red Hat Enterprise Linux CoreOS (RHCOS) 镜像，并将其上传到可访问的位置。

流程

1. 创建 install-config.yaml 文件。

a. 进入包含安装程序的目录并运行以下命令：

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

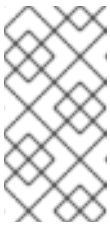
1 对于 <installation_directory>，请指定要存储安装程序创建的文件的目录名称。

在指定目录时：

- 验证该目录是否具有执行权限。在安装目录中运行 Terraform 二进制文件需要这个权限。
- 使用空目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

b. 在提示符处，提供云的配置详情：

i. 可选：选择用于访问集群机器的 SSH 密钥。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 ssh-agent 进程使用的 SSH 密钥。

ii. 选择 openstack 作为目标平台。

iii. 指定用于安装集群的 Red Hat OpenStack Platform(RHOSP)外部网络名称。

iv. 指定用于从外部访问 OpenShift API 的浮动 IP 地址。

v. 指定至少有 16 GB RAM 用于 control plane 节点，以及计算节点的 8 GB RAM。

vi. 选择集群要部署到的基域。所有 DNS 记录都将是这个基域的子域，并且还包括集群名称。

vii. 为集群输入一个名称。名称长度必须为 14 个或更少字符。

2. 在 install-config.yaml 文件中，将 platform.openstack.clusterOSImage 的值设置为镜像位置或名称。例如：

```
platform:
 openstack:
   clusterOSImage: http://mirror.example.com/images/rhcos-43.81.201912131630.0-
openstack.x86_64.qcow2.gz?
sha256=ffebbd68e8a1f2a245ca19522c16c86f67f9ac8e4e0c1f0a812b068b16f7265d
```

3. 编辑 install-config.yaml 文件，以提供在受限网络中安装所需的额外信息。

a. 更新 pullSecret 值，使其包含 registry 的身份验证信息：

```
pullSecret: '{"auths":{"<mirror_host_name>:5000": {"auth": "
<credentials>","email": "you@example.com"}}}'
```

对于 <mirror_host_name>，请指定 您在镜像 registry 证书中指定的 registry 域名 ；对于 <credentials>， 请指定您的镜像 registry 的 base64 编码用户名和密码。

b. 添加 additionalTrustBundle 参数和值。

```
additionalTrustBundle: |
 -----BEGIN CERTIFICATE-----

ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
 -----END CERTIFICATE-----
```

该值必须是您用于镜像 registry 的证书文件内容。证书文件可以是现有的可信证书颁发机构，也可以是您为镜像 registry 生成的自签名证书。

c. 添加镜像内容资源，类似于以下 YAML 摘录：

```
imageContentSources:
- mirrors:
  - <mirror_host_name>:5000/<repo_name>/release
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <mirror_host_name>:5000/<repo_name>/release
  source: registry.redhat.io/ocp/release
```

对于这些值，请使用您在创建镜像 registry 时记录的 imageContentSources。

d. 可选：将发布策略设置为 Internal：

```
publish: Internal
```

通过设置这个选项，您可以创建一个内部 Ingress Controller 和一个私有负载均衡器。

4. 对您需要的 install-config.yaml 文件进行任何其他修改。

有关参数的更多信息，请参阅"安装配置参数"。

5. 备份 install-config.yaml 文件，以便您可以使用它安装多个集群。

重要

install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用此文件，必须现在备份。

其他资源

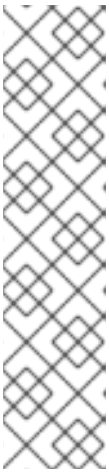
- OpenStack 的安装配置参数

21.5.9.1. 在安装过程中配置集群范围的代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 install-config.yaml 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 install-config.yaml 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 Proxy 对象的 spec.noProxy 字段中添加站点来绕过代理。



注意

Proxy 对象 status.noProxy 字段使用安装配置中的 networking.machineNetwork[].cidr、networking.clusterNetwork[].cidr 和 networking.serviceNetwork[] 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，Proxy 对象 status.noProxy 字段也会使用实例元数据端点填充(169.254.169.254)。

流程

1. 编辑 install-config.yaml 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 http。
2.

用于创建集群外 HTTPS 连接的代理 URL。

3

要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 . 以仅匹配子域。例如，.y.com 匹配 x.y.com，但不匹配 y.com。使用 * 绕过所有目的地的代理。

4

如果提供，安装程序会在 openshift-config 命名空间中生成名为 user-ca-bundle 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 trusted-ca-bundle 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， Proxy 对象的 trustedCA 字段中也会引用此配置映射。additionalTrustBundle 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

5

可选：决定 Proxy 对象的配置以引用 trustedCA 字段中 user-ca-bundle 配置映射的策略。允许的值是 Proxyonly 和 Always。仅在配置了 http/https 代理时，使用 Proxyonly 引用 user-ca-bundle 配置映射。使用 Always 始终引用 user-ca-bundle 配置映射。默认值为 Proxyonly。

注意

安装程序不支持代理的 readinessEndpoints 字段。

注意

如果安装程序超时，重启并使用安装程序的 wait-for 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. 保存该文件并在安装 OpenShift Container Platform 时引用。

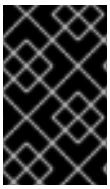
安装程序会创建一个名为 cluster 的集群范围代理，该代理 使用 提供的 install-config.yaml 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 cluster Proxy 对象，但它会有一个空 spec。

注意

只支持名为 cluster 的 Proxy 对象，且无法创建额外的代理。

21.5.9.2. 受限 OpenStack 安装的自定义 install-config.yaml 文件示例

这个示例 install-config.yaml 演示了所有可能的 Red Hat OpenStack Platform(RHOSP)自定义选项。



重要

此示例文件仅供参考。您必须使用安装程序来获取 install-config.yaml 文件。

```
apiVersion: v1
baseDomain: example.com
controlPlane:
  name: master
  platform: {}
  replicas: 3
compute:
- name: worker
  platform:
    openstack:
      type: ml.large
  replicas: 3
metadata:
  name: example
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  serviceNetwork:
  - 172.30.0.0/16
  networkType: OVNKubernetes
platform:
  openstack:
    region: region1
    cloud: mycloud
    externalNetwork: external
    computeFlavor: m1.xlarge
    apiFloatingIP: 128.0.0.1
fips: false
pullSecret: '{"auths": ...}'
sshKey: ssh-ed25519 AAAA...
additionalTrustBundle: |

  -----BEGIN CERTIFICATE-----
```

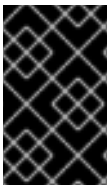
```
ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ

-----END CERTIFICATE-----

imageContentSources:
- mirrors:
  - <mirror_registry>/<repo_name>/release
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <mirror_registry>/<repo_name>/release
  source: quay.io/openshift-release-dev/ocp-v4.0-art-dev
```

21.5.10. 为集群节点 SSH 访问生成密钥对

在 OpenShift Container Platform 安装过程中，您可以为安装程序提供 SSH 公钥。密钥通过它们的 Ignition 配置文件传递给 Red Hat Enterprise Linux CoreOS(RHCOS)节点，用于验证对节点的 SSH 访问。密钥添加到每个节点上 core 用户的 ~/.ssh/authorized_keys 列表中，这将启用免密码身份验证。

将密钥传递给节点后，您可以使用密钥对作为用户 核心 通过 SSH 连接到 RHCOS 节点。若要通过 SSH 访问节点，必须由 SSH 为您的本地用户管理私钥身份。

如果要通过 SSH 连接到集群节点来执行安装调试或灾难恢复，则必须在安装过程中提供 SSH 公钥。./openshift-install gather 命令还需要在集群节点上设置 SSH 公钥。



重要

不要在生产环境中跳过这个过程，在生产环境中需要灾难恢复和调试。

注意

您必须使用本地密钥，而不是使用特定平台方法配置 的密钥，如 AWS 密钥对。

流程

1. 如果您在本地计算机上没有可用于在集群节点上进行身份验证的现有 SSH 密钥对，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1

注意

如果您计划在 x86_64、ppc64le 和 s390x 架构上安装使用 RHEL 加密库（这些加密库已提交给 NIST 用于 FIPS 140-2/140-3 验证）的 OpenShift Container Platform 集群，则不要创建使用 ed25519 算法的密钥。相反，创建一个使用 rsa 或 ecdsa 算法的密钥。

2. 查看公共 SSH 密钥：

```
$ cat <path>/<file_name>.pub
```

例如，运行以下命令来查看 ~/.ssh/id_ed25519.pub 公钥：

```
$ cat ~/.ssh/id_ed25519.pub
```

3. 将 SSH 私钥身份添加到本地用户的 SSH 代理（如果尚未添加）。在集群节点上，或者要使用 ./openshift-install gather 命令，需要对该密钥进行 SSH 代理管理，才能在集群节点上进行免密码 SSH 身份验证。

注意

在某些发行版中，自动管理默认 SSH 私钥身份，如 ~/.ssh/id_rsa 和 ~/.ssh/id_dsa。

a. 如果 ssh-agent 进程尚未为您的本地用户运行，请将其作为后台任务启动：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果集群处于 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

4. 将 SSH 私钥添加到 ssh-agent ：

```
$ ssh-add <path>/<file_name> 1
```

1 指定 SSH 私钥的路径和文件名，如 ~/.ssh/id_ed25519.pub

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

后续步骤

- 安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

21.5.11. 启用对环境的访问

在部署时，所有 OpenShift Container Platform 机器都是在 Red Hat OpenStack Platform(RHOSP) 租户网络中创建的。因此，大多数 RHOSP 部署中都无法直接访问它们。

您可以在安装过程中使用浮动 IP 地址(FIP)来配置 OpenShift Container Platform API 和应用程序访问。您还可以在没有配置 FIP 的情况下完成安装，但安装程序不会配置一种从外部访问 API 或应用程序的方法。

21.5.11.1. 启用通过浮动 IP 地址进行访问

创建浮动 IP(FIP)地址，以便从外部访问 OpenShift Container Platform API 和集群应用程序。

流程

1. 使用 Red Hat OpenStack Platform(RHOSP)CLI 创建 API FIP：

```
$ openstack floating ip create --description "API <cluster_name>.<base_domain>"
<external_network>
```

2. 使用 Red Hat OpenStack Platform(RHOSP)CLI，创建应用程序或 Ingress，FIP：

```
$ openstack floating ip create --description "Ingress <cluster_name>.<base_domain>"
<external_network>
```

3. 在 API 和 Ingress FIP 的 DNS 服务器中添加符合这些模式的记录：

```
api.<cluster_name>.<base_domain>.  IN  A  <API_FIP>
*.apps.<cluster_name>.<base_domain>. IN  A <apps_FIP>
```

注意

如果您不控制 DNS 服务器，可以通过将集群域名（如以下内容）添加到 /etc/hosts 文件中来访问集群：

- **<api_floating_ip> api.<cluster_name>.<base_domain>**
- **<application_floating_ip> grafana-openshift-monitoring.apps.<cluster_name>.<base_domain>**
- **<application_floating_ip> prometheus-k8s-openshift-monitoring.apps.<cluster_name>.<base_domain>**
- **<application_floating_ip> oauth-openshift.apps.<cluster_name>.<base_domain>**
- **<application_floating_ip> console-openshift-console.apps.<cluster_name>.<base_domain>**
- **application_floating_ip integrated-oauth-server-openshift-authentication.apps.<cluster_name>.<base_domain>**

/etc/hosts 文件中的集群域名授予对本地集群的 Web 控制台和监控界面的访问权限。您还可以使用 kubectl 或 oc。您可以使用指向 <application_floating_ip> 的额外条目来访问用户应用程序。此操作使 API 和应用程序可供您访问，不适合于生产部署，但允许对开发和测试进行安装。

4. **将 FIP 添加到 install-config.yaml 文件中，作为以下参数的值：**

   - **platform.openstack.ingressFloatingIP**
   - **platform.openstack.apiFloatingIP**

如果使用这些值，还必须在 install-config.yaml 文件中输入一个外部网络作为

platform.openstack.externalNetwork 参数的值。

提示

您可以通过分配浮动 IP 地址并更新防火墙配置，使 OpenShift Container Platform 资源在集群外可用。

21.5.11.2. 完成没有浮动 IP 地址的安装

您可以在 Red Hat OpenStack Platform(RHOSP)上安装 OpenShift Container Platform，而无需提供浮动 IP 地址。

在 install-config.yaml 文件中，不要定义以下参数：

- platform.openstack.ingressFloatingIP
- platform.openstack.apiFloatingIP

如果无法提供外部网络，也可以将 platform.openstack.externalNetwork 留 空。如果没有为 platform.openstack.externalNetwork 提供值，则不会为您创建路由器，如果没有额外的操作，安装程序将无法从 Glance 检索镜像。您必须自行配置外部连接。

如果您在因为缺少浮动 IP 地址或名称解析而无法访问集群 API 的系统中运行安装程序，安装会失败。在这些情况下，您可以使用代理网络或者从与机器位于相同网络的系统中运行安装程序。

注意

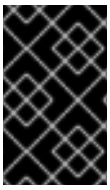
您可以通过为 API 和 Ingress 端口创建 DNS 记录来启用名称解析。例如：

```
api.<cluster_name>.<base_domain>.  IN  A  <api_port_IP>
*.apps.<cluster_name>.<base_domain>. IN  A <ingress_port_IP>
```

如果您不控制 DNS 服务器，您可以将记录添加到 /etc/hosts 文件中。此操作使 API 可供您自己访问，不适合于生产部署，而是允许安装以进行开发和测试。

21.5.12. 部署集群

您可以在兼容云平台上安装 OpenShift Container Platform。



重要

在初始安装过程中，您只能运行安装程序的 create cluster 命令一次。

先决条件

- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。
- 已确认主机上的云供应商帐户具有部署集群的正确权限。权限不正确的帐户会导致安装过程失败，并显示包括缺失权限的错误消息。

流程

- 进入包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

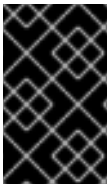
1. 对于 <installation_directory>，请指定自定义 ./install-config.yaml 文件的位置。
2. 要查看不同的安装详情，请指定 warn、debug 或 error，而不是 info。

验证

当集群部署成功完成时：

- 终端会显示用于访问集群的说明，包括指向 Web 控制台和 kubeadmin 用户的凭证的链接。

- 凭证信息还会输出到 <installation_directory>/.openshift_install.log.



重要

不要删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 node-bootstrapper 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。

- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

21.5.13. 验证集群状态

您可以在安装过程中或安装后验证 OpenShift Container Platform 集群的状态。

流程

1. 在集群环境中，导出管理员的 kubeconfig 文件：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 <installation_directory>，请指定安装文件保存到的目录的路径。

kubeconfig 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。

2. 查看部署后创建的 control plane 和计算机器：

```
$ oc get nodes
```

3. 查看集群的版本：

```
$ oc get clusterversion
```

4. 查看 Operator 的状态：

```
$ oc get clusteroperator
```

5. 查看集群中的所有正在运行的 pod:

```
$ oc get pods -A
```

21.5.14. 使用 CLI 登录集群

您可以通过导出集群 kubeconfig 文件，以默认系统用户身份登录集群。kubeconfig 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。

- 已安装 oc CLI。

流程

1. 导出 kubeadmin 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 <installation_directory>，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 oc 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

其他资源

- 如需有关 访问和了解 OpenShift Container Platform Web 控制台的更多详情，请参阅 访问 Web 控制台。

21.5.15. 禁用默认的 OperatorHub 目录源

在 OpenShift Container Platform 安装过程中，默认为 OperatorHub 配置由红帽和社区项目提供的源内容的 operator 目录。在受限网络环境中，必须以集群管理员身份禁用默认目录。

流程

- 通过在 OperatorHub 对象中添加 disableAllDefaultSources: true 来 禁用默认目录的源：

```
$ oc patch OperatorHub cluster --type json \
    -p '[{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]'
```

提示

或者，您可以使用 Web 控制台管理目录源。在 Administration → Cluster Settings → Configuration → OperatorHub 页面中，点 Sources 选项卡，您可以在其中创建、更新、删除、禁用和启用单独的源。

21.5.16. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控

21.5.17. 后续步骤

- 自定义集群。
- 如果您用来安装集群的镜像 registry 具有可信任的 CA，请通过 配置额外的信任存储将其添加到集群中。
- 如果需要，您可以选择 不使用远程健康报告。
- 如果需要，请参阅 注册断开连接的集群
- 为 Cluster Samples Operator 和 must-gather 工具 配置镜像流。

- 了解如何在 受限网络中使用 Operator Lifecycle Manager(OLM)。

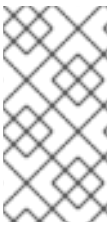
- 如果您没有将 RHOSP 配置为接受通过浮动 IP 地址的应用程序流量，请使用 浮动 IP 地址配置 RHOSP 访问。

21.6. OPENSTACK CLOUD CONTROLLER MANAGER 参考指南

21.6.1. OpenStack Cloud Controller Manager

从 OpenShift Container Platform 4.12 开始，在 Red Hat OpenStack Platform (RHOSP) 上运行的集群从旧的 OpenStack 云供应商切换到外部 OpenStack Cloud Controller Manager (CCM)。此更改遵循 Kubernetes 的变化，它从 in-tree（传统的云供应商）变为使用 Cloud Controller Manager 实施的外部云供应商。

要为旧的云供应商保留用户定义的配置，现有配置作为迁移过程的一部分映射到新的配置。它会在 openshift-config 命名空间中搜索名为 cloud-provider-config 的配置。



注意

配置映射名称 cloud-provider-config 没有静态配置。它源自 infrastructure/cluster CRD 中的 spec.cloudConfig.name 值。

找到的配置与 openshift-cloud-controller-manager 命名空间中的 cloud-conf 配置映射同步。

作为此同步的一部分，OpenStack CCM Operator 会更改新的配置映射，以便其属性与外部云供应商兼容。该文件通过以下方式更改：

- [Global] secret-name,[Global] secret-namespace, 和 [Global] kubeconfig-path 选项会被删除。它们不适用于外部云供应商。

- 添加 [Global] use-clouds,[Global] clouds-file, 和 [Global] cloud 选项。

- 整个 [BlockStorage] 部分已被删除。外部云供应商不再执行存储操作。块存储配置由 Cinder CSI 驱动程序管理。

另外，CCM Operator 会强制执行多个默认选项。这些选项的值始终被覆盖，如下所示：

```
[Global]
use-clouds = true
clouds-file = /etc/openstack/secret/clouds.yaml
cloud = openstack
...

[LoadBalancer]
enabled = true
```

clouds-value 值 /etc/openstack/secret/clouds.yaml 映射到 openshift-cloud-controller-manager 命名空间中的 openstack-cloud-credentials 配置。您可以像执行任何其他 clouds.yaml 文件一样修改此文件中的 RHOSP 云。

21.6.2. OpenStack Cloud Controller Manager (CCM) 配置映射

OpenStack CCM 配置映射定义集群如何与 RHOSP 云交互。默认情况下，此配置存储在 openshift-cloud-controller-manager 命名空间中的 cloud-conf 配置映射中的 cloud.conf 键下。

重要

cloud-conf 配置映射从 openshift-config 命名空间中的 cloud-provider-config 配置映射生成。

要更改 cloud-conf 配置映射描述的设置，请修改 cloud-provider-config 配置映射。

作为此同步的一部分，CCM Operator 会覆盖一些选项。如需更多信息，请参阅 "RHOSP Cloud Controller Manager"。

例如：

cloud-conf 配置映射示例

```
apiVersion: v1
data:
  cloud.conf: |
    [Global] 1
```

```
    secret-name = openstack-credentials
    secret-namespace = kube-system
    region = regionOne
    [LoadBalancer]
    enabled = True
kind: ConfigMap
metadata:
  creationTimestamp: "2022-12-20T17:01:08Z"
  name: cloud-conf
  namespace: openshift-cloud-controller-manager
  resourceVersion: "2519"
  uid: cbbeedaf-41ed-41c2-9f37-4885732d3677
```

1

使用 clouds.yaml 文件而不是修改配置映射来设置全局选项。

配置映射中存在以下选项。除另有说明时，对于在 RHOSP 上运行的集群，它们是必需的。

21.6.2.1. 负载均衡器选项

CCM 为使用 Octavia 的部署支持多个负载均衡器选项。

注意

neutron-rhcs 支持已弃用。

选项	描述
enabled	是否启用 **LoadBalancer** 类型的服务集成。默认值为 **true**。
floating-network-id	可选。用于为负载均衡器虚拟 IP 地址 (VIP) 创建浮动 IP 地址的外部网络。如果云中有多个外部网络，则必须设置此选项，或者用户必须在服务注解中指定 **loadbalancer.openstack.org/floating-network-id**。

选项	描述
floating-subnet-id	可选。用于为负载均衡器 VIP 创建浮动 IP 地址的外部网络子网。可以被服务注解 **loadbalancer.openstack.org/floating-subnet-id** 覆盖。
floating-subnet	可选。一个名称特征（如果以 ~ 开始为 glob 或正则表达式），用于为负载均衡器 VIP 创建浮动 IP 地址的外部网络子网。可以被服务注解 **loadbalancer.openstack.org/floating-subnet** 覆盖。如果多个子网与模式匹配，则使用第一个具有可用 IP 地址的子网。
floating-subnet-tags	可选。用于为负载均衡器 VIP 创建浮动 IP 地址的外部网络子网的标签。可以被服务注解 **loadbalancer.openstack.org/floating-subnet-tags** 覆盖。如果多个子网与这些标签匹配，则使用第一个具有可用 IP 地址的子网。 如果 RHOSP 网络被配置为禁用共享，例如，创建过程中使用的 **--no-share** 标志，则不支持这个选项。将网络设置为共享来使用这个选项。
lb-method	用于创建负载均衡器池的负载平衡算法。对于 Amphora 供应商，值可以是 **ROUND_ROBIN**、**LEAST_CONNECTIONS** 或 **SOURCE_IP**。默认值为 **ROUND_ROBIN**。 对于 OVN 提供程序，只支持 **SOURCE_IP_PORT** 算法。 对于 Amphora 供应商，如果使用 **LEAST_CONNECTIONS** 或 **SOURCE_IP** 方法，在 **openshift-config** 命名空间的 **cloud-provider-config** 配置映射中将 **create-monitor** 选项配置为 **true**，以便在负载均衡器类型服务上的 **ETP:Local** 以允许客户端中的平衡算法强制进行服务端点连接。
lb-provider	可选。用于指定负载均衡器的供应商，如 **amphora** 或 **octavia**。仅支持 Amphora 和 Octavia 供应商。
lb-version	可选。负载均衡器 API 版本。仅支持 **"v2"**。
subnet-id	创建负载均衡器 VIP 的网络服务子网的 ID。对于双堆栈部署，请保留此选项未设置。OpenStack 云提供商自动选择要用于负载平衡器的子网。
network-id	创建负载均衡器 VIP 的网络服务网络的 ID。如果设置了 **subnet-id**，则不需要。如果没有设置此属性，则会根据集群节点使用的网络自动选择网络。

选项	描述
create-monitor	是否为服务负载均衡器创建运行状况监控器。声明 **externalTrafficPolicy: Local** 的服务需要健康监控器。默认值为 **false**。 如果您使用早于 **ovn** 供应商的版本 17 的 RHOSP，则不支持这个选项。
monitor-delay	将探测发送到负载均衡器成员的时间间隔（以秒为单位）。默认值为 **5**。
monitor-max-retries	将负载均衡器成员的操作状态更改为 **ONLINE** 所需的成功检查数量。有效范围为 **1** 到 **10**，默认值为 **1**。
monitor-timeout	监控器在超时前等待连接到后端的时间（以秒为单位）。默认值为 **3**。
internal-lb	是否在没有浮动 IP 地址的情况下创建内部负载均衡器。默认值为 **false**。
LoadBalancerClass "ClassName"	这是一个 config 部分，其中包含一组选项： ● **floating-network-id** ● **floating-subnet-id** ● **floating-subnet** ● **floating-subnet-tags** ● **network-id** ● **subnet-id** 这些选项的行为与 CCM 配置文件的负载均衡器部分相同的命名选项的行为相同。 您可以通过指定服务注解 **loadbalancer.openstack.org/class** 来设置 **ClassName** 值。
max-shared-lb	可以共享负载均衡器的最大服务数。默认值为 **2**。

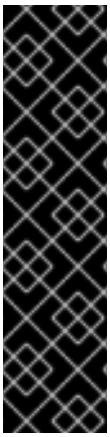
21.6.2.2. Operator 覆盖的选项

CCM Operator 覆盖以下选项，您可以识别其配置 RHOSP。不要自行配置它们。本文档中包含它们仅用于信息。

选项	描述
auth-url	RHOSP Identity 服务 URL。例如：**http://128.110.154.166/identity**。
os-endpoint-type	从服务目录使用的端点类型。
username	Identity 服务用户名。
password	Identity 服务用户密码。
domain-id	Identity 服务用户域 ID。
domain-name	Identity 服务用户域名。
tenant-id	Identity 服务项目 ID。如果您使用 Identity 服务应用程序凭证，请保留这个选项。 在 Identity API 的版本 3 中，其标识符 **tenant** 改为 **project**，**tenant-id** 的值会自动映射到 API 中的项目结构。
tenant-name	Identity 服务项目名称。
tenant-domain-id	Identity service 项目域 ID。
tenant-domain-name	Identity service 项目域名。
user-domain-id	Identity 服务用户域 ID。
user-domain-name	Identity 服务用户域名。
use-clouds	是否在 **clouds.yaml** 文件中获取授权凭证。本节中设置的选项优先于 **clouds.yaml** 文件中读取的值。 CCM 在以下位置搜索文件： 1. **clouds-file** 选项的值。 2. 存储在环境变量 **OS_CLIENT_CONFIG_FILE** 中的文件路径。 3. **pkg/openstack** 目录。 4. **~/.config/openstack** 目录。 5. **/etc/openstack** 目录。
clouds-file	**clouds.yaml** 文件的文件路径。如果将 **use-clouds** 选项设置为 **true**，则会使用它。

选项	描述
cloud	要使用的 **clouds.yaml** 文件中命名的云。如果将 **use-clouds** 选项设置为 **true**，则会使用它。

21.7. 在本地磁盘上使用 ROOTVOLUME 和 ETCD 部署 OPENSTACK



重要

在本地磁盘上使用 rootVolume 和 etcd 在 Red Hat OpenStack Platform (RHOSP)上部署只是一个技术预览功能。技术预览功能不受红帽产品服务等级协议（SLA）支持，且功能可能并不完整。红帽不推荐在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。

有关红帽技术预览功能支持范围的更多信息，请参阅技术预览功能支持范围。

作为第 2 天操作，您可以通过将 etcd 从根卷（由 OpenStack Cinder 提供）移到专用的临时本地磁盘，解决并防止 Red Hat OpenStack Platform (RHOSP)安装的性能问题。

21.7.1. 在本地磁盘上部署 RHOSP

如果您有一个现有的 RHOSP 云，您可以将 etcd 从该云移到专用临时本地磁盘。

警告

此流程仅用于在本地磁盘中测试 etcd，不应在生产环境中使用。在某些情况下，可能会发生 control plane 的完整丢失。如需更多信息，请参阅"备份和恢复"下的"备份和恢复操作"。

先决条件

- 您有一个带有正常工作的 Cinder 的 OpenStack 云。
- 您的 OpenStack 云至少有 75 GB 可用存储，以适应 OpenShift control plane 的 3 个根

卷。

- OpenStack 云使用 Nova 临时存储进行部署，该存储使用本地存储后端而不是 rbd。

流程

1. 运行以下命令，为 control plane 创建至少有 10 GB 临时磁盘的 Nova 类别，并基于您的环境替换 --ram、--disk 和 <flavor_name> 的值：

```
$ openstack flavor create --<ram 16384> --<disk 0> --ephemeral 10 --vcpus 4
<flavor_name>
```

2. 使用 control plane 的 root 卷部署集群，例如：

YAML 文件示例

```
# ...
controlPlane:
  name: master
  platform:
    openstack:
      type: ${CONTROL_PLANE_FLAVOR}
      rootVolume:
        size: 25
        types:
        - ${CINDER_TYPE}
  replicas: 3
# ...
```

3. 运行以下命令部署您创建的集群：

```
$ openshift-install create cluster --dir <installation_directory> 1
```

1. 对于 <installation_directory>，请指定之前创建的自定义 ./install-config.yaml 文件的位置。

4. 运行以下命令，在继续下一步前验证您部署的集群是否健康：

```
$ oc wait clusteroperators --all --for=condition=Progressing=false (1)
```

(1) 确保集群操作器完成进行，且集群没有部署或更新。

5. 运行以下命令，编辑 ControlPlaneMachineSet (CPMS) 来添加 etcd 使用的额外块设备：

```
$ oc patch ControlPlaneMachineSet/cluster -n openshift-machine-api --type json -p '
(1)
[
  {
    "op": "add",
    "path":
"/spec/template/machines_v1beta1_machine_openshift_io/spec/providerSpec/value/additionalBlockDevices", (2)
    "value": [
      {
        "name": "etcd",
        "sizeGiB": 10,
        "storage": {
          "type": "Local" (3)
        }
      }
    ]
  }
]
'
```

(1) 将 JSON 补丁应用到 ControlPlaneMachineSet 自定义资源 (CR)。

(2) 指定添加 additionalBlockDevices 的路径。

(3) 在集群中添加至少 10 GB 的 etcd 设备。只要 etcd 设备适合 Nova 类别，就可以指定大于 10 GB 的值。例如，如果 Nova 类别有 15 GB，您可以创建具有 12 GB 的 etcd 设备。

6. 使用以下步骤验证 control plane 机器是否健康：

a. 运行以下命令，等待 control plane 机器集更新完成：

```
$ oc wait --timeout=90m --for=condition=Progressing=false
controlplanemachineset.machine.openshift.io -n openshift-machine-api cluster
```

b. 运行以下命令验证 3 个 control plane 机器集是否已更新：

```
$ oc wait --timeout=90m --for=jsonpath='{.status.updatedReplicas}'=3
controlplanemachineset.machine.openshift.io -n openshift-machine-api cluster
```

c. 运行以下命令，验证 3 个 control plane 机器集是否健康：

```
$ oc wait --timeout=90m --for=jsonpath='{.status.replicas}'=3
controlplanemachineset.machine.openshift.io -n openshift-machine-api cluster
```

d. 运行以下命令，验证 ClusterOperators 是否没有在集群中进行：

```
$ oc wait clusteroperators --timeout=30m --all --for=condition=Progressing=false
```

e. 运行以下脚本，验证 3 个 control plane 机器是否都有之前创建的额外块设备：

```
$ cp_machines=$(oc get machines -n openshift-machine-api --
selector='machine.openshift.io/cluster-api-machine-role=master' --no-headers -o
custom-columns=NAME:.metadata.name) 1


if [[ $(echo "${cp_machines}" | wc -l) -ne 3 ]]; then
  exit 1
fi 2


for machine in ${cp_machines}; do
  if ! oc get machine -n openshift-machine-api "${machine}" -o
jsonpath='{.spec.providerSpec.value.additionalBlockDevices}' | grep -q 'etcd'; then
 exit 1
  fi 3
done
```

1

检索集群中运行的 control plane 机器。

2

迭代带有名称 etcd 的 additionalBlockDevices 条目的机器。

3

输出每个 control plane 机器的名称，该机器具有一个名为 etcd 的 additionalBlockDevice。

7. 使用以下 YAML 文件，创建一个名为 98-var-lib-etcd.yaml 的文件：

警告

此流程用于在本地磁盘上测试 etcd，不应在生产环境中使用。在某些情况下，可能会发生 control plane 的完整丢失。如需更多信息，请参阅"备份和恢复"下的"备份和恢复操作"。

```
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: master
  name: 98-var-lib-etcd
spec:
  config:
    ignition:
      version: 3.4.0
    systemd:
      units:
      - contents: |
          [Unit]
          Description=Mount local-etcd to /var/lib/etcd

          [Mount]
          What=/dev/disk/by-label/local-etcd 1
          Where=/var/lib/etcd
          Type=xfs
          Options=defaults,prjquota

          [Install]
```

```
      WantedBy=local-fs.target
    enabled: true
    name: var-lib-etcd.mount
  - contents: |
      [Unit]
      Description=Create local-etcd filesystem
      DefaultDependencies=no
      After=local-fs-pre.target
      ConditionPathIsSymbolicLink=!/dev/disk/by-label/local-etcd 2

      [Service]
      Type=oneshot
      RemainAfterExit=yes
      ExecStart=/bin/bash -c "[ -L /dev/disk/by-label/ephemeral0 ] || ( >&2 echo
Ephemeral disk does not exist; /usr/bin/false )"
      ExecStart=/usr/sbin/mkfs.xfs -f -L local-etcd /dev/disk/by-label/ephemeral0 3

      [Install]
      RequiredBy=dev-disk-by\x2dlabel-local\x2detcd.device
    enabled: true
    name: create-local-etcd.service
  - contents: |
      [Unit]
      Description=Migrate existing data to local etcd
      After=var-lib-etcd.mount
      Before=crio.service 4

      Requisite=var-lib-etcd.mount
      ConditionPathExists=!/var/lib/etcd/member
      ConditionPathIsDirectory=/sysroot/ostree/deploy/rhcos/var/lib/etcd/member 5

      [Service]
      Type=oneshot
      RemainAfterExit=yes

      ExecStart=/bin/bash -c "if [ -d /var/lib/etcd/member.migrate ]; then rm -rf
/var/lib/etcd/member.migrate; fi" 6

      ExecStart=/usr/bin/cp -aZ /sysroot/ostree/deploy/rhcos/var/lib/etcd/member/
/var/lib/etcd/member.migrate
      ExecStart=/usr/bin/mv /var/lib/etcd/member.migrate /var/lib/etcd/member 7

      [Install]
      RequiredBy=var-lib-etcd.mount
    enabled: true
    name: migrate-to-local-etcd.service
  - contents: |
      [Unit]
      Description=Relabel /var/lib/etcd

      After=migrate-to-local-etcd.service
      Before=crio.service

      [Service]
      Type=oneshot
```

```
    RemainAfterExit=yes

    ExecCondition=/bin/bash -c "[ -n \"$(restorecon -nv /var/lib/etcd)\" ]" 8

    ExecStart=/usr/sbin/restorecon -R /var/lib/etcd

    [Install]
    RequiredBy=var-lib-etcd.mount
  enabled: true
  name: relabel-var-lib-etcd.service
```

1

etcd 数据库必须由设备挂载，而不是标签，以确保 systemd 生成此配置中使用的设备依赖项来触发文件系统创建。

2

如果文件系统 dev/disk/by-label/local-etcd 已存在，则不要运行。

3

如果 /dev/disk/by-label/ephemeral0 不存在，则会失败并显示警报消息。

4

将现有数据迁移到本地 etcd 数据库。这个配置会在挂载 /var/lib/etcd 后进行，但在 CRI-O 启动前，etcd 还没有运行。

5

要求挂载 etcd，且不包含成员目录，但 ostree 会被挂载。

6

清理任何以前的迁移状态。

7

复制和移动单独的步骤，以确保创建完整的成员目录。

8

在执行完整的递归重新标记前，对挂载点目录执行快速检查。如果文件路径 /var/lib/etcd 中的 restorecon 无法重命名目录，则不会执行递归重命名。

8. 运行以下命令来创建新的 MachineConfig 对象：

```
$ oc create -f 98-var-lib-etcd.yaml
```

注意

将 etcd 数据库移到每个 control plane 机器的本地磁盘需要时间。

9. 运行以下命令，验证 etcd 数据库是否已传输到每个 control plane 的本地磁盘：

a. 运行以下命令验证集群是否仍然在更新：

```
$ oc wait --timeout=45m --for=condition=Updating=false
machineconfigpool/master
```

b. 运行以下命令验证集群是否已就绪：

```
$ oc wait node --selector='node-role.kubernetes.io/master' --for condition=Ready --
timeout=30s
```

c. 运行以下命令，验证集群 Operator 是否在集群中运行：

```
$ oc wait clusteroperators --timeout=30m --all --for=condition=Progressing=false
```

21.7.2. 其他资源

- 推荐的 etcd 实践
- 备份和恢复选项概述

21.8. 在 OPENSTACK 上卸载集群

您可以删除部署到 Red Hat OpenStack Platform(RHOSP)中的集群。

21.8.1. 删除使用安装程序置备的基础架构的集群

您可以从云中删除使用安装程序置备的基础架构的集群。

注意

卸载后，检查云供应商是否有未正确删除的资源，特别是在用户置备基础架构(UPI)集群中。可能存在安装程序未创建或安装程序无法访问的资源。

先决条件

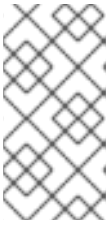
- 有用于部署集群的安装程序副本。
- 有创建集群时安装程序生成的文件。

流程

1. 在用来安装集群的计算机中包含安装程序的目录中，运行以下命令：

```
$ ./openshift-install destroy cluster \
--dir <installation_directory> --log-level info 1 2
```

1. 对于 <installation_directory>，请指定安装文件保存到的目录的路径。
2. 要查看不同的详情，请指定 warn、debug 或 error，而不是 info。



注意

您必须为集群指定包含集群定义文件的目录。安装程序需要此目录中的 metadata.json 文件来删除集群。

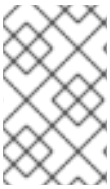
2. 可选：删除 <installation_directory> 目录和 OpenShift Container Platform 安装程序。

21.9. 从您自己的基础架构中卸载 RHOSP 上的集群

您可以删除在用户置备的基础架构上部署到 Red Hat OpenStack Platform(RHOSP)中的集群。

21.9.1. 下载 playbook 依赖项

用于简化用户置备的基础架构删除过程的 Ansible playbook 需要几个 Python 模块。在您要运行该过程的机器上，添加模块的存储库，然后下载它们。



注意

这些说明假设您使用 Red Hat Enterprise Linux(RHEL)8。

先决条件

- Python 3 已安装在您的机器上。

流程

1. 在命令行中添加软件仓库：
   a. 使用 Red Hat Subscription Manager 注册：

```
$ sudo subscription-manager register # If not done already
```

   b. 获取最新的订阅数据：

```
$ sudo subscription-manager attach --pool=$YOUR_POOLID # If not done already
```

   c. 禁用当前的软件仓库：

```
$ sudo subscription-manager repos --disable=* # If not done already
```

   d. 添加所需的软件仓库：

```
$ sudo subscription-manager repos \
  --enable=rhel-8-for-x86_64-baseos-rpms \
  --enable=openstack-16-tools-for-rhel-8-x86_64-rpms \
  --enable=ansible-2.9-for-rhel-8-x86_64-rpms \
  --enable=rhel-8-for-x86_64-appstream-rpms
```

2. 安装模块：

```
$ sudo yum install python3-openstackclient ansible python3-openstacksdk
```

3. 确保 python 命令指向 python 3:

```
$ sudo alternatives --set python /usr/bin/python3
```

21.9.2. 从使用您自己的基础架构的 RHOSP 中删除集群

您可以在使用您自己的基础架构的 Red Hat OpenStack Platform(RHOSP)上删除 OpenShift Container Platform 集群。要快速完成移除过程，请运行多个 Ansible playbook。

先决条件

- Python 3 已安装在您的机器上。
- 下载了"下载 playbook 依赖项"中的模块。
- 您有用于安装集群的 playbook。
- 您修改了前缀为 down- 的 playbook，以反映您对相应安装 playbook 所做的任何更改。例如，对 bootstrap.yaml 文件的更改会反映在 down-bootstrap.yaml 文件中。
- 所有 playbook 都位于一个通用目录中。

流程

1. 在命令行中运行您下载的 playbook：

```
$ ansible-playbook -i inventory.yaml  \
 down-bootstrap.yaml      \
 down-control-plane.yaml  \
 down-compute-nodes.yaml  \
 down-load-balancers.yaml \
 down-network.yaml        \
 down-security-groups.yaml
```

2. 删除您为 OpenShift Container Platform 安装所做的任何 DNS 记录更改。

OpenShift Container Platform 从您的基础架构中删除。

21.10. OPENSTACK 的安装配置参数

在 Red Hat OpenStack Platform (RHOSP) 上部署 OpenShift Container Platform 集群前，您可以提供参数来自定义集群及其托管平台。在创建 install-config.yaml 文件时，您可以通过命令行为所需参数提供值。然后，您可以修改 install-config.yaml 文件以进一步自定义集群。

21.10.1. OpenStack 可用的安装配置参数

下表指定您可以在安装过程中设置所需的、可选和 OpenStack 安装配置参数。

注意

安装后，您无法在 install-config.yaml 文件中修改这些参数。

21.10.1.1. 所需的配置参数

下表描述了所需的安装配置参数：

表 21.6. 所需的参数

参数	描述	值
apiVersion:	**install-config.yaml** 内容的 API 版本。当前版本为 **v1**。安装程序可能还支持旧的 API 版本。	字符串

参数	描述	值
baseDomain:	云供应商的基域。基域用于创建到 OpenShift Container Platform 集群组件的路由。集群的完整 DNS 名称是 **baseDomain** 和 **metadata.name** 参数值的组合，其格式为 **<metadata.name>.<baseDomain>**。	完全限定域名或子域名，如 **example.com**。
metadata:	Kubernetes 资源 **ObjectMeta**，其中只消耗 **name** 参数。	对象
metadata: name:	集群的名称。集群的 DNS 记录是 **{{.metadata.name}}.{{.baseDomain}}** 的子域。	小写字母、连字符(**-**)和句点(**.**)字符串，如 **dev**。字符串长度必须为 14 个字符或更少。
platform:	对于特定平台的配置取决于执行安装的环境： **aws**, **baremetal**, **azure**, **gcp**, **ibmcloud**, **nutanix**, **openstack**, **powervs**, **vsphere**, 或 **{}**.有关 **platform.<platform>** 参数的更多信息，请参考下表中您的特定平台。	对象

pullSecret:	从 Red Hat OpenShift Cluster Manager 获取 pull secret，验证从 Quay.io 等服务中下载 OpenShift Container Platform 组件的容器镜像。	<pre>{
		  "auths":{
		  "cloud.openshift.com":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  },
		  "quay.io":{
		  "auth":"b3Blb=",
		  "email":"you@example.com"
		  }
		  }
		}</pre>

21.10.1.2. 网络配置参数

您可以根据现有网络基础架构的要求自定义安装配置。例如，您可以扩展集群网络的 IP 地址块，或者提供不同于默认值的不同 IP 地址块。

仅支持 IPv4 地址。

注意

Red Hat OpenShift Data Foundation 灾难恢复解决方案不支持 Globalnet。对于区域灾难恢复场景，请确保为每个集群中的集群和服务网络使用非重叠的专用 IP 地址。

表 21.7. 网络参数

参数	描述	值
networking:	集群网络的配置。	对象  **注意** 您无法在安装后修改 **网络** 对象指定的参数。
networking: networkType:	要安装的 Red Hat OpenShift Networking 网络插件。	**OVNKubernetes**。**OVNKubernetes** 是 Linux 网络和包含 Linux 和 Windows 服务器的混合网络的 CNI 插件。默认值为 **OVNKubernetes**。
networking: clusterNetwork:	pod 的 IP 地址块。 默认值为 **10.128.0.0/14**，主机前缀为 **/23**。 如果您指定了多个 IP 地址块，块不得重叠。	对象数组。例如： networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23
networking: clusterNetwork: cidr:	使用 **networking.clusterNetwork** 时需要此项。IP 地址块。 IPv4 网络。	无类别域间路由(CIDR)表示法中的 IP 地址块。IPv4 块的前缀长度介于 **0 到 32** 之间。
networking: clusterNetwork: hostPrefix:	分配给每个节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从 given **cidr** 中分配 a/**23** 子网。**hostPrefix** 值 **23** 提供 510（2^(32 - 23)- 2）pod IP 地址。	子网前缀。 默认值为 **23**。

参数	描述	值
networking: serviceNetwork:	服务的 IP 地址块。默认值为 **172.30.0.0/16**。 OVN-Kubernetes 网络插件只支持服务网络的一个 IP 地址块。	CIDR 格式具有 IP 地址块的数组。例如： networking: serviceNetwork: - 172.30.0.0/16
networking: machineNetwork:	机器的 IP 地址块。 如果您指定了多个 IP 地址块，块不得重叠。	对象数组。例如： networking: machineNetwork: - cidr: 10.0.0.0/16
networking: machineNetwork: cidr:	使用 **networking.machineNetwork** 时需要此项。IP 地址块。对于 libvirt 和 IBM Power® Virtual Server 以外的所有平台，默认值为 **10.0.0.0/16**。对于 libvirt，默认值 为 **192.168.126.0/24**。对于 IBM Power® Virtual Server，默认值为 **192.168.0.0/24**。	CIDR 表示法中的 IP 网络块。 例如： **10.0.0.0/16**。 **注意** 将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

21.10.1.3. 可选的配置参数

下表描述了可选的安装配置参数：

表 21.8. 可选参数

参数	描述	值
additionalTrustBundle:	添加到节点可信证书存储中的 PEM 编码 X.509 证书捆绑包。配置了代理时，也可以使用此信任捆绑包。	字符串
capabilities:	控制可选核心组件的安装。您可以通过禁用可选组件来减少 OpenShift Container Platform 集群的空间。如需更多信息，请参阅 *安装*中的"集群功能"页面。	字符串数组

参数	描述	值
capabilities: baselineCapabilitySet:	选择要启用的一组初始可选功能。有效值为 **None**、**v4.11**、**v4.12** 和 **vCurrent**。默认值为 **vCurrent**。	字符串
capabilities: additionalEnabledCapabilities:	将可选功能集合扩展到您在 **baselineCapabilitySet** 中指定的范围。您可以在此参数中指定多个功能。	字符串数组
cpuPartitioningMode:	启用工作负载分区，它会隔离 OpenShift Container Platform 服务、集群管理工作负载和基础架构 pod，以便在保留的一组 CPU 上运行。工作负载分区只能在安装过程中启用，且在安装后无法禁用。虽然此字段启用工作负载分区，但它不会将工作负载配置为使用特定的 CPU。如需更多信息，请参阅 *Scalability and Performance* 部分中的 *Workload partitioning* 页面。	**None** 或 **AllNodes**.**None** 是默认值。
compute:	组成计算节点的机器的配置。	**MachinePool** 对象的数组。
compute: architecture:	决定池中机器的指令集合架构。目前，不支持具有不同架构的集群。所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
compute: hyperthreading:	是否在计算机器上启用或禁用并发多**线程或超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。	**enabled** 或 **Disabled**
compute: name:	使用 **compute** 时需要此项。机器池的名称。	**worker**

参数	描述	值
compute: platform:	使用 **compute** 时需要此项。使用此参数指定托管 worker 机器的云供应商。此参数值必须与 **controlPlane.platform** 参数值匹配。	**aws**, **azure**, **gcp**, **ibmcloud**, **nutanix**, **openstack**, **powervs**, **vsphere**, 或 **{}**
compute: replicas:	要置备的计算机器数量，也称为 worker 机器。	大于或等于 **2** 的正整数。默认值为 **3**。
featureSet:	为功能集启用集群。功能集是 OpenShift Container Platform 功能的集合，默认情况下不启用。有关在安装过程中启用功能集的更多信息，请参阅"使用功能门启用功能"。	字符串.要启用的功能集的名称，如 **TechPreviewNoUpgrade**。
controlPlane:	组成 control plane 的机器的配置。	**MachinePool** 对象的数组。
controlPlane: architecture:	决定池中机器的指令集合架构。目前，不支持具有不同架构的集群。所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
controlPlane: hyperthreading:	是否在 control plane 机器上启用或禁用并发多 **线程或超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。	**enabled** 或 **Disabled**
controlPlane: name:	使用 **controlPlane** 时需要此项。机器池的名称。	**master**
controlPlane: platform:	使用 **controlPlane** 时需要此项。使用此参数指定托管 control plane 机器的云供应商。此参数值必须与 **compute.platform** 参数值匹配。	**aws**, **azure**, **gcp**, **ibmcloud**, **nutanix**, **openstack**, **powervs**, **vsphere**, 或 **{}**

参数	描述	值
controlPlane: replicas:	要置备的 control plane 机器数量。	部署单节点 OpenShift 时支持的值为 **3** 或 **1**。
credentialsMode:	Cloud Credential Operator(CCO)模式。如果没有指定模式，CCO 会动态尝试决定提供的凭证的功能，在支持多个模式的平台上首选 mint 模式。	**Mint**、**Passthrough**、**Manual** 或空字符串(**""**)。[1]
fips:	启用或禁用 FIPS 模式。默认值为 **false**（禁用）。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。  **重要** 要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 Red Hat Enterprise Linux (RHEL) 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅在 FIPS 模式中安装该系统。 当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS) 时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。  **注意** 如果使用 Azure File 存储，则无法启用 FIPS 模式。	**false** 或 **true**

参数	描述	值
imageContentSources:	release-image 内容的源和存储库。	对象数组。包括一个 **source** 以及可选的 **mirrors**，如本表的以下行所述。
imageContentSources: source:	使用 **imageContentSources** 时需要此项。指定用户在镜像拉取规格中引用的存储库。	字符串
imageContentSources: mirrors:	指定可能还包含同一镜像的一个或多个仓库。	字符串数组
publish:	如何发布或公开集群的面向用户的端点，如 Kubernetes API、OpenShift 路由。	**内部或外部 .**默认值为 **External**。 在非云平台上不支持将此字段设置为 **Internal**。 **重要** 如果将字段的值设为 **Internal**，集群将无法运行。如需更多信息，请参阅 BZ#1953035。
sshKey:	用于验证对集群机器的访问的 SSH 密钥。 **注意** 对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。	例如，**sshKey: ssh-ed25519 AAAA..**。

1. **不是所有 CCO 模式都支持所有云供应商。有关 CCO 模式的更多信息，请参阅*身份验证和授权*内容中的"管理云供应商凭证"条目。**

21.10.1.4. 其他 Red Hat OpenStack Platform(RHOSP)配置参数

下表中描述了其他 RHOSP 配置参数：

表 21.9. 其他 RHOSP 参数

参数	描述	值
compute: platform: openstack: rootVolume: size:	对于计算机器，以 GB 为单位的根卷大小。如果没有设置这个值，机器将使用临时存储。	整数，如 **30**。
compute: platform: openstack: rootVolume: types:	对于计算机器，根卷类型。	字符串列表，如 {**performance-host1,performance-host2,performance-host3**}。[1]
compute: platform: openstack: rootVolume: type:	对于计算机器，根卷的类型。此属性已弃用，并被 **compute.platform.openstack.rootVolume.types** 替代。	字符串，例如，**performance**。[2]
compute: platform: openstack: rootVolume: zones:	对于计算机器，Cinder 可用区在其中安装根卷。如果没有为此参数设置值，安装程序会选择默认的可用性区域。当定义了 **compute.platform.openstack.zones** 时，此参数是必需的。	字符串列表，如 **["zone-1", "zone-2"]**。
controlPlane: platform: openstack: rootVolume: size:	对于 control plane 机器，以 GB 为单位表示的根卷大小。如果没有设置这个值，机器将使用临时存储。	整数，如 **30**。

参数	描述	值
controlPlane: platform: openstack: rootVolume: types:	对于 control plane 机器，根卷类型。	字符串列表，如 {**performance-host1**,**performance-host2**,**performance-host3**}。[1]
controlPlane: platform: openstack: rootVolume: type:	对于 control plane 机器，根卷的类型。此属性已弃用，并被 **compute.platform.openstack.rootVolume.types** 替代。	字符串，例如，**performance**。[2]
controlPlane: platform: openstack: rootVolume: zones:	对于 control plane 机器，安装根卷的 Cinder 可用区。如果没有设置这个值，安装程序会选择默认可用区。当定义了 **controlPlane.platform.openstack.zones** 时，此参数是必须的。	字符串列表，如 **["zone-1", "zone-2"]**。
platform: openstack: cloud:	要使用的 RHOSP 云名称，来自于 **clouds.yaml** 文件中的云列表。 在 **clouds.yaml** 文件中的云配置中，如果可能，请使用应用程序凭证而不是用户名和密码组合。使用应用程序凭证可以避免因遵循用户名和密码轮转的 secret propogation 中断。	字符串，如 **MyCloud**。
platform: openstack: externalNetwork:	用于安装的 RHOSP 外部网络名称。	字符串，如 **external**。

参数	描述	值
platform: openstack: computeFlavor:	用于 control plane 和计算机器的 RHOSP 类别。 此属性已弃用。要将 flavor 用作所有机器池的默认值，请将其添加为 **platform.openstack.defaultMachinePlatform** 属性中的 **type** 键的值。您还可以分别为每个机器池设置 flavor 值。	字符串，如 **m1.xlarge**。

1. **如果机器池定义了 zones，则类型的数量可以是单个项目，也可以是与 zones 中的项目数量匹配。例如，如果 zones 中有 3 个项目，则类型的数量不能为 2。**

2. **如果您对此属性有任何现有引用，安装程序会在 controlPlane.platform.openstack.rootVolume.types 字段中填充对应的值。**

21.10.1.5. 可选的 RHOSP 配置参数

下表描述了可选的 RHOSP 配置参数：

表 21.10. 可选的 RHOSP 参数

参数	描述	值

compute:
  platform:
  openstack:
 	
additionalNetworkIDs:
	与计算机器关联的其他网络。不为额外网络创建允许的地址对。	一个或多个 UUID 的列表作为字符串。例如：**fa806b2f-ac49-4bce-b9db-124bc64209bf**。

compute:
  platform:
  openstack:
 	
additionalSecurityGroupIDs:
	与计算机器关联的其他安全组。	一个或多个 UUID 的列表作为字符串。例如，**7ee219f3-d2e9-48a1-96c2-e7429f1b0da7**。

参数	描述	值
compute: platform: openstack: zones:	RHOSP Compute(Nova)可用区(AZ)在其中安装机器。如果没有设置此参数，安装程序会依赖于配置了 RHOSP 管理员的 Nova 的默认设置。	字符串列表。例如：**["zone-1", "zone-2"]**。
compute: platform: openstack: serverGroupPolicy:	要应用到包含池中的计算机器的组策略。您不能在创建后更改服务器组策略或关系。支持的选项包括 **anti-affinity**, **soft-affinity**, 和 **soft-anti-affinity**默认值为 **soft-anti-affinity**。 **affinity** 策略可防止迁移，因此会影响 RHOSP 升级。不支持 **affinity** 策略。 如果使用严格的 **anti-affinity** 策略，实例迁移过程中需要额外的 RHOSP 主机。	应用到机器池的服务器组策略。例如，**soft-affinity**。
controlPlane: platform: openstack: additionalNetworkIDs:	与 control plane 机器关联的额外网络。不为额外网络创建允许的地址对。 附加到 control plane 机器的额外网络也会附加到 bootstrap 节点。	一个或多个 UUID 的列表作为字符串。例如：**fa806b2f-ac49-4bce-b9db-124bc64209bf**。
controlPlane: platform: openstack: additionalSecurityGroupIDs:	与 control plane 机器关联的其他安全组。	一个或多个 UUID 的列表作为字符串。例如，**7ee219f3-d2e9-48a1-96c2-e7429f1b0da7**。
controlPlane: platform: openstack: zones:	RHOSP Compute(Nova)可用区(AZ)在其中安装机器。如果没有设置此参数，安装程序会依赖于配置了 RHOSP 管理员的 Nova 的默认设置。	字符串列表。例如：**["zone-1", "zone-2"]**。

参数	描述	值
controlPlane: platform: openstack: serverGroupPolicy:	要应用到池中包含 control plane 机器的组的服务器组策略。您不能在创建后更改服务器组策略或关系。支持的选项包括 **anti-affinity**, **soft-affinity**, 和 **soft-anti-affinity**默认值为 **soft-anti-affinity**。 **affinity关联性** 策略会阻止迁移，因此会影响 RHOSP 升级。不支持 **affinity** 策略。 如果使用严格的 **anti-affinity** 策略，实例迁移过程中需要额外的 RHOSP 主机。	应用到机器池的服务器组策略。例如，**soft-affinity**。
platform: openstack: clusterOSImage:	安装程序从中下载 RHCOS 镜像的位置。 您必须设置此参数才能在受限网络中执行安装。	HTTP 或 HTTPS URL，可选使用 SHA-256 校验和。 例如： **http://mirror.example.com/images/rhcos-43.81.201912131630.0-openstack.x86_64.qcow2.gz?sha256=ffebbd68e8a1f2a245ca19522c16c86f67f9ac8e4e0c1f0a812b068b16f7265d**。该值也可以是现有 Glance 镜像的名称，如 **my-rhcos**。
platform: openstack: clusterOSImageProperties:	在 Glance 中添加至安装程序上传的 ClusterOSImage 的属性。如果将 **platform.openstack.clusterOSImage** 设置为现有的 Glance 镜像，则此属性将被忽略。 您可以使用此属性超过每个节点 26 个 PV 的 RHOSP 的默认持久性卷(PV)限值。要超过这个限制，请将 virtio-**scsi_model** 属性值设置为 **virtio-scsi，** 并将 **hw_disk_bus** 值设置为 **scsi**。 您还可以使用此属性通过包含值为 **yes** 的 **hw_qemu_guest_agent** 属性来启用 QEMU 客户机代理。	键值字符串对列表。例如，**["hw_scsi_model": "virtio-scsi", "hw_disk_bus": "scsi"]**。

参数	描述	值

platform:
  openstack:
 	
defaultMachinePlatform:
	默认机器池平台配置。	<pre>{
		  "type": "ml.large",
		  "rootVolume": {
		  "size": 30,
		  "type": "performance"
		  }
		}</pre>

platform: openstack: ingressFloatingIP:	与 Ingress 端口关联的现有浮动 IP 地址。要使用此属性，还必须定义 **platform.openstack.externalNetwork** 属性。	IP 地址，如 **128.0.0.1**。
platform: openstack: apiFloatingIP:	与 API 负载均衡器关联的现有浮动 IP 地址。要使用此属性，还必须定义 **platform.openstack.externalNetwork** 属性。	IP 地址，如 **128.0.0.1**。
platform: openstack: externalDNS:	集群实例用于 DNS 解析的外部 DNS 服务器的 IP 地址。	IP 地址列表作为字符串。例如，**["8.8.8.8", "192.168.1.12"]**。
platform: openstack: loadbalancer:	是否使用默认的内部负载均衡器。如果值设为 **UserManaged**，则此默认负载均衡器被禁用，以便您可以部署使用外部用户管理的负载均衡器的集群。如果没有设置该参数，或者值为 **OpenShiftManagedDefault**，集群将使用默认负载均衡器。	**UserManaged** 或 **OpenShiftManagedDefault**.

参数	描述	值
platform: openstack: machinesSubnet:	集群节点使用的 RHOSP 子网的 UUID。在这个子网上创建节点和虚拟 IP(VIP)端口。 **networking.machineNetwork** 中的第一个项必须与 machine **Subnet** 的值匹配。 如果部署到自定义子网中，则无法将外部 DNS 服务器指定到 OpenShift Container Platform 安装程序。相反，将 DNS 添加到 RHOSP 中的子网中。	作为字符串的 UUID。例如： **fa806b2f-ac49-4bce-b9db-124bc64209bf**。

第 22 章 在 OCI 上安装

22.1. 使用辅助安装程序在 ORACLE CLOUD INFRASTRUCTURE (OCI) 上安装集群

在 OpenShift Container Platform 4.16 及更新的版本中，您可以使用 Assisted Installer 使用您提供的基础架构在 Oracle® Cloud Infrastructure (OCI)上安装集群。

22.1.1. 支持的安装程序和 OCI 概述

您可以在支持专用、混合、公共和多个云环境的 Oracle® 云基础架构 (OCI) 基础架构上运行集群工作负载。红帽和 Oracle 都测试、验证和支持在 OCI 上的 OpenShift Container Platform 集群中运行 OCI。

Assisted Installer 支持 OCI 平台，您可以使用 Assisted Installer 访问直观的互动工作流，以便在 OCI 上自动化集群安装任务。

OCI 提供可满足您的法规合规性、性能和成本效益的服务。您可以访问 OCI 资源管理器配置来置备和配置 OCI 资源。

重要

置备 OCI 资源的步骤仅作为示例。您还可以选择通过其他方法创建所需的资源；脚本只是一个示例。使用您提供的基础架构安装集群需要了解云供应商和 OpenShift Container Platform 上的安装过程。您可以访问 OCI Resource Manager 配置来完成这些步骤，或使用配置来建模您自己的自定义脚本。

按照 *使用 Assisted Installer 文档在 Oracle Cloud Infrastructure (OCI)* 上安装集群以了解如何使用 Assisted Installer 在 OCI 上安装 OpenShift Container Platform 集群的步骤。本文档演示了使用 OCI Cloud Controller Manager (CCM) 和 Oracle 的 Container Storage Interface (CSI) 对象来将 OpenShift Container Platform 集群与 OCI API 链接。

重要

为确保在 OCI 上运行的集群工作负载的最佳性能条件，请确保为您的工作负载调整块卷性能单元(VPU)。以下列表提供了选择特定性能需要 VPU 的指导：

- 测试或概念验证环境：100 GB，20 到 30 个 VPU。
- 基本环境：500 GB 和 60 个 VPU。
- 大型生产环境：500 GB 和 100 个或更多 VPU。

考虑保留额外的 VPU，以便为更新和扩展活动提供足够的容量。有关 VPU 的更多信息，请参阅卷性能单元 (Oracle 文档)。

如果您不熟悉 OpenShift Container Platform Assisted Installer，请参阅 "Assisted Installer for OpenShift Container Platform"。

其他资源

- OpenShift Container Platform 支持的安装程序
- OpenShift Container Platform 互联网访问
- 卷性能单元 (Oracle 文档)
- 在 OCI 节点 (Oracle) 文档中为 OpenShift Container Platform 大小建议实例

22.1.2. 创建 OCI 资源和服务

创建 Oracle® Cloud Infrastructure (OCI) 资源和服务，以便您可以建立符合您机构要求的监管标准的基础架构。

先决条件

- 已将 OCI 帐户配置为托管集群。请参阅 先决条件(Oracle 文档)。

流程

1. 使用管理员权限登录到您的 Oracle Cloud Infrastructure (OCI) 帐户。

2. 从 Oracle 资源下载存档文件。归档文件包含用于创建集群资源和自定义清单的文件。归档文件还包括脚本，运行该脚本时，脚本会创建 OCI 资源，如 DNS 记录、实例等。如需更多信息，请参阅配置文件(Oracle 文档)。

22.1.3. 使用辅助安装程序生成 OCI 兼容的发现 ISO 镜像

生成一个发现 ISO 镜像，并将镜像上传到 Oracle® Cloud Infrastructure (OCI)，以便代理可以在 OCI 上安装 OpenShift Container Platform 集群前执行硬件和网络验证检查。

在 OCI web 控制台中创建以下资源：

- 用于更好地组织、限制访问和设置 OCI 资源使用限制的比较。

- 用于安全存储发现 ISO 镜像的对象存储存储桶。您可以在稍后阶段访问镜像，用于引导实例，以便您可以创建集群。

先决条件

- 您在 OCI 上创建子 compartment 和对象存储桶。请参阅 Oracle 文档中的置备云基础架构 (OCI Console)。

- 您可以参阅有关 OpenShift Container Platform 安装和更新流程的详细信息。

- 如果使用防火墙并计划使用 Telemetry 服务，则将防火墙配置为允许 OpenShift Container Platform 访问所需的站点。

- 在创建虚拟机 (VM) 前，请参阅云实例类型（红帽生态系统目录门户） 来识别支持的 OCI 虚拟机。

流程

1. **在混合云控制台上的 Install OpenShift with the Assisted Installer 页面中，通过完成所有必要的 Assisted Installer 步骤来生成发现 ISO 镜像。**

   a. **在 Cluster Details 步骤中，完成以下字段：**

字段	所需的操作
集群名称	指定集群的名称，如 **ocidemo**。
基域	指定集群的基域，如 **splat-oci.devcluster.openshift.com**。您之前在 OCI 上创建 compartment 时，您可以通过进入 **DNS management → Zones → List 范围** 来获取此信息，然后选择父 compartment。您的基域应当会显示在**公共区域**选项卡下。
OpenShift version	指定 **OpenShift 4.16** 或更高版本。
CPU 架构	指定 **x86_64** 或 **Arm64**。
与外部合作伙伴平台集成	指定 **Oracle Cloud Infrastructure**。 指定这个值后，默认选择 **Include custom manifests** 复选框。

   b. **在 Operators 页面上，单击 Next。**

   c. **在 Host Discovery 页面上，点 Add hosts。**

   d. **对于 SSH 公钥 字段，请从本地系统添加 SSH 密钥。**

   提示

   您可以使用 ssh-keygen 工具创建 SSH 身份验证密钥对。

   e. **点 Generate Discovery ISO 生成发现 ISO 镜像文件。**

f. 将文件下载到您的本地系统。

2. 将发现 ISO 镜像上传到 OCI 存储桶。请参阅将对象存储对象上传到 Bucket (Oracle 文档)。

a. 您必须为上传的发现 ISO 镜像创建一个预先验证的请求。确保您记下了预验证请求的 URL，因为在创建 OCI 堆栈时，您必须在稍后阶段指定 URL。

其他资源

- 安装和更新
- 配置防火墙

22.1.4. 为集群置备 OCI 基础架构

通过使用 Assisted Installer 为 OpenShift Container Platform 集群创建详情，您可以在堆栈中指定这些详情。堆栈(stack)是一个 OCI 功能，您可以自动置备所有所需的 OCI 基础架构资源，如自定义镜像，这是在 OCI 上安装 OpenShift Container Platform 集群所需的。

Oracle® Cloud Infrastructure (OCI) Compute Service 在 OCI 上创建虚拟机(VM)实例。然后，这个实例可以自动附加到虚拟网络(VNC)子网中的虚拟网络接口控制器(vNIC)。在自定义清单模板文件中指定 OpenShift Container Platform 集群的 IP 地址时，OCI 实例可以通过 VNC 与集群通信。

先决条件

- 将发现 ISO 镜像上传到 OCI 存储桶。如需更多信息，请参阅"使用辅助安装程序来生成 OCI 兼容的发现 ISO 镜像"。

流程

1. 为 OpenShift Container Platform 集群置备 OCI 基础架构的步骤。请参阅使用资源管理器(Oracle 文档)创建 OpenShift Container Platform 基础架构。

2. 创建一个堆栈，然后根据编辑 OpenShift 自定义清单 (Oracle 文档)中的步骤编辑自定义清单文件。

22.1.5. 完成剩余的 Assisted Installer 步骤

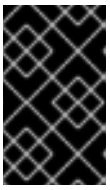
置备 Oracle® Cloud Infrastructure (OCI) 资源并将 OpenShift Container Platform 自定义清单文件上传到 OCI 后，您必须在 Assisted Installer 上完成剩余的集群安装步骤，然后才能创建实例 OCI。

先决条件

- 您在 OCI 上创建资源堆栈，其中包含自定义清单文件和 OCI 资源管理器配置资源。请参阅 "Provisioning OCI infrastructure for your cluster"。

流程

1. 在 Red Hat Hybrid Cloud Console web 控制台中进入 Host discovery 页面。
2. 在 Role 列下，为每个目标主机名选择 Control plane 节点或 Worker。



重要

在以前的版本中，您可以继续后续步骤，等待每个节点达到 Ready 状态。

3. 接受 Storage 和 Networking 步骤的默认设置，然后点 Next。
4. 在 Custom manifests 页面上的 Folder 字段中，选择 manifest。这是您要保存自定义清单文件的 Assisted Installer 文件夹。
   a. 在 File name 字段中，输入一个值，如 oci-ccm.yml。
   b. 在 Content 部分，点 Browse，然后从您的驱动器中选择位于 custom_manifest/manifests/oci-ccm.yml 的 CCM 清单。
5. 展开下一个 Custom manifest 部分，并为以下清单重复相同的步骤：
   - CSI 驱动程序清单：custom_manifest/manifests/oci-csi.yml

- CCM 机器配置： custom_manifest/openshift/machineconfig-ccm.yml
- CSI 驱动程序机器配置：custom_manifest/openshift/machineconfig-csi.yml

6. 在 Review and create 页面中，点 Install cluster 在 OCI 上创建 OpenShift Container Platform 集群。

在集群安装和初始化操作后，辅助安装程序表示集群安装操作完成。如需更多信息，请参阅 *Assisted Installer for OpenShift Container Platform* 文档中的"完成安装"部分。

其他资源

- OpenShift Container Platform 支持的安装程序

22.1.6. 在 OCI 上验证集群安装是否成功

验证集群是否已安装并在 Oracle® Cloud Infrastructure (OCI) 上有效运行。

流程

1. 在 Hybrid Cloud Console 中，进入 Clusters > Assisted Clusters 并选择集群名称。
2. 检查安装进度条是否处于 100%，并显示 "Installation completed successfully" 信息。
3. 要访问 OpenShift Container Platform Web 控制台，请点提供的 Web 控制台 URL。
4. 前往 Nodes 菜单页面。
5. 从 Nodes 表中查找您的节点。
6. 在 Overview 选项卡中，检查您的节点的状态是否为 Ready。

7. 选择 **YAML** 选项卡。

8. 检查 **labels** 参数，并验证列出的标签是否应用到您的配置。例如，**topology.kubernetes.io/region=us-sanjose-1** 标签指示节点部署的 OCI 区域。

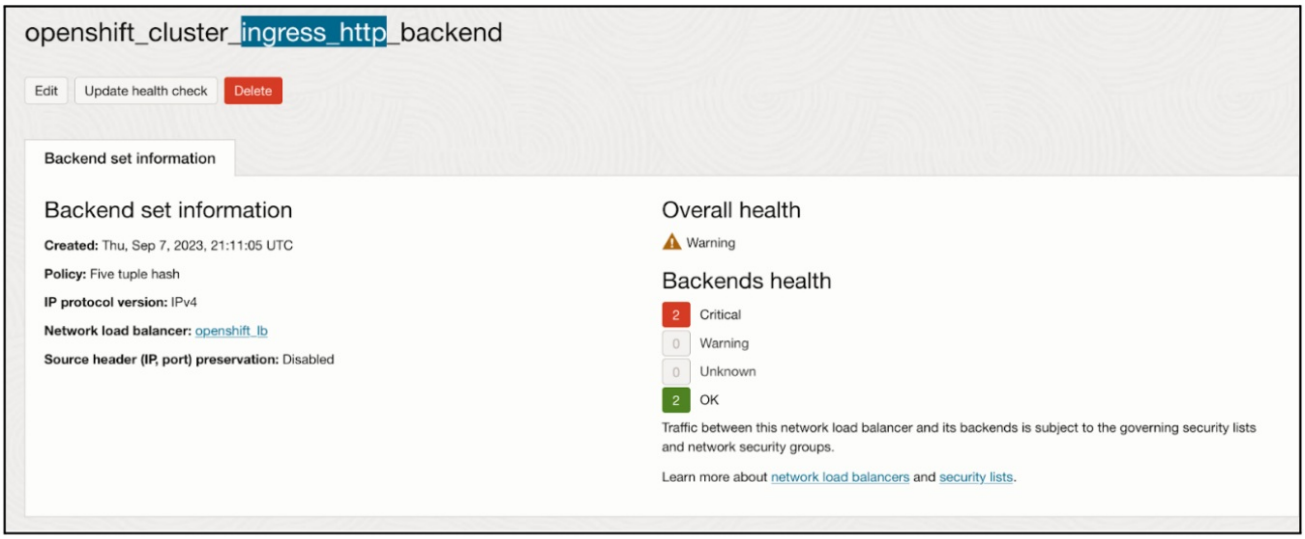
22.1.7. 对在 OCI 上安装集群进行故障排除

如果您在使用 Assisted Installer 在 Oracle® Cloud Infrastructure (OCI)上安装 OpenShift Container Platform 集群时遇到问题，请阅读以下部分以排除常见问题。

OCI 中的 Ingress Load Balancer 处于健康状态

此问题被归类为 Warning，因为通过使用 Resource Manager 创建堆栈，您创建了计算节点池 3，默认情况下会自动添加为 Ingress Load Balancer 的后端监听程序。默认情况下，OpenShift Container Platform 部署 2 个路由器 Pod，它们基于 OpenShift Container Platform 清单文件中的默认值。Warning 是正常的，因为存在与可用路由器 Pod 数量时出现的不匹配(2 个)可在三个计算节点上运行。

图 22.1. OCI 上 后端设置信息选项卡下的 Warning 消息示例：



您不需要修改 Ingress Load Balancer 配置。相反，您可以将 Ingress Load Balancer 指向在 OpenShift Container Platform 上在集群中运行的特定计算节点。要做到这一点，您需要在 OpenShift Container Platform 上使用放置机制，如注解，以确保路由器 pod 仅在您最初作为后端监听程序配置的计算节点上运行。

OCI 创建堆栈操作失败，并显示 Error: 400-InvalidParameter 消息

尝试在 OCI 上创建堆栈时，您确定作业的 Logs 部分会输出错误消息。例如：

```
Error: 400-InvalidParameter, DNS Label oci-demo does not follow Oracle requirements
```

> **Suggestion: Please update the parameter(s) in the Terraform config as per error message DNS Label oci-demo does not follow Oracle requirements Documentation: https://registry.terraform.io/providers/oracle/oci/latest/docs/resources/core_vcn**

进入混合云控制台上的使用 Assisted Installer 安装 OpenShift，并检查 Cluster Details 步骤中的 Cluster name 字段。从名称中删除任何特殊字符，如连字符(-)，因为这些特殊字符与 OCI 命名约定不兼容。例如，将 oci-demo 更改为 ocidemo。

其他资源

- 对 OCI 上的 OpenShift Container Platform 进行故障排除 (Oracle 文档)
- 使用 Assisted Installer 安装内部集群

22.2. 使用基于代理的安装程序在 ORACLE CLOUD INFRASTRUCTURE (OCI) 上安装集群

在 OpenShift Container Platform 4.16 中，您可以使用基于代理的安装程序在 Oracle® Cloud Infrastructure (OCI) 上安装集群，以便您可以在支持专用、混合、公共和多个云环境的基础架构上运行集群工作负载。

22.2.1. 基于代理的安装程序和 OCI 概述

您可以使用基于代理的安装程序在 Oracle® Cloud Infrastructure (OCI) 上安装 OpenShift Container Platform 集群。红帽和 Oracle 测试、验证和支持在 OCI 上的 OpenShift Container Platform 集群中运行 OCI 和 Oracle® Cloud VMware Solution (OCVS) 工作负载。

基于代理的安装程序提供了辅助安装服务的易用性，但可以在连接或断开连接的环境中安装集群。

下图显示了连接的和断开连接的环境的工作流：

图 22.2. 在连接的环境中使用基于代理的安装程序在 OCI 上安装集群的工作流

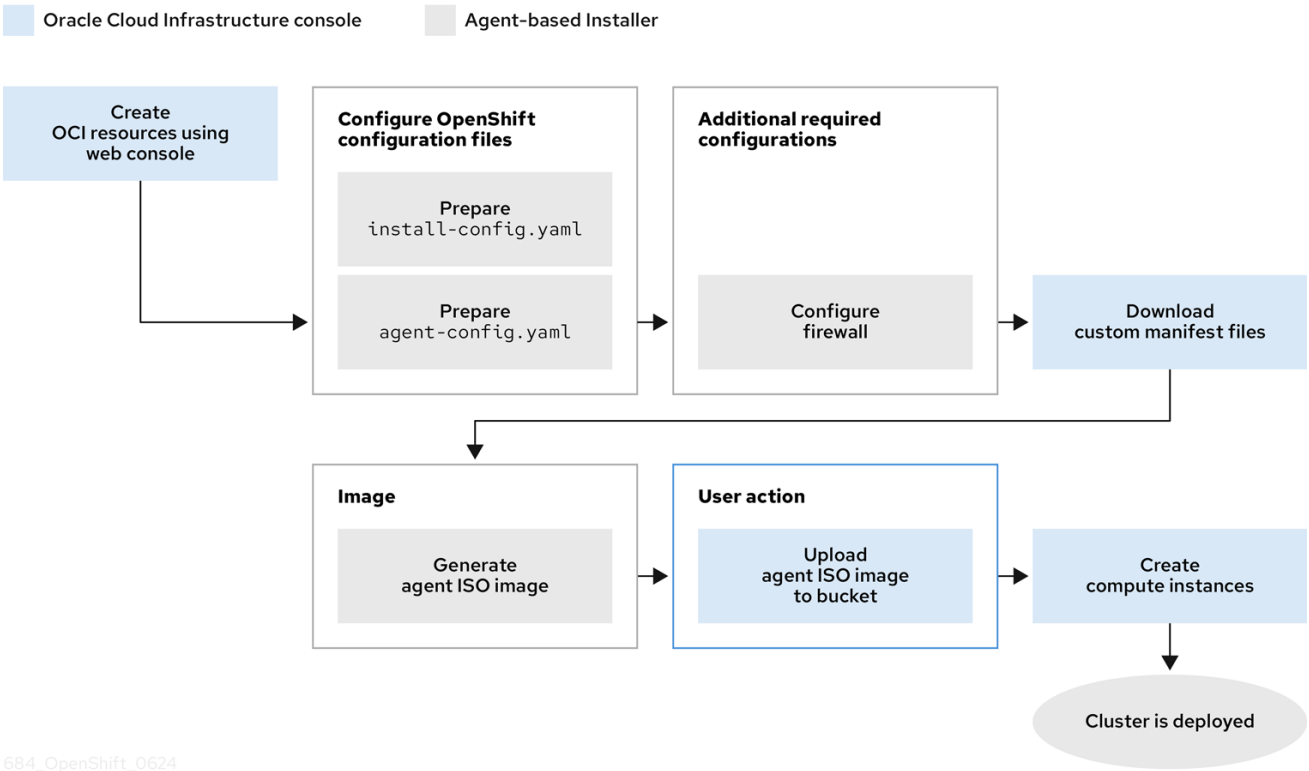
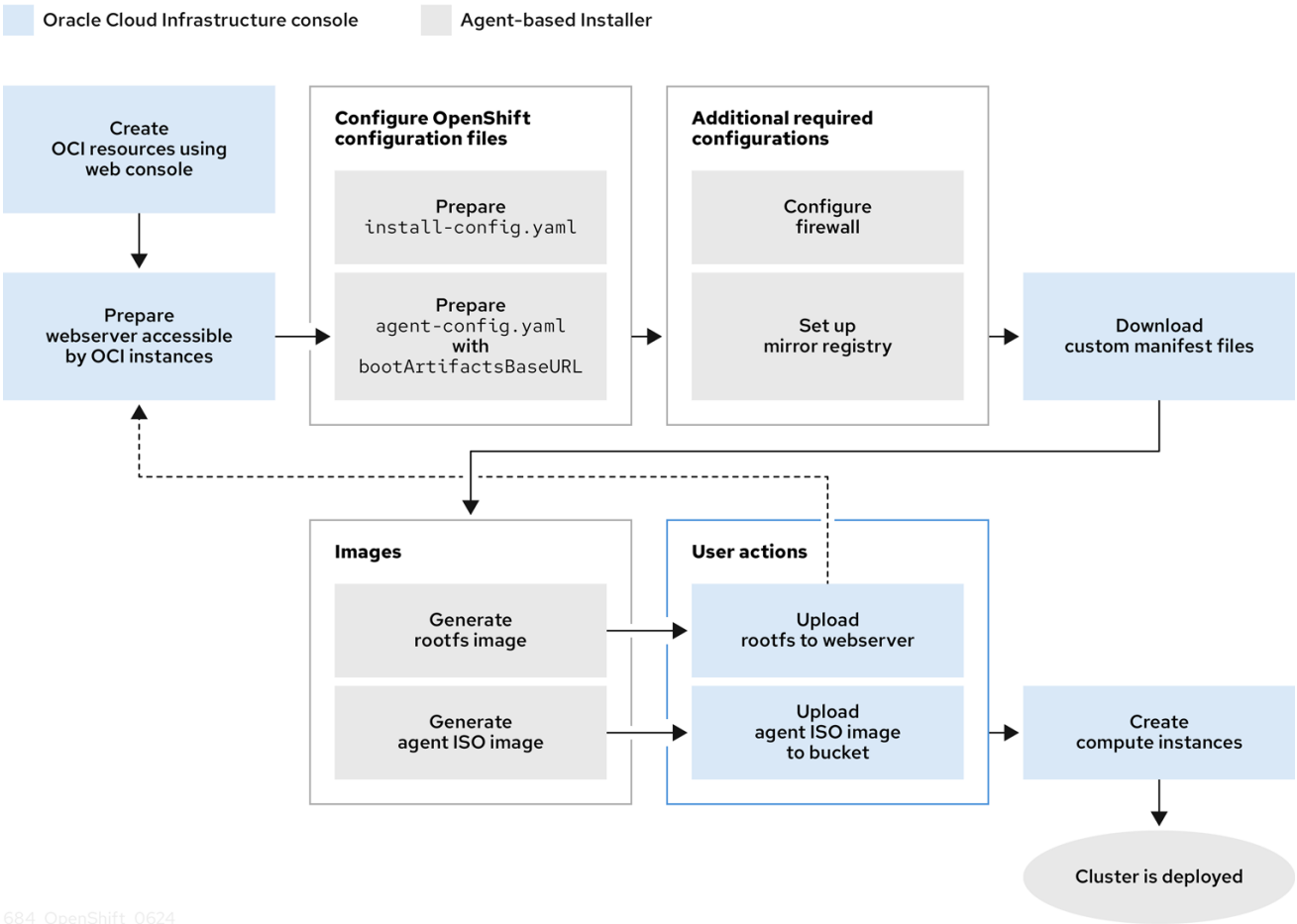


图 22.3. 在断开连接的环境中使用基于代理的安装程序在 OCI 上安装集群的工作流



OCI 提供可满足您的法规合规性、性能和符合成本效益的服务。OCI 支持 64 位 x86 实例和 64 位 ARM 实例。另外，OCI 提供了一个 OCVS 服务，您可以在其中使用最小应用程序重新架构将 VMware 工作负载移到 OCI 中。

注意

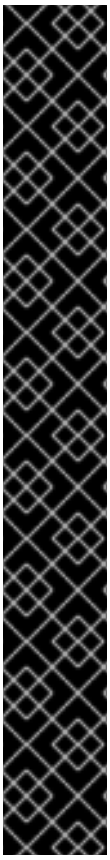
考虑为您的引导磁盘选择一个非易失性内存表达(NVMe)驱动器或固态驱动器(SSD)，因为这些驱动器为您的引导磁盘提供低延迟和高吞吐量功能。

通过在 OCI 上运行 OpenShift Container Platform 集群，您可以访问以下功能：

- 计算灵活性形成，您可以在其中自定义虚拟机的 Oracle® CPU (OCPU)和内存资源的数量。通过访问这个功能，集群的工作负载可以在资源平衡的环境中执行操作。您可以通过进入红帽生态系统目录门户上的 Oracle 页面来找到所有 RHEL 认证的 OCI 结构。

- 块卷存储，您可以在其中为存储卷配置扩展和自动调整设置，以便块卷服务自动调整性能级别以优化性能。

- OCVS，您可以在其中在 VMware® vSphere 软件定义的数据中心 (SDDC) 上运行的公共云环境中部署集群。继续保持对 VMware vSphere 环境的完全管理控制，但您可以使用 OCI 服务在灵活、可扩展和安全基础架构上提高应用程序。



重要

为确保在 OCI 和 OCVS 服务上运行的集群工作负载的最佳性能条件，请确保您的块卷的卷性能单元(VPU)的大小是针对您的工作负载的大小。以下列表提供了一些有关选择特定性能需要 VPU 的指导信息：

- 测试或概念验证环境：100 GB，20 到 30 个 VPU。
- 基本环境：500 GB 和 60 个 VPU。
- 大型生产环境：500 GB 和 100 个或更多 VPU。

考虑保留额外的 VPU，以便为更新和扩展活动提供足够的容量。有关 VPU 的更多信息，请参阅卷性能单元 (Oracle 文档)。

其他资源

- 安装过程
- OpenShift Container Platform 互联网访问
- 了解基于代理的安装程序
- Compute 服务概述 (Oracle 文档)
- 卷性能单元 (Oracle 文档)
- 在 OCI 节点上为 OpenShift Container Platform 大小建议实例(Oracle 文档)

22.2.2. 创建 OCI 基础架构资源和服务

您必须在虚拟机(VM)上创建一个 OCI 环境。通过创建此环境，您可以安装 OpenShift Container Platform，并在支持广泛的云选项和强大的安全策略的基础架构上部署集群。预先了解 OCI 组件的相关

知识可帮助您了解 OCI 资源的概念以及如何配置它们来满足您的机构需求。

在 OCI 上安装 OpenShift Container Platform 集群的基于 Agent 的安装程序方法需要您手动创建 OCI 资源和服务。

重要

为确保与 OpenShift Container Platform 兼容，您必须将 A 设置为每个 DNS 记录和名称记录的记录类型，如下所示：

- api.<cluster_name>.<base_domain>，它以 API 负载均衡器的 apiVIP 参数为目标。
- api-int.<cluster_name>.<base_domain>，它以 API 负载均衡器的 apiVIP 参数为目标。
- *.apps.<cluster_name>.<base_domain>，它以 Ingress 负载均衡器的 ingressVIP 参数为目标。

api.* 和 api-int.* DNS 记录与 control plane 机器相关，因此您必须确保安装的 OpenShift Container Platform 集群中的所有节点都可以访问这些 DNS 记录。

先决条件

- 已将 OCI 帐户配置为托管 OpenShift Container Platform 集群。请参阅 先决条件(Oracle 文档)。

流程

- 创建所需的 OCI 资源和服务。请参阅使用基于代理的安装程序(Oracle 文档)所需的 OCI 资源。

其他资源

- 了解 Oracle 云基础知识(Oracle 文档)

22.2.3. 创建用于在 OCI 上安装集群的配置文件

您需要创建 install-config.yaml 和 agent-config.yaml 配置文件，以便您可以使用基于 Agent 的安装程序来生成可引导 ISO 镜像。基于代理的安装包含一个可引导 ISO，它带有辅助发现代理和辅助服务。这两个组件都需要执行集群安装，但后者的组件仅在其中一个主机上运行。

在以后的阶段，您必须按照 Oracle 文档中的步骤将生成的代理 ISO 镜像上传到 Oracle 的默认对象存储存储桶，这是在 Oracle® Cloud Infrastructure (OCI)上集成 OpenShift Container Platform 集群的初始步骤。

注意

您还可以使用基于代理的安装程序来生成或接受 Zero Touch Provisioning (ZTP)自定义资源。

先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新流程的详细信息。
- 您可以阅读有关选择集群安装方法的文档，并为用户准备相关的环境。
- 您已阅读了"准备基于代理的安装程序"文档。
- 您已从 Red Hat Hybrid Cloud Console 下载了基于代理的安装程序和命令行界面 (CLI)。
- 已使用管理员权限登录到 OpenShift Container Platform。

流程

1. 对于断开连接的环境，将 Red Hat OpenShift 的镜像 registry 镜像到本地容器镜像 registry。

重要

检查您的 openshift-install 二进制版本是否与本地镜像容器 registry 相关，而不是共享的 registry，如 Red Hat Quay。

```
$ ./openshift-install version
```

共享 registry 二进制文件的输出示例

```
./openshift-install 4.16.0
built from commit ae7977b7d1ca908674a0d45c5c243c766fa4b2ca
release image registry.ci.openshift.org/origin/release:4.16ocp-
release@sha256:0da6316466d60a3a4535d5fed3589feb0391989982fba59d
47d4c729912d6363
release architecture amd64
```

2. 配置 install-config.yaml 配置文件以满足您的机构需求。

演示设置外部平台的 install-config.yaml 配置文件示例

```
# install-config.yaml
apiVersion: v1
baseDomain: <base_domain> (1)
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  network type: OVNKubernetes
  machineNetwork:
  - cidr: <ip_address_from_cidr> (2)
  serviceNetwork:
  - 172.30.0.0/16
compute:
  - architecture: amd64 (3)
  hyperthreading: Enabled
  name: worker
  replicas: 0
controlPlane:
  architecture: amd64 (4)
  hyperthreading: Enabled
  name: master
```

```
  replicas: 3
platform:
  external:
   platformName: oci 5
   cloudControllerManager: External
sshKey: <public_ssh_key> 6
pullSecret: '<pull_secret>' 7
# ...
```

1

云供应商的基域。

2

CIDR 分配给网络上操作的资源和组件的虚拟网络 (VCN) 的 IP 地址。

3 4

根据您的基础架构，您可以选择 x86_64 或 amd64。

5

将 OCI 设置为外部平台，以便 OpenShift Container Platform 能够与 OCI 集成。

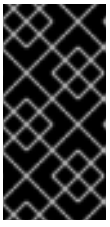
6

指定 SSH 公钥。

7

为 OpenShift Container Platform 组件和服务（如 Quay.io）下载容器镜像时，您需要验证目的的 pull secret。请参阅 通过 Red Hat Hybrid Cloud Console 安装 OpenShift Container Platform 4。

3. 在本地系统上创建一个名为 openshift 的目录。



重要

不要将 install-config.yaml 和 agent-config.yaml 配置文件移到 openshift 目录中。

4. 完成 Oracle 文档的 "配置文件" 部分中的步骤，以下载 *Oracle* Cloud Controller Manager (CCM)和 Oracle Container Storage Interface (CSI)清单作为归档文件，并将存档文件保存到 openshift 目录中。您需要 Oracle CCM 清单以在集群安装过程中部署 Oracle CCM，以便 OpenShift Container Platform 可以连接到外部 OCI 平台。在集群安装过程中部署 Oracle CSI 驱动程序需要 Oracle CSI 自定义清单，以便 OpenShift Container Platform 可以从 OCI 声明所需的对象。

5. 访问 *Oracle* 文档的" 配置文件" 部分中提供的自定义清单文件。

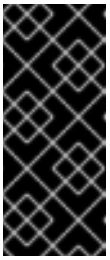
   a. 更改 oci-cloud-controller-manager secret，该 secret 在 oci-ccm.yml 配置文件中定义，以匹配您的机构的区域、比较 OCID、VCN OCID 和子网 OCID。

6. 在 OpenShift Container Platform CLI 中输入以下命令来，使用基于 Agent 的安装程序来生成最小 ISO 镜像（不包括 rootfs 镜像）。您可以在稍后使用此镜像引导所有集群节点。

```
$ ./openshift-install agent create image --log-level debug
```

   该命令还会完成以下操作：

   - 创建子目录 ./<installation_directory>/auth directory: : 将 kubeadmin-password 和 kubeconfig 文件放在 子目录中。
   - 根据您在 agent-config.yaml 配置文件中指定的 IP 地址，创建一个 rendezvousIP 文件。
   - 可选：您对 agent-config.yaml 和 install-config.yaml 配置文件所做的任何修改都会导入到 Zero Touch Provisioning (ZTP)自定义资源。



重要

基于代理的安装程序使用 Red Hat Enterprise Linux CoreOS (RHCOS)。启动、恢复和修复您的操作系统需要在后续列出的项中提到的 rootfs 镜像。

7. 配置 agent-config.yaml 配置文件，以满足您的机构要求。

为 IPv4 格式的网络设置值的 agent-config.yaml 配置文件示例。

```
apiVersion: v1alpha1
metadata:
 name: <cluster_name> 1
 namespace: <cluster_namespace> 2
rendezvousIP: <ip_address_from_CIDR> 3
bootArtifactsBaseURL: <server_URL> 4
# ...
```

1 您在 DNS 记录中指定的集群名称。

2 OpenShift Container Platform 上集群的命名空间。

3 如果您使用 IPv4 作为网络 IP 地址格式，请确保将 rendezvousIP 参数设置为在网络上分配的 VCN 的无类别域间路由 (CIDR) 方法的 IPv4 地址。另外，确保至少有一个实例来自您使用 ISO 引导的实例池中，与您为 rendezvousIP 设置的 IP 地址值匹配。

4 要上传 rootfs 镜像的服务器的 URL。

8. 对 agent-config.yaml 配置文件应用以下两个更新之一：

- 对于断开连接的网络：运行该命令以生成最小 ISO 镜像后，基于代理的安装程序会将 rootfs 镜像保存到本地系统的 ./<installation_directory>/boot-artifacts 目录中。使用您首选的 Web 服务器，如任何 Hypertext 传输协议守护进程(httpd)，将 rootfs 上传到 agent-config.yaml 配置文件中的 bootArtifactsBaseURL 参数中声明的位置。

  例如，如果 bootArtifactsBaseURL 参数状态 http://192.168.122.20，您可以将生成的 rootfs 镜像上传到此位置，以便基于代理的安装程序可以从 http://192.168.122.20/agent.x86_64-rootfs.img 访问镜像。在基于代理的安装程序为外部平

台引导最小 ISO 后，基于代理的安装程序将 rootfs 镜像从 http://192.168.122.20/agent.x86_64-rootfs.img 位置下载到系统内存中。

注意

基于代理的安装程序还将 bootArtifactsBaseURL 的值添加到最小 ISO 镜像的配置中，以便在 Operator 引导集群节点时，基于代理的安装程序会将 rootfs 镜像下载到系统内存中。

- 对于连接的网络： 您不需要在 agent-config.yaml 配置文件中指定 bootArtifactsBaseURL 参数。基于代理的安装程序的默认行为从 https://rhcos.mirror.openshift.com 读取 rootfs URL 位置。在基于代理的安装程序为外部平台引导最小 ISO 后，基于代理的安装程序，然后从默认的 RHCOS URL 将 rootfs 文件下载到您的系统内存中。

重要

考虑超过 1 GB 的完整 ISO 镜像包括 rootfs 镜像。镜像大于最小 ISO 镜像，该镜像通常小于 150 MB。

其他资源

- 关于 OpenShift Container Platform 安装
- 选择集群安装类型
- 准备使用基于代理的安装程序安装
- 下载基于代理的安装程序
- 镜像 OpenShift Container Platform 镜像存储库
- 可选： 使用 ZTP 清单

22.2.4. 为 OpenShift Container Platform 配置防火墙

在安装 OpenShift Container Platform 前，您必须配置防火墙，以授予 OpenShift Container Platform 所需站点的访问权限。在使用防火墙时，为防火墙提供额外的配置，以便 OpenShift Container Platform 可以访问正常工作所需的站点。

对于断开连接的环境，您必须从 Red Hat 和 Oracle 镜像内容。此环境要求您创建防火墙规则，将防火墙公开给特定端口和 registry。

注意

如果您的环境在 OpenShift Container Platform 集群前面有一个专用的负载均衡器，请查看防火墙和负载均衡器之间的允许列表，以防止对集群造成不必要的网络限制。

流程

1. 为您的防火墙的允许列表设置以下 registry URL：

URL	port	功能
registry.redhat.io	443	提供核心容器镜像
access.redhat.com [1]	443	托管存储在 Red Hat Ecosytem Catalog 中的所有容器镜像，包括核心容器镜像。
quay.io	443	提供核心容器镜像
cdn.quay.io	443	提供核心容器镜像
cdn01.quay.io	443	提供核心容器镜像
cdn02.quay.io	443	提供核心容器镜像
cdn03.quay.io	443	提供核心容器镜像
cdn04.quay.io	443	提供核心容器镜像
cdn05.quay.io	443	提供核心容器镜像
cdn06.quay.io	443	提供核心容器镜像
sso.redhat.com	443	**https://console.redhat.com** 站点使用来自 **sso.redhat.com** 的身份验证

1.

在防火墙环境中，确保 access.redhat.com 资源位于允许列表中。此资源托管容器客户端在从 registry.access.redhat.com 中拉取镜像时验证镜像所需的签名存储。

您可以在 allowlist 中使用通配符 lfquay.io 和 lfopenshiftapps.com 而不是 cdn.quay.io 和 cdn0[1-3].quay.io。在 allowlist 中添加站点（如 quay.io ）时，不要向 denylist 添加通配符条目，如 *.quay.io。在大多数情况下，镜像 registry 使用内容交付网络（CDN）来提供镜像。如果防火墙阻止访问，则初始下载请求重定向到一个主机名（如 cdn01.quay.io ）时，镜像下载将被拒绝。

2. 将防火墙的允许列表设置为包含为构建所需的语言或框架提供资源的任何站点。

3. 如果不禁用 Telemetry，您必须授予对以下 URL 的访问权限，以访问 Red Hat Insights：

URL	port	功能
cert-api.access.redhat.com	443	Telemetry 所需
api.access.redhat.com	443	Telemetry 所需
infogw.api.openshift.com	443	Telemetry 所需
console.redhat.com	443	Telemetry 和 **insights-operator**需要

4. 将防火墙的允许列表设置为包含以下 registry URL：

URL	port	功能
api.openshift.com	443	集群令牌需要，并检查集群是否有可用的更新。
rhcos.mirror.openshift.com	443	需要此项以下载 Red Hat Enterprise Linux CoreOS(RHCOS)镜像。

5. 将防火墙的允许列表设置为包含以下外部 URL：每个存储库 URL 都托管 OCI 容器。考虑将镜像镜像到几个存储库，以减少任何性能问题。

URL	port	功能

URL	port	功能
k8s.gcr.io	port	为基于社区的镜像 registry 托管容器镜像的 Kubernetes registry。此镜像 registry 托管在自定义 Google Container Registry (GCR)域中。
ghcr.io	port	一个 GitHub 镜像 registry，您可以在其中存储和管理开放容器项目镜像。需要访问令牌发布、安装和删除私有、内部和公共软件包。
storage.googleapis.com	443	发行版本镜像签名源，但 Cluster Version Operator 只需要一个可正常工作的源。
registry.k8s.io	port	替换 **k8s.gcr.io** 镜像 registry，因为 **k8s.gcr.io** 镜像 registry 不支持其他平台和供应商。

22.2.5. 在 OCI 上运行集群

要在 Oracle® Cloud Infrastructure (OCI) 上运行集群，您必须将生成的代理 ISO 镜像上传到 OCI 上的默认 Object Storage 存储桶。另外，您必须从提供的基础镜像创建一个计算实例，以便 OpenShift Container Platform 和 OCI 能够相互通信，以便在 OCI 上运行集群。

注意

OCI 支持以下 OpenShift Container Platform 集群拓扑：

- **在单一节点上安装 OpenShift Container Platform 集群。**
- **高可用性集群至少有三个 control plane 实例和两个计算实例。**
- **紧凑的三节点集群，至少有三个 control plane 实例。**

先决条件

- **您生成了一个代理 ISO 镜像。请参阅"创建配置文件以在 OCI 上安装集群"部分。**

流程

1. 将代理 ISO 镜像上传到 Oracle 的默认 Object Storage 存储桶，并将代理 ISO 镜像作为自定义镜像导入到此存储桶。确保将自定义镜像配置为以统一可扩展固件接口(UEFI)模式引导。如需更多信息，请参阅创建 OpenShift Container Platform ISO 镜像(Oracle 文档)。

2. 从集群拓扑提供的基础镜像创建计算实例。请参阅在 OCI (Oracle 文档)上创建 OpenShift Container Platform 集群。

重要

在创建计算实例前，请检查您有足够的内存和磁盘资源。另外，请确保至少有一个计算实例与 agent-config.yaml 文件中 rendezvousIP 下声明的地址相同的 IP 地址。

其他资源

- 拓扑的建议资源
- 在 OCI 节点上为 OpenShift Container Platform 大小建议实例(Oracle 文档)
- 对 OCI 上的 OpenShift Container Platform 进行故障排除 (Oracle 文档)

22.2.6. 验证您的基于代理的集群安装是否在 OCI 上运行

验证集群是否已安装并在 Oracle® Cloud Infrastructure (OCI) 上有效运行。

先决条件

- 您创建了所有必需的 OCI 资源和服务。请参阅"创建 OCI 基础架构资源和服务"部分。
- 已创建 install-config.yaml 和 agent-config.yaml 配置文件。请参阅"创建配置文件以在 OCI 上安装集群"部分。
- 您已将代理 ISO 镜像上传到 Oracle 的默认 Object Storage 存储桶，并在 OCI 上创建计算实例。如需更多信息，请参阅"在 OCI 上运行集群"。

流程

在 OpenShift Container Platform 集群的自管理节点上部署计算实例后，您可以选择以下选项之一来监控集群的状态：

- **在 OpenShift Container Platform CLI 中输入以下命令：**

```
$ ./openshift-install agent wait-for install-complete --log-level debug
```

检查运行 bootstrap 节点的 rendezvous 主机节点的状态。主机重启后，集群的主机表单部分。

- **使用 kubeconfig API 检查各种 OpenShift Container Platform 组件的状态。对于 KUBECONFIG 环境变量，请设置集群的 kubeconfig 配置文件的相对路径：**

```
$  export KUBECONFIG=~/auth/kubeconfig
```

检查每个集群的自我管理的节点的状态。CCM 对每个节点应用标签，以指定在 OCI 上集群中运行的节点。

```
$ oc get nodes -A
```

输出示例

```
NAME                                    STATUS ROLES                AGE VERSION
main-0.private.agenttest.oraclevcn.com Ready  control-plane, master 7m
v1.27.4+6eeca63
main-1.private.agenttest.oraclevcn.com Ready  control-plane, master 15m
v1.27.4+d7fa83f
main-2.private.agenttest.oraclevcn.com Ready  control-plane, master 15m
v1.27.4+d7fa83f
```

检查每个集群的 Operator 的状态，其中 CCM Operator 状态是集群正在运行的良好指示。

```
$ oc get co
```

截断的输出示例

```
NAME          VERSION    AVAILABLE  PROGRESSING    DEGRADED   SINCE
MESSAGE
authentication 4.16.0-0    True       False          False      6m18s
baremetal      4.16.0-0    True       False          False      2m42s
network        4.16.0-0    True       True           False      5m58s  Progressing: …
    …
```

其他资源

- 从基于代理的安装收集日志数据

第 23 章 在 VSPHERE 上安装

23.1. 安装方法

您可以使用各种不同的安装方法在 vSphere 上安装 OpenShift Container Platform 集群。每种方法都有可能使其更适合不同用例的质量，比如在断开连接的环境中安装集群，或使用最小配置和置备安装集群。

23.1.1. 支持的安装程序

您可以使用 Assisted Installer 安装 OpenShift Container Platform。这个方法不需要安装程序的设置，对于 vSphere 等连接的环境来说是理想的选择。使用 Assisted Installer 安装还提供与 vSphere 集成，从而启用自动扩展。如需了解更多详细信息，请参阅使用 Assisted Installer 安装内部集群。

23.1.2. 基于代理的安装程序

您可以使用基于代理的安装程序在 vSphere 上安装 OpenShift Container Platform 集群。基于代理的安装程序可以用来使用可引导镜像在断开连接的环境中引导内部服务器。使用基于代理的安装程序，用户还可以灵活地配置基础架构、自定义网络配置并在断开连接的环境中自定义安装。如需了解更多详细信息，请参阅准备使用基于代理的安装程序安装。

23.1.3. 安装程序置备的基础架构安装

您可以使用安装程序置备的基础架构在 vSphere 上安装 OpenShift Container Platform。安装程序置备的基础架构允许安装程序预配置和自动置备 OpenShift Container Platform 所需的资源。安装程序置备的基础架构适用于在使用断开连接的网络的环境中安装，安装程序为集群置备底层基础架构。

- 在 vSphere 上安装集群 ： 您可以使用安装程序置备的基础架构安装在 vSphere 上安装 OpenShift Container Platform。
- 使用自定义在 vSphere 上安装集群 ： 您可以使用安装程序置备的基础架构安装和默认自定义选项在 vSphere 上安装 OpenShift Container Platform。
- 使用自定义网络在 vSphere 上安装集群：您可以使用网络自定义 在安装程序置备的 vSphere 基础架构上安装 OpenShift Container Platform。您可以在安装过程中自定义 OpenShift Container Platform 网络配置，以便集群可以与现有 IP 地址分配共存并遵循您的网络要求。
-

在受限网络中的 vSphere 上安装集群 ：您可以通过创建安装发行内容的内部镜像在受限网络中的 VMware vSphere 基础架构上安装集群。您可以使用此方法在互联网不可见的内部网络中部署 OpenShift Container Platform。

23.1.4. 用户置备的基础架构安装

您可以使用用户置备的基础架构在 vSphere 上安装 OpenShift Container Platform。用户置备的基础架构要求用户置备 OpenShift Container Platform 所需的所有资源。如果不使用安装程序置备的基础架构，您必须自己管理和维护集群资源。

- 使用用户置备的基础架构在 vSphere 上安装集群： 您可以在您置备的 VMware vSphere 基础架构上安装 OpenShift Container Platform。
- 使用用户置备的基础架构在 vSphere 上安装集群：您可以使用自定义的网络配置选项在 VMware vSphere 基础架构上安装 OpenShift Container Platform。
- 在带有用户置备的受限网络中的 vSphere 上安装集群： OpenShift Container Platform 可以在受限网络中置备的 VMware vSphere 基础架构上安装。

重要

执行用户置备的基础架构安装的步骤仅作为示例。使用您提供的基础架构安装集群需要了解 vSphere 平台和 OpenShift Container Platform 的安装过程。使用用户置备的基础架构安装说明作为指南；您可以通过其他方法创建所需的资源。

23.1.5. 其他资源

- 安装过程

23.2. 安装程序置备的基础架构

23.2.1. vSphere 安装要求

在使用安装程序置备的基础架构开始安装前，请确保您的 vSphere 环境满足以下安装要求。

23.2.1.1. VMware vSphere 基础架构要求

您必须在满足您使用的组件要求的 VMware vSphere 实例之一上安装 OpenShift Container Platform 集群：

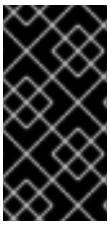
- 版本 7.0 更新 2 或更高版本
- 版本 8.0 更新 1 或更高版本

这两个版本都支持 Container Storage Interface (CSI) 迁移，它在 OpenShift Container Platform 4.16 中默认启用。

您可以在内部或 VMware Cloud 验证的供应商中托管 VMware vSphere 基础架构，以满足下表中概述的要求：

表 23.1. vSphere 虚拟环境的版本要求

虚拟环境产品	所需的版本
VMware 虚拟硬件	15 或更高版本
vSphere ESXi 主机	7.0 更新 2 或更高版本; 8.0 更新 1 或更高版本
vCenter 主机	7.0 更新 2 或更高版本; 8.0 更新 1 或更高版本



重要

您必须确保在安装 OpenShift Container Platform 前同步 ESXi 主机上的时间。请参阅 VMware 文档中的 编辑主机时间配置。

表 23.2. VMware 组件支持的最低 vSphere 版本

组件	最低支持版本	描述
虚拟机监控程序（Hypervisor）	vSphere 7.0 更新 2 （或更新版本）或 vSphere 8.0 更新 1 （或更新版本）带有虚拟硬件版本 15	此 hypervisor 版本是 Red Hat Enterprise Linux CoreOS (RHCOS) 支持的最低版本。有关与 RHCOS 兼容的 Red Hat Enterprise Linux (RHEL) 最新版本中支持的硬件的更多信息，请参阅红帽客户门户网站中的硬件。

组件	最低支持版本	描述
可选： Networking(NSX-T)	vSphere 7.0 更新 2 或更高版本; vSphere 8.0 更新 1 或更高版本	OpenShift Container Platform 至少需要 vSphere 7.0 Update 2 或 vSphere 8.0 Update 1。有关 NSX 和 OpenShift Container Platform 兼容性的更多信息，请参阅 VMware 的 NSX 容器插件文档中的发行注记部分。
CPU 微架构	x86-64-v2 或更高版本	OpenShift 4.13 及更高版本基于 RHEL 9.2 主机操作系统，这提高了 x86-64-v2 的微架构要求。请参阅 RHEL Microarchitecture 要求文档。您可以按照这个 KCS 文章中介绍的步骤验证兼容性。

重要

为确保在 Oracle® Cloud Infrastructure (OCI) 和 Oracle® Cloud VMware Solution (OCVS) 服务上运行的集群工作负载的最佳性能条件，请确保块卷的卷性能单元 (VPU) 为您的工作负载的大小。

以下列表提供了一些有关选择特定性能需要 VPU 的指导信息：

- **测试或概念验证环境：100 GB，20 到 30 个 VPU。**
- **基础生产环境：500 GB 和 60 个 VPU.**
- **高度使用生产环境：超过 500 GB，100 个或更多 VPU。**

考虑分配额外的 VPU，以便为更新和扩展活动提供足够的容量。请参阅块卷性能级别 (Oracle 文档)。

23.2.1.2. 网络连接要求

您必须配置机器之间的网络连接，以允许 OpenShift Container Platform 集群组件进行通信。

查看有关所需网络端口的以下详细信息。

表 23.3. 用于全机器到所有机器通信的端口

协议	port	描述
VRRP	N/A	keepalived 需要
ICMP	N/A	网络可访问性测试
TCP	**1936**	指标
	9000-9999	主机级别的服务，包括端口 9 **100-9101 上的节点导出器**，以及端口 **9099** 上的 Cluster Version Operator。
	10250-10259	Kubernetes 保留的默认端口
	10256	openshift-sdn
UDP	**4789**	虚拟可扩展 LAN (VXLAN)
	6081	Geneve
	9000-9999	主机级别的服务，包括端口 **9100**到**9101** 上的节点导出器。
	500	IPsec IKE 数据包
	4500	IPsec NAT-T 数据包
TCP/UDP	**30000-32767**	Kubernetes 节点端口
ESP	N/A	IPsec Encapsulating Security Payload(ESP)

表 23.4. 用于所有机器控制平面通信的端口

协议	port	描述
TCP	**6443**	Kubernetes API

表 23.5. control plane 机器用于 control plane 机器通信的端口

协议	port	描述
TCP	**2379-2380**	etcd 服务器和对等端口

23.2.1.3. VMware vSphere CSI Driver Operator 要求

要安装 vSphere Container Storage Interface (CSI) Driver Operator，必须满足以下要求：

- VMware vSphere 版本： 7.0 更新 2 或更高版本； 8.0 更新 1 或更高版本
- vCenter 版本： 7.0 更新 2 或更高版本； 8.0 更新 1 或更高版本
- 硬件版本 15 或更高版本的虚拟机
- 集群中还没有安装第三方 vSphere CSI 驱动程序

如果集群中存在第三方 vSphere CSI 驱动程序，OpenShift Container Platform 不会覆盖它。存在第三方 vSphere CSI 驱动程序可防止 OpenShift Container Platform 更新到 OpenShift Container Platform 4.13 或更高版本。

注意

只有在安装清单中使用 platform: vsphere 部署的集群中才支持 VMware vSphere CSI Driver Operator。

您可以为 Container Storage Interface (CSI) 驱动程序、vSphere CSI Driver Operator 和 vSphere Problem Detector Operator 创建自定义角色。自定义角色可以包含为每个 vSphere 对象分配最小权限集的权限集。这意味着 CSI 驱动程序、vSphere CSI Driver Operator 和 vSphere Problem Detector Operator 可以建立与这些对象的基本交互。

重要

在 vCenter 中安装 OpenShift Container Platform 集群会根据完整权限列表进行测试，如 "Required vCenter account privileges" 部分所述。通过遵循完整的特权列表，您可以减少创建具有一组受限权限的自定义角色时可能会出现意外和不支持的行为的可能性。

其他资源

- 要删除第三方 vSphere CSI 驱动程序，请参阅 删除第三方 vSphere CSI 驱动程序。
- 要为您的 vSphere 节点更新硬件版本，请参阅在 vSphere 中运行的节点上更新硬件。
- 存储组件的最低权限

23.2.1.4. vCenter 要求

在使用安装程序置备的基础架构的 vCenter 上安装 OpenShift Container Platform 集群前，您必须准备自己的环境。

所需的 vCenter 帐户权限

要在 vCenter 中安装 OpenShift Container Platform 集群，安装程序需要访问具有特权的帐户来读取和创建所需资源。使用具有全局管理特权的帐户是访问所有所需权限的最简单方法。

如果无法使用具有全局管理特权的帐户，则必须创建角色来授予 OpenShift Container Platform 集群安装所需的权限。虽然大多数权限都是需要的，但只有在安装程序计划置备一个文件夹来包含 vCenter 实例上的 OpenShift Container Platform 集群时，才需要一些权限，这是默认行为。您必须为指定对象创建或调整 vSphere 角色，才能授予所需的特权。

如果安装程序要创建 vSphere 虚拟机文件夹，则需要额外的角色。

例 23.1. 在 vSphere API 中安装所需的角色和权限

适用于角色的 vSphere 对象	必要时	vSphere API 中所需的权限

适用于角色的 vSphere 对象	必要时	vSphere API 中所需的权限
vSphere vCenter	Always	**Cns.Searchable** **InventoryService.Tagging.AttachTag** **InventoryService.Tagging.CreateCategory** **InventoryService.Tagging.CreateTag** **InventoryService.Tagging.DeleteCategory** **InventoryService.Tagging.DeleteTag** **InventoryService.Tagging.EditCategory** **InventoryService.Tagging.EditTag** **Sessions.ValidateSession** **StorageProfile.Update** **StorageProfile.View**
vSphere vCenter 集群	如果在集群 root 中创建虚拟机	**Host.Config.Storage** **Resource.AssignVMToPool** **VApp.AssignResourcePool** **VApp.Import** **VirtualMachine.Config.AddNewDisk**
vSphere vCenter 资源池	如果提供了现有的资源池	**Resource.AssignVMToPool** **VApp.AssignResourcePool** **VApp.Import** **VirtualMachine.Config.AddNewDisk**
vSphere Datastore	Always	**Datastore.AllocateSpace** **Datastore.Browse** **Datastore.FileManagement** **InventoryService.Tagging.ObjectAttachable**
vSphere 端口组	Always	**Network.Assign**
虚拟机文件夹	Always	**InventoryService.Tagging.ObjectAttachable** **Resource.AssignVMToPool** **VApp.Import** **VirtualMachine.Config.AddExistingDisk** **VirtualMachine.Config.AddNewDisk** **VirtualMachine.Config.AddRemoveDevice**

适用于角色的 vSphere 对象	必要时	vSphere API 中所需的权限
		VirtualMachine.Config.AdvancedConfig **VirtualMachine.Config.Annotation** **VirtualMachine.Config.CPUCount** **VirtualMachine.Config.DiskExtend** **VirtualMachine.Config.DiskLease** **VirtualMachine.Config.EditDevice** **VirtualMachine.Config.Memory** **VirtualMachine.Config.RemoveDisk** **VirtualMachine.Config.Rename** **Host.Config.Storage** **VirtualMachine.Config.ResetGuestInfo** **VirtualMachine.Config.Resource** **VirtualMachine.Config.Settings** **VirtualMachine.Config.UpgradeVirtualHardware** **VirtualMachine.Interact.GuestControl** **VirtualMachine.Interact.PowerOff** **VirtualMachine.Interact.PowerOn** **VirtualMachine.Interact.Reset** **VirtualMachine.Inventory.Create** **VirtualMachine.Inventory.CreateFromExisting** **VirtualMachine.Inventory.Delete** **VirtualMachine.Provisioning.Clone** **VirtualMachine.Provisioning.MarkAsTemplate** **VirtualMachine.Provisioning.DeployTemplate**
vSphere vCenter Datacenter	如果安装程序创建虚拟机文件夹对于用户置备的基础架构，如果集群没有使用 Machine API，则 **VirtualMachine.Inventory.Create** 和 **VirtualMachine.Inventory.Delete** 权限是可选的。请参阅"机器 API 的最小权限"表。	**InventoryService.Tagging.ObjectAttachable** **Resource.AssignVMToPool** **VirtualMachine.Config.AddExistingDisk** **VirtualMachine.Config.AddNewDisk** **VirtualMachine.Config.Add**

适用于角色的 vSphere 对象	必要时	vSphere API 中所需的权限
		RemoveDevice VirtualMachine.Config.AdvancedConfig VirtualMachine.Config.Annotation VirtualMachine.Config.CPUCount VirtualMachine.Config.DiskExtend VirtualMachine.Config.DiskLease VirtualMachine.Config.EditDevice VirtualMachine.Config.Memory VirtualMachine.Config.RemoveDisk VirtualMachine.Config.Rename VirtualMachine.Config.ResetGuestInfo VirtualMachine.Config.Resource VirtualMachine.Config.Settings VirtualMachine.Config.UpgradeVirtualHardware VirtualMachine.Interact.GuestControl VirtualMachine.Interact.PowerOff VirtualMachine.Interact.PowerOn VirtualMachine.Interact.Reset VirtualMachine.Inventory.Create VirtualMachine.Inventory.CreateFromExisting VirtualMachine.Inventory.Delete VirtualMachine.Provisioning.Clone VirtualMachine.Provisioning.DeployTemplate VirtualMachine.Provisioning.MarkAsTemplate Folder.Create Folder.Delete

例 23.2. 在 vCenter 图形用户界面 (GUI) 中安装所需的角色和权限

适用于角色的 vSphere 对象	必要时	vCenter GUI 中所需的权限
vSphere vCenter	Always	**Cns.Searchable "vSphere Tagging"."Assign or Unassign vSphere Tag" "vSphere Tagging"."Create vSphere Tag Category" "vSphere Tagging"."Create vSphere Tag" vSphere Tagging"."Delete vSphere Tag Category" "vSphere Tagging"."Delete vSphere Tag" "vSphere Tagging"."Edit vSphere Tag Category" "vSphere Tagging"."Edit vSphere Tag" Sessions."Validate session" "Profile-driven storage"."Profile-driven storage update" "Profile-driven storage"."Profile-driven storage view"**
vSphere vCenter 集群	如果在集群 root 中创建虚拟机	**Host.Configuration."Storage partition configuration" Resource."Assign virtual machine to resource pool" VApp."Assign resource pool" VApp.Import "Virtual machine"."Change Configuration"."Add new disk"**
vSphere vCenter 资源池	如果提供了现有的资源池	**Host.Configuration."Storage partition configuration" Resource."Assign virtual machine to resource pool" VApp."Assign resource pool" VApp.Import "Virtual machine"."Change Configuration"."Add new disk"**
vSphere Datastore	Always	**Datastore."Allocate space" Datastore."Browse datastore" Datastore."Low level file operations" "vSphere Tagging"."Assign or Unassign vSphere Tag on Object"**
vSphere 端口组	Always	**Network."Assign network"**

适用于角色的 vSphere 对象	必要时	vCenter GUI 中所需的权限
虚拟机文件夹	Always	"vSphere Tagging"."Assign or Unassign vSphere Tag on Object" Resource."Assign virtual machine to resource pool" VApp.Import "Virtual machine"."Change Configuration"."Add existing disk" "Virtual machine"."Change Configuration"."Add new disk" "Virtual machine"."Change Configuration"."Add or remove device" "Virtual machine"."Change Configuration"."Advanced configuration" "Virtual machine"."Change Configuration"."Set annotation" "Virtual machine"."Change Configuration"."Change CPU count" "Virtual machine"."Change Configuration"."Extend virtual disk" "Virtual machine"."Change Configuration"."Acquire disk lease" "Virtual machine"."Change Configuration"."Modify device settings" "Virtual machine"."Change Configuration"."Change Memory" "Virtual machine"."Change Configuration"."Remove disk" "Virtual machine"."Change Configuration".Rename "Virtual machine"."Change Configuration"."Reset guest information" "Virtual machine"."Change Configuration"."Change resource" "Virtual machine"."Change Configuration"."Change Settings" "Virtual machine"."Change Configuration"."Upgrade virtual machine compatibility" "Virtual machine".Interaction."Guest operating system management by VIX API" "Virtual machine".Interaction."Power

适用于角色的 vSphere 对象	必要时	vCenter GUI 中所需的权限
		off" "Virtual machine".Interaction."Power on" "Virtual machine".Interaction.Reset "Virtual machine"."Edit Inventory"."Create new" "Virtual machine"."Edit Inventory"."Create from existing" "Virtual machine"."Edit Inventory"."Remove" "Virtual machine".Provisioning."Clone virtual machine" "Virtual machine".Provisioning."Mark as template" "Virtual machine".Provisioning."Deploy template"
vSphere vCenter Datacenter	如果安装程序创建虚拟机文件夹对于用户置备的基础架构，如果集群没有使用 Machine API，则 **VirtualMachine.Inventory.Create** 和 **VirtualMachine.Inventory.Delete** 权限是可选的。	"vSphere Tagging"."Assign or Unassign vSphere Tag on Object" Resource."Assign virtual machine to resource pool" VApp.Import "Virtual machine"."Change Configuration"."Add existing disk" "Virtual machine"."Change Configuration"."Add new disk" "Virtual machine"."Change Configuration"."Add or remove device" "Virtual machine"."Change Configuration"."Advanced configuration" "Virtual machine"."Change Configuration"."Set annotation" "Virtual machine"."Change Configuration"."Change CPU count" "Virtual machine"."Change Configuration"."Extend virtual disk" "Virtual machine"."Change Configuration"."Acquire disk lease" "Virtual machine"."Change Configuration"."Modify device

适用于角色的 vSphere 对象	必要时	vCenter GUI 中所需的权限
		settings" "Virtual machine"."Change Configuration"."Change Memory" "Virtual machine"."Change Configuration"."Remove disk" "Virtual machine"."Change Configuration".Rename "Virtual machine"."Change Configuration"."Reset guest information" "Virtual machine"."Change Configuration"."Change resource" "Virtual machine"."Change Configuration"."Change Settings" "Virtual machine"."Change Configuration"."Upgrade virtual machine compatibility" "Virtual machine".Interaction."Guest operating system management by VIX API" "Virtual machine".Interaction."Power off" "Virtual machine".Interaction."Power on" "Virtual machine".Interaction.Reset "Virtual machine"."Edit Inventory"."Create new" "Virtual machine"."Edit Inventory"."Create from existing" "Virtual machine"."Edit Inventory"."Remove" "Virtual machine".Provisioning."Clone virtual machine" "Virtual machine".Provisioning."Deploy template" "Virtual machine".Provisioning."Mark as template" Folder."Create folder" Folder."Delete folder"

此外，用户需要一些 ReadOnly 权限，一些角色需要相应的权限来将权限代理到子对象。这些设置会因您是否将集群安装到现有文件夹而有所不同。

例 23.3. 所需的权限和传播设置

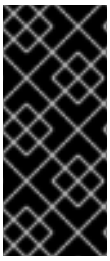
vSphere object	必要时	传播到子对象	所需的权限
vSphere vCenter	Always	False	列出所需的权限
vSphere vCenter Datacenter	现有文件夹	False	**只读** 权限
	安装程序创建文件夹	True	列出所需的权限
vSphere vCenter 集群	现有资源池	False	**只读** 权限
	集群 root 中的虚拟机	True	列出所需的权限
vSphere vCenter 数据存储	Always	False	列出所需的权限
vSphere Switch	Always	False	**只读** 权限
vSphere 端口组	Always	False	列出所需的权限
vSphere vCenter 虚拟机文件夹	现有文件夹	True	列出所需的权限
vSphere vCenter 资源池	现有资源池	True	列出所需的权限

有关只使用所需权限创建帐户的更多信息，请参阅 vSphere 文档中的 vSphere 权限和用户管理任务。

最低所需的 vCenter 帐户权限

创建自定义角色并为其分配特权后，您可以通过选择特定的 vSphere 对象来创建权限，然后为每个对象将自定义角色分配给用户或组。

在为 vSphere 对象创建权限或请求创建权限前，请确定将什么最小权限应用到 vSphere 对象。通过执行此任务，您可以确保 vSphere 对象和 OpenShift Container Platform 架构之间存在基本交互。



重要

如果您创建一个自定义角色，且没有为其分配特权，vSphere 服务器默认会为自定义角色分配一个 Read Only 角色。请注意，对于云供应商 API，自定义角色只需要继承 Read Only 角色的权限。

当具有全局管理特权的帐户不满足您的需要时，请考虑创建自定义角色。

重要

不支持使用所需权限配置的帐户。在 vCenter 中安装 OpenShift Container Platform 集群会根据完整权限列表进行测试，如 "Required vCenter account privileges" 部分所述。通过遵循完整的特权列表，您可以降低创建具有受限特权的自定义角色时可能会出现意外行为的可能性。

下表列出了与特定 OpenShift Container Platform 架构交互的 vSphere 对象的最低权限。

例 23.4. 安装程序置备的基础架构上的最低权限

适用于角色的 vSphere 对象	必要时	所需的权限
vSphere vCenter	Always	**Cns.Searchable InventoryService.Tagging.AttachTag InventoryService.Tagging.CreateCategory InventoryService.Tagging.CreateTag InventoryService.Tagging.DeleteCategory InventoryService.Tagging.DeleteTag InventoryService.Tagging.EditCategory InventoryService.Tagging.EditTag Sessions.ValidateSession StorageProfile.Update StorageProfile.View**

适用于角色的 vSphere 对象	必要时	所需的权限
vSphere vCenter 集群	如果要在集群 root 中创建虚拟机	**Host.Config.Storage Resource.AssignVMT oPool VApp.AssignResourc ePool VApp.Import VirtualMachine.Config .AddNewDisk**
vSphere vCenter 资源池	如果您在 **install-config.yaml** 文件中提供现有的资源池	**Datastore.Browse Datastore.FileManage ment Host.Config.Storage InventoryService.Tag ging.ObjectAttachable Resource.AssignVMT oPool VApp.AssignResourc ePool VApp.Import`minimu m**
vSphere 端口组	Always	**Network.Assign**
虚拟机文件夹	Always	**InventoryService.Tag ging.ObjectAttachable Resource.AssignVMT oPool VApp.Import VirtualMachine.Config .AddExistingDisk VirtualMachine.Config .AddNewDisk VirtualMachine.Config .AddRemoveDevice VirtualMachine.Config .AdvancedConfig VirtualMachine.Config .Annotation VirtualMachine.Config .CPUCount VirtualMachine.Config .DiskExtend VirtualMachine.Config .DiskLease VirtualMachine.Config .EditDevice VirtualMachine.Config .Memory VirtualMachine.Config .RemoveDisk**

适用于角色的 vSphere 对象	必要时	所需的权限
		VirtualMachine.Config.Rename **VirtualMachine.Config.ResetGuestInfo** **VirtualMachine.Config.Resource** **VirtualMachine.Config.Settings** **VirtualMachine.Config.UpgradeVirtualHardware** **VirtualMachine.Interact.GuestControl** **VirtualMachine.Interact.PowerOff** **VirtualMachine.Interact.PowerOn** **VirtualMachine.Interact.Reset** **VirtualMachine.Inventory.Create** **VirtualMachine.Inventory.CreateFromExisting** **VirtualMachine.Inventory.Delete** **VirtualMachine.Provisioning.Clone** **VirtualMachine.Provisioning.MarkAsTemplate** **VirtualMachine.Provisioning.DeployTemplate**
vSphere vCenter Datacenter	如果安装程序创建虚拟机文件夹对于用户置备的基础架构，如果集群没有使用 Machine API，则 **VirtualMachine.Inventory.Create** 和 **VirtualMachine.Inventory.Delete** 权限是可选的。如果您的集群使用 Machine API，并且要为 API 设置最小权限集，请参阅 Machine API 的"最小权限"表。	**Folder.Create** **Folder.Delete** **InventoryService.Tagging.ObjectAttachable** **Resource.AssignVMToPool** **VApp.Import** **VirtualMachine.Config.AddExistingDisk** **VirtualMachine.Config.AddNewDisk** **VirtualMachine.Config.AddRemoveDevice** **VirtualMachine.Config.AdvancedConfig** **VirtualMachine.Config.Annotation** **VirtualMachine.Config.CPUCount** **VirtualMachine.Config.DiskExtend**

适用于角色的 vSphere 对象	必要时	所需的权限
		VirtualMachine.Config.DiskLease VirtualMachine.Config.EditDevice VirtualMachine.Config.Memory VirtualMachine.Config.RemoveDisk VirtualMachine.Config.Rename VirtualMachine.Config.ResetGuestInfo VirtualMachine.Config.Resource VirtualMachine.Config.Settings VirtualMachine.Config.UpgradeVirtualHardware VirtualMachine.Interact.GuestControl VirtualMachine.Interact.PowerOff VirtualMachine.Interact.PowerOn VirtualMachine.Interact.Reset VirtualMachine.Inventory.Create VirtualMachine.Inventory.CreateFromExisting VirtualMachine.Inventory.Delete VirtualMachine.Provisioning.Clone VirtualMachine.Provisioning.DeployTemplate VirtualMachine.Provisioning.MarkAsTemplate

例 23.5. 安装后管理组件的最小权限

适用于角色的 vSphere 对象	必要时	所需的权限

适用于角色的 vSphere 对象	必要时	所需的权限
vSphere vCenter	Always	**Cns.Searchable** **InventoryService.Tagging.AttachTag** **InventoryService.Tagging.CreateCategory** **InventoryService.Tagging.CreateTag** **InventoryService.Tagging.DeleteCategory** **InventoryService.Tagging.DeleteTag** **InventoryService.Tagging.EditCategory** **InventoryService.Tagging.EditTag** **Sessions.ValidateSession** **StorageProfile.Update** **StorageProfile.View**
vSphere vCenter 集群	如果要在集群 root 中创建虚拟机	**Host.Config.Storage** **Resource.AssignVMToPool**
vSphere vCenter 资源池	如果您在 **install-config.yaml** 文件中提供现有的资源池	**Host.Config.Storage**
vSphere Datastore	Always	**Datastore.AllocateSpace** **Datastore.Browse** **Datastore.FileManagement** **InventoryService.Tagging.ObjectAttachable**
vSphere 端口组	Always	**Network.Assign**

适用于角色的 vSphere 对象	必要时	所需的权限
虚拟机文件夹	Always	**VirtualMachine.Config.AddExistingDisk** **VirtualMachine.Config.AddRemoveDevice** **VirtualMachine.Config.AdvancedConfig** **VirtualMachine.Config.Annotation** **VirtualMachine.Config.CPUCount** **VirtualMachine.Config.DiskExtend** **VirtualMachine.Config.Memory** **VirtualMachine.Config.Settings** **VirtualMachine.Interact.PowerOff** **VirtualMachine.Interact.PowerOn** **VirtualMachine.Inventory.CreateFromExisting** **VirtualMachine.Inventory.Delete** **VirtualMachine.Provisioning.Clone** **VirtualMachine.Provisioning.DeployTemplate**
vSphere vCenter Datacenter	如果安装程序创建虚拟机文件夹对于用户置备的基础架构，如果集群没有使用 Machine API，则 **VirtualMachine.Inventory.Create** 和 **VirtualMachine.Inventory.Delete** 权限是可选的。如果您的集群使用 Machine API，并且要为 API 设置最小权限集，请参阅 Machine API 的"最小权限"表。	**Resource.AssignVMToPool** **VirtualMachine.Config.AddExistingDisk** **VirtualMachine.Config.AddRemoveDevice** **VirtualMachine.Interact.PowerOff** **VirtualMachine.Interact.PowerOn** **VirtualMachine.Provisioning.DeployTemplate**

例 23.6. 存储组件的最低权限

适用于角色的 vSphere 对象	必要时	所需的权限
vSphere vCenter	Always	**Cns.Searchable InventoryService.Tagging.CreateCategory InventoryService.Tagging.CreateTag InventoryService.Tagging.EditCategory InventoryService.Tagging.EditTag StorageProfile.Update StorageProfile.View**
vSphere vCenter 集群	如果要在集群 root 中创建虚拟机	**Host.Config.Storage**
vSphere vCenter 资源池	如果您在 **install-config.yaml** 文件中提供现有的资源池	**Host.Config.Storage**
vSphere Datastore	Always	**Datastore.Browse Datastore.FileManagement InventoryService.Tagging.ObjectAttachable**
vSphere 端口组	Always	**只读**
虚拟机文件夹	Always	**VirtualMachine.Config.AddExistingDisk VirtualMachine.Config.AddRemoveDevice**
vSphere vCenter Datacenter	如果安装程序创建虚拟机文件夹对于用户置备的基础架构，如果集群没有使用 Machine API，则 **VirtualMachine.Inventory.Create** 和 **VirtualMachine.Inventory.Delete** 权限是可选的。如果您的集群使用 Machine API，并且要为 API 设置最小权限集，请参阅 Machine API 的"最小权限"表。	**VirtualMachine.Config.AddExistingDisk VirtualMachine.Config.AddRemoveDevice**

例 23.7. Machine API 的最低权限

适用于角色的 vSphere 对象	必要时	所需的权限

适用于角色的 vSphere 对象	必要时	所需的权限
vSphere vCenter	Always	**InventoryService.Tagging.AttachTag InventoryService.Tagging.CreateCategory InventoryService.Tagging.CreateTag InventoryService.Tagging.DeleteCategory InventoryService.Tagging.DeleteTag InventoryService.Tagging.EditCategory InventoryService.Tagging.EditTag Sessions.ValidateSession StorageProfile.Update StorageProfile.View**
vSphere vCenter 集群	如果要在集群 root 中创建虚拟机	**Resource.AssignVMToPool**
vSphere vCenter 资源池	如果您在 **install-config.yaml** 文件中提供现有的资源池	只读
vSphere Datastore	Always	**Datastore.AllocateSpace Datastore.Browse**
vSphere 端口组	Always	**Network.Assign**

适用于角色的 vSphere 对象	必要时	所需的权限
虚拟机文件夹	Always	**VirtualMachine.Config.AddRemoveDevice** **VirtualMachine.Config.AdvancedConfig** **VirtualMachine.Config.Annotation** **VirtualMachine.Config.CPUCount** **VirtualMachine.Config.DiskExtend** **VirtualMachine.Config.Memory** **VirtualMachine.Config.Settings** **VirtualMachine.Interact.PowerOff** **VirtualMachine.Interact.PowerOn** **VirtualMachine.Inventory.CreateFromExisting** **VirtualMachine.Inventory.Delete** **VirtualMachine.Provisioning.Clone** **VirtualMachine.Provisioning.DeployTemplate**
vSphere vCenter Datacenter	如果安装程序创建虚拟机文件夹对于用户置备的基础架构，如果集群没有使用 Machine API，则 **VirtualMachine.Inventory.Create** 和 **VirtualMachine.Inventory.Delete** 权限是可选的。	**Resource.AssignVMToPool** **VirtualMachine.Interact.PowerOff** **VirtualMachine.Interact.PowerOn** **VirtualMachine.Provisioning.DeployTemplate**

将 OpenShift Container Platform 与 vMotion 搭配使用

如果要在 vSphere 环境中使用 vMotion，请在安装 OpenShift Container Platform 集群前考虑以下内容。

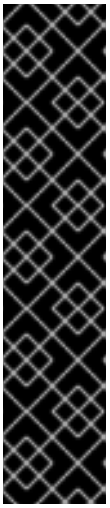
- **OpenShift Container Platform 通常支持 compute-only vMotion，其中*通常*意味着您满足所有 VMware 最佳实践进行 vMotion。**

为了帮助确保计算和 control plane 节点的正常运行时间，请确保遵循 VMware 最佳实践进行 vMotion，并使用 VMware 反关联性规则提高 OpenShift Container Platform 在维护或硬件问题期间的可用性。

有关 vMotion 和 anti-affinity 规则的更多信息，请参阅 VMware vSphere 文档以了解 vMotion 网络要求和虚拟机反关联性规则。

- 使用 Storage vMotion 可能会导致问题且不受支持。如果您在 pod 中使用 vSphere 卷，请手动或通过 Storage vMotion 在数据存储间迁移虚拟机，这会导致 OpenShift Container Platform 持久性卷(PV)对象中的无效引用，这可能会导致数据丢失。

- OpenShift Container Platform 不支持在数据存储间有选择地迁移 VMDK，使用数据存储集群进行虚拟机置备或动态或静态置备 PV，或使用作为数据存储集群一部分的数据存储来动态或静态置备 PV。



重要

您可以指定数据存储集群中存在的任何数据存储路径。默认情况下，使用 Storage vMotion 的存储分布式资源调度程序(SDRS)会自动为数据存储集群启用。红帽不支持 Storage vMotion，因此您必须禁用 Storage DRS 以避免 OpenShift Container Platform 集群的数据丢失问题。

如果需要在多个数据存储间指定虚拟机，请使用 数据存储 对象在集群 install-config.yaml 配置文件中指定故障域。如需更多信息，请参阅"VMware vSphere 区域和区启用"。

集群资源

当您部署使用安装程序置备的基础架构的 OpenShift Container Platform 集群时，安装程序必须能够在 vCenter 实例中创建多个资源。

标准 OpenShift Container Platform 安装会创建以下 vCenter 资源：

- 1 个文件夹

- 1 标签类别

- 1 标签
- 虚拟机：
  - 1 个模板
  - 1 个临时 bootstrap 节点
  - 3 个 control plane 节点
  - 3 个计算机器

虽然这些资源使用 856 GB 存储，但 bootstrap 节点会在集群安装过程中销毁。使用标准集群至少需要 800 GB 存储。

如果部署更多计算机器，OpenShift Container Platform 集群将使用更多存储。

集群限制

可用资源因集群而异。vCenter 中可能的集群数量主要受可用存储空间以及对所需资源数量的限制。确保考虑集群创建的 vCenter 资源的限制和部署集群所需的资源，如 IP 地址和网络。

网络要求

对网络使用动态主机配置协议(DHCP)，并确保 DHCP 服务器配置为为集群机器提供持久的 IP 地址。

注意

如果要使用静态 IP 地址置备节点，则不需要将 DHCP 用于网络。

将默认网关配置为使用 DHCP 服务器。所有节点必须位于同一 VLAN 中。您不能将第二 VLAN 用作第 2 天操作来缩放集群。

您必须为网络使用动态主机配置协议 (DHCP)，并确保 DHCP 服务器被配置为为集群机器提供持久的 IP 地址。在 DHCP 租期中，您必须将 DHCP 配置为使用默认网关。所有节点必须位于同一 VLAN 中。您不能将第二 VLAN 用作第 2 天操作来缩放集群。

如果要安装到受限环境中，受限网络中的虚拟机必须有权访问 vCenter，以便它可以置备和管理节点、持久性卷声明 (PVC) 和其他资源。

另外，在安装 OpenShift Container Platform 集群前，您必须创建以下网络资源：

注意

建议集群中的每个 OpenShift Container Platform 节点都必须有权访问可通过 DHCP 发现的网络时间协议(NTP)服务器。没有 NTP 服务器即可安装。但是，异步服务器时钟将导致错误，NTP 服务器会阻止。

所需的 IP 地址

对于使用 DHCP 的网络，安装程序置备的 vSphere 安装需要两个静态 IP 地址：

- API 地址用于访问集群 API。
- Ingress 地址用于集群入口流量。

安装 OpenShift Container Platform 集群时，必须向安装程序提供这些 IP 地址。

DNS 记录

您必须在适当的 DNS 服务器中为托管 OpenShift Container Platform 集群的 vCenter 实例创建两个静态 IP 地址的 DNS 记录。在每个记录中，<cluster_name> 是集群名称，<base_domain> 是您 在安装集群时指定的集群基域。完整的 DNS 记录采用以下形式： <component>.<cluster_name>.<base_domain>.。

表 23.6. 所需的 DNS 记录

组件	记录	描述

组件	记录	描述
API VIP	**api.<cluster_name>.<base_domain>.**	此 DNS A/AAAA 或 CNAME (Canonical Name) 记录必须指向 control plane 机器的负载均衡器。此记录必须由集群外的客户端和集群中的所有节点解析。
Ingress VIP	***.apps.<cluster_name>.<base_domain>.**	通配符 DNS A/AAAA 或 CNAME 记录，指向以运行入口路由器 Pod 的机器（默认为 worker 节点）为目标的负载均衡器。此记录必须由集群外的客户端和集群中的所有节点解析。

vSphere 节点的静态 IP 地址

您可以在没有动态主机配置协议(DHCP)的环境中置备 bootstrap、control plane 和计算节点，以使用静态 IP 地址配置。要配置此环境，您必须为 install-config.yaml 文件中的 platform.vsphere.hosts.role 参数提供值。

默认情况下，安装程序被配置为使用 DHCP 作为网络，但此网络具有有限的可配置功能。

在 install-config.yaml 文件中定义一个或多个机器池后，您可以为网络上的节点定义网络定义。确保网络定义数量与您为集群配置的机器池数量匹配。

以下示例显示了具有 compute 角色的节点的网络配置：

```
---
platform:
  vsphere:
    hosts:
    - role: compute 1
      networkDevice:
        ipAddrs:
        - 192.168.204.10/24 2
        gateway: 192.168.204.1 3
        nameservers:
        - 192.168.204.1 4
---
```

1

有效的网络定义值包括 bootstrap、control-plane 和 compute。您必须在 install-config.yaml 配置文件中至少列出一个 bootstrap 网络定义。

2

列出安装程序传递给网络接口的 IPv4、IPv6 或两个 IP 地址。机器 API 控制器将所有配置的 IP 地址分配给默认网络接口。

3

网络接口的默认网关。

4

最多列出 3 个 DNS 名称服务器。

在部署集群以使用静态 IP 地址运行节点后，您可以扩展机器以使用这些静态 IP 地址之一。另外，您可以使用机器集将机器配置为使用配置的静态 IP 地址之一。

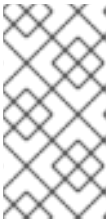
其他资源

- 扩展机器以使用静态 IP 地址
- 使用机器集扩展带有配置的静态 IP 地址的机器

23.2.2. 准备使用安装程序置备的基础架构安装集群

您可以通过完成以下步骤，准备在 vSphere 上安装 OpenShift Container Platform 集群：

- 下载安装程序。



注意

如果您要在断开连接的环境中安装，您可以从镜像内容中提取安装程序。如需更多信息，请参阅为断开连接的安装镜像镜像。

- 安装 OpenShift CLI (oc)。

注意

如果要在断开连接的环境中安装，请将 oc 安装到镜像主机上。

- 生成 SSH 密钥对。您可以在部署后使用此密钥对在 OpenShift Container Platform 集群的节点上进行身份验证。

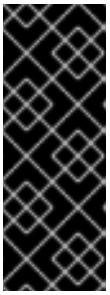
- 将 vCenter 的可信 root CA 证书添加到您的系统信任中。

23.2.2.1. 获取安装程序

在安装 OpenShift Container Platform 前，将安装文件下载到您用于安装的主机上。

先决条件

- 您有一个运行 Linux 的机器，如 Red Hat Enterprise Linux 8，本地磁盘空间为 500 MB。



重要

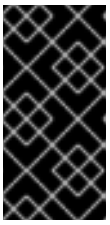
如果您试图在 macOS 上运行安装程序，与 golang 编译器相关的已知问题会导致 OpenShift Container Platform 集群的安装失败。有关此问题的更多信息，请参阅 *OpenShift Container Platform 4.16 发行注记*文档中的"已知问题"部分。

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐户，请使用您的凭证登录。如果没有，请创建一个帐户。

2. 选择您的基础架构供应商。

3. 进入到安装类型的页面，下载与您的主机操作系统和架构对应的安装程序，并将该文件放在您要存储安装配置文件的目录中。



重要

安装程序会在用来安装集群的计算机上创建几个文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。

重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，请为特定云供应商完成 OpenShift Container Platform 卸载流程。

4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret。此 pull secret 允许您与所含授权机构提供的服务进行身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

23.2.2.2. 安装 OpenShift CLI

您可以安装 OpenShift CLI(oc)来使用命令行界面与 OpenShift Container Platform 进行交互。您可以在 Linux、Windows 或 macOS 上安装 oc。

重要

如果安装了旧版本的 oc，则可能无法使用 OpenShift Container Platform 4.16 中的所有命令。下载并安装新版本的 oc。

在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI(oc)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。

2. 从 产品变体 下拉列表中选择架构。

3. 从 版本 下拉列表中选择适当的版本。

4. 点 OpenShift v4.16 Linux Client 条目旁的 Download Now 来保存文件。

5. 解包存档：

```
$ tar xvf <file>
```

6. 将 oc 二进制文件放到 PATH 中的目录中。

   要查看您的 PATH，请执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 oc 命令：

```
$ oc <command>
```

在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI(oc)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。

2. 从 版本 下拉列表中选择适当的版本。

3. 点 OpenShift v4.16 Windows Client 条目旁的 Download Now 来保存文件。

4. 使用 ZIP 程序解压存档。

5. 将 oc 二进制文件移到 PATH 中的目录中。

   要查看您的 PATH，请打开命令提示并执行以下命令：

```
C:\> path
```

验证

- 安装 OpenShift CLI 后，可以使用 oc 命令：

```
C:\> oc <command>
```

在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI(oc)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。

2. 从 版本 下拉列表中选择适当的版本。

3. 点 OpenShift v4.16 macOS Client 条目旁的 Download Now 来保存文件。

注意

对于 macOS arm64，请选择 OpenShift v4.16 macOS arm64 Client 条目。

4. 解包和解压存档。

5. 将 oc 二进制文件移到 PATH 的目录中。

要查看您的 PATH，请打开终端并执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 oc 命令：

```
$ oc <command>
```

23.2.2.3. 为集群节点 SSH 访问生成密钥对

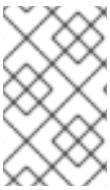
在 OpenShift Container Platform 安装过程中，您可以为安装程序提供 SSH 公钥。密钥通过它们的 Ignition 配置文件传递给 Red Hat Enterprise Linux CoreOS(RHCOS)节点，用于验证对节点的 SSH 访问。密钥添加到每个节点上 core 用户的 ~/.ssh/authorized_keys 列表中，这将启用免密码身份验证。

将密钥传递给节点后，您可以使用密钥对作为用户 核心 通过 SSH 连接到 RHCOS 节点。若要通过 SSH 访问节点，必须由 SSH 为您的本地用户管理私钥身份。

如果要通过 SSH 连接到集群节点来执行安装调试或灾难恢复，则必须在安装过程中提供 SSH 公钥。./openshift-install gather 命令还需要在集群节点上设置 SSH 公钥。

重要

不要在生产环境中跳过这个过程，在生产环境中需要灾难恢复和调试。



注意

您必须使用本地密钥，而不是使用特定平台方法配置 的密钥，如 AWS 密钥对。

流程

1. 如果您在本地计算机上没有可用于在集群节点上进行身份验证的现有 SSH 密钥对，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1

指定新 SSH 密钥的路径和文件名，如 ~/.ssh/id_ed25519。如果您已有密钥对，请确保您的公钥位于 ~/.ssh 目录中。

注意

如果您计划在 x86_64、ppc64le 和 s390x 架构上安装使用 RHEL 加密库（这些加密库已提交给 NIST 用于 FIPS 140-2/140-3 验证）的 OpenShift Container Platform 集群，则不要创建使用 ed25519 算法的密钥。相反，创建一个使用 rsa 或 ecdsa 算法的密钥。

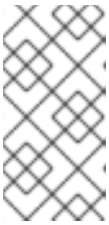
2. 查看公共 SSH 密钥：

```
$ cat <path>/<file_name>.pub
```

例如，运行以下命令来查看 ~/.ssh/id_ed25519.pub 公钥：

```
$ cat ~/.ssh/id_ed25519.pub
```

3. 将 SSH 私钥身份添加到本地用户的 SSH 代理（如果尚未添加）。在集群节点上，或者要使用 ./openshift-install gather 命令，需要对该密钥进行 SSH 代理管理，才能在集群节点上进行免密码 SSH 身份验证。



注意

在某些发行版中，自动管理默认 SSH 私钥身份，如 ~/.ssh/id_rsa 和 ~/.ssh/id_dsa。

a. 如果 ssh-agent 进程尚未为您的本地用户运行，请将其作为后台任务启动：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果集群处于 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

4. 将 SSH 私钥添加到 ssh-agent ：

```
$ ssh-add <path>/<file_name> 1
```

1 指定 SSH 私钥的路径和文件名，如 ~/.ssh/id_ed25519.pub

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

后续步骤

- 安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。

23.2.2.4. 在您的系统信任中添加 vCenter root CA 证书

因为安装程序需要访问 vCenter 的 API，所以您必须在安装 OpenShift Container Platform 集群前将 vCenter 的可信 root CA 证书添加到系统信任中。

流程

1. 在 vCenter 主页中下载 vCenter 的 root CA 证书。在 vSphere Web Services SDK 部分点 Download trusted root CA 证书。<vCenter>/certs/download.zip 文件下载。

2. 提取包含 vCenter root CA 证书的压缩文件。压缩文件的内容类似以下文件结构：

```
certs
├── lin
│   ├── 108f4d17.0
│   ├── 108f4d17.r1
│   ├── 7e757f6a.0
│   ├── 8e4f8471.0
│   └── 8e4f8471.r0
├── mac
│   ├── 108f4d17.0
│   ├── 108f4d17.r1
│   ├── 7e757f6a.0
│   ├── 8e4f8471.0
│   └── 8e4f8471.r0
└── win
    ├── 108f4d17.0.crt
    ├── 108f4d17.r1.crl
    ├── 7e757f6a.0.crt
    ├── 8e4f8471.0.crt
    └── 8e4f8471.r0.crl

3 directories, 15 files
```

3. 将您的操作系统的文件添加到系统信任中。例如，在 Fedora 操作系统中运行以下命令：

```
# cp certs/lin/* /etc/pki/ca-trust/source/anchors
```

4. 更新您的系统信任关系。例如，在 Fedora 操作系统中运行以下命令：

```
# update-ca-trust extract
```

23.2.3. 在 vSphere 上安装集群

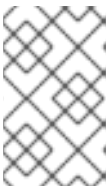
在 OpenShift Container Platform 版本 4.16 中，您可以使用安装程序置备的基础架构在 VMware vSphere 实例上安装集群。

注意

OpenShift Container Platform 支持将集群部署到单个 VMware vCenter 中。不支持在多个 vCenter 上使用机器/机器集部署集群。

23.2.3.1. 先决条件

- 您已完成了 准备使用安装程序置备的基础架构安装集群 中的任务。
- 您检查了 VMware 平台许可证。红帽不会对 VMware 许可证产生任何限制，但有些 VMware 基础架构组件需要许可。
- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读有关 选择集群安装方法的文档，并为用户准备它。
- 已为集群置备了 持久性存储。要部署私有镜像 registry，您的存储必须提供 ReadWriteMany 访问模式。
- OpenShift Container Platform 安装程序需要访问 vCenter 和 ESXi 主机上的端口 443。您确认可以访问端口 443。
- 如果您使用防火墙，您与管理员确认可以访问端口 443。control plane 节点必须能够通过端口 443 访问 vCenter 和 ESXi 主机，才能成功安装。
- 如果使用防火墙，则会 将其配置为允许集群需要访问的站点。



注意

如果要配置代理，请务必查看此站点列表。

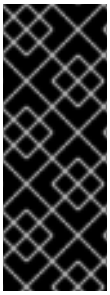
23.2.3.2. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。

- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。



重要

如果您的集群无法直接访问互联网，则可以在置备的某些类型的基础架构上执行受限网络安装。在此过程中，您可以下载所需的内容，并使用它为镜像 registry 填充安装软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群前，您要更新镜像 registry 的内容。

23.2.3.3. 部署集群

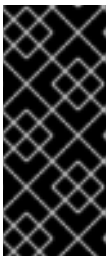
您可以在兼容云平台上安装 OpenShift Container Platform。

重要

在初始安装过程中，您只能运行安装程序的 create cluster 命令一次。

先决条件

- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。
- 已确认主机上的云供应商帐户具有部署集群的正确权限。权限不正确的帐户会导致安装过程失败，并显示包括缺失权限的错误消息。
- 可选：在创建集群时，配置外部负载均衡器来代替默认负载均衡器。



重要

您不需要为安装程序指定 API 和 Ingress 静态地址。如果选择此配置，则必须采取额外的操作来定义接受每个引用的 vSphere 子网的 IP 地址的网络目标。请参阅"配置用户管理的负载均衡器"部分。

流程

1. 进入包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1 对于 <installation_directory>，请指定要存储安装程序创建的文件的目录名称。

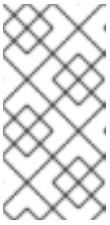
2 要查看不同的安装详情，请指定 warn、debug 或 error，而不是 info。

在指定目录时：

- 验证该目录是否具有执行权限。在安装目录中运行 Terraform 二进制文件需要这个权限。
- 使用空目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

2. 在提示符处提供值：

a. 可选：选择用于访问集群机器的 SSH 密钥。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 ssh-agent 进程使用的 SSH 密钥。

b. Select vsphere 作为目标平台。

c. 指定 vCenter 实例的名称。

d.

指定创建集群所需的权限的 vCenter 帐户的用户名和密码。

安装程序连接到您的 vCenter 实例。

重要

有些带有活动目录(AD)集成的 VMware vCenter Single Sign-On (SSO)环境可能主要要求您使用传统的登录方法，这需要 <domain>\ 构造。

要确保 vCenter 帐户权限正确检查，请考虑使用 User Principal Name (UPN)登录方法，如 <username>@<fully_qualified_domainname>。

e.

选择要连接的 vCenter 实例中的数据中心。

f.

选择要使用的默认 vCenter 数据存储。

注意

数据存储和集群名称不能超过 60 个字符，因此请确保组合字符串长度不超过 60 个字符的限制。

g.

选择要在其中安装 OpenShift Container Platform 集群的 vCenter 集群。安装程序使用 vSphere 集群的 root 资源池作为默认资源池。

h.

选择包含您配置的虚拟 IP 地址和 DNS 记录的 vCenter 实例中的网络。

i.

输入您为 control plane API 访问配置的虚拟 IP 地址。

j.

输入您为集群入口配置的虚拟 IP 地址。

k.

输入基域。这个基域必须与您配置的 DNS 记录中使用的域相同。

l. 为集群输入描述性名称。集群名称必须与您配置的 DNS 记录中使用的相同。

注意

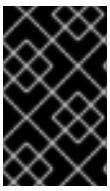
数据存储和集群名称不能超过 60 个字符，因此请确保组合字符串长度不超过 60 个字符的限制。

m. 粘贴 Red Hat OpenShift Cluster Manager 中的 pull secret。

验证

当集群部署成功完成时：

- 终端会显示用于访问集群的说明，包括指向 Web 控制台和 kubeadmin 用户的凭证的链接。
- 凭证信息还会输出到 <installation_directory>/.openshift_install.log.

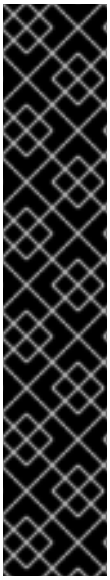


重要

不要删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 node-bootstrapper 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。

- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

23.2.3.4. 使用 CLI 登录集群

您可以通过导出集群 kubeconfig 文件，以默认系统用户身份登录集群。kubeconfig 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。
- 已安装 oc CLI。

流程

1. 导出 kubeadmin 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 <installation_directory>，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 oc 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

23.2.3.5. 创建 registry 存储

安装集群后，必须为 registry Operator 创建存储。

23.2.3.5.1. 安装过程中删除的镜像 registry

在不提供可共享对象存储的平台上，OpenShift Image Registry Operator bootstraps 本身为 Removed。这允许 openshift-installer 在这些平台类型上完成安装。

安装后，您必须编辑 Image Registry Operator 配置，将 managementState 从 Removed 切换到 Managed。完成此操作后，您必须配置存储。

23.2.3.5.2. 镜像 registry 存储配置

对于不提供默认存储的平台，Image Registry Operator 最初不可用。安装后，您必须将 registry 配置为使用存储，以便 Registry Operator 可用。

显示配置生产集群所需的持久性卷的说明。如果适用，显示有关将空目录配置为存储位置的说明，这仅适用于非生产集群。

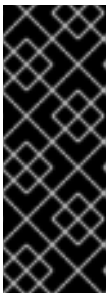
提供了在升级过程中使用 Recreate rollout 策略来允许镜像 registry 使用块存储类型的说明。

23.2.3.5.2.1. 为 VMware vSphere 配置 registry 存储

作为集群管理员，在安装后需要配置 registry 来使用存储。

先决条件

- 集群管理员权限。
- VMware vSphere 上有一个集群。
- 为集群置备的持久性存储，如 Red Hat OpenShift Data Foundation。



重要

当您只有一个副本时，OpenShift Container Platform 支持对镜像 registry 存储的 ReadWriteOnce 访问。ReadWriteOnce 访问还要求 registry 使用 Recreate rollout 策略。要部署支持高可用性的镜像 registry，需要两个或多个副本，ReadWriteMany 访问。

- 必须具有"100Gi"容量.

重要

测试显示在 RHEL 中使用 NFS 服务器作为核心服务的存储后端的问题。这包括 OpenShift Container Registry 和 Quay，Prometheus 用于监控存储，以及 Elasticsearch 用于日志存储。因此，不建议使用 RHEL NFS 作为 PV 后端用于核心服务。

市场上的其他 NFS 实现可能没有这些问题。如需了解更多与此问题相关的信息，请联络相关的 NFS 厂商。

流程

1. 要将 registry 配置为使用存储，修改 configs.imageregistry/cluster 资源中的 spec.storage.pvc。

注意

使用共享存储时，请查看您的安全设置以防止外部访问。

2. 验证您没有 registry pod:

```
$ oc get pod -n openshift-image-registry -l docker-registry=default
```

输出示例

```
No resourses found in openshift-image-registry namespace
```

注意

如果您的输出中有一个 registry pod，则不需要继续这个过程。

3. 检查 registry 配置：

```
$ oc edit configs.imageregistry.operator.openshift.io
```

输出示例

```
storage:
  pvc:
    claim: 1
```

1

将 claim 字段留空以允许自动创建 image-registry-storage 持久性卷声明(PVC)。PVC 基于默认存储类生成。但请注意，默认存储类可能会提供 ReadWriteOnce (RWO) 卷，如 RADOS 块设备(RBD)，这可能会在复制到多个副本时导致问题。

4. 检查 clusteroperator 状态：

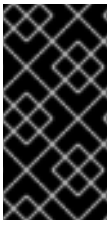
```
$ oc get clusteroperator image-registry
```

输出示例

```
NAME             VERSION                              AVAILABLE   PROGRESSING   DEGRADED
SINCE   MESSAGE
image-registry   4.7                                  True        False         False     6h50m
```

23.2.3.5.2.2. 为 VMware vSphere 配置块 registry 存储

要允许镜像 registry 在作为集群管理员升级过程中使用块存储类型，如 vSphere Virtual Machine Disk(VMDK)，您可以使用 Recreate rollout 策略。



重要

支持块存储卷，但不建议在生产环境中用于镜像 registry。在块存储上配置 registry 的安装不具有高可用性，因为 registry 无法具有多个副本。

流程

1. 输入以下命令将镜像 registry 存储设置为块存储类型，对 registry 进行补丁，使其使用 Recreate rollout 策略，并只使用一个副本运行：

```
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec":
{"rolloutStrategy":"Recreate","replicas":1}}'
```

2. 为块存储设备置备 PV，并为该卷创建 PVC。请求的块卷使用 ReadWriteOnce(RWO)访问模式。

   a. 创建包含以下内容的 pvc.yaml 文件以定义 VMware vSphere PersistentVolumeClaim 对象：

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: image-registry-storage (1)
  namespace: openshift-image-registry (2)
```

```
spec:
  accessModes:
  - ReadWriteOnce 3
  resources:
    requests:
      storage: 100Gi 4
```

1 代表 PersistentVolumeClaim 对象的唯一名称。

2 PersistentVolumeClaim 对象的命名空间，即 openshift-image-registry。

3 持久性卷声明的访问模式。使用 ReadWriteOnce 时，单个节点可以通过读写权限挂载该卷。

4 持久性卷声明的大小。

b. 输入以下命令从文件创建 PersistentVolumeClaim 对象：

```
$ oc create -f pvc.yaml -n openshift-image-registry
```

3. 输入以下命令编辑 registry 配置，使其引用正确的 PVC：

```
$ oc edit config.imageregistry.operator.openshift.io -o yaml
```

输出示例

```
storage:
  pvc:
    claim: 1
```

1

有关配置 registry 存储以便引用正确的 PVC 的说明，请参阅 为 vSphere 配置 registry。

23.2.3.6. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

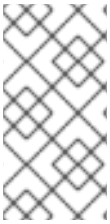
- 有关 Telemetry 服务的更多信息，请参阅关于 远程健康监控

23.2.3.7. 后续步骤

- 自定义集群。
- 如果需要，您可以选择 不使用远程健康报告。
- 设置 registry 并配置 registry 存储。
- 可选： 查看 vSphere 问题检测器 Operator 中的事件，以确定集群是否有权限或存储配置问题。

23.2.4. 使用自定义在 vSphere 上安装集群

在 OpenShift Container Platform 版本 4.16 中，您可以使用安装程序置备的基础架构在 VMware vSphere 实例上安装集群。要自定义安装，请在安装集群前修改 install-config.yaml 文件中的参数。

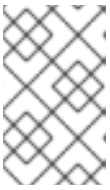


注意

OpenShift Container Platform 支持将集群部署到单个 VMware vCenter 中。不支持在多个 vCenter 上使用机器/机器集部署集群。

23.2.4.1. 先决条件

- 您已完成了 准备使用安装程序置备的基础架构安装集群 中的任务。
- 您检查了 VMware 平台许可证。红帽不会对 VMware 许可证产生任何限制，但有些 VMware 基础架构组件需要许可。
- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读有关 选择集群安装方法的文档，并为用户准备它。
- 已为集群置备了 持久性存储。要部署私有镜像 registry，您的存储必须提供 ReadWriteMany 访问模式。
- OpenShift Container Platform 安装程序需要访问 vCenter 和 ESXi 主机上的端口 443。您确认可以访问端口 443。
- 如果您使用防火墙，您与管理员确认可以访问端口 443。control plane 节点必须能够通过端口 443 访问 vCenter 和 ESXi 主机，才能成功安装。
- 如果使用防火墙，则会 将其配置为允许集群需要访问的站点。



注意

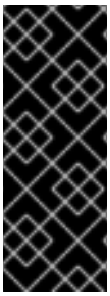
如果要配置代理，请务必查看此站点列表。

23.2.4.2. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

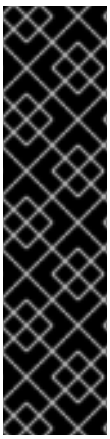


重要

如果您的集群无法直接访问互联网，则可以在置备的某些类型的基础架构上执行受限网络安装。在此过程中，您可以下载所需的内容，并使用它为镜像 registry 填充安装软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群前，您要更新镜像 registry 的内容。

23.2.4.3. VMware vSphere 区域和区启用

您可以将 OpenShift Container Platform 集群部署到在单个 VMware vCenter 中运行的多个 vSphere 数据中心。每个数据中心都可以运行多个集群。此配置降低了导致集群失败的硬件故障或网络中断的风险。要启用区域和区域，您必须为 OpenShift Container Platform 集群定义多个故障域。



重要

VMware vSphere 区域和区启用功能需要 vSphere Container Storage Interface (CSI) 驱动程序作为集群中的默认存储驱动程序。因此，这个功能只在新安装的集群中可用。

对于从上一版本升级的集群，您必须为集群启用 CSI 自动迁移。然后，您可以为升级的集群配置多个区域和区域。

默认安装配置将集群部署到单个 vSphere 数据中心。如果要将集群部署到多个 vSphere 数据中心，您必须创建一个启用地区和区功能的安装配置文件。

默认 install-config.yaml 文件包含 vcenters 和 failureDomains 字段，您可以在其中为 OpenShift Container Platform 集群指定多个 vSphere 数据中心和集群。如果要在由单个数据中心组成的 vSphere

环境中安装 OpenShift Container Platform 集群，您可以将这些字段留空。

以下列表描述了为集群定义区和区域相关的术语：

- 故障域：建立地区和区域之间的关系。您可以使用 vCenter 对象（如 datastore 对象）定义故障域。故障域定义 OpenShift Container Platform 集群节点的 vCenter 位置。
- Region ：指定 vCenter 数据中心。您可以使用 openshift-region 标签类别中的标签来定义区域。
- Zone：指定一个 vCenter 集群。您可以使用 openshift-zone 标签类别中的标签来定义区。

注意

如果您计划在 install-config.yaml 文件中指定多个故障域，则必须在创建配置文件前创建标签类别、区域标签和区域标签。

您必须为每个代表一个区域的 vCenter 数据中心创建一个 vCenter 标签。另外，您必须为比数据中心（代表一个区）中运行的每个集群创建一个 vCenter 标签。创建标签后，您必须将每个标签附加到对应的数据中心和集群。

下表概述了在单个 VMware vCenter 中运行的多个 vSphere 数据中心的区域、区域和标签之间的关系示例。

数据中心（区域）	集群（区）	Tags
us-east	us-east-1	us-east-1a
		us-east-1b
	us-east-2	us-east-2a
		us-east-2b
us-west	us-west-1	us-west-1a
		us-west-1b

数据中心（区域）	集群（区）	Tags
	us-west-2	us-west-2a
		us-west-2b

其他资源

- 其他 VMware vSphere 配置参数
- 弃用的 VMware vSphere 配置参数
- vSphere 自动迁移
- VMware vSphere CSI Driver Operator

23.2.4.4. 创建安装配置文件

您可以自定义在 VMware vSphere 上安装的 OpenShift Container Platform 集群。

先决条件

- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。

流程

1. 创建 install-config.yaml 文件。
   a. 进入包含安装程序的目录并运行以下命令：

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

1 对于 <installation_directory>，请指定要存储安装程序创建的文件的目录名称。

在指定目录时：

- 验证该目录是否具有执行权限。在安装目录中运行 Terraform 二进制文件需要这个权限。

- 使用空目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

b. 在提示符处，提供云的配置详情：

i. 可选：选择用于访问集群机器的 SSH 密钥。

注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 ssh-agent 进程使用的 SSH 密钥。

ii. Select vsphere 作为目标平台。

iii. 指定 vCenter 实例的名称。

iv. 指定创建集群所需的权限的 vCenter 帐户的用户名和密码。

安装程序连接到您的 vCenter 实例。

v. 选择要连接的 vCenter 实例中的数据中心。

注意

创建安装配置文件后，您可以修改该文件以创建多个 vSphere 数据中心环境。这意味着您可以将 OpenShift Container Platform 集群部署到在单个 VMware vCenter 中运行的多个 vSphere 数据中心。有关创建此环境的更多信息，请参阅名为 *VMware vSphere 区域和区启用*的部分。

vi. 选择要使用的默认 vCenter 数据存储。

警告

您可以指定数据存储集群中存在的任何数据存储路径。默认情况下，使用 Storage vMotion 的存储分布式资源调度程序(SDRS)会自动为数据存储集群启用。红帽不支持 Storage vMotion，因此您必须禁用 Storage DRS 以避免 OpenShift Container Platform 集群的数据丢失问题。

您不能指定多个数据存储路径。如果需要在多个数据存储间指定虚拟机，请使用 数据存储 对象在集群 install-config.yaml 配置文件中指定故障域。如需更多信息，请参阅"VMware vSphere 区域和区启用"。

vii. 选择要在其中安装 OpenShift Container Platform 集群的 vCenter 集群。安装程序使用 vSphere 集群的 root 资源池作为默认资源池。

viii. 选择包含您配置的虚拟 IP 地址和 DNS 记录的 vCenter 实例中的网络。

ix. 输入您为 control plane API 访问配置的虚拟 IP 地址。

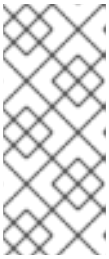
x. 输入您为集群入口配置的虚拟 IP 地址。

xi. 输入基域。这个基域必须与您配置的 DNS 记录中使用的域相同。

xii. 为集群输入描述性名称。

您输入的集群名称必须与您在配置 DNS 记录时指定的集群名称匹配。

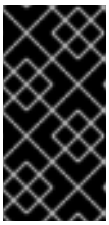
2. 修改 install-config.yaml 文件。您可以在"安装配置参数"部分找到有关可用参数的更多信息。



注意

如果要安装三节点集群，请确保将 compute.replicas 参数设置为 0。这样可确保集群的 control plane 可以调度。如需更多信息，请参阅"在 vSphere 上安装三节点集群"。

3. 备份 install-config.yaml 文件，以便您可以使用它安装多个集群。



重要

install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用此文件，必须现在备份。

其他资源

- 安装配置参数

23.2.4.4.1. 安装程序置备的 VMware vSphere 集群的 install-config.yaml 文件示例

您可以自定义 install-config.yaml 文件，以指定有关 OpenShift Container Platform 集群平台的更多详情，或修改所需参数的值。

```
apiVersion: v1
baseDomain: example.com (1)
compute: (2)
- architecture: amd64
  name:  <worker_node>
  platform: {}
  replicas: 3
controlPlane: (3)
  architecture: amd64
  name: <parent_node>
```

```
  platform: {}
  replicas: 3
metadata:
  creationTimestamp: null
  name: test 4
platform:
  vsphere: 5
    apiVIPs:
      - 10.0.0.1
    failureDomains: 6
    - name: <failure_domain_name>
      region: <default_region_name>
      server: <fully_qualified_domain_name>
      topology:
        computeCluster: "/<datacenter>/host/<cluster>"
        datacenter: <datacenter>
        datastore: "/<datacenter>/datastore/<datastore>" 7
        networks:
        - <VM_Network_name>
        resourcePool: "/<datacenter>/host/<cluster>/Resources/<resourcePool>" 8
        folder: "/<datacenter_name>/vm/<folder_name>/<subfolder_name>"
        tagIDs: 9
        - <tag_id> 10
      zone: <default_zone_name>
    ingressVIPs:
    - 10.0.0.2
    vcenters:
    - datacenters:
      - <datacenter>
      password: <password>
      port: 443
      server: <fully_qualified_domain_name>
      user: administrator@vsphere.local
    diskType: thin 11
fips: false
pullSecret: '{"auths": ...}'
sshKey: 'ssh-ed25519 AAAA...'
```

1

集群的基域。所有 DNS 记录都必须是这个基域的子域，并包含集群名称。

2 3

controlPlane 部分是一个单个映射，但 compute 部分是一系列映射。为满足不同数据结构的要求，compute 部分的第一行必须以连字符 - 开头，controlPlane 部分 的第一行则不以连字符开头。仅使用一个 control plane 池。

4

您在 DNS 记录中指定的集群名称。

5

可选：为 compute 和 control plane 机器提供额外的机器池参数配置。

6

建立地区和区域之间的关系。您可以使用 vCenter 对象（如 datastore 对象）定义故障域。故障域定义 OpenShift Container Platform 集群节点的 vCenter 位置。

7

保存虚拟机文件、模板和 ISO 镜像的 vSphere 数据存储路径。

重要

您可以指定数据存储集群中存在的任何数据存储路径。默认情况下，Storage vMotion 会自动为数据存储集群启用。红帽不支持 Storage vMotion，因此您必须禁用 Storage vMotion 以避免 OpenShift Container Platform 集群的数据丢失问题。

如果需要在多个数据存储间指定虚拟机，请使用 数据存储 对象在集群 install-config.yaml 配置文件中指定故障域。如需更多信息，请参阅"VMware vSphere 区域和区启用"。

8

可选：为创建机器提供现有资源池。如果没有指定值，安装程序将使用 vSphere 集群的 root 资源池。

9

可选：由 OpenShift Container Platform 创建的每个虚拟机都会被分配一个特定于集群的唯一标签。分配的标签可让安装程序在集群停用时识别和删除关联的虚拟机。您可以最多列出十个额外标签 ID，以附加到安装程序置备的虚拟机。

10

安装程序关联的标签的 ID。例如，urn:vmomi:InventoryServiceTag:208e713c-cae3-4b7f-918e-4051ca7d1f97:GLOBAL.有关确定标签 ID 的更多信息，请参阅 vSphere 标签和属性文档。

11

vSphere 磁盘置备方法。

注意

在 OpenShift Container Platform 4.12 及更新的版本中，apiVIP 和 ingressVIP 配置设置已弃用。反之，使用列表格式在 apiVIPs 和 ingressVIPs 配置设置中输入值。

23.2.4.4.2. 在安装过程中配置集群范围的代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 install-config.yaml 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 install-config.yaml 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 Proxy 对象的 spec.noProxy 字段中添加站点来绕过代理。

注意

Proxy 对象 status.noProxy 字段使用安装配置中的 networking.machineNetwork[].cidr、networking.clusterNetwork[].cidr 和 networking.serviceNetwork[] 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，Proxy 对象 status.noProxy 字段也会使用实例元数据端点填充(169.254.169.254)。

流程

1. 编辑 install-config.yaml 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
```

```
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 http。

2 用于创建集群外 HTTPS 连接的代理 URL。

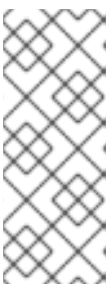
3 要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 . 以仅匹配子域。例如，.y.com 匹配 x.y.com，但不匹配 y.com。使用 * 绕过所有目的地的代理。您必须包含 vCenter 的 IP 地址以及用于其机器的 IP 范围。

4 如果提供，安装程序会在 openshift-config 命名空间中生成名为 user-ca-bundle 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 trusted-ca-bundle 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， Proxy 对象的 trustedCA 字段中也会引用此配置映射。additionalTrustBundle 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

5 可选：决定 Proxy 对象的配置以引用 trustedCA 字段中 user-ca-bundle 配置映射的策略。允许的值是 Proxyonly 和 Always。仅在配置了 http/https 代理时，使用 Proxyonly 引用 user-ca-bundle 配置映射。使用 Always 始终引用 user-ca-bundle 配置映射。默认值为 Proxyonly。

注意

安装程序不支持代理的 readinessEndpoints 字段。



注意

如果安装程序超时，重启并使用安装程序的 wait-for 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. 保存该文件并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 cluster 的集群范围代理，该代理 使用 提供的 install-config.yaml 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 cluster Proxy 对象，但它会有一个空 spec。

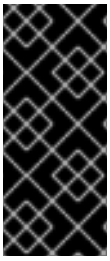
注意

只支持名为 cluster 的 Proxy 对象，且无法创建额外的代理。

23.2.4.4.3. 为 VMware vCenter 配置区域和区域

您可以修改默认安装配置文件，以便您可以将 OpenShift Container Platform 集群部署到在单个 VMware vCenter 中运行的多个 vSphere 数据中心。

之前版本的 OpenShift Container Platform 的默认 install-config.yaml 文件配置已弃用。您可以继续使用已弃用的默认配置，但 openshift-installer 会提示您显示在配置文件中已弃用字段的警告信息。



重要

这个示例使用 govc 命令。govc 命令是 VMware 提供的开源命令；它不是红帽提供的。红帽支持团队不维护 govc 命令。有关下载和安装 govc 的说明，请参阅 VMware 文档网站

先决条件

- 您有一个现有的 install-config.yaml 安装配置文件。

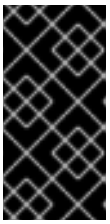
重要

您必须为 OpenShift Container Platform 集群指定一个故障域，以便您可以为 VMware vCenter 服务器置备数据中心对象。如果您需要在不同的数据中心、集群、数据存储和其他组件中置备虚拟机节点，请考虑指定多个故障域。要启用区域和区域，您必须为 OpenShift Container Platform 集群定义多个故障域。

流程

1. 输入以下 govc 命令行工具命令，以创建 openshift-region 和 openshift-zone vCenter 标

签类别：



重要

如果为 **openshift-region** 和 **openshift-zone vCenter** 标签类别指定不同的名称，**OpenShift Container Platform** 集群的安装会失败。

```
$ govc tags.category.create -d "OpenShift region" openshift-region
```

```
$ govc tags.category.create -d "OpenShift zone" openshift-zone
```

2. 要为您要部署集群的每个区域 **vSphere** 数据中心创建一个 **region** 标签，请在终端中输入以下命令：

```
$ govc tags.create -c <region_tag_category> <region_tag>
```

3. 要为您要部署集群的每个 **vSphere** 集群创建一个区标签，请输入以下命令：

```
$ govc tags.create -c <zone_tag_category> <zone_tag>
```

4. 输入以下命令将区域标签附加到每个 **vCenter** 数据中心对象：

```
$ govc tags.attach -c <region_tag_category> <region_tag_1> /<datacenter_1>
```

5. 输入以下命令将区标签附加到每个 **vCenter** 数据中心对象：

```
$ govc tags.attach -c <zone_tag_category> <zone_tag_1> /<datacenter_1>/host/vcs-mdcnc-workload-1
```

6. 进入包含安装程序的目录，并根据您选择的安装要求初始化集群部署。

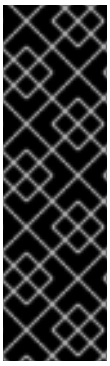
在 vSphere 数据中心中定义的多个数据中心的 install-config.yaml 文件示例

```
---
compute:
---
```

```
  vsphere:
    zones:
      - "<machine_pool_zone_1>"
      - "<machine_pool_zone_2>"
---
controlPlane:
---
vsphere:
    zones:
      - "<machine_pool_zone_1>"
      - "<machine_pool_zone_2>"
---
platform:
  vsphere:
    vcenters:
---
    datacenters:
      - <datacenter1_name>
      - <datacenter2_name>
    failureDomains:
    - name: <machine_pool_zone_1>
      region: <region_tag_1>
      zone: <zone_tag_1>
      server: <fully_qualified_domain_name>
      topology:
        datacenter: <datacenter1>
        computeCluster: "/<datacenter1>/host/<cluster1>"
        networks:
        - <VM_Network1_name>
        datastore: "/<datacenter1>/datastore/<datastore1>"
        resourcePool: "/<datacenter1>/host/<cluster1>/Resources/<resourcePool1>"
        folder: "/<datacenter1>/vm/<folder1>"
    - name: <machine_pool_zone_2>
      region: <region_tag_2>
      zone: <zone_tag_2>
      server: <fully_qualified_domain_name>
      topology:
        datacenter: <datacenter2>
        computeCluster: "/<datacenter2>/host/<cluster2>"
        networks:
        - <VM_Network2_name>
        datastore: "/<datacenter2>/datastore/<datastore2>"
        resourcePool: "/<datacenter2>/host/<cluster2>/Resources/<resourcePool2>"
        folder: "/<datacenter2>/vm/<folder2>"
---
```

23.2.4.5. 用户管理的负载均衡器的服务

您可以将 OpenShift Container Platform 集群配置为使用用户管理的负载均衡器来代替默认负载均衡器。



重要

配置用户管理的负载均衡器取决于您的厂商的负载均衡器。

本节中的信息和示例仅用于指导目的。有关供应商负载均衡器的更多信息，请参阅供应商文档。

红帽支持用户管理的负载均衡器的以下服务：

- Ingress Controller
- OpenShift API
- OpenShift MachineConfig API

您可以选择是否要为用户管理的负载均衡器配置一个或多个所有服务。仅配置 Ingress Controller 服务是一个通用的配置选项。要更好地了解每个服务，请查看以下图表：

图 23.1. 显示 OpenShift Container Platform 环境中运行的 Ingress Controller 的网络工作流示例

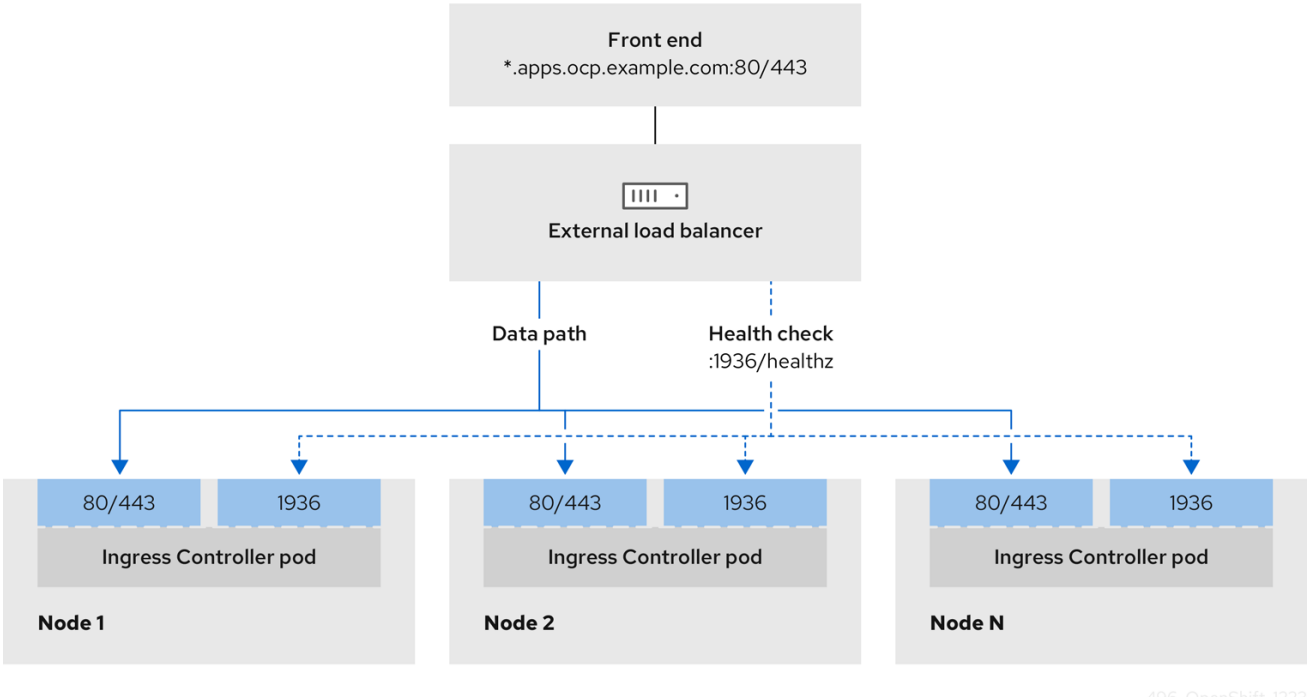


图 23.2. 显示 OpenShift Container Platform 环境中运行的 OpenShift API 的网络工作流示例

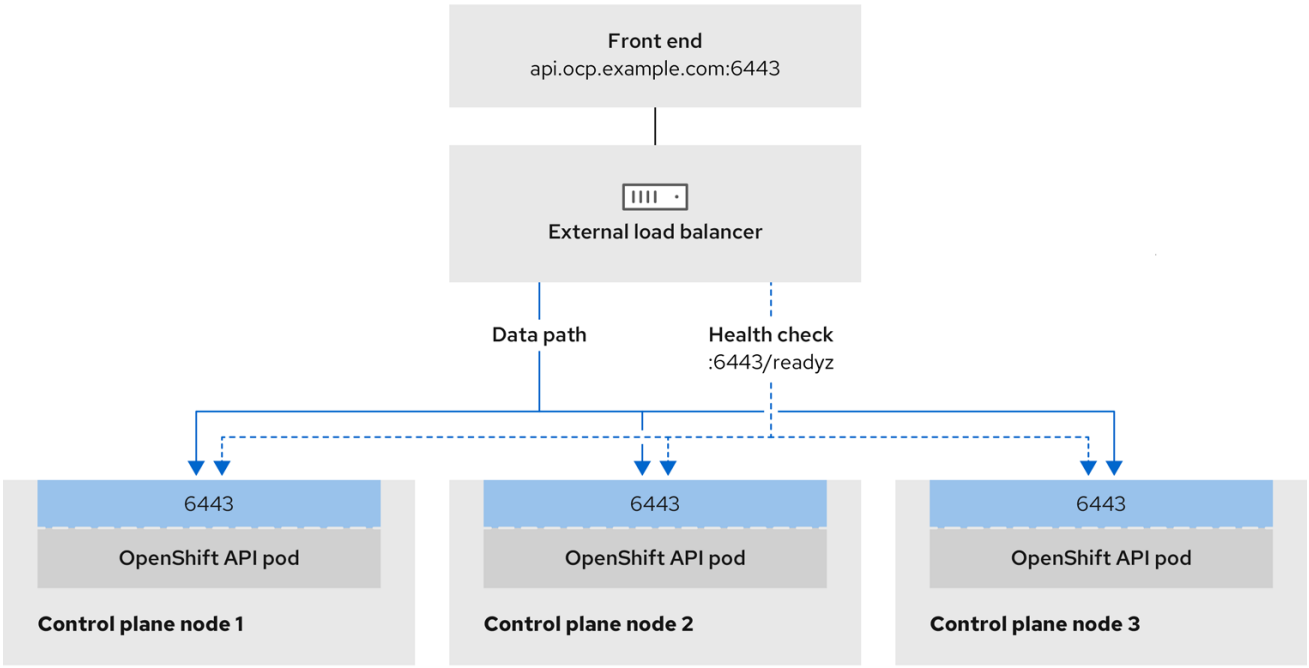
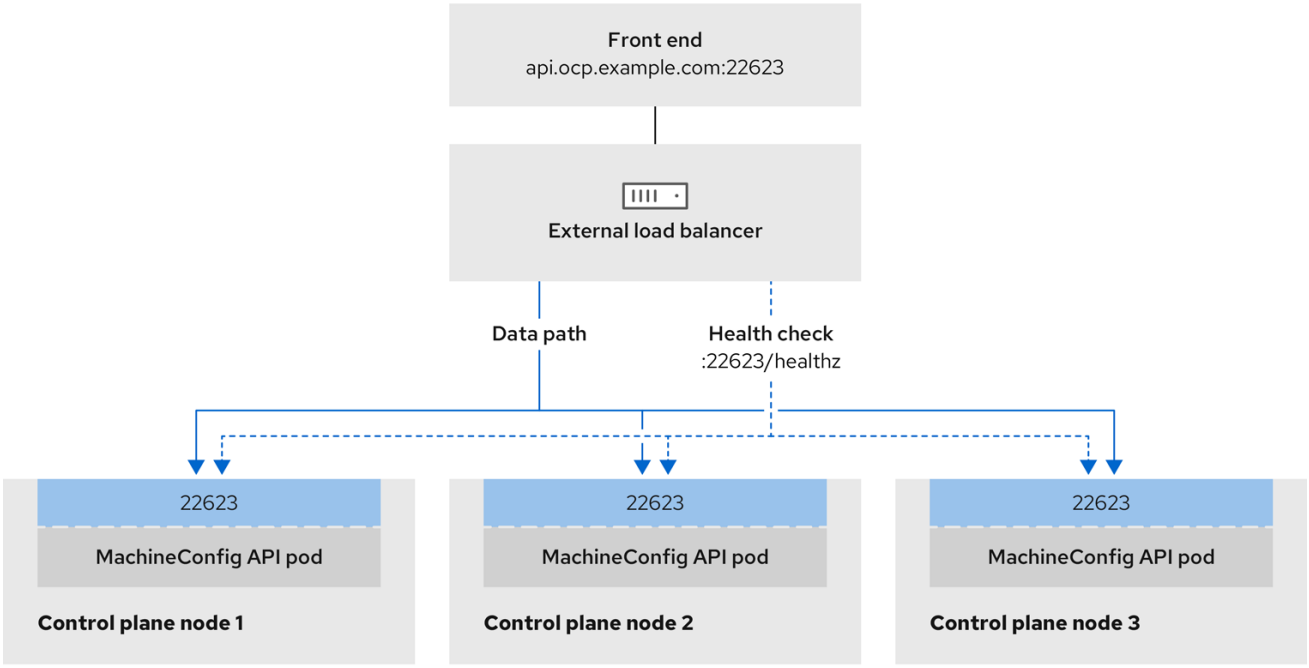


图 23.3. 显示 OpenShift Container Platform 环境中运行的 OpenShift MachineConfig API 的网络工作流示例



用户管理的负载均衡器支持以下配置选项：

- 使用节点选择器将 Ingress Controller 映射到一组特定的节点。您必须为这个集合中的每个节点分配一个静态 IP 地址，或者将每个节点配置为从动态主机配置协议(DHCP)接收相同的 IP 地

址。基础架构节点通常接收这种类型的配置。

- 以子网上的所有 IP 地址为目标。此配置可减少维护开销，因为您可以在这些网络中创建和销毁节点，而无需重新配置负载均衡器目标。如果您使用较小的网络上的机器集来部署入口 pod，如 /27 或 /28，您可以简化负载均衡器目标。

  提示

  您可以通过检查机器配置池的资源来列出网络中存在的所有 IP 地址。

在为 OpenShift Container Platform 集群配置用户管理的负载均衡器前，请考虑以下信息：

- 对于前端 IP 地址，您可以对前端 IP 地址、Ingress Controller 的负载均衡器和 API 负载均衡器使用相同的 IP 地址。查看厂商的文档以获取此功能的相关信息。

- 对于后端 IP 地址，请确保 OpenShift Container Platform control plane 节点的 IP 地址在用户管理的负载均衡器生命周期内不会改变。您可以通过完成以下操作之一来实现此目的：

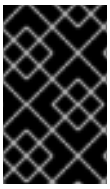
  - 为每个 control plane 节点分配一个静态 IP 地址。

  - 将每个节点配置为在每次节点请求 DHCP 租期时从 DHCP 接收相同的 IP 地址。根据供应商，DHCP 租期可能采用 IP 保留或静态 DHCP 分配的形式。

- 在 Ingress Controller 后端服务的用户管理的负载均衡器中手动定义运行 Ingress Controller 的每个节点。例如，如果 Ingress Controller 移到未定义节点，则可能会出现连接中断。

23.2.4.5.1. 配置用户管理的负载均衡器

您可以将 OpenShift Container Platform 集群配置为使用用户管理的负载均衡器来代替默认负载均衡器。



重要

在配置用户管理的负载均衡器前，请确保阅读用户管理的负载均衡器部分。

阅读适用于您要为用户管理的负载均衡器配置的服务的以下先决条件。

注意

MetalLB，在集群中运行，充当用户管理的负载均衡器。

OpenShift API 的先决条件

- 您定义了前端 IP 地址。
- TCP 端口 6443 和 22623 在负载均衡器的前端 IP 地址上公开。检查以下项：
  - 端口 6443 提供对 OpenShift API 服务的访问。
  - 端口 22623 可以为节点提供 ignition 启动配置。
- 前端 IP 地址和端口 6443 可以被您的系统的所有用户访问，其位置为 OpenShift Container Platform 集群外部。
- 前端 IP 地址和端口 22623 只能被 OpenShift Container Platform 节点访问。
- 负载均衡器后端可以在端口 6443 和 22623 上与 OpenShift Container Platform control plane 节点通信。

Ingress Controller 的先决条件

- 您定义了前端 IP 地址。

- TCP 端口 443 和 80 在负载均衡器的前端 IP 地址上公开。

- 前端 IP 地址、端口 80 和端口 443 可以被您的系统所有用户访问，以及 OpenShift Container Platform 集群外部的位置。

- 前端 IP 地址、端口 80 和端口 443 可被 OpenShift Container Platform 集群中运行的所有节点访问。

- 负载均衡器后端可以在端口 80、443 和 1936 上与运行 Ingress Controller 的 OpenShift Container Platform 节点通信。

健康检查 URL 规格的先决条件

您可以通过设置健康检查 URL 来配置大多数负载均衡器，以确定服务是否可用或不可用。OpenShift Container Platform 为 OpenShift API、Machine Configuration API 和 Ingress Controller 后端服务提供这些健康检查。

以下示例显示了之前列出的后端服务的健康检查规格：

Kubernetes API 健康检查规格示例

```
Path: HTTPS:6443/readyz
Healthy threshold: 2
Unhealthy threshold: 2
Timeout: 10
Interval: 10
```

Machine Config API 健康检查规格示例

```
Path: HTTPS:22623/healthz
Healthy threshold: 2
Unhealthy threshold: 2
Timeout: 10
Interval: 10
```

Ingress Controller 健康检查规格示例

```
Path: HTTP:1936/healthz/ready
Healthy threshold: 2
Unhealthy threshold: 2
Timeout: 5
Interval: 10
```

流程

1. **配置 HAProxy Ingress Controller，以便您可以在端口 6443、22623、443 和 80 上从负载均衡器访问集群。根据您的需要，您可以在 HAProxy 配置中指定来自多个子网的单个子网或 IP 地址的 IP 地址。**

   带有列出子网的 HAProxy 配置示例

```
# ...
listen my-cluster-api-6443
    bind 192.168.1.100:6443
    mode tcp
    balance roundrobin
  option httpchk
  http-check connect
  http-check send meth GET uri /readyz
  http-check expect status 200
    server my-cluster-master-2 192.168.1.101:6443 check inter 10s rise 2 fall 2
    server my-cluster-master-0 192.168.1.102:6443 check inter 10s rise 2 fall 2
    server my-cluster-master-1 192.168.1.103:6443 check inter 10s rise 2 fall 2

listen my-cluster-machine-config-api-22623
    bind 192.168.1.100:22623
    mode tcp
    balance roundrobin
  option httpchk
  http-check connect
  http-check send meth GET uri /healthz
  http-check expect status 200
    server my-cluster-master-2 192.168.1.101:22623 check inter 10s rise 2 fall 2
    server my-cluster-master-0 192.168.1.102:22623 check inter 10s rise 2 fall 2
    server my-cluster-master-1 192.168.1.103:22623 check inter 10s rise 2 fall 2
```

```
listen my-cluster-apps-443
    bind 192.168.1.100:443
    mode tcp
    balance roundrobin
  option httpchk
  http-check connect
  http-check send meth GET uri /healthz/ready
  http-check expect status 200
    server my-cluster-worker-0 192.168.1.111:443 check port 1936 inter 10s rise 2 fall 2
    server my-cluster-worker-1 192.168.1.112:443 check port 1936 inter 10s rise 2 fall 2
    server my-cluster-worker-2 192.168.1.113:443 check port 1936 inter 10s rise 2 fall 2

listen my-cluster-apps-80
   bind 192.168.1.100:80
   mode tcp
   balance roundrobin
  option httpchk
  http-check connect
  http-check send meth GET uri /healthz/ready
  http-check expect status 200
    server my-cluster-worker-0 192.168.1.111:80 check port 1936 inter 10s rise 2 fall 2
    server my-cluster-worker-1 192.168.1.112:80 check port 1936 inter 10s rise 2 fall 2
    server my-cluster-worker-2 192.168.1.113:80 check port 1936 inter 10s rise 2 fall 2
# ...
```

带有多个列出子网的 HAProxy 配置示例

```
# ...
listen api-server-6443
    bind *:6443
    mode tcp
      server master-00 192.168.83.89:6443 check inter 1s
      server master-01 192.168.84.90:6443 check inter 1s
      server master-02 192.168.85.99:6443 check inter 1s
      server bootstrap 192.168.80.89:6443 check inter 1s

listen machine-config-server-22623
    bind *:22623
    mode tcp
      server master-00 192.168.83.89:22623 check inter 1s
      server master-01 192.168.84.90:22623 check inter 1s
      server master-02 192.168.85.99:22623 check inter 1s
      server bootstrap 192.168.80.89:22623 check inter 1s

listen ingress-router-80
    bind *:80
    mode tcp
    balance source
```

```
    server worker-00 192.168.83.100:80 check inter 1s
    server worker-01 192.168.83.101:80 check inter 1s

listen ingress-router-443
  bind *:443
  mode tcp
  balance source
    server worker-00 192.168.83.100:443 check inter 1s
    server worker-01 192.168.83.101:443 check inter 1s

listen ironic-api-6385
  bind *:6385
  mode tcp
  balance source
    server master-00 192.168.83.89:6385 check inter 1s
    server master-01 192.168.84.90:6385 check inter 1s
    server master-02 192.168.85.99:6385 check inter 1s
    server bootstrap 192.168.80.89:6385 check inter 1s

listen inspector-api-5050
  bind *:5050
  mode tcp
  balance source
    server master-00 192.168.83.89:5050 check inter 1s
    server master-01 192.168.84.90:5050 check inter 1s
    server master-02 192.168.85.99:5050 check inter 1s
    server bootstrap 192.168.80.89:5050 check inter 1s
# ...
```

2. 使用 curl CLI 命令验证用户管理的负载均衡器及其资源是否正常运行：

   a. 运行以下命令并查看响应，验证集群机器配置 API 是否可以被 Kubernetes API 服务器资源访问：

```
$ curl https://<loadbalancer_ip_address>:6443/version --insecure
```

   如果配置正确，您会收到 JSON 对象的响应：

```
{
  "major": "1",
  "minor": "11+",
  "gitVersion": "v1.11.0+ad103ed",
  "gitCommit": "ad103ed",
  "gitTreeState": "clean",
  "buildDate": "2019-01-09T06:44:10Z",
  "goVersion": "go1.10.3",
```

```
  "compiler": "gc",
  "platform": "linux/amd64"
}
```

b. 运行以下命令并观察输出，验证集群机器配置 API 是否可以被 Machine 配置服务器资源访问：

```
$ curl -v https://<loadbalancer_ip_address>:22623/healthz --insecure
```

如果配置正确，命令的输出会显示以下响应：

```
HTTP/1.1 200 OK
Content-Length: 0
```

c. 运行以下命令并观察输出，验证控制器是否可以被端口 80 上的 Ingress Controller 资源访问：

```
$ curl -I -L -H "Host: console-openshift-console.apps.<cluster_name>.
<base_domain>" http://<load_balancer_front_end_IP_address>
```

如果配置正确，命令的输出会显示以下响应：

```
HTTP/1.1 302 Found
content-length: 0
location: https://console-openshift-console.apps.ocp4.private.opequon.net/
cache-control: no-cache
```

d. 运行以下命令并观察输出，验证控制器是否可以被端口 443 上的 Ingress Controller 资源访问：

```
$ curl -I -L --insecure --resolve console-openshift-console.apps.<cluster_name>.
<base_domain>:443:<Load Balancer Front End IP Address> https://console-
openshift-console.apps.<cluster_name>.<base_domain>
```

如果配置正确，命令的输出会显示以下响应：

```
HTTP/1.1 200 OK
referrer-policy: strict-origin-when-cross-origin
set-cookie: csrf-
token=UlYWOyQ62LWjw2h003xtYSKlh1a0Py2hhctw0WmV2YEdhJjFyQwWcGBsja2
61dGLgaYO0nxzVErhiXt6QepA7g==; Path=/; Secure; SameSite=Lax
x-content-type-options: nosniff
```

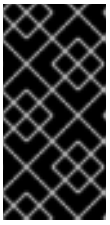
```
x-dns-prefetch-control: off
x-frame-options: DENY
x-xss-protection: 1; mode=block
date: Wed, 04 Oct 2023 16:29:38 GMT
content-type: text/html; charset=utf-8
set-cookie:
1e2670d92730b515ce3a1bb65da45062=1bf5e9573c9a2760c964ed1659cc1673;
path=/; HttpOnly; Secure; SameSite=None
cache-control: private
```

3. 配置集群的 DNS 记录，使其以用户管理的负载均衡器的前端 IP 地址为目标。您必须在负载均衡器上将记录更新为集群 API 和应用程序的 DNS 服务器。

修改 DNS 记录示例

```
<load_balancer_ip_address>  A  api.<cluster_name>.<base_domain>
A record pointing to Load Balancer Front End
```

```
<load_balancer_ip_address>   A apps.<cluster_name>.<base_domain>
A record pointing to Load Balancer Front End
```



重要

DNS 传播可能需要一些时间才能获得每个 DNS 记录。在验证每个记录前，请确保每个 DNS 记录传播。

4. 要使 OpenShift Container Platform 集群使用用户管理的负载均衡器，您必须在集群的 install-config.yaml 文件中指定以下配置：

```
# ...
platform:
  vsphere:
    loadBalancer:
      type: UserManaged 1
      apiVIPs:
      - <api_ip> 2
      ingressVIPs:
      - <ingress_ip> 3
# ...
```

1

为 type 参数设置 UserManaged，为集群指定用户管理的负载均衡器。参数默认为 OpenShiftManagedDefault，它表示默认的内部负载均衡器。对于 openshift-kni-infra 命名空间中定义的服务，用户管理的负载均衡器可将 coredns 服务部署到集群中的 pod，但忽略 keepalived 和 haproxy 服务。

2

指定用户管理的负载均衡器时所需的参数。指定用户管理的负载均衡器的公共 IP 地址，以便 Kubernetes API 可以与用户管理的负载均衡器通信。

3

指定用户管理的负载均衡器时所需的参数。指定用户管理的负载均衡器的公共 IP 地址，以便用户管理的负载均衡器可以管理集群的入口流量。

验证

1. 使用 curl CLI 命令验证用户管理的负载均衡器和 DNS 记录配置是否正常工作：

   a. 运行以下命令并查看输出，验证您可以访问集群 API：

   ```
   $ curl https://api.<cluster_name>.<base_domain>:6443/version --insecure
   ```

   如果配置正确，您会收到 JSON 对象的响应：

   ```
   {
    "major": "1",
    "minor": "11+",
    "gitVersion": "v1.11.0+ad103ed",
    "gitCommit": "ad103ed",
    "gitTreeState": "clean",
    "buildDate": "2019-01-09T06:44:10Z",
    "goVersion": "go1.10.3",
    "compiler": "gc",
    "platform": "linux/amd64"
    }
   ```

   b. 运行以下命令并查看输出，验证您可以访问集群机器配置：

   ```
   $ curl -v https://api.<cluster_name>.<base_domain>:22623/healthz --insecure
   ```

如果配置正确，命令的输出会显示以下响应：

```
HTTP/1.1 200 OK
Content-Length: 0
```

c. 运行以下命令并查看输出，验证您可以在端口上访问每个集群应用程序：

```
$ curl http://console-openshift-console.apps.<cluster_name>.<base_domain> -I -L
--insecure
```

如果配置正确，命令的输出会显示以下响应：

```
HTTP/1.1 302 Found
content-length: 0
location: https://console-openshift-console.apps.<cluster-name>.<base domain>/
cache-control: no-cacheHTTP/1.1 200 OK
referrer-policy: strict-origin-when-cross-origin
set-cookie: csrf-
token=39HoZgztDnzjJkq/JuLJMeoKNXlfiVv2YgZc09c3TBOBU4Nl6kDXaJH1LdicNh
N1UsQWzon4Dor9GWGfopaTEQ==; Path=/; Secure
x-content-type-options: nosniff
x-dns-prefetch-control: off
x-frame-options: DENY
x-xss-protection: 1; mode=block
date: Tue, 17 Nov 2020 08:42:10 GMT
content-type: text/html; charset=utf-8
set-cookie:
1e2670d92730b515ce3a1bb65da45062=9b714eb87e93cf34853e87a92d6894be;
path=/; HttpOnly; Secure; SameSite=None
cache-control: private
```

d. 运行以下命令并查看输出，验证您可以在端口 443 上访问每个集群应用程序：

```
$ curl https://console-openshift-console.apps.<cluster_name>.<base_domain> -I -
L --insecure
```

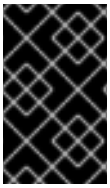
如果配置正确，命令的输出会显示以下响应：

```
HTTP/1.1 200 OK
referrer-policy: strict-origin-when-cross-origin
set-cookie: csrf-
token=UlYWOyQ62LWjw2h003xtYSKlh1a0Py2hhctw0WmV2YEdhJjFyQwWcGBsja2
61dGLgaYO0nxzVErhiXt6QepA7g==; Path=/; Secure; SameSite=Lax
x-content-type-options: nosniff
```

```
x-dns-prefetch-control: off
x-frame-options: DENY
x-xss-protection: 1; mode=block
date: Wed, 04 Oct 2023 16:29:38 GMT
content-type: text/html; charset=utf-8
set-cookie:
1e2670d92730b515ce3a1bb65da45062=1bf5e9573c9a2760c964ed1659cc1673;
path=/; HttpOnly; Secure; SameSite=None
cache-control: private
```

23.2.4.6. 部署集群

您可以在兼容云平台上安装 OpenShift Container Platform。



重要

在初始安装过程中，您只能运行安装程序的 create cluster 命令一次。

先决条件

- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。
- 已确认主机上的云供应商帐户具有部署集群的正确权限。权限不正确的帐户会导致安装过程失败，并显示包括缺失权限的错误消息。
- 可选：在创建集群时，配置外部负载均衡器来代替默认负载均衡器。

重要

您不需要为安装程序指定 API 和 Ingress 静态地址。如果选择此配置，则必须采取额外的操作来定义接受每个引用的 vSphere 子网的 IP 地址的网络目标。请参阅"配置用户管理的负载均衡器"部分。

流程

- 进入包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

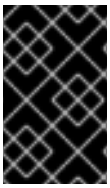
1 对于 <installation_directory>，请指定自定义 ./install-config.yaml 文件的位置。

2 要查看不同的安装详情，请指定 warn、debug 或 error，而不是 info。

验证

当集群部署成功完成时：

- 终端会显示用于访问集群的说明，包括指向 Web 控制台和 kubeadmin 用户的凭证的链接。
- 凭证信息还会输出到 <installation_directory>/.openshift_install.log.

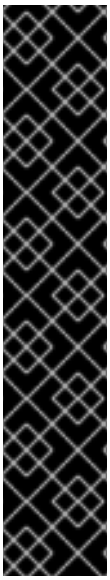


重要

不要删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```



重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 node-bootstrapper 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。

- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

23.2.4.7. 使用 CLI 登录集群

您可以通过导出集群 kubeconfig 文件，以默认系统用户身份登录集群。kubeconfig 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。
- 已安装 oc CLI。

流程

1. 导出 kubeadmin 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 <installation_directory>，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 oc 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

23.2.4.8. 创建 registry 存储

安装集群后，必须为 registry Operator 创建存储。

23.2.4.8.1. 安装过程中删除的镜像 registry

在不提供可共享对象存储的平台上，OpenShift Image Registry Operator bootstraps 本身为 Removed。这允许 openshift-installer 在这些平台类型上完成安装。

安装后，您必须编辑 Image Registry Operator 配置，将 managementState 从 Removed 切换到 Managed。完成此操作后，您必须配置存储。

23.2.4.8.2. 镜像 registry 存储配置

对于不提供默认存储的平台，Image Registry Operator 最初不可用。安装后，您必须将 registry 配置为使用存储，以便 Registry Operator 可用。

显示配置生产集群所需的持久性卷的说明。如果适用，显示有关将空目录配置为存储位置的说明，这仅适用于非生产集群。

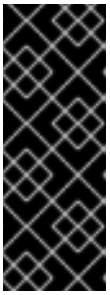
提供了在升级过程中使用 Recreate rollout 策略来允许镜像 registry 使用块存储类型的说明。

23.2.4.8.2.1. 为 VMware vSphere 配置 registry 存储

作为集群管理员，在安装后需要配置 registry 来使用存储。

先决条件

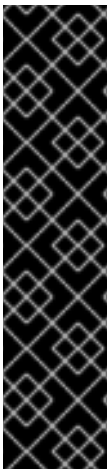
- 集群管理员权限。
- VMware vSphere 上有一个集群。
- 为集群置备的持久性存储，如 Red Hat OpenShift Data Foundation。



重要

当您只有一个副本时，OpenShift Container Platform 支持对镜像 registry 存储的 ReadWriteOnce 访问。ReadWriteOnce 访问还要求 registry 使用 Recreate rollout 策略。要部署支持高可用性的镜像 registry，需要两个或多个副本，ReadWriteMany 访问。

- 必须具有"100Gi"容量.



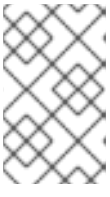
重要

测试显示在 RHEL 中使用 NFS 服务器作为核心服务的存储后端的问题。这包括 OpenShift Container Registry 和 Quay，Prometheus 用于监控存储，以及 Elasticsearch 用于日志存储。因此，不建议使用 RHEL NFS 作为 PV 后端用于核心服务。

市场上的其他 NFS 实现可能没有这些问题。如需了解更多与此问题相关的信息，请联络相关的 NFS 厂商。

流程

1. 要将 registry 配置为使用存储，修改 configs.imageregistry/cluster 资源中的 spec.storage.pvc。



注意

使用共享存储时，请查看您的安全设置以防止外部访问。

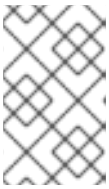
2.

2. 验证您没有 registry pod:

```
$ oc get pod -n openshift-image-registry -l docker-registry=default
```

输出示例

```
No resourses found in openshift-image-registry namespace
```



注意

如果您的输出中有一个 registry pod，则不需要继续这个过程。

3. 检查 registry 配置：

```
$ oc edit configs.imageregistry.operator.openshift.io
```

输出示例

```
storage:
  pvc:
    claim: 1
```

1

将 claim 字段留空以允许自动创建 image-registry-storage 持久性卷声明(PVC)。PVC 基于默认存储类生成。但请注意，默认存储类可能会提供 ReadWriteOnce (RWO) 卷，如 RADOS 块设备(RBD)，这可能会在复制到多个副本时导致问题。

4. 检查 clusteroperator 状态：

```
$ oc get clusteroperator image-registry
```

输出示例

```
NAME              VERSION                             AVAILABLE   PROGRESSING   DEGRADED
SINCE   MESSAGE
image-registry    4.7                                 True        False         False     6h50m
```

23.2.4.8.2.2. 为 VMware vSphere 配置块 registry 存储

要允许镜像 registry 在作为集群管理员升级过程中使用块存储类型，如 vSphere Virtual Machine Disk(VMDK)，您可以使用 Recreate rollout 策略。

重要

支持块存储卷，但不建议在生产环境中用于镜像 registry。在块存储上配置 registry 的安装不具有高可用性，因为 registry 无法具有多个副本。

流程

1. 输入以下命令将镜像 registry 存储设置为块存储类型，对 registry 进行补丁，使其使用 Recreate rollout 策略，并只使用一个副本运行：

```
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec":
{"rolloutStrategy":"Recreate","replicas":1}}'
```

2. 为块存储设备置备 PV，并为该卷创建 PVC。请求的块卷使用 ReadWriteOnce(RWO)访问模式。

   a. 创建包含以下内容的 pvc.yaml 文件以定义 VMware vSphere PersistentVolumeClaim 对象：

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: image-registry-storage 1
  namespace: openshift-image-registry 2
```

```
spec:
  accessModes:
  - ReadWriteOnce (3)
  resources:
    requests:
      storage: 100Gi (4)
```

(1) 代表 PersistentVolumeClaim 对象的唯一名称。

(2) PersistentVolumeClaim 对象的命名空间，即 openshift-image-registry。

(3) 持久性卷声明的访问模式。使用 ReadWriteOnce 时，单个节点可以通过读写权限挂载该卷。

(4) 持久性卷声明的大小。

b. 输入以下命令从文件创建 PersistentVolumeClaim 对象：

```
$ oc create -f pvc.yaml -n openshift-image-registry
```

3. 输入以下命令编辑 registry 配置，使其引用正确的 PVC：

```
$ oc edit config.imageregistry.operator.openshift.io -o yaml
```

输出示例

```
storage:
  pvc:
    claim: (1)
```

(1)

有关配置 registry 存储以便引用正确的 PVC 的说明，请参阅 为 vSphere 配置 registry。

23.2.4.9. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于 远程健康监控

23.2.4.10. 后续步骤

- 自定义集群。
- 如果需要，您可以选择 不使用远程健康报告。
- 设置 registry 并配置 registry 存储。
- 可选： 查看 vSphere 问题检测器 Operator 中的事件，以确定集群是否有权限或存储配置问题。

23.2.5. 使用自定义网络在 vSphere 上安装集群

在 OpenShift Container Platform 版本 4.16 中，您可以使用安装程序置备的基础架构和自定义的网络配置选项在 VMware vSphere 实例上安装集群。通过自定义网络配置，您的集群可以与环境中现有的 IP 地址分配共存，并与现有的 MTU 和 VXLAN 配置集成。要自定义安装，请在安装集群前修改 install-config.yaml 文件中的参数。

您必须在安装过程中设置大多数网络配置参数，且您只能在正在运行的集群中修改 kubeProxy 配置参数。

注意

OpenShift Container Platform 支持将集群部署到单个 VMware vCenter 中。不支持在多个 vCenter 上使用机器/机器集部署集群。

23.2.5.1. 先决条件

- 您已完成了 准备使用安装程序置备的基础架构安装集群 中的任务。
- 您检查了 VMware 平台许可证。红帽不会对 VMware 许可证产生任何限制，但有些 VMware 基础架构组件需要许可。
- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读有关 选择集群安装方法的文档，并为用户准备它。
- 已为集群置备了 持久性存储。要部署私有镜像 registry，您的存储必须提供 ReadWriteMany 访问模式。
- OpenShift Container Platform 安装程序需要访问 vCenter 和 ESXi 主机上的端口 443。您确认可以访问端口 443。
- 如果您使用防火墙，请与管理员一起确认可以访问端口 443。control plane 节点必须能够通过端口 443 访问 vCenter 和 ESXi 主机，才能成功安装。
- 如果使用防火墙，则会 将其配置为允许集群需要访问的站点。

注意

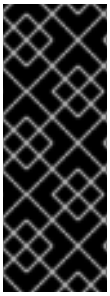
如果要配置代理，请务必查看此站点列表。

23.2.5.2. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

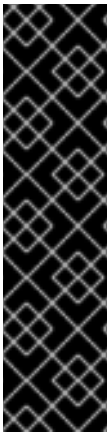


重要

如果您的集群无法直接访问互联网，则可以在置备的某些类型的基础架构上执行受限网络安装。在此过程中，您可以下载所需的内容，并使用它为镜像 registry 填充安装软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群前，您要更新镜像 registry 的内容。

23.2.5.3. VMware vSphere 区域和区启用

您可以将 OpenShift Container Platform 集群部署到在单个 VMware vCenter 中运行的多个 vSphere 数据中心。每个数据中心都可以运行多个集群。此配置降低了导致集群失败的硬件故障或网络中断的风险。要启用区域和区域，您必须为 OpenShift Container Platform 集群定义多个故障域。



重要

VMware vSphere 区域和区启用功能需要 vSphere Container Storage Interface (CSI) 驱动程序作为集群中的默认存储驱动程序。因此，这个功能只在新安装的集群中可用。

对于从上一版本升级的集群，您必须为集群启用 CSI 自动迁移。然后，您可以为升级的集群配置多个区域和区域。

默认安装配置将集群部署到单个 vSphere 数据中心。如果要将集群部署到多个 vSphere 数据中心，

您必须创建一个启用地区和区功能的安装配置文件。

默认 install-config.yaml 文件包含 vcenters 和 failureDomains 字段，您可以在其中为 OpenShift Container Platform 集群指定多个 vSphere 数据中心和集群。如果要在由单个数据中心组成的 vSphere 环境中安装 OpenShift Container Platform 集群，您可以将这些字段留空。

以下列表描述了为集群定义区和区域相关的术语：

- 故障域：建立地区和区域之间的关系。您可以使用 vCenter 对象（如 datastore 对象）定义故障域。故障域定义 OpenShift Container Platform 集群节点的 vCenter 位置。
- Region ：指定 vCenter 数据中心。您可以使用 openshift-region 标签类别中的标签来定义区域。
- Zone：指定一个 vCenter 集群。您可以使用 openshift-zone 标签类别中的标签来定义区。

注意

如果您计划在 install-config.yaml 文件中指定多个故障域，则必须在创建配置文件前创建标签类别、区域标签和区域标签。

您必须为每个代表一个区域的 vCenter 数据中心创建一个 vCenter 标签。另外，您必须为比数据中心（代表一个区）中运行的每个集群创建一个 vCenter 标签。创建标签后，您必须将每个标签附加到对应的数据中心和集群。

下表概述了在单个 VMware vCenter 中运行的多个 vSphere 数据中心的区域、区域和标签之间的关系示例。

数据中心（区域）	集群（区）	Tags
us-east	us-east-1	us-east-1a
		us-east-1b
	us-east-2	us-east-2a

数据中心（区域）	集群（区）	Tags
		us-east-2b
us-west	us-west-1	us-west-1a
		us-west-1b
	us-west-2	us-west-2a
		us-west-2b

其他资源

- 其他 VMware vSphere 配置参数
- 弃用的 VMware vSphere 配置参数
- vSphere 自动迁移
- VMware vSphere CSI Driver Operator

23.2.5.4. 创建安装配置文件

您可以自定义在 VMware vSphere 上安装的 OpenShift Container Platform 集群。

先决条件

- **您有 OpenShift Container Platform 安装程序和集群的 pull secret。**

流程

1. **创建 install-config.yaml 文件。**
   a. **进入包含安装程序的目录并运行以下命令：**

```
$ ./openshift-install create install-config --dir <installation_directory> 1
```

1

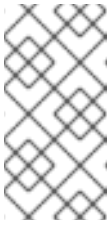
对于 <installation_directory>，请指定要存储安装程序创建的文件的目录名称。

在指定目录时：

- 验证该目录是否具有执行权限。在安装目录中运行 Terraform 二进制文件需要这个权限。
- 使用空目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

b. 在提示符处，提供云的配置详情：

i. 可选：选择用于访问集群机器的 SSH 密钥。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 ssh-agent 进程使用的 SSH 密钥。

ii. Select vsphere 作为目标平台。

iii. 指定 vCenter 实例的名称。

iv. 指定创建集群所需的权限的 vCenter 帐户的用户名和密码。

安装程序连接到您的 vCenter 实例。

v. 选择要连接的 vCenter 实例中的数据中心。

注意

创建安装配置文件后，您可以修改该文件以创建多个 vSphere 数据中心环境。这意味着您可以将 OpenShift Container Platform 集群部署到在单个 VMware vCenter 中运行的多个 vSphere 数据中心。有关创建此环境的更多信息，请参阅名为 *VMware vSphere 区域和区启用*的部分。

vi. 选择要使用的默认 vCenter 数据存储。

警告

您可以指定数据存储集群中存在的任何数据存储路径。默认情况下，使用 Storage vMotion 的存储分布式资源调度程序(SDRS)会自动为数据存储集群启用。红帽不支持 Storage vMotion，因此您必须禁用 Storage DRS 以避免 OpenShift Container Platform 集群的数据丢失问题。

您不能指定多个数据存储路径。如果需要在多个数据存储间指定虚拟机，请使用 数据存储 对象在集群 install-config.yaml 配置文件中指定故障域。如需更多信息，请参阅"VMware vSphere 区域和区启用"。

vii. 选择要在其中安装 OpenShift Container Platform 集群的 vCenter 集群。安装程序使用 vSphere 集群的 root 资源池作为默认资源池。

viii. 选择包含您配置的虚拟 IP 地址和 DNS 记录的 vCenter 实例中的网络。

ix. 输入您为 control plane API 访问配置的虚拟 IP 地址。

x. 输入您为集群入口配置的虚拟 IP 地址。

xi. 输入基域。这个基域必须与您配置的 DNS 记录中使用的域相同。

xii. 为集群输入描述性名称。

您输入的集群名称必须与您在配置 DNS 记录时指定的集群名称匹配。

2. 修改 install-config.yaml 文件。您可以在"安装配置参数"部分找到有关可用参数的更多信息。

3. 备份 install-config.yaml 文件，以便您可以使用它安装多个集群。

重要

install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用此文件，必须现在备份。

其他资源

- 安装配置参数

23.2.5.4.1. 安装程序置备的 VMware vSphere 集群的 install-config.yaml 文件示例

您可以自定义 install-config.yaml 文件，以指定有关 OpenShift Container Platform 集群平台的更多详情，或修改所需参数的值。

```
apiVersion: v1
baseDomain: example.com (1)
compute: (2)
- architecture: amd64
  name:  <worker_node>
  platform: {}
  replicas: 3
controlPlane: (3)
  architecture: amd64
  name: <parent_node>
  platform: {}
```

```
  replicas: 3
metadata:
  creationTimestamp: null
  name: test 4
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 10.0.0.0/16
  networkType: OVNKubernetes 5
  serviceNetwork:
  - 172.30.0.0/16
platform:
  vsphere: 6
    apiVIPs:
      - 10.0.0.1
    failureDomains: 7
    - name: <failure_domain_name>
      region: <default_region_name>
      server: <fully_qualified_domain_name>
      topology:
        computeCluster: "/<datacenter>/host/<cluster>"
        datacenter: <datacenter>
        datastore: "/<datacenter>/datastore/<datastore>" 8
        networks:
        - <VM_Network_name>
        resourcePool: "/<datacenter>/host/<cluster>/Resources/<resourcePool>" 9
        folder: "/<datacenter_name>/vm/<folder_name>/<subfolder_name>"
        tagIDs: 10
        - <tag_id> 11
      zone: <default_zone_name>
    ingressVIPs:
    - 10.0.0.2
    vcenters:
    - datacenters:
      - <datacenter>
      password: <password>
      port: 443
      server: <fully_qualified_domain_name>
      user: administrator@vsphere.local
    diskType: thin 12
fips: false
pullSecret: '{"auths": ...}'
sshKey: 'ssh-ed25519 AAAA...'
```

1

集群的基域。所有 DNS 记录都必须是这个基域的子域，并包含集群名称。

2 3

controlPlane 部分是一个单个映射，但 compute 部分是一系列映射。为满足不同数据结构的要求，compute 部分的第一行必须以连字符 - 开头，controlPlane 部分 的第一行则不以连字符开

头。仅使用一个 control plane 池。

4

您在 DNS 记录中指定的集群名称。

6

可选：为 compute 和 control plane 机器提供额外的机器池参数配置。

7

建立地区和区域之间的关系。您可以使用 vCenter 对象（如 datastore 对象）定义故障域。故障域定义 OpenShift Container Platform 集群节点的 vCenter 位置。

8

保存虚拟机文件、模板和 ISO 镜像的 vSphere 数据存储路径。

重要

您可以指定数据存储集群中存在的任何数据存储路径。默认情况下，Storage vMotion 会自动为数据存储集群启用。红帽不支持 Storage vMotion，因此您必须禁用 Storage vMotion 以避免 OpenShift Container Platform 集群的数据丢失问题。

如果需要在多个数据存储间指定虚拟机，请使用 数据存储 对象在集群 install-config.yaml 配置文件中指定故障域。如需更多信息，请参阅"VMware vSphere 区域和区启用"。

9

可选：为创建机器提供现有资源池。如果没有指定值，安装程序将使用 vSphere 集群的 root 资源池。

10

可选：由 OpenShift Container Platform 创建的每个虚拟机都会被分配一个特定于集群的唯一标签。分配的标签可让安装程序在集群停用时识别和删除关联的虚拟机。您可以最多列出十个额外标签 ID，以附加到安装程序置备的虚拟机。

11

安装程序关联的标签的 ID。例如，urn:vmomi:InventoryServiceTag:208e713c-cae3-4b7f-918e-4051ca7d1f97:GLOBAL.有关确定标签 ID 的更多信息，请参阅 vSphere 标签和属性文档。

12

vSphere 磁盘置备方法。

5

要安装的集群网络插件。默认值 OVNKubernetes 是唯一支持的值。

注意

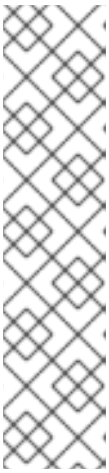
在 OpenShift Container Platform 4.12 及更新的版本中，apiVIP 和 ingressVIP 配置设置已弃用。反之，使用列表格式在 apiVIPs 和 ingressVIPs 配置设置中输入值。

23.2.5.4.2. 在安装过程中配置集群范围的代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 install-config.yaml 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 install-config.yaml 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 Proxy 对象的 spec.noProxy 字段中添加站点来绕过代理。



注意

Proxy 对象 status.noProxy 字段使用安装配置中的 networking.machineNetwork[].cidr、networking.clusterNetwork[].cidr 和 networking.serviceNetwork[] 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，Proxy 对象 status.noProxy 字段也会使用实例元数据端点填充(169.254.169.254)。

流程

1. 编辑 install-config.yaml 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 http。

2 用于创建集群外 HTTPS 连接的代理 URL。

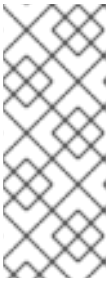
3 要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 . 以仅匹配子域。例如，.y.com 匹配 x.y.com，但不匹配 y.com。使用 * 绕过所有目的地的代理。您必须包含 vCenter 的 IP 地址以及用于其机器的 IP 范围。

4 如果提供，安装程序会在 openshift-config 命名空间中生成名为 user-ca-bundle 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 trusted-ca-bundle 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， Proxy 对象的 trustedCA 字段中也会引用此配置映射。additionalTrustBundle 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

5 可选：决定 Proxy 对象的配置以引用 trustedCA 字段中 user-ca-bundle 配置映射的策略。允许的值是 Proxyonly 和 Always。仅在配置了 http/https 代理时，使用 Proxyonly 引用 user-ca-bundle 配置映射。使用 Always 始终引用 user-ca-bundle 配置映射。默认值为 Proxyonly。

注意

安装程序不支持代理的 readinessEndpoints 字段。



注意

如果安装程序超时，重启并使用安装程序的 wait-for 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. 保存该文件并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 cluster 的集群范围代理，该代理 使用 提供的 install-config.yaml 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 cluster Proxy 对象，但它会有一个空 spec。

注意

只支持名为 cluster 的 Proxy 对象，且无法创建额外的代理。

23.2.5.4.3. 可选：使用双栈网络部署

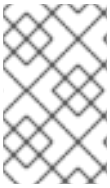
对于 OpenShift Container Platform 集群中的双栈网络，您可以为集群节点配置 IPv4 和 IPv6 地址端点。要为集群节点配置 IPv4 和 IPv6 地址端点，请编辑 install-config.yaml 文件中的 machineNetwork、clusterNetwork 和 serviceNetwork 配置设置。每个设置必须分别有两个 CIDR 条目。对于将 IPv4 系列用作主地址系列的集群，请首先指定 IPv4 设置。对于将 IPv6 系列用作主地址系列的集群，请首先指定 IPv6 设置。

```
machineNetwork:
- cidr: {{ extcidrnet }}
- cidr: {{ extcidrnet6 }}
clusterNetwork:
- cidr: 10.128.0.0/14
  hostPrefix: 23
- cidr: fd02::/48
  hostPrefix: 64
serviceNetwork:
- 172.30.0.0/16
- fd03::/112
```

要为使用 IPv4 和 IPv6 地址的应用程序提供接口，请为 Ingress VIP 和 API VIP 服务配置 IPv4 和 IPv6 虚拟 IP (VIP) 地址端点。要配置 IPv4 和 IPv6 地址端点，请编辑 install-config.yaml 文件中的

apiVIPs 和 ingressVIPs 配置设置。apiVIPs 和 ingressVIPs 配置设置使用列表格式。列表的顺序决定了每个服务的主 VIP 地址和次 VIP 地址。

```
platform:
  vsphere:
    apiVIPs:
      - <api_ipv4>
      - <api_ipv6>
    ingressVIPs:
      - <wildcard_ipv4>
      - <wildcard_ipv6>
```



注意

对于具有双栈网络配置的集群，您必须将 IPv4 和 IPv6 地址分配到同一接口。

23.2.5.4.4. 为 VMware vCenter 配置区域和区域

您可以修改默认安装配置文件，以便您可以将 OpenShift Container Platform 集群部署到在单个 VMware vCenter 中运行的多个 vSphere 数据中心。

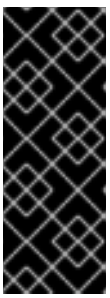
之前版本的 OpenShift Container Platform 的默认 install-config.yaml 文件配置已弃用。您可以继续使用已弃用的默认配置，但 openshift-installer 会提示您显示在配置文件中已弃用字段的警告信息。

重要

这个示例使用 govc 命令。govc 命令是 VMware 提供的开源命令；它不是红帽提供的。红帽支持团队不维护 govc 命令。有关下载和安装 govc 的说明，请参阅 VMware 文档网站

先决条件

- 您有一个现有的 install-config.yaml 安装配置文件。

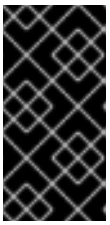


重要

您必须为 OpenShift Container Platform 集群指定一个故障域，以便您可以为 VMware vCenter 服务器置备数据中心对象。如果您需要在不同的数据中心、集群、数据存储和其他组件中置备虚拟机节点，请考虑指定多个故障域。要启用区域和区域，您必须为 OpenShift Container Platform 集群定义多个故障域。

流程

1. 输入以下 govc 命令行工具命令，以创建 openshift-region 和 openshift-zone vCenter 标签类别：



重要

如果为 openshift-region 和 openshift-zone vCenter 标签类别指定不同的名称，OpenShift Container Platform 集群的安装会失败。

```
$ govc tags.category.create -d "OpenShift region" openshift-region
```

```
$ govc tags.category.create -d "OpenShift zone" openshift-zone
```

2. 要为您要部署集群的每个区域 vSphere 数据中心创建一个 region 标签，请在终端中输入以下命令：

```
$ govc tags.create -c <region_tag_category> <region_tag>
```

3. 要为您要部署集群的每个 vSphere 集群创建一个区标签，请输入以下命令：

```
$ govc tags.create -c <zone_tag_category> <zone_tag>
```

4. 输入以下命令将区域标签附加到每个 vCenter 数据中心对象：

```
$ govc tags.attach -c <region_tag_category> <region_tag_1> /<datacenter_1>
```

5. 输入以下命令将区标签附加到每个 vCenter 数据中心对象：

```
$ govc tags.attach -c <zone_tag_category> <zone_tag_1> /<datacenter_1>/host/vcs-mdcnc-workload-1
```

6. 进入包含安装程序的目录，并根据您选择的安装要求初始化集群部署。

在 vSphere 数据中心中定义的多个数据中心的 install-config.yaml 文件示例

```
---
compute:
---
  vsphere:
     zones:
       - "<machine_pool_zone_1>"
       - "<machine_pool_zone_2>"
---
controlPlane:
---
vsphere:
     zones:
       - "<machine_pool_zone_1>"
       - "<machine_pool_zone_2>"
---
platform:
  vsphere:
    vcenters:
---
    datacenters:
      - <datacenter1_name>
      - <datacenter2_name>
    failureDomains:
    - name: <machine_pool_zone_1>
      region: <region_tag_1>
      zone: <zone_tag_1>
      server: <fully_qualified_domain_name>
      topology:
        datacenter: <datacenter1>
        computeCluster: "/<datacenter1>/host/<cluster1>"
        networks:
        - <VM_Network1_name>
        datastore: "/<datacenter1>/datastore/<datastore1>"
        resourcePool: "/<datacenter1>/host/<cluster1>/Resources/<resourcePool1>"
        folder: "/<datacenter1>/vm/<folder1>"
    - name: <machine_pool_zone_2>
      region: <region_tag_2>
      zone: <zone_tag_2>
      server: <fully_qualified_domain_name>
      topology:
        datacenter: <datacenter2>
        computeCluster: "/<datacenter2>/host/<cluster2>"
        networks:
        - <VM_Network2_name>
        datastore: "/<datacenter2>/datastore/<datastore2>"
        resourcePool: "/<datacenter2>/host/<cluster2>/Resources/<resourcePool2>"
        folder: "/<datacenter2>/vm/<folder2>"
---
```

23.2.5.5. 网络配置阶段

OpenShift Container Platform 安装前有两个阶段，您可以在其中自定义网络配置。

第 1 阶段

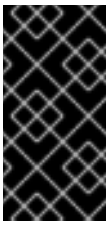
在创建清单文件前，您可以自定义 install-config.yaml 文件中的以下与网络相关的字段：

- networking.networkType
- networking.clusterNetwork
- networking.serviceNetwork
- networking.machineNetwork

有关这些字段的更多信息，请参阅 *安装配置参数*。

注意

将 networking.machineNetwork 设置为与首选 NIC 所在的 CIDR 匹配。



重要

CIDR 范围 172.17.0.0/16 由 libVirt 保留。对于集群中的任何网络，您无法使用此范围或与这个范围重叠的范围。

第 2 阶段

运行 openshift-install create 清单创建 清单文件后，您可以只使用您要修改的字段定义自定义 Cluster Network Operator 清单。您可以使用 清单指定高级网络配置。

您不能覆盖在 stage 2 阶段 1 中在 install-config.yaml 文件中指定的值。但是，您可以在第 2 阶段进一步自定义网络插件。

23.2.5.6. 指定高级网络配置

您可以使用网络插件的高级网络配置将集群集成到现有网络环境中。您只能在安装集群前指定高级网络配置。

重要

不支持通过修改安装程序创建的 OpenShift Container Platform 清单文件来自定义网络配置。支持应用您创建的清单文件，如以下流程中所示。

先决条件

- 您已创建 install-config.yaml 文件并完成对其所做的任何修改。

流程

1. 进入包含安装程序的目录并创建清单：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 <installation_directory> 指定包含集群的 install-config.yaml 文件的目录名称。

2. 在 <installation_directory>/manifests/ 目录中 为高级网络配置创建一个名为 cluster-network-03-config.yml 的 stub 清单文件：

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
```

3. 在 cluster-network-03-config.yml 文件中指定集群的高级网络配置，如下例所示：

为 OVN-Kubernetes 网络供应商启用 IPsec

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
```

```
  name: cluster
spec:
  defaultNetwork:
    ovnKubernetesConfig:
      ipsecConfig:
        mode: Full
```

4. 可选：备份 manifests/cluster-network-03-config.yml 文件。创建 Ignition 配置文件时，安装程序会使用 manifests/ 目录。

5. 删除定义 control plane 机器的 Kubernetes 清单文件和计算 machineSets：

```
$ rm -f openshift/99_openshift-cluster-api_master-machines-*.yaml
openshift/99_openshift-cluster-api_worker-machineset-*.yaml
```

由于您要自行创建和管理这些资源，因此不必初始化这些资源。

- 您可以使用机器 API 保留 MachineSet 文件以创建计算机器，但您必须更新对它们的引用以匹配您的环境。

23.2.5.7. Cluster Network Operator 配置

集群网络的配置作为 Cluster Network Operator(CNO)配置的一部分指定，并存储在名为 cluster 的自定义资源(CR)对象中。CR 指定 operator.openshift.io API 组中的 Network API 的字段。

CNO 配置在集群安装过程中从 Network.config.openshift.io API 组中的 Network API 继承以下字段：

clusterNetwork

从中分配 Pod IP 地址的 IP 地址池。

serviceNetwork

服务的 IP 地址池.

defaultNetwork.type

集群网络插件。OVNKubernetes 是安装期间唯一支持的插件。

您可以通过在名为 cluster 的 CNO 对象中设置 defaultNetwork 对象的字段来为集群指定集群网络插件配置。

23.2.5.7.1. Cluster Network Operator 配置对象

下表中描述了 Cluster Network Operator(CNO)的字段：

表 23.7. Cluster Network Operator 配置对象

字段	类型	描述
metadata.name	字符串	CNO 对象的名称。这个名称始终是 集群。
spec.clusterNetwork	array	用于指定从哪些 IP 地址块分配 Pod IP 地址以及集群中每个节点的子网前缀长度的列表。例如： spec: clusterNetwork: - cidr: 10.128.0.0/19 hostPrefix: 23 - cidr: 10.128.32.0/19 hostPrefix: 23
spec.serviceNetwork	array	服务的 IP 地址块。OpenShift SDN 和 OVN-Kubernetes 网络插件只支持服务网络的一个 IP 地址块。例如： spec: serviceNetwork: - 172.30.0.0/14 您只能在创建清单前在 install-config.yaml 文件中自定义此字段。该值在清单文件中是只读的。
spec.defaultNetwork	object	为集群网络配置网络插件。
spec.kubeProxyConfig	object	此对象的字段指定 kube-proxy 配置。如果使用 OVN-Kubernetes 集群网络供应商，则 kube-proxy 配置不会起作用。

defaultNetwork 对象配置

下表列出了 defaultNetwork 对象的值：

表 23.8. defaultNetwork 对象

字段	类型	描述
type	**字符串**	**OVNKubernetes**。Red Hat OpenShift Networking 网络插件在安装过程中被选择。此值在集群安装后无法更改。  **注意** OpenShift Container Platform 默认使用 OVN-Kubernetes 网络插件。OpenShift SDN 不再作为新集群的安装选择提供。
ovnKubernetesConfig	**object**	此对象仅对 OVN-Kubernetes 网络插件有效。

配置 OVN-Kubernetes 网络插件

下表描述了 OVN-Kubernetes 网络插件的配置字段：

表 23.9. ovnKubernetesConfig object

字段	类型	描述
mtu	**integer**	Geneve（通用网络虚拟化封装）覆盖网络的最大传输单元 (MTU)。这根据主网络接口的 MTU 自动探测。您通常不需要覆盖检测到的 MTU。 如果自动探测的值不是您期望的值，请确认节点上主网络接口上的 MTU 是否正确。您不能使用这个选项更改节点上主网络接口的 MTU 值。 如果集群中不同节点需要不同的 MTU 值，则必须将此值设置为 **比** 集群中的最低 MTU 值小 100。例如，如果集群中的某些节点的 MTU 为 **9001**，而某些节点的 MTU 为 **1500**，则必须将此值设置为 **1400**。
genevePort	**integer**	用于所有 Geneve 数据包的端口。默认值为 **6081**。此值在集群安装后无法更改。
ipsecConfig	**object**	指定用于自定义 IPsec 配置的配置对象。
ipv4	**object**	为 IPv4 设置指定配置对象。
ipv6	**object**	为 IPv6 设置指定配置对象。
policyAuditConfig	**object**	指定用于自定义网络策略审计日志的配置对象。如果未设置，则使用默认的审计日志设置。

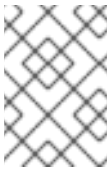
字段	类型	描述
gatewayConfig	**object**	可选：指定一个配置对象来自定义如何将出口流量发送到节点网关。  **注意** 在迁移出口流量时，工作负载和服务流量会受到一定影响，直到 Cluster Network Operator (CNO) 成功推出更改。

表 23.10. ovnKubernetesConfig.ipv4 object

字段	类型	描述
internalTransitSwitchSubnet	字符串	如果您的现有网络基础架构与 **100.88.0.0/16** IPv4 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。启用东西流量的分布式传输交换机的子网。此子网不能与 OVN-Kubernetes 或主机本身使用的任何其他子网重叠。必须足够大，以适应集群中的每个节点一个 IP 地址。 默认值为 **100.88.0.0/16**。
internalJoinSubnet	字符串	如果您的现有网络基础架构与 **100.64.0.0/16** IPv4 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。您必须确保 IP 地址范围没有与 OpenShift Container Platform 安装使用的任何其他子网重叠。IP 地址范围必须大于可添加到集群的最大节点数。例如，如果 **clusterNetwork.cidr** 值为 **10.128.0.0/14**，并且 **clusterNetwork.hostPrefix** 值为 **/23**，则最大节点数量为 **2^(23-14)=512**。 默认值为 **100.64.0.0/16**。

表 23.11. ovnKubernetesConfig.ipv6 object

字段	类型	描述
internalTransitSwitchSubnet	字符串	如果您的现有网络基础架构与 **fd97::/64** IPv6 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。启用东西流量的分布式传输交换机的子网。此子网不能与 OVN-Kubernetes 或主机本身使用的任何其他子网重叠。必须足够大，以适应集群中的每个节点一个 IP 地址。 默认值为 **fd97::/64**。
internalJoinSubnet	字符串	如果您的现有网络基础架构与 **fd98::/64** IPv6 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。您必须确保 IP 地址范围没有与 OpenShift Container Platform 安装使用的任何其他子网重叠。IP 地址范围必须大于可添加到集群的最大节点数。 默认值为 **fd98::/64**。

表 23.12. policyAuditConfig object

字段	类型	描述
rateLimit	整数	每个节点每秒生成一次的消息数量上限。默认值为每秒 **20** 条消息。
maxFileSize	整数	审计日志的最大大小，以字节为单位。默认值为 **50000000** 或 50 MB。
maxLogFiles	整数	保留的日志文件的最大数量。
目的地	字符串	以下附加审计日志目标之一： **libc** 主机上的 journald 进程的 libc **syslog（）** 函数。 **UDP:<host>:<port>** 一个 syslog 服务器。将 **<host>:<port>** 替换为 **syslog 服务器的主机** 和端口。 **Unix:<file>** 由 **<file>** 指定的 Unix 域套接字文件。 **null** 不要将审计日志发送到任何其他目标。
syslogFacility	字符串	syslog 工具，如 as **kern**，如 RFC5424 定义。默认值为 **local0。**

表 23.13. gatewayConfig object

字段	类型	描述
routingViaHost	**布尔值**	将此字段设置为 **true**，将来自 pod 的出口流量发送到主机网络堆栈。对于依赖于在内核路由表中手动配置路由的高级别安装和应用程序，您可能需要将出口流量路由到主机网络堆栈。默认情况下，出口流量在 OVN 中进行处理以退出集群，不受内核路由表中的特殊路由的影响。默认值为 **false**。 此字段与 Open vSwitch 硬件卸载功能有交互。如果将此字段设置为 **true**，则不会获得卸载的性能优势，因为主机网络堆栈会处理出口流量。
ipForwarding	**object**	您可以使用 **Network** 资源中的 **ipForwarding** 规格来控制 OVN-Kubernetes 管理接口上所有流量的 IP 转发。指定 **Restricted** 只允许 Kubernetes 相关流量的 IP 转发。指定 **Global** 以允许转发所有 IP 流量。对于新安装，默认值为 **Restricted**。对于到 OpenShift Container Platform 4.14 或更高版本的更新，默认值为 **Global**。
ipv4	**object**	可选：指定一个对象来为主机配置内部 OVN-Kubernetes 伪装地址，以服务 IPv4 地址的流量。

字段	类型	描述
ipv6	**object**	可选：指定一个对象来为主机配置内部 OVN-Kubernetes 伪装地址，以服务 IPv6 地址的流量。

表 23.14. gatewayConfig.ipv4 对象

字段	类型	描述
internalMasqueradeSubnet	**字符串**	内部使用的伪装 IPv4 地址，以启用主机服务流量。主机配置了这些 IP 地址和共享网关网桥接口。默认值为 **169.254.169.0/29**。

表 23.15. gatewayConfig.ipv6 对象

字段	类型	描述
internalMasqueradeSubnet	**字符串**	内部使用的伪装 IPv6 地址，以启用主机服务流量。主机配置了这些 IP 地址和共享网关网桥接口。默认值为 **fd69::/125**。

表 23.16. ipsecConfig 对象

字段	类型	描述
模式	**字符串**	指定 IPsec 实现的行为。必须是以下值之一： • **Disabled**: 在集群节点上不启用 IPsec。 • **External** ：对于带有外部主机的网络流量，启用 IPsec。 • **Full**: IPsec 为带有外部主机的 pod 流量和网络流量启用 IPsec。

启用 IPSec 的 OVN-Kubernetes 配置示例

```
defaultNetwork:
  type: OVNKubernetes
  ovnKubernetesConfig:
    mtu: 1400
    genevePort: 6081
      ipsecConfig:
        mode: Full
```

重要

使用 OVNKubernetes 可能会导致 IBM Power® 上的堆栈耗尽问题。

kubeProxyConfig 对象配置（仅限 OpenShiftSDN 容器网络接口）

kubeProxyConfig 对象的值在下表中定义：

表 23.17. kubeProxyConfig object

字段	类型	描述
iptablesSyncPeriod	**字符串**	**iptables** 规则的刷新周期。默认值为 **30s**。有效的后缀包括 **s**、**m** 和 **h**，具体参见 Go 时间 包 文档。  **注意** 由于 OpenShift Container Platform 4.3 及更高版本中引进了性能改进，不再需要调整 **iptablesSyncPeriod** 参数。
proxyArguments.iptables-min-sync-period	**array**	刷新 **iptables** 规则前的最短持续时间。此字段确保刷新的频率不会过于频繁。有效的后缀包括 **s**、**m** 和 **h**，具体参见 Go **time** 软件包。默认值为： kubeProxyConfig: proxyArguments: iptables-min-sync-period: - 0s

23.2.5.8. 用户管理的负载均衡器的服务

您可以将 OpenShift Container Platform 集群配置为使用用户管理的负载均衡器来代替默认负载均衡器。

重要

配置用户管理的负载均衡器取决于您的厂商的负载均衡器。

本节中的信息和示例仅用于指导目的。有关供应商负载均衡器的更多信息，请参阅供应商文档。

红帽支持用户管理的负载均衡器的以下服务：

- Ingress Controller
- OpenShift API
- OpenShift MachineConfig API

您可以选择是否要为用户管理的负载均衡器配置一个或多个所有服务。仅配置 Ingress Controller 服务是一个通用的配置选项。要更好地了解每个服务，请查看以下图表：

图 23.4. 显示 OpenShift Container Platform 环境中运行的 Ingress Controller 的网络工作流示例

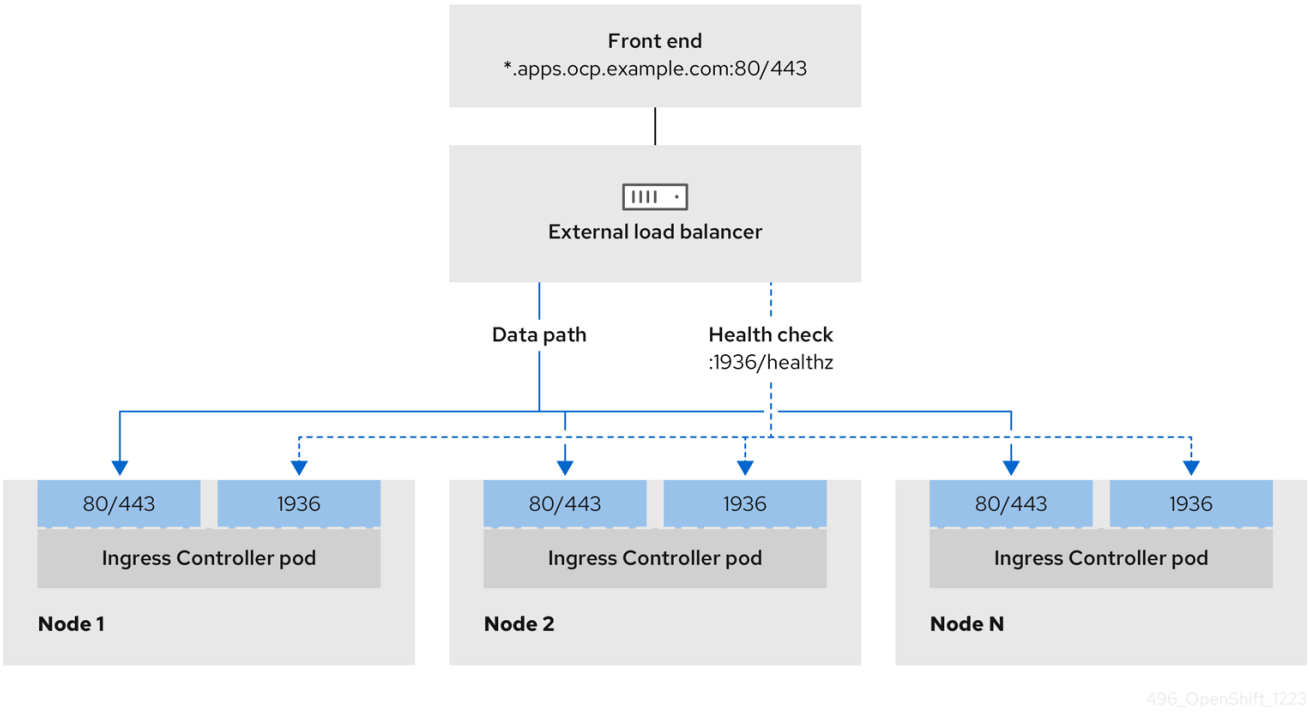


图 23.5. 显示 OpenShift Container Platform 环境中运行的 OpenShift API 的网络工作流示例

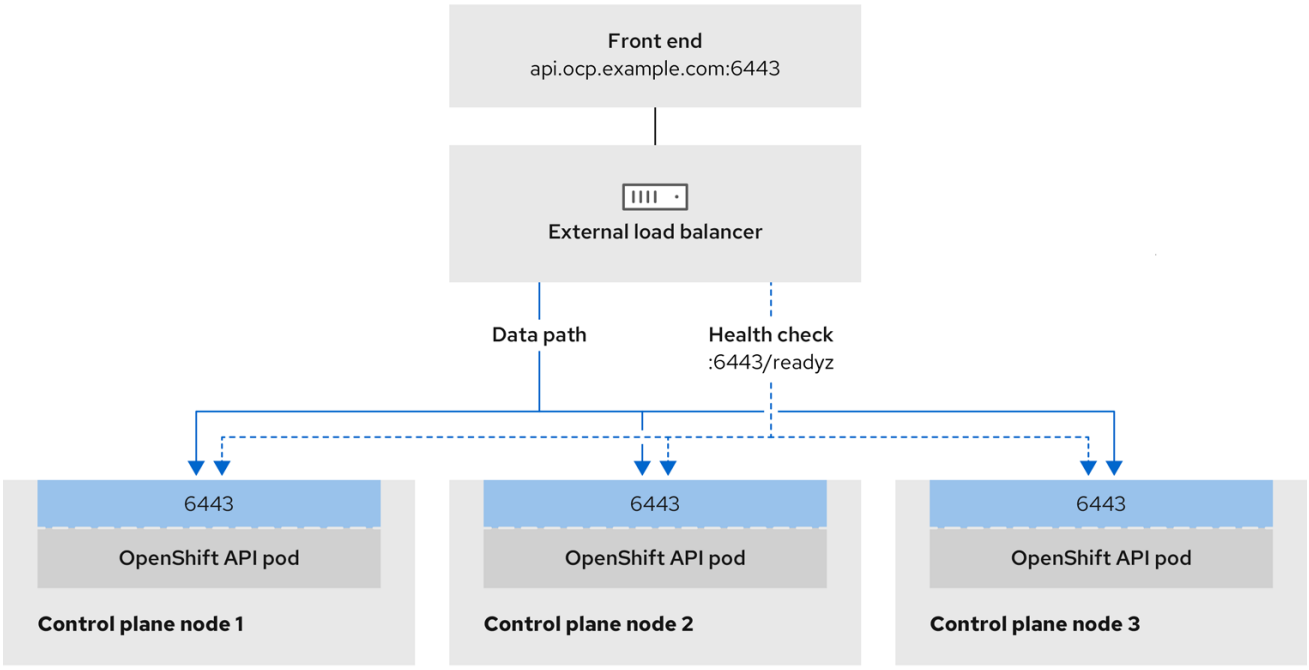
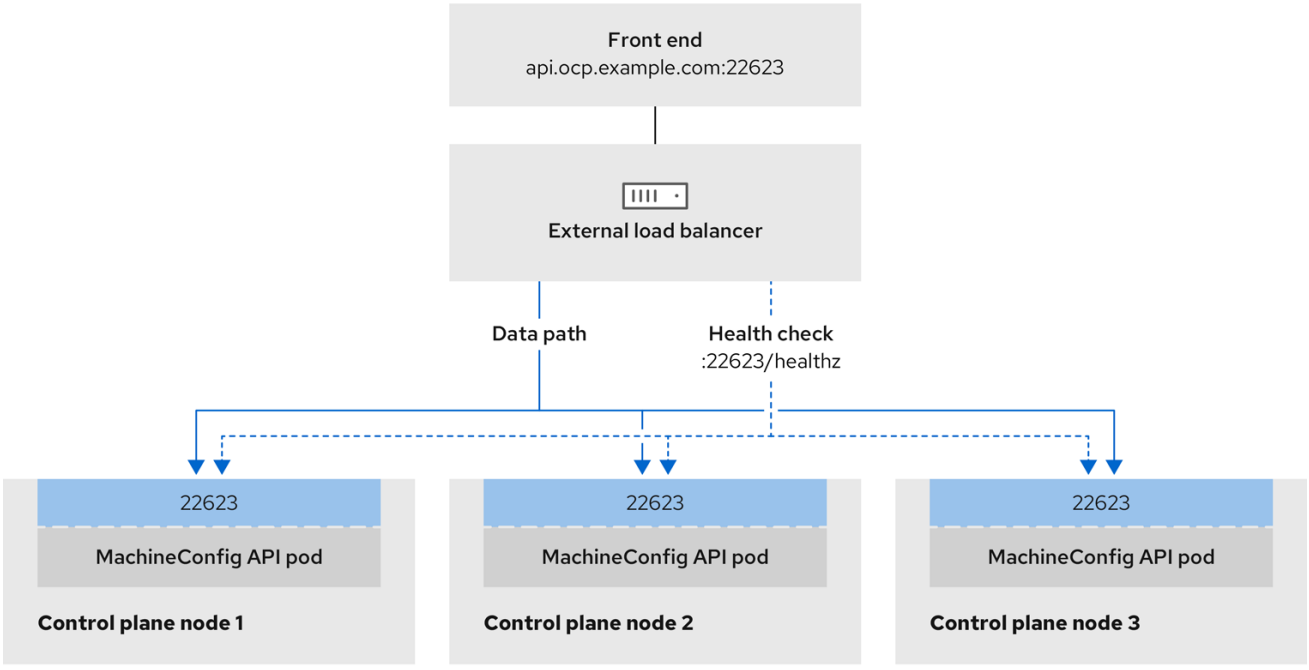


图 23.6. 显示 OpenShift Container Platform 环境中运行的 OpenShift MachineConfig API 的网络工作流示例



用户管理的负载均衡器支持以下配置选项：

- 使用节点选择器将 Ingress Controller 映射到一组特定的节点。您必须为这个集合中的每个节点分配一个静态 IP 地址，或者将每个节点配置为从动态主机配置协议(DHCP)接收相同的 IP 地

址。基础架构节点通常接收这种类型的配置。

- 以子网上的所有 IP 地址为目标。此配置可减少维护开销，因为您可以在这些网络中创建和销毁节点，而无需重新配置负载均衡器目标。如果您使用较小的网络上的机器集来部署入口 pod，如 /27 或 /28，您可以简化负载均衡器目标。

  提示

  您可以通过检查机器配置池的资源来列出网络中存在的所有 IP 地址。

在为 OpenShift Container Platform 集群配置用户管理的负载均衡器前，请考虑以下信息：

- 对于前端 IP 地址，您可以对前端 IP 地址、Ingress Controller 的负载均衡器和 API 负载均衡器使用相同的 IP 地址。查看厂商的文档以获取此功能的相关信息。

- 对于后端 IP 地址，请确保 OpenShift Container Platform control plane 节点的 IP 地址在用户管理的负载均衡器生命周期内不会改变。您可以通过完成以下操作之一来实现此目的：

  - 为每个 control plane 节点分配一个静态 IP 地址。

  - 将每个节点配置为在每次节点请求 DHCP 租期时从 DHCP 接收相同的 IP 地址。根据供应商，DHCP 租期可能采用 IP 保留或静态 DHCP 分配的形式。

- 在 Ingress Controller 后端服务的用户管理的负载均衡器中手动定义运行 Ingress Controller 的每个节点。例如，如果 Ingress Controller 移到未定义节点，则可能会出现连接中断。

23.2.5.8.1. 配置用户管理的负载均衡器

您可以将 OpenShift Container Platform 集群配置为使用用户管理的负载均衡器来代替默认负载均衡器。

重要

在配置用户管理的负载均衡器前，请确保阅读用户管理的负载均衡器部分。

阅读适用于您要为用户管理的负载均衡器配置的服务的以下先决条件。

注意

MetalLB，在集群中运行，充当用户管理的负载均衡器。

OpenShift API 的先决条件

- 您定义了前端 IP 地址。
- TCP 端口 6443 和 22623 在负载均衡器的前端 IP 地址上公开。检查以下项：
  - 端口 6443 提供对 OpenShift API 服务的访问。
  - 端口 22623 可以为节点提供 ignition 启动配置。
- 前端 IP 地址和端口 6443 可以被您的系统的所有用户访问，其位置为 OpenShift Container Platform 集群外部。
- 前端 IP 地址和端口 22623 只能被 OpenShift Container Platform 节点访问。
- 负载均衡器后端可以在端口 6443 和 22623 上与 OpenShift Container Platform control plane 节点通信。

Ingress Controller 的先决条件

- 您定义了前端 IP 地址。

- TCP 端口 443 和 80 在负载均衡器的前端 IP 地址上公开。
- 前端 IP 地址、端口 80 和端口 443 可以被您的系统所有用户访问，以及 OpenShift Container Platform 集群外部的位置。
- 前端 IP 地址、端口 80 和端口 443 可被 OpenShift Container Platform 集群中运行的所有节点访问。
- 负载均衡器后端可以在端口 80、443 和 1936 上与运行 Ingress Controller 的 OpenShift Container Platform 节点通信。

健康检查 URL 规格的先决条件

您可以通过设置健康检查 URL 来配置大多数负载均衡器，以确定服务是否可用或不可用。OpenShift Container Platform 为 OpenShift API、Machine Configuration API 和 Ingress Controller 后端服务提供这些健康检查。

以下示例显示了之前列出的后端服务的健康检查规格：

Kubernetes API 健康检查规格示例

```
Path: HTTPS:6443/readyz
Healthy threshold: 2
Unhealthy threshold: 2
Timeout: 10
Interval: 10
```

Machine Config API 健康检查规格示例

```
Path: HTTPS:22623/healthz
Healthy threshold: 2
Unhealthy threshold: 2
Timeout: 10
Interval: 10
```

Ingress Controller 健康检查规格示例

```
Path: HTTP:1936/healthz/ready
Healthy threshold: 2
Unhealthy threshold: 2
Timeout: 5
Interval: 10
```

流程

1. 配置 HAProxy Ingress Controller，以便您可以在端口 6443、22623、443 和 80 上从负载均衡器访问集群。根据您的需要，您可以在 HAProxy 配置中指定来自多个子网的单个子网或 IP 地址的 IP 地址。

   带有列出子网的 HAProxy 配置示例

```
# ...
listen my-cluster-api-6443
    bind 192.168.1.100:6443
    mode tcp
    balance roundrobin
  option httpchk
  http-check connect
  http-check send meth GET uri /readyz
  http-check expect status 200
    server my-cluster-master-2 192.168.1.101:6443 check inter 10s rise 2 fall 2
    server my-cluster-master-0 192.168.1.102:6443 check inter 10s rise 2 fall 2
    server my-cluster-master-1 192.168.1.103:6443 check inter 10s rise 2 fall 2

listen my-cluster-machine-config-api-22623
    bind 192.168.1.100:22623
    mode tcp
    balance roundrobin
  option httpchk
  http-check connect
  http-check send meth GET uri /healthz
  http-check expect status 200
    server my-cluster-master-2 192.168.1.101:22623 check inter 10s rise 2 fall 2
    server my-cluster-master-0 192.168.1.102:22623 check inter 10s rise 2 fall 2
    server my-cluster-master-1 192.168.1.103:22623 check inter 10s rise 2 fall 2
```

```
listen my-cluster-apps-443
    bind 192.168.1.100:443
    mode tcp
    balance roundrobin
   option httpchk
   http-check connect
   http-check send meth GET uri /healthz/ready
   http-check expect status 200
     server my-cluster-worker-0 192.168.1.111:443 check port 1936 inter 10s rise 2 fall 2
     server my-cluster-worker-1 192.168.1.112:443 check port 1936 inter 10s rise 2 fall 2
     server my-cluster-worker-2 192.168.1.113:443 check port 1936 inter 10s rise 2 fall 2

listen my-cluster-apps-80
    bind 192.168.1.100:80
    mode tcp
    balance roundrobin
   option httpchk
   http-check connect
   http-check send meth GET uri /healthz/ready
   http-check expect status 200
     server my-cluster-worker-0 192.168.1.111:80 check port 1936 inter 10s rise 2 fall 2
     server my-cluster-worker-1 192.168.1.112:80 check port 1936 inter 10s rise 2 fall 2
     server my-cluster-worker-2 192.168.1.113:80 check port 1936 inter 10s rise 2 fall 2
# ...
```

带有多个列出子网的 HAProxy 配置示例

```
# ...
listen api-server-6443
    bind *:6443
    mode tcp
      server master-00 192.168.83.89:6443 check inter 1s
      server master-01 192.168.84.90:6443 check inter 1s
      server master-02 192.168.85.99:6443 check inter 1s
      server bootstrap 192.168.80.89:6443 check inter 1s

listen machine-config-server-22623
    bind *:22623
    mode tcp
      server master-00 192.168.83.89:22623 check inter 1s
      server master-01 192.168.84.90:22623 check inter 1s
      server master-02 192.168.85.99:22623 check inter 1s
      server bootstrap 192.168.80.89:22623 check inter 1s

listen ingress-router-80
    bind *:80
    mode tcp
    balance source
```

```
    server worker-00 192.168.83.100:80 check inter 1s
    server worker-01 192.168.83.101:80 check inter 1s

listen ingress-router-443
  bind *:443
  mode tcp
  balance source
    server worker-00 192.168.83.100:443 check inter 1s
    server worker-01 192.168.83.101:443 check inter 1s

listen ironic-api-6385
  bind *:6385
  mode tcp
  balance source
    server master-00 192.168.83.89:6385 check inter 1s
    server master-01 192.168.84.90:6385 check inter 1s
    server master-02 192.168.85.99:6385 check inter 1s
    server bootstrap 192.168.80.89:6385 check inter 1s

listen inspector-api-5050
  bind *:5050
  mode tcp
  balance source
    server master-00 192.168.83.89:5050 check inter 1s
    server master-01 192.168.84.90:5050 check inter 1s
    server master-02 192.168.85.99:5050 check inter 1s
    server bootstrap 192.168.80.89:5050 check inter 1s
# ...
```

2. 使用 curl CLI 命令验证用户管理的负载均衡器及其资源是否正常运行：

   a. 运行以下命令并查看响应，验证集群机器配置 API 是否可以被 Kubernetes API 服务器资源访问：

```
$ curl https://<loadbalancer_ip_address>:6443/version --insecure
```

   如果配置正确，您会收到 JSON 对象的响应：

```
{
  "major": "1",
  "minor": "11+",
  "gitVersion": "v1.11.0+ad103ed",
  "gitCommit": "ad103ed",
  "gitTreeState": "clean",
  "buildDate": "2019-01-09T06:44:10Z",
  "goVersion": "go1.10.3",
```

```
  "compiler": "gc",
  "platform": "linux/amd64"
}
```

b. 运行以下命令并观察输出，验证集群机器配置 API 是否可以被 Machine 配置服务器资源访问：

```
$ curl -v https://<loadbalancer_ip_address>:22623/healthz --insecure
```

如果配置正确，命令的输出会显示以下响应：

```
HTTP/1.1 200 OK
Content-Length: 0
```

c. 运行以下命令并观察输出，验证控制器是否可以被端口 80 上的 Ingress Controller 资源访问：

```
$ curl -I -L -H "Host: console-openshift-console.apps.<cluster_name>.
<base_domain>" http://<load_balancer_front_end_IP_address>
```

如果配置正确，命令的输出会显示以下响应：

```
HTTP/1.1 302 Found
content-length: 0
location: https://console-openshift-console.apps.ocp4.private.opequon.net/
cache-control: no-cache
```

d. 运行以下命令并观察输出，验证控制器是否可以被端口 443 上的 Ingress Controller 资源访问：

```
$ curl -I -L --insecure --resolve console-openshift-console.apps.<cluster_name>.
<base_domain>:443:<Load Balancer Front End IP Address> https://console-
openshift-console.apps.<cluster_name>.<base_domain>
```

如果配置正确，命令的输出会显示以下响应：

```
HTTP/1.1 200 OK
referrer-policy: strict-origin-when-cross-origin
set-cookie: csrf-
token=UlYWOyQ62LWjw2h003xtYSKlh1a0Py2hhctw0WmV2YEdhJjFyQwWcGBsja2
61dGLgaYO0nxzVErhiXt6QepA7g==; Path=/; Secure; SameSite=Lax
x-content-type-options: nosniff
```

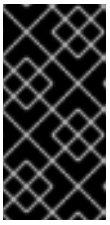
```
x-dns-prefetch-control: off
x-frame-options: DENY
x-xss-protection: 1; mode=block
date: Wed, 04 Oct 2023 16:29:38 GMT
content-type: text/html; charset=utf-8
set-cookie:
1e2670d92730b515ce3a1bb65da45062=1bf5e9573c9a2760c964ed1659cc1673;
path=/; HttpOnly; Secure; SameSite=None
cache-control: private
```

3. 配置集群的 DNS 记录，使其以用户管理的负载均衡器的前端 IP 地址为目标。您必须在负载均衡器上将记录更新为集群 API 和应用程序的 DNS 服务器。

修改 DNS 记录示例

```
<load_balancer_ip_address>  A  api.<cluster_name>.<base_domain>
A record pointing to Load Balancer Front End
```

```
<load_balancer_ip_address>  A apps.<cluster_name>.<base_domain>
A record pointing to Load Balancer Front End
```



重要

DNS 传播可能需要一些时间才能获得每个 DNS 记录。在验证每个记录前，请确保每个 DNS 记录传播。

4. 要使 OpenShift Container Platform 集群使用用户管理的负载均衡器，您必须在集群的 install-config.yaml 文件中指定以下配置：

```
# ...
platform:
  vsphere:
    loadBalancer:
      type: UserManaged 1
      apiVIPs:
      - <api_ip> 2
      ingressVIPs:
      - <ingress_ip> 3
# ...
```

1

为 type 参数设置 UserManaged，为集群指定用户管理的负载均衡器。参数默认为 OpenShiftManagedDefault，它表示默认的内部负载均衡器。对于 openshift-kni-infra 命名空间中定义的服务，用户管理的负载均衡器可将 coredns 服务部署到集群中的 pod，但忽略 keepalived 和 haproxy 服务。

2

指定用户管理的负载均衡器时所需的参数。指定用户管理的负载均衡器的公共 IP 地址，以便 Kubernetes API 可以与用户管理的负载均衡器通信。

3

指定用户管理的负载均衡器时所需的参数。指定用户管理的负载均衡器的公共 IP 地址，以便用户管理的负载均衡器可以管理集群的入口流量。

验证

1. 使用 curl CLI 命令验证用户管理的负载均衡器和 DNS 记录配置是否正常工作：

   a. 运行以下命令并查看输出，验证您可以访问集群 API：

   ```
   $ curl https://api.<cluster_name>.<base_domain>:6443/version --insecure
   ```

   如果配置正确，您会收到 JSON 对象的响应：

   ```
   {
    "major": "1",
    "minor": "11+",
    "gitVersion": "v1.11.0+ad103ed",
    "gitCommit": "ad103ed",
    "gitTreeState": "clean",
    "buildDate": "2019-01-09T06:44:10Z",
    "goVersion": "go1.10.3",
    "compiler": "gc",
    "platform": "linux/amd64"
    }
   ```

   b. 运行以下命令并查看输出，验证您可以访问集群机器配置：

   ```
   $ curl -v https://api.<cluster_name>.<base_domain>:22623/healthz --insecure
   ```

如果配置正确，命令的输出会显示以下响应：

```
HTTP/1.1 200 OK
Content-Length: 0
```

c. 运行以下命令并查看输出，验证您可以在端口上访问每个集群应用程序：

```
$ curl http://console-openshift-console.apps.<cluster_name>.<base_domain> -I -L
--insecure
```

如果配置正确，命令的输出会显示以下响应：

```
HTTP/1.1 302 Found
content-length: 0
location: https://console-openshift-console.apps.<cluster-name>.<base domain>/
cache-control: no-cacheHTTP/1.1 200 OK
referrer-policy: strict-origin-when-cross-origin
set-cookie: csrf-
token=39HoZgztDnzjJkq/JuLJMeoKNXlfiVv2YgZc09c3TBOBU4Nl6kDXaJH1LdicNh
N1UsQWzon4Dor9GWGfopaTEQ==; Path=/; Secure
x-content-type-options: nosniff
x-dns-prefetch-control: off
x-frame-options: DENY
x-xss-protection: 1; mode=block
date: Tue, 17 Nov 2020 08:42:10 GMT
content-type: text/html; charset=utf-8
set-cookie:
1e2670d92730b515ce3a1bb65da45062=9b714eb87e93cf34853e87a92d6894be;
path=/; HttpOnly; Secure; SameSite=None
cache-control: private
```

d. 运行以下命令并查看输出，验证您可以在端口 443 上访问每个集群应用程序：

```
$ curl https://console-openshift-console.apps.<cluster_name>.<base_domain> -I -
L --insecure
```

如果配置正确，命令的输出会显示以下响应：

```
HTTP/1.1 200 OK
referrer-policy: strict-origin-when-cross-origin
set-cookie: csrf-
token=UlYWOyQ62LWjw2h003xtYSKlh1a0Py2hhctw0WmV2YEdhJjFyQwWcGBsja2
61dGLgaYO0nxzVErhiXt6QepA7g==; Path=/; Secure; SameSite=Lax
x-content-type-options: nosniff
```

```
x-dns-prefetch-control: off
x-frame-options: DENY
x-xss-protection: 1; mode=block
date: Wed, 04 Oct 2023 16:29:38 GMT
content-type: text/html; charset=utf-8
set-cookie:
1e2670d92730b515ce3a1bb65da45062=1bf5e9573c9a2760c964ed1659cc1673;
path=/; HttpOnly; Secure; SameSite=None
cache-control: private
```

23.2.5.9. 部署集群

您可以在兼容云平台上安装 OpenShift Container Platform。

重要

在初始安装过程中，您只能运行安装程序的 create cluster 命令一次。

先决条件

- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。
- 已确认主机上的云供应商帐户具有部署集群的正确权限。权限不正确的帐户会导致安装过程失败，并显示包括缺失权限的错误消息。
- 可选：在创建集群时，配置外部负载均衡器来代替默认负载均衡器。

重要

您不需要为安装程序指定 API 和 Ingress 静态地址。如果选择此配置，则必须采取额外的操作来定义接受每个引用的 vSphere 子网的 IP 地址的网络目标。请参阅"配置用户管理的负载均衡器"部分。

流程

- 进入包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
    --log-level=info 2
```

1 对于 <installation_directory>，请指定自定义 ./install-config.yaml 文件的位置。

2 要查看不同的安装详情，请指定 warn、debug 或 error，而不是 info。

验证

当集群部署成功完成时：

- 终端会显示用于访问集群的说明，包括指向 Web 控制台和 kubeadmin 用户的凭证的链接。
- 凭证信息还会输出到 <installation_directory>/.openshift_install.log.

重要

不要删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 node-bootstrapper 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。

- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

23.2.5.10. 使用 CLI 登录集群

您可以通过导出集群 kubeconfig 文件，以默认系统用户身份登录集群。kubeconfig 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。
- 已安装 oc CLI。

流程

1. 导出 kubeadmin 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 <installation_directory>，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 oc 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

23.2.5.11. 创建 registry 存储

安装集群后，必须为 registry Operator 创建存储。

23.2.5.11.1. 安装过程中删除的镜像 registry

在不提供可共享对象存储的平台上，OpenShift Image Registry Operator bootstraps 本身为 Removed。这允许 openshift-installer 在这些平台类型上完成安装。

安装后，您必须编辑 Image Registry Operator 配置，将 managementState 从 Removed 切换到 Managed。完成此操作后，您必须配置存储。

23.2.5.11.2. 镜像 registry 存储配置

对于不提供默认存储的平台，Image Registry Operator 最初不可用。安装后，您必须将 registry 配置为使用存储，以便 Registry Operator 可用。

显示配置生产集群所需的持久性卷的说明。如果适用，显示有关将空目录配置为存储位置的说明，这仅适用于非生产集群。

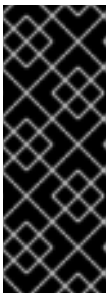
提供了在升级过程中使用 Recreate rollout 策略来允许镜像 registry 使用块存储类型的说明。

23.2.5.11.2.1. 为 VMware vSphere 配置 registry 存储

作为集群管理员，在安装后需要配置 registry 来使用存储。

先决条件

- 集群管理员权限。
- VMware vSphere 上有一个集群。
- 为集群置备的持久性存储，如 Red Hat OpenShift Data Foundation。



重要

当您只有一个副本时，OpenShift Container Platform 支持对镜像 registry 存储的 ReadWriteOnce 访问。ReadWriteOnce 访问还要求 registry 使用 Recreate rollout 策略。要部署支持高可用性的镜像 registry，需要两个或多个副本，ReadWriteMany 访问。

- 必须具有"100Gi"容量.

重要

测试显示在 RHEL 中使用 NFS 服务器作为核心服务的存储后端的问题。这包括 OpenShift Container Registry 和 Quay，Prometheus 用于监控存储，以及 Elasticsearch 用于日志存储。因此，不建议使用 RHEL NFS 作为 PV 后端用于核心服务。

市场上的其他 NFS 实现可能没有这些问题。如需了解更多与此问题相关的信息，请联络相关的 NFS 厂商。

流程

1. 要将 registry 配置为使用存储，修改 configs.imageregistry/cluster 资源中的 spec.storage.pvc。

注意

使用共享存储时，请查看您的安全设置以防止外部访问。

2. 验证您没有 registry pod:

```
$ oc get pod -n openshift-image-registry -l docker-registry=default
```

输出示例

```
No resourses found in openshift-image-registry namespace
```

注意

如果您的输出中有一个 registry pod，则不需要继续这个过程。

3. 检查 registry 配置：

```
$ oc edit configs.imageregistry.operator.openshift.io
```

输出示例

```
storage:
  pvc:
    claim: 1
```

1

将 claim 字段留空以允许自动创建 image-registry-storage 持久性卷声明(PVC)。PVC 基于默认存储类生成。但请注意，默认存储类可能会提供 ReadWriteOnce (RWO) 卷，如 RADOS 块设备(RBD)，这可能会在复制到多个副本时导致问题。

4. 检查 clusteroperator 状态：

```
$ oc get clusteroperator image-registry
```

输出示例

```
NAME           VERSION                              AVAILABLE   PROGRESSING   DEGRADED
SINCE   MESSAGE
image-registry   4.7                                True        False         False      6h50m
```

23.2.5.11.2.2. 为 VMware vSphere 配置块 registry 存储

要允许镜像 registry 在作为集群管理员升级过程中使用块存储类型，如 vSphere Virtual Machine Disk(VMDK)，您可以使用 Recreate rollout 策略。

重要

支持块存储卷，但不建议在生产环境中用于镜像 registry。在块存储上配置 registry 的安装不具有高可用性，因为 registry 无法具有多个副本。

流程

1. 输入以下命令将镜像 registry 存储设置为块存储类型，对 registry 进行补丁，使其使用 Recreate rollout 策略，并只使用一个副本运行：

```
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec":
{"rolloutStrategy":"Recreate","replicas":1}}'
```

2. 为块存储设备置备 PV，并为该卷创建 PVC。请求的块卷使用 ReadWriteOnce(RWO)访问模式。

   a. 创建包含以下内容的 pvc.yaml 文件以定义 VMware vSphere PersistentVolumeClaim 对象：

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: image-registry-storage (1)
  namespace: openshift-image-registry (2)
```

```
spec:
 accessModes:
 - ReadWriteOnce 3
 resources:
  requests:
   storage: 100Gi 4
```

1 代表 PersistentVolumeClaim 对象的唯一名称。

2 PersistentVolumeClaim 对象的命名空间，即 openshift-image-registry。

3 持久性卷声明的访问模式。使用 ReadWriteOnce 时，单个节点可以通过读写权限挂载该卷。

4 持久性卷声明的大小。

b. 输入以下命令从文件创建 PersistentVolumeClaim 对象：

```
$ oc create -f pvc.yaml -n openshift-image-registry
```

3. 输入以下命令编辑 registry 配置，使其引用正确的 PVC：

```
$ oc edit config.imageregistry.operator.openshift.io -o yaml
```

输出示例

```
storage:
  pvc:
    claim: 1
```

1

有关配置 registry 存储以便引用正确的 PVC 的说明，请参阅 为 vSphere 配置 registry。

23.2.5.12. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于 远程健康监控

23.2.5.13. 配置要在 control plane 上运行的网络组件

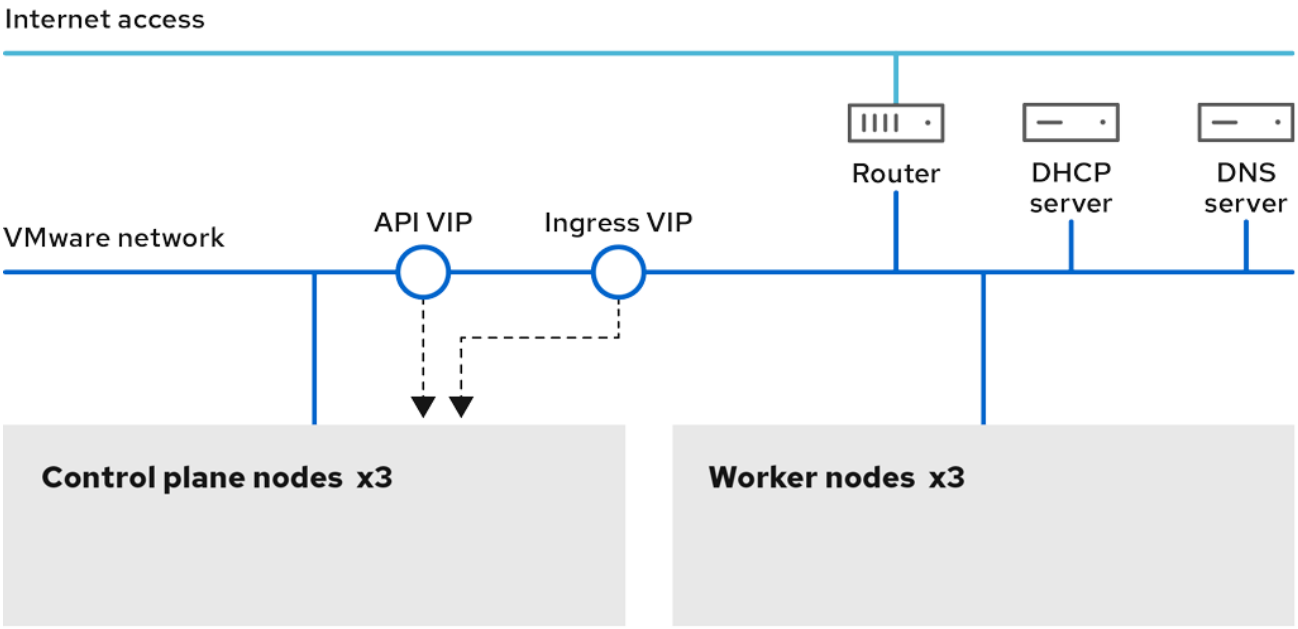
您可以配置网络组件，使其仅在 control plane 节点上运行。默认情况下，OpenShift Container Platform 允许机器配置池中的任何节点托管 ingressVIP 虚拟 IP 地址。但是，有些环境在与 control plane 节点独立的子网中部署计算节点，这需要将 ingressVIP 虚拟 IP 地址配置为在 control plane 节点上运行。

注意

您可以通过在单独的子网中创建计算机器集来扩展远程节点。

重要

在单独的子网中部署远程节点时，您必须将 ingressVIP 虚拟 IP 地址专门用于 control plane 节点。



流程

1. 进入存储 **install-config.yaml** 文件的目录：

```
$ cd ~/clusterconfigs
```

2. 切换到 **manifests** 子目录：

```
$ cd manifests
```

3. 创建名为 **cluster-network-avoid-workers-99-config.yaml** 的文件：

```
$ touch cluster-network-avoid-workers-99-config.yaml
```

4. 在编辑器中打开 **cluster-network-avoid-workers-99-config.yaml** 文件，并输入描述 Operator 配置的自定义资源(CR)：

```
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  name: 50-worker-fix-ipi-rwn
  labels:
    machineconfiguration.openshift.io/role: worker
spec:
  config:
    ignition:
```

```
    version: 3.2.0
  storage:
    files:
      - path: /etc/kubernetes/manifests/keepalived.yaml
        mode: 0644
        contents:
          source: data:,
```

此清单将 ingressVIP 虚拟 IP 地址放在 control plane 节点上。另外，此清单仅在 control plane 节点上部署以下进程：

- openshift-ingress-operator
- keepalived

5. 保存 cluster-network-avoid-workers-99-config.yaml 文件。

6. 创建 manifests/cluster-ingress-default-ingresscontroller.yaml 文件：

```
apiVersion: operator.openshift.io/v1
kind: IngressController
metadata:
  name: default
  namespace: openshift-ingress-operator
spec:
  nodePlacement:
    nodeSelector:
      matchLabels:
        node-role.kubernetes.io/master: ""
```

7. 考虑备份 manifests 目录。在创建集群时，安装程序会删除 manifests/ 目录。

8. 通过将 mastersSchedulable 字段设置为 true 来修改 cluster-scheduler-02-config.yml 清单，使 control plane 节点可以调度。默认情况下，control plane 节点不可调度。例如：

```
$ sed -i "s;mastersSchedulable: false;mastersSchedulable: true;g"
clusterconfigs/manifests/cluster-scheduler-02-config.yml
```

注意

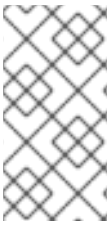
如果在完成此步骤后 control plane 节点不可调度，则部署集群将失败。

23.2.5.14. 后续步骤

- 自定义集群。
- 如果需要，您可以选择 不使用远程健康报告。
- 设置 registry 并配置 registry 存储。
- 可选： 查看 vSphere 问题检测器 Operator 中的事件，以确定集群是否有权限或存储配置问题。

23.2.6. 在受限网络中的 vSphere 上安装集群

在 OpenShift Container Platform 4.16 中，您可以通过创建安装发行内容的内部镜像在受限网络中的 VMware vSphere 基础架构上安装集群。



注意

OpenShift Container Platform 支持将集群部署到单个 VMware vCenter 中。不支持在多个 vCenter 上使用机器/机器集部署集群。

23.2.6.1. 先决条件

- 您已完成了 准备使用安装程序置备的基础架构安装集群 中的任务。
- 您检查了 VMware 平台许可证。红帽不会对 VMware 许可证产生任何限制，但有些 VMware 基础架构组件需要许可。
- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。

- 您可以阅读有关 选择集群安装方法的文档，并为用户准备它。

- 您在镜像主机上创建 registry，并获取您的 OpenShift Container Platform 版本的 imageContentSources 数据。

重要

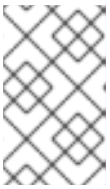
由于安装介质位于镜像主机上，因此您可以使用该计算机完成所有安装步骤。

- 已为集群置备了 持久性存储。要部署私有镜像 registry，您的存储必须提供 ReadWriteMany 访问模式。

- OpenShift Container Platform 安装程序需要访问 vCenter 和 ESXi 主机上的端口 443。您确认可以访问端口 443。

- 如果您使用防火墙，您与管理员确认可以访问端口 443。control plane 节点必须能够通过端口 443 访问 vCenter 和 ESXi 主机，才能成功安装。

- 如果您使用防火墙并计划使用 Telemetry 服务，则将防火墙配置为允许集群需要访问的站点。



注意

如果要配置代理，请务必查看此站点列表。

23.2.6.2. 关于在受限网络中安装

在 OpenShift Container Platform 4.16 中，可以执行不需要有效的互联网连接来获取软件组件的安装。受限网络安装可以使用安装程序置备的基础架构或用户置备的基础架构完成，具体取决于您要安装集群的云平台。

如果您选择在云平台中执行受限网络安装，您仍需要访问其云 API。有些云功能，比如 Amazon Web Service 的 Route 53 DNS 和 IAM 服务，需要访问互联网。根据您的网络，在裸机硬件、Nutanix 或 VMware vSphere 上安装可能需要较少的互联网访问。

要完成受限网络安装，您必须创建一个 registry，以镜像 OpenShift 镜像 registry 的内容并包含安装介质。您可以在镜像主机上创建此 registry，该主机可同时访问互联网和您的封闭网络，也可以使用满足您的限制条件的其他方法。

23.2.6.2.1. 其他限制

受限网络中的集群有以下额外限制和限制：

- ClusterVersion 状态包含一个 Unable to retrieve available updates 错误。
- 默认情况下，您无法使用 Developer Catalog 的内容，因为您无法访问所需的镜像流标签。

23.2.6.3. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来获得用来安装集群的镜像。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

23.2.6.4. 为受限网络安装创建 RHCOS 镜像

下载 Red Hat Enterprise Linux CoreOS(RHCOS)镜像，以在受限网络 VMware vSphere 环境上安装 OpenShift Container Platform。

先决条件

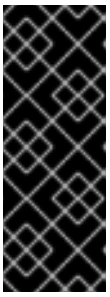
- 获取 OpenShift Container Platform 安装程序。对于受限网络安装，该程序位于您的镜像

registry 主机上。

流程

1. 登录到红帽客户门户网站的产品 下载页面。

2. 在 Version 下，为 RHEL 8 选择 OpenShift Container Platform 4.16 的最新发行版本。



重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每个发行版本而改变。您必须下载最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。如果可用，请使用与 OpenShift Container Platform 版本匹配的镜像版本。

3. 下载 Red Hat Enterprise Linux CoreOS(RHCOS)- vSphere 镜像。

4. 将下载的镜像上传到堡垒服务器可访问的位置。

该镜像现在可用于受限安装。记录 OpenShift Container Platform 部署中使用的镜像名称或位置。

23.2.6.5. VMware vSphere 区域和区启用

您可以将 OpenShift Container Platform 集群部署到在单个 VMware vCenter 中运行的多个 vSphere 数据中心。每个数据中心都可以运行多个集群。此配置降低了导致集群失败的硬件故障或网络中断的风险。要启用区域和区域，您必须为 OpenShift Container Platform 集群定义多个故障域。

重要

VMware vSphere 区域和区启用功能需要 vSphere Container Storage Interface (CSI) 驱动程序作为集群中的默认存储驱动程序。因此，这个功能只在新安装的集群中可用。

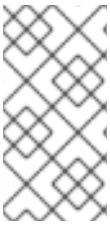
对于从上一版本升级的集群，您必须为集群启用 CSI 自动迁移。然后，您可以为升级的集群配置多个区域和区域。

默认安装配置将集群部署到单个 vSphere 数据中心。如果要将集群部署到多个 vSphere 数据中心，您必须创建一个启用地区和区功能的安装配置文件。

默认 install-config.yaml 文件包含 vcenters 和 failureDomains 字段，您可以在其中为 OpenShift Container Platform 集群指定多个 vSphere 数据中心和集群。如果要在由单个数据中心组成的 vSphere 环境中安装 OpenShift Container Platform 集群，您可以将这些字段留空。

以下列表描述了为集群定义区和区域相关的术语：

- 故障域：建立地区和区域之间的关系。您可以使用 vCenter 对象（如 datastore 对象）定义故障域。故障域定义 OpenShift Container Platform 集群节点的 vCenter 位置。
- Region ：指定 vCenter 数据中心。您可以使用 openshift-region 标签类别中的标签来定义区域。
- Zone：指定一个 vCenter 集群。您可以使用 openshift-zone 标签类别中的标签来定义区。



注意

如果您计划在 install-config.yaml 文件中指定多个故障域，则必须在创建配置文件前创建标签类别、区域标签和区域标签。

您必须为每个代表一个区域的 vCenter 数据中心创建一个 vCenter 标签。另外，您必须为比数据中心（代表一个区）中运行的每个集群创建一个 vCenter 标签。创建标签后，您必须将每个标签附加到对应的数据中心和集群。

下表概述了在单个 VMware vCenter 中运行的多个 vSphere 数据中心的区域、区域和标签之间的关系示例。

数据中心（区域）	集群（区）	Tags
us-east	us-east-1	us-east-1a
		us-east-1b

数据中心（区域）	集群（区）	Tags
	us-east-2	us-east-2a
		us-east-2b
us-west	us-west-1	us-west-1a
		us-west-1b
	us-west-2	us-west-2a
		us-west-2b

其他资源

- 其他 VMware vSphere 配置参数
- 弃用的 VMware vSphere 配置参数
- vSphere 自动迁移
- VMware vSphere CSI Driver Operator

23.2.6.6. 创建安装配置文件

您可以自定义在 VMware vSphere 上安装的 OpenShift Container Platform 集群。

先决条件

- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。对于受限网络安装，这些文件位于您的镜像主机上。
- 您有创建镜像 registry 期间生成的 imageContentSources 值。
- 您已获取了镜像 registry 的证书内容。

- 您已检索了 Red Hat Enterprise Linux CoreOS (RHCOS) 镜像，并将其上传到可访问的位置。

流程

1. 创建 install-config.yaml 文件。

   a. 进入包含安装程序的目录并运行以下命令：

   ```
   $ ./openshift-install create install-config --dir <installation_directory> 1
   ```

   1 对于 <installation_directory>，请指定要存储安装程序创建的文件的目录名称。

   在指定目录时：

   - 验证该目录是否具有执行权限。在安装目录中运行 Terraform 二进制文件需要这个权限。
   - 使用空目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

   b. 在提示符处，提供云的配置详情：

      i. 可选：选择用于访问集群机器的 SSH 密钥。

      注意

      对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 ssh-agent 进程使用的 SSH 密钥。

ii. Select vsphere 作为目标平台。

iii. 指定 vCenter 实例的名称。

iv. 指定创建集群所需的权限的 vCenter 帐户的用户名和密码。

安装程序连接到您的 vCenter 实例。

v. 选择要连接的 vCenter 实例中的数据中心。

注意

创建安装配置文件后，您可以修改该文件以创建多个 vSphere 数据中心环境。这意味着您可以将 OpenShift Container Platform 集群部署到在单个 VMware vCenter 中运行的多个 vSphere 数据中心。有关创建此环境的更多信息，请参阅名为 *VMware vSphere 区域和区启用*的部分。

vi. 选择要使用的默认 vCenter 数据存储。

警告

您可以指定数据存储集群中存在的任何数据存储路径。默认情况下，使用 Storage vMotion 的存储分布式资源调度程序(SDRS)会自动为数据存储集群启用。红帽不支持 Storage vMotion，因此您必须禁用 Storage DRS 以避免 OpenShift Container Platform 集群的数据丢失问题。

您不能指定多个数据存储路径。如果需要在多个数据存储间指定虚拟机，请使用 数据存储 对象在集群 install-config.yaml 配置文件中指定故障域。如需更多信息，请参阅"VMware vSphere 区域和区启用"。

vii. 选择要在其中安装 OpenShift Container Platform 集群的 vCenter 集群。安装程序使用 vSphere 集群的 root 资源池作为默认资源池。

viii. 选择包含您配置的虚拟 IP 地址和 DNS 记录的 vCenter 实例中的网络。

ix. 输入您为 control plane API 访问配置的虚拟 IP 地址。

x. 输入您为集群入口配置的虚拟 IP 地址。

xi. 输入基域。这个基域必须与您配置的 DNS 记录中使用的域相同。

xii. 为集群输入描述性名称。

您输入的集群名称必须与您在配置 DNS 记录时指定的集群名称匹配。

2. 在 install-config.yaml 文件中，将 platform.vsphere.clusterOSImage 的值设置为镜像位置或名称。例如：

```
platform:
  vsphere:
    clusterOSImage: http://mirror.example.com/images/rhcos-43.81.201912131630.0-
vmware.x86_64.ova?
sha256=ffebbd68e8a1f2a245ca19522c16c86f67f9ac8e4e0c1f0a812b068b16f7265d
```

3. 编辑 install-config.yaml 文件，以提供在受限网络中安装所需的额外信息。

a. 更新 pullSecret 值，使其包含 registry 的身份验证信息：

```
pullSecret: '{"auths":{"<mirror_host_name>:5000": {"auth": "
<credentials>","email": "you@example.com"}}}'
```

对于 <mirror_host_name>，请指定 您在镜像 registry 证书中指定的 registry 域名 ；对于 <credentials>， 请指定您的镜像 registry 的 base64 编码用户名和密码。

b.

b.

添加 additionalTrustBundle 参数和值。

```
additionalTrustBundle: |
  -----BEGIN CERTIFICATE-----

ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
  -----END CERTIFICATE-----
```

该值必须是您用于镜像 registry 的证书文件内容。证书文件可以是现有的可信证书颁发机构，也可以是您为镜像 registry 生成的自签名证书。

c.

添加镜像内容资源，类似于以下 YAML 摘录：

```
imageContentSources:
- mirrors:
  - <mirror_host_name>:5000/<repo_name>/release
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <mirror_host_name>:5000/<repo_name>/release
  source: registry.redhat.io/ocp/release
```

对于这些值，请使用您在创建镜像 registry 时记录的 imageContentSources。

d.

可选：将发布策略设置为 Internal：

```
publish: Internal
```

通过设置这个选项，您可以创建一个内部 Ingress Controller 和一个私有负载均衡器。

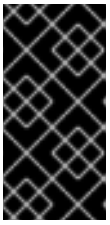
4.

对您需要的 install-config.yaml 文件进行任何其他修改。

有关参数的更多信息，请参阅"安装配置参数"。

5.

备份 install-config.yaml 文件，以便您可以使用它安装多个集群。



重要

install-config.yaml 文件会在安装过程中消耗掉。如果要重复使用此文件，必须现在备份。

其他资源

- 安装配置参数

23.2.6.6.1. 安装程序置备的 VMware vSphere 集群的 install-config.yaml 文件示例

您可以自定义 install-config.yaml 文件，以指定有关 OpenShift Container Platform 集群平台的更多详情，或修改所需参数的值。

```
apiVersion: v1
baseDomain: example.com (1)
compute: (2)
- architecture: amd64
  name:  <worker_node>
  platform: {}
  replicas: 3
controlPlane: (3)
  architecture: amd64
  name: <parent_node>
  platform: {}
  replicas: 3
metadata:
  creationTimestamp: null
  name: test (4)
platform:
  vsphere: (5)
    apiVIPs:
      - 10.0.0.1
    failureDomains: (6)
    - name: <failure_domain_name>
      region: <default_region_name>
      server: <fully_qualified_domain_name>
      topology:
        computeCluster: "/<datacenter>/host/<cluster>"
        datacenter: <datacenter>
        datastore: "/<datacenter>/datastore/<datastore>" (7)
        networks:
        - <VM_Network_name>
        resourcePool: "/<datacenter>/host/<cluster>/Resources/<resourcePool>" (8)
        folder: "/<datacenter_name>/vm/<folder_name>/<subfolder_name>"
        tagIDs: (9)
        - <tag_id> (10)
      zone: <default_zone_name>
```

```
    ingressVIPs:
    - 10.0.0.2
    vcenters:
    - datacenters:
      - <datacenter>
      password: <password>
      port: 443
      server: <fully_qualified_domain_name>
      user: administrator@vsphere.local
    diskType: thin (11)
    clusterOSImage: http://mirror.example.com/images/rhcos-47.83.202103221318-0-
vmware.x86_64.ova (12)
fips: false
pullSecret: '{"auths":{"<local_registry>": {"auth": "<credentials>","email":
"you@example.com"}}}' (13)
sshKey: 'ssh-ed25519 AAAA...'
additionalTrustBundle: | (14)
  -----BEGIN CERTIFICATE-----
  ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
  -----END CERTIFICATE-----
imageContentSources: (15)
- mirrors:
  - <mirror_host_name>:<mirror_port>/<repo_name>/release
  source: <source_image_1>
- mirrors:
  - <mirror_host_name>:<mirror_port>/<repo_name>/release-images
  source: <source_image_2>
```

(1)

集群的基域。所有 DNS 记录都必须是这个基域的子域，并包含集群名称。

(2) (3)

controlPlane 部分是一个单个映射，但 compute 部分是一系列映射。为满足不同数据结构的要求，compute 部分的第一行必须以连字符 - 开头，controlPlane 部分 的第一行则不以连字符开头。仅使用一个 control plane 池。

(4)

您在 DNS 记录中指定的集群名称。

(5)

可选：为 compute 和 control plane 机器提供额外的机器池参数配置。

(6)

建立地区和区域之间的关系。您可以使用 vCenter 对象（如 datastore 对象）定义故障域。故障域定义 OpenShift Container Platform 集群节点的 vCenter 位置。

7

保存虚拟机文件、模板和 ISO 镜像的 vSphere 数据存储路径。

重要

您可以指定数据存储集群中存在的任何数据存储路径。默认情况下，Storage vMotion 会自动为数据存储集群启用。红帽不支持 Storage vMotion，因此您必须禁用 Storage vMotion 以避免 OpenShift Container Platform 集群的数据丢失问题。

如果需要在多个数据存储间指定虚拟机，请使用 数据存储 对象在集群 install-config.yaml 配置文件中指定故障域。如需更多信息，请参阅"VMware vSphere 区域和区启用"。

8

可选：为创建机器提供现有资源池。如果没有指定值，安装程序将使用 vSphere 集群的 root 资源池。

9

可选：由 OpenShift Container Platform 创建的每个虚拟机都会被分配一个特定于集群的唯一标签。分配的标签可让安装程序在集群停用时识别和删除关联的虚拟机。您可以最多列出十个额外标签 ID，以附加到安装程序置备的虚拟机。

10

安装程序关联的标签的 ID。例如，urn:vmomi:InventoryServiceTag:208e713c-cae3-4b7f-918e-4051ca7d1f97:GLOBAL.有关确定标签 ID 的更多信息，请参阅 vSphere 标签和属性文档。

11

vSphere 磁盘置备方法。

12

可从 bastion 服务器访问的 Red Hat Enterprise Linux CoreOS(RHCOS)镜像的位置。

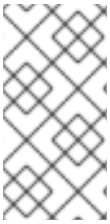
13

对于 <local_registry>，请指定 registry 域名，以及您的镜像 registry 用来提供内容的可选端口。例如 registry.example.com 或 registry.example.com:5000。对于 <credentials>，请为您的镜像 registry 指定 base64 编码的用户名和密码。

14

15

提供命令输出中的 imageContentSources 部分来 镜像存储库。



注意

在 OpenShift Container Platform 4.12 及更新的版本中，apiVIP 和 ingressVIP 配置设置已弃用。反之，使用列表格式在 apiVIPs 和 ingressVIPs 配置设置中输入值。

23.2.6.6.2. 在安装过程中配置集群范围的代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 install-config.yaml 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 install-config.yaml 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 Proxy 对象的 spec.noProxy 字段中添加站点来绕过代理。

注意

Proxy 对象 status.noProxy 字段使用安装配置中的 networking.machineNetwork[].cidr、networking.clusterNetwork[].cidr 和 networking.serviceNetwork[] 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，Proxy 对象 status.noProxy 字段也会使用实例元数据端点填充(169.254.169.254)。

流程

1. 编辑 install-config.yaml 文件并添加代理设置。例如：

```
apiVersion: v1
```

```
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1

用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 http。

2

用于创建集群外 HTTPS 连接的代理 URL。

3

要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 . 以仅匹配子域。例如，.y.com 匹配 x.y.com，但不匹配 y.com。使用 * 绕过所有目的地的代理。您必须包含 vCenter 的 IP 地址以及用于其机器的 IP 范围。

4

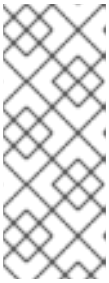
如果提供，安装程序会在 openshift-config 命名空间中生成名为 user-ca-bundle 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 trusted-ca-bundle 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， Proxy 对象的 trustedCA 字段中也会引用此配置映射。additionalTrustBundle 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

5

可选：决定 Proxy 对象的配置以引用 trustedCA 字段中 user-ca-bundle 配置映射的策略。允许的值是 Proxyonly 和 Always。仅在配置了 http/https 代理时，使用 Proxyonly 引用 user-ca-bundle 配置映射。使用 Always 始终引用 user-ca-bundle 配置映射。默认值为 Proxyonly。

注意

安装程序不支持代理的 readinessEndpoints 字段。



注意

如果安装程序超时，重启并使用安装程序的 wait-for 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. 保存该文件并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 cluster 的集群范围代理，该代理 使用 提供的 install-config.yaml 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 cluster Proxy 对象，但它会有一个空 spec。

注意

只支持名为 cluster 的 Proxy 对象，且无法创建额外的代理。

23.2.6.6.3. 为 VMware vCenter 配置区域和区域

您可以修改默认安装配置文件，以便您可以将 OpenShift Container Platform 集群部署到在单个 VMware vCenter 中运行的多个 vSphere 数据中心。

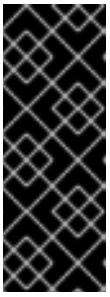
之前版本的 OpenShift Container Platform 的默认 install-config.yaml 文件配置已弃用。您可以继续使用已弃用的默认配置，但 openshift-installer 会提示您显示在配置文件中已弃用字段的警告信息。

重要

这个示例使用 govc 命令。govc 命令是 VMware 提供的开源命令；它不是红帽提供的。红帽支持团队不维护 govc 命令。有关下载和安装 govc 的说明，请参阅 VMware 文档网站

先决条件

- 您有一个现有的 install-config.yaml 安装配置文件。



重要

您必须为 OpenShift Container Platform 集群指定一个故障域，以便您可以为 VMware vCenter 服务器置备数据中心对象。如果您需要在不同的数据中心、集群、数据存储和其他组件中置备虚拟机节点，请考虑指定多个故障域。要启用区域和区域，您必须为 OpenShift Container Platform 集群定义多个故障域。

流程

1. **输入以下 govc 命令行工具命令，以创建 openshift-region 和 openshift-zone vCenter 标签类别：**

重要

如果为 openshift-region 和 openshift-zone vCenter 标签类别指定不同的名称，OpenShift Container Platform 集群的安装会失败。

```
$ govc tags.category.create -d "OpenShift region" openshift-region
```

```
$ govc tags.category.create -d "OpenShift zone" openshift-zone
```

2. **要为您要部署集群的每个区域 vSphere 数据中心创建一个 region 标签，请在终端中输入以下命令：**

```
$ govc tags.create -c <region_tag_category> <region_tag>
```

3. **要为您要部署集群的每个 vSphere 集群创建一个区标签，请输入以下命令：**

```
$ govc tags.create -c <zone_tag_category> <zone_tag>
```

4. **输入以下命令将区域标签附加到每个 vCenter 数据中心对象：**

```
$ govc tags.attach -c <region_tag_category> <region_tag_1> /<datacenter_1>
```

5. **输入以下命令将区标签附加到每个 vCenter 数据中心对象：**

```
$ govc tags.attach -c <zone_tag_category> <zone_tag_1> /<datacenter_1>/host/vcs-mdcnc-workload-1
```

6. 进入包含安装程序的目录，并根据您选择的安装要求初始化集群部署。

在 vSphere 数据中心中定义的多个数据中心的 install-config.yaml 文件示例

```
---
compute:
---
  vsphere:
    zones:
      - "<machine_pool_zone_1>"
      - "<machine_pool_zone_2>"
---
controlPlane:
---
vsphere:
    zones:
      - "<machine_pool_zone_1>"
      - "<machine_pool_zone_2>"
---
platform:
  vsphere:
    vcenters:
---
    datacenters:
      - <datacenter1_name>
      - <datacenter2_name>
    failureDomains:
    - name: <machine_pool_zone_1>
      region: <region_tag_1>
      zone: <zone_tag_1>
      server: <fully_qualified_domain_name>
      topology:
        datacenter: <datacenter1>
        computeCluster: "/<datacenter1>/host/<cluster1>"
        networks:
        - <VM_Network1_name>
        datastore: "/<datacenter1>/datastore/<datastore1>"
        resourcePool: "/<datacenter1>/host/<cluster1>/Resources/<resourcePool1>"
        folder: "/<datacenter1>/vm/<folder1>"
    - name: <machine_pool_zone_2>
      region: <region_tag_2>
      zone: <zone_tag_2>
      server: <fully_qualified_domain_name>
      topology:
        datacenter: <datacenter2>
        computeCluster: "/<datacenter2>/host/<cluster2>"
        networks:
        - <VM_Network2_name>
        datastore: "/<datacenter2>/datastore/<datastore2>"
```

```
        resourcePool: "/<datacenter2>/host/<cluster2>/Resources/<resourcePool2>"
        folder: "/<datacenter2>/vm/<folder2>"
---
```

23.2.6.7. 用户管理的负载均衡器的服务

您可以将 OpenShift Container Platform 集群配置为使用用户管理的负载均衡器来代替默认负载均衡器。

重要

配置用户管理的负载均衡器取决于您的厂商的负载均衡器。

本节中的信息和示例仅用于指导目的。有关供应商负载均衡器的更多信息，请参阅供应商文档。

红帽支持用户管理的负载均衡器的以下服务：

- Ingress Controller
- OpenShift API
- OpenShift MachineConfig API

您可以选择是否要为用户管理的负载均衡器配置一个或多个所有服务。仅配置 Ingress Controller 服务是一个通用的配置选项。要更好地了解每个服务，请查看以下图表：

图 23.7. 显示 OpenShift Container Platform 环境中运行的 Ingress Controller 的网络工作流示例

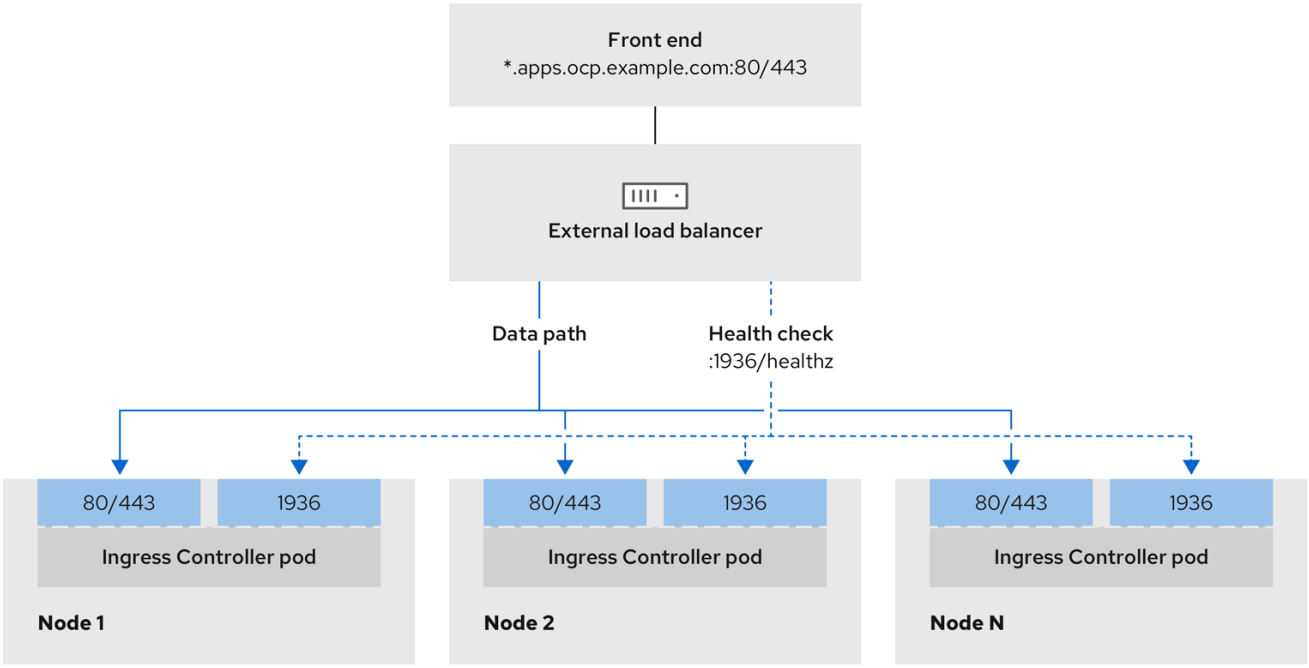


图 23.8. 显示 OpenShift Container Platform 环境中运行的 OpenShift API 的网络工作流示例

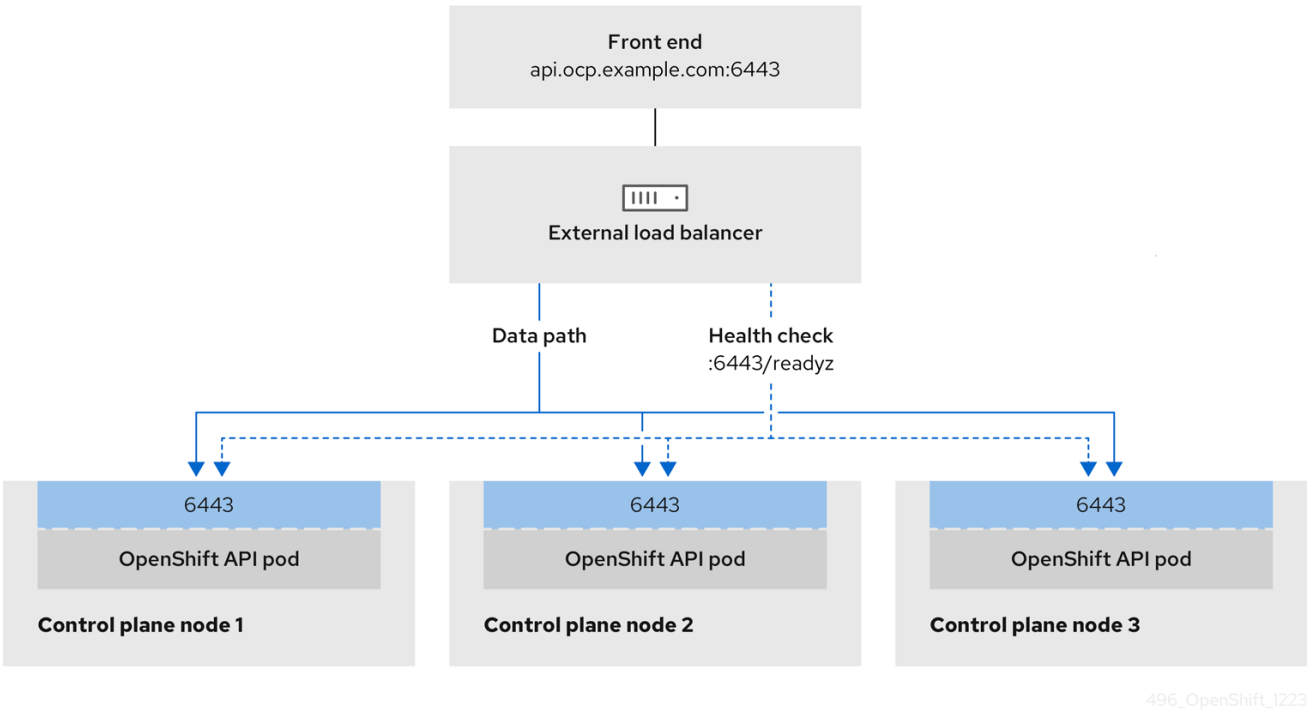
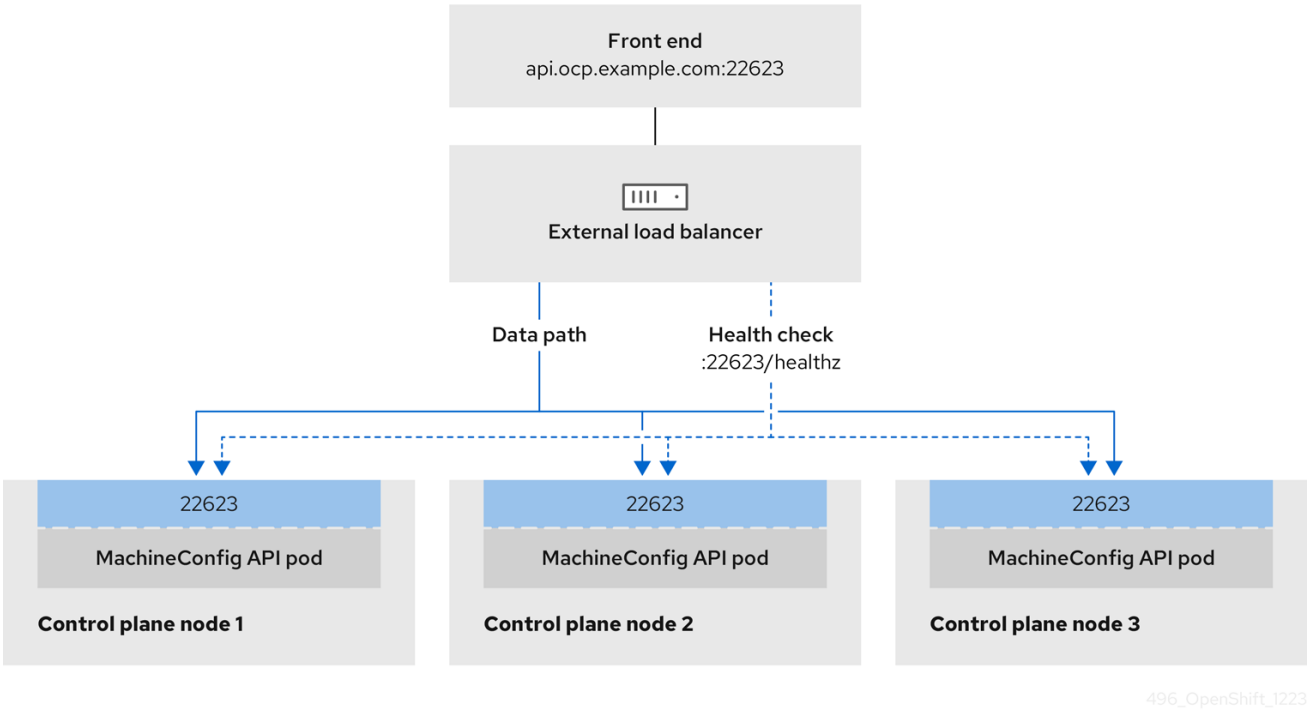


图 23.9. 显示 OpenShift Container Platform 环境中运行的 OpenShift MachineConfig API 的网络工作流示例



用户管理的负载均衡器支持以下配置选项：

- 使用节点选择器将 Ingress Controller 映射到一组特定的节点。您必须为这个集合中的每个节点分配一个静态 IP 地址，或者将每个节点配置为从动态主机配置协议(DHCP)接收相同的 IP 地址。基础架构节点通常接收这种类型的配置。

- 以子网上的所有 IP 地址为目标。此配置可减少维护开销，因为您可以在这些网络中创建和销毁节点，而无需重新配置负载均衡器目标。如果您使用较小的网络上的机器集来部署入口 pod，如 /27 或 /28，您可以简化负载均衡器目标。

  提示

  您可以通过检查机器配置池的资源来列出网络中存在的所有 IP 地址。

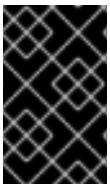
在为 OpenShift Container Platform 集群配置用户管理的负载均衡器前，请考虑以下信息：

- 对于前端 IP 地址，您可以对前端 IP 地址、Ingress Controller 的负载均衡器和 API 负载均衡器使用相同的 IP 地址。查看厂商的文档以获取此功能的相关信息。

- 对于后端 IP 地址，请确保 OpenShift Container Platform control plane 节点的 IP 地址在用户管理的负载均衡器生命周期内不会改变。您可以通过完成以下操作之一来实现此目的：
  - 为每个 control plane 节点分配一个静态 IP 地址。
  - 将每个节点配置为在每次节点请求 DHCP 租期时从 DHCP 接收相同的 IP 地址。根据供应商，DHCP 租期可能采用 IP 保留或静态 DHCP 分配的形式。
- 在 Ingress Controller 后端服务的用户管理的负载均衡器中手动定义运行 Ingress Controller 的每个节点。例如，如果 Ingress Controller 移到未定义节点，则可能会出现连接中断。

23.2.6.7.1. 配置用户管理的负载均衡器

您可以将 OpenShift Container Platform 集群配置为使用用户管理的负载均衡器来代替默认负载均衡器。



重要

在配置用户管理的负载均衡器前，请确保阅读用户管理的负载均衡器部分。

阅读适用于您要为用户管理的负载均衡器配置的服务的以下先决条件。

注意

MetalLB，在集群中运行，充当用户管理的负载均衡器。

OpenShift API 的先决条件

- 您定义了前端 IP 地址。
- TCP 端口 6443 和 22623 在负载均衡器的前端 IP 地址上公开。检查以下项：

- 端口 6443 提供对 OpenShift API 服务的访问。
- 端口 22623 可以为节点提供 ignition 启动配置。

- 前端 IP 地址和端口 6443 可以被您的系统的所有用户访问，其位置为 OpenShift Container Platform 集群外部。
- 前端 IP 地址和端口 22623 只能被 OpenShift Container Platform 节点访问。
- 负载均衡器后端可以在端口 6443 和 22623 上与 OpenShift Container Platform control plane 节点通信。

Ingress Controller 的先决条件

- 您定义了前端 IP 地址。
- TCP 端口 443 和 80 在负载均衡器的前端 IP 地址上公开。
- 前端 IP 地址、端口 80 和端口 443 可以被您的系统所有用户访问，以及 OpenShift Container Platform 集群外部的位置。
- 前端 IP 地址、端口 80 和端口 443 可被 OpenShift Container Platform 集群中运行的所有节点访问。
- 负载均衡器后端可以在端口 80、443 和 1936 上与运行 Ingress Controller 的 OpenShift Container Platform 节点通信。

健康检查 URL 规格的先决条件

您可以通过设置健康检查 URL 来配置大多数负载均衡器，以确定服务是否可用或不可用。OpenShift Container Platform 为 OpenShift API、Machine Configuration API 和 Ingress Controller 后端服务提供这些健康检查。

以下示例显示了之前列出的后端服务的健康检查规格：

Kubernetes API 健康检查规格示例

```
Path: HTTPS:6443/readyz
Healthy threshold: 2
Unhealthy threshold: 2
Timeout: 10
Interval: 10
```

Machine Config API 健康检查规格示例

```
Path: HTTPS:22623/healthz
Healthy threshold: 2
Unhealthy threshold: 2
Timeout: 10
Interval: 10
```

Ingress Controller 健康检查规格示例

```
Path: HTTP:1936/healthz/ready
Healthy threshold: 2
Unhealthy threshold: 2
Timeout: 5
Interval: 10
```

流程

1. 配置 HAProxy Ingress Controller，以便您可以在端口 6443、22623、443 和 80 上从负载均衡器访问集群。根据您的需要，您可以在 HAProxy 配置中指定来自多个子网的单个子网或 IP 地址的 IP 地址。

带有列出子网的 HAProxy 配置示例

```
# ...
listen my-cluster-api-6443
    bind 192.168.1.100:6443
    mode tcp
    balance roundrobin
  option httpchk
  http-check connect
  http-check send meth GET uri /readyz
  http-check expect status 200
    server my-cluster-master-2 192.168.1.101:6443 check inter 10s rise 2 fall 2
    server my-cluster-master-0 192.168.1.102:6443 check inter 10s rise 2 fall 2
    server my-cluster-master-1 192.168.1.103:6443 check inter 10s rise 2 fall 2

listen my-cluster-machine-config-api-22623
    bind 192.168.1.100:22623
    mode tcp
    balance roundrobin
  option httpchk
  http-check connect
  http-check send meth GET uri /healthz
  http-check expect status 200
    server my-cluster-master-2 192.168.1.101:22623 check inter 10s rise 2 fall 2
    server my-cluster-master-0 192.168.1.102:22623 check inter 10s rise 2 fall 2
    server my-cluster-master-1 192.168.1.103:22623 check inter 10s rise 2 fall 2

listen my-cluster-apps-443
    bind 192.168.1.100:443
    mode tcp
    balance roundrobin
  option httpchk
  http-check connect
  http-check send meth GET uri /healthz/ready
  http-check expect status 200
    server my-cluster-worker-0 192.168.1.111:443 check port 1936 inter 10s rise 2 fall 2
    server my-cluster-worker-1 192.168.1.112:443 check port 1936 inter 10s rise 2 fall 2
    server my-cluster-worker-2 192.168.1.113:443 check port 1936 inter 10s rise 2 fall 2

listen my-cluster-apps-80
   bind 192.168.1.100:80
   mode tcp
   balance roundrobin
  option httpchk
  http-check connect
  http-check send meth GET uri /healthz/ready
  http-check expect status 200
    server my-cluster-worker-0 192.168.1.111:80 check port 1936 inter 10s rise 2 fall 2
    server my-cluster-worker-1 192.168.1.112:80 check port 1936 inter 10s rise 2 fall 2
    server my-cluster-worker-2 192.168.1.113:80 check port 1936 inter 10s rise 2 fall 2
# ...
```

带有多个列出子网的 HAProxy 配置示例

```
# ...
listen api-server-6443
  bind *:6443
  mode tcp
    server master-00 192.168.83.89:6443 check inter 1s
    server master-01 192.168.84.90:6443 check inter 1s
    server master-02 192.168.85.99:6443 check inter 1s
    server bootstrap 192.168.80.89:6443 check inter 1s

listen machine-config-server-22623
  bind *:22623
  mode tcp
    server master-00 192.168.83.89:22623 check inter 1s
    server master-01 192.168.84.90:22623 check inter 1s
    server master-02 192.168.85.99:22623 check inter 1s
    server bootstrap 192.168.80.89:22623 check inter 1s

listen ingress-router-80
  bind *:80
  mode tcp
  balance source
    server worker-00 192.168.83.100:80 check inter 1s
    server worker-01 192.168.83.101:80 check inter 1s

listen ingress-router-443
  bind *:443
  mode tcp
  balance source
    server worker-00 192.168.83.100:443 check inter 1s
    server worker-01 192.168.83.101:443 check inter 1s

listen ironic-api-6385
  bind *:6385
  mode tcp
  balance source
    server master-00 192.168.83.89:6385 check inter 1s
    server master-01 192.168.84.90:6385 check inter 1s
    server master-02 192.168.85.99:6385 check inter 1s
    server bootstrap 192.168.80.89:6385 check inter 1s

listen inspector-api-5050
  bind *:5050
  mode tcp
  balance source
    server master-00 192.168.83.89:5050 check inter 1s
    server master-01 192.168.84.90:5050 check inter 1s
```

```
    server master-02 192.168.85.99:5050 check inter 1s
    server bootstrap 192.168.80.89:5050 check inter 1s
# ...
```

2. 使用 curl CLI 命令验证用户管理的负载均衡器及其资源是否正常运行：

a. 运行以下命令并查看响应，验证集群机器配置 API 是否可以被 Kubernetes API 服务器资源访问：

```
$ curl https://<loadbalancer_ip_address>:6443/version --insecure
```

如果配置正确，您会收到 JSON 对象的响应：

```
{
  "major": "1",
  "minor": "11+",
  "gitVersion": "v1.11.0+ad103ed",
  "gitCommit": "ad103ed",
  "gitTreeState": "clean",
  "buildDate": "2019-01-09T06:44:10Z",
  "goVersion": "go1.10.3",
  "compiler": "gc",
  "platform": "linux/amd64"
}
```

b. 运行以下命令并观察输出，验证集群机器配置 API 是否可以被 Machine 配置服务器资源访问：

```
$ curl -v https://<loadbalancer_ip_address>:22623/healthz --insecure
```

如果配置正确，命令的输出会显示以下响应：

```
HTTP/1.1 200 OK
Content-Length: 0
```

c. 运行以下命令并观察输出，验证控制器是否可以被端口 80 上的 Ingress Controller 资源访问：

```
$ curl -I -L -H "Host: console-openshift-console.apps.<cluster_name>.
<base_domain>" http://<load_balancer_front_end_IP_address>
```

如果配置正确，命令的输出会显示以下响应：

```
HTTP/1.1 302 Found
content-length: 0
location: https://console-openshift-console.apps.ocp4.private.opequon.net/
cache-control: no-cache
```

d. 运行以下命令并观察输出，验证控制器是否可以被端口 443 上的 Ingress Controller 资源访问：

```
$ curl -I -L --insecure --resolve console-openshift-console.apps.<cluster_name>.
<base_domain>:443:<Load Balancer Front End IP Address> https://console-
openshift-console.apps.<cluster_name>.<base_domain>
```

如果配置正确，命令的输出会显示以下响应：

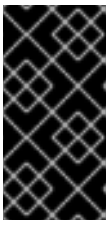
```
HTTP/1.1 200 OK
referrer-policy: strict-origin-when-cross-origin
set-cookie: csrf-
token=UlYWOyQ62LWjw2h003xtYSKlh1a0Py2hhctw0WmV2YEdhJjFyQwWcGBsja2
61dGLgaYO0nxzVErhiXt6QepA7g==; Path=/; Secure; SameSite=Lax
x-content-type-options: nosniff
x-dns-prefetch-control: off
x-frame-options: DENY
x-xss-protection: 1; mode=block
date: Wed, 04 Oct 2023 16:29:38 GMT
content-type: text/html; charset=utf-8
set-cookie:
1e2670d92730b515ce3a1bb65da45062=1bf5e9573c9a2760c964ed1659cc1673;
path=/; HttpOnly; Secure; SameSite=None
cache-control: private
```

3. 配置集群的 DNS 记录，使其以用户管理的负载均衡器的前端 IP 地址为目标。您必须在负载均衡器上将记录更新为集群 API 和应用程序的 DNS 服务器。

修改 DNS 记录示例

```
<load_balancer_ip_address>  A  api.<cluster_name>.<base_domain>
A record pointing to Load Balancer Front End
```

```
<load_balancer_ip_address>  A apps.<cluster_name>.<base_domain>
A record pointing to Load Balancer Front End
```



重要

DNS 传播可能需要一些时间才能获得每个 DNS 记录。在验证每个记录前，请确保每个 DNS 记录传播。

4.

要使 OpenShift Container Platform 集群使用用户管理的负载均衡器，您必须在集群的 install-config.yaml 文件中指定以下配置：

```
# ...
platform:
  vsphere:
    loadBalancer:
      type: UserManaged 1
      apiVIPs:
      - <api_ip> 2
      ingressVIPs:
      - <ingress_ip> 3
# ...
```

1

为 type 参数设置 UserManaged，为集群指定用户管理的负载均衡器。参数默认为 OpenShiftManagedDefault，它表示默认的内部负载均衡器。对于 openshift-kni-infra 命名空间中定义的服务，用户管理的负载均衡器可将 coredns 服务部署到集群中的 pod，但忽略 keepalived 和 haproxy 服务。

2

指定用户管理的负载均衡器时所需的参数。指定用户管理的负载均衡器的公共 IP 地址，以便 Kubernetes API 可以与用户管理的负载均衡器通信。

3

指定用户管理的负载均衡器时所需的参数。指定用户管理的负载均衡器的公共 IP 地址，以便用户管理的负载均衡器可以管理集群的入口流量。

验证

1. 使用 curl CLI 命令验证用户管理的负载均衡器和 DNS 记录配置是否正常工作：

   a. 运行以下命令并查看输出，验证您可以访问集群 API：

   ```
   $ curl https://api.<cluster_name>.<base_domain>:6443/version --insecure
   ```

   如果配置正确，您会收到 JSON 对象的响应：

   ```
   {
    "major": "1",
    "minor": "11+",
    "gitVersion": "v1.11.0+ad103ed",
    "gitCommit": "ad103ed",
    "gitTreeState": "clean",
    "buildDate": "2019-01-09T06:44:10Z",
    "goVersion": "go1.10.3",
    "compiler": "gc",
    "platform": "linux/amd64"
    }
   ```

   b. 运行以下命令并查看输出，验证您可以访问集群机器配置：

   ```
   $ curl -v https://api.<cluster_name>.<base_domain>:22623/healthz --insecure
   ```

   如果配置正确，命令的输出会显示以下响应：

   ```
   HTTP/1.1 200 OK
   Content-Length: 0
   ```

   c. 运行以下命令并查看输出，验证您可以在端口上访问每个集群应用程序：

   ```
   $ curl http://console-openshift-console.apps.<cluster_name>.<base_domain> -I -L
   --insecure
   ```

   如果配置正确，命令的输出会显示以下响应：

   ```
   HTTP/1.1 302 Found
   content-length: 0
   location: https://console-openshift-console.apps.<cluster-name>.<base domain>/
   cache-control: no-cacheHTTP/1.1 200 OK
   ```

```
referrer-policy: strict-origin-when-cross-origin
set-cookie: csrf-
token=39HoZgztDnzjJkq/JuLJMeoKNXlfiVv2YgZc09c3TBOBU4Nl6kDXaJH1LdicNh
N1UsQWzon4Dor9GWGfopaTEQ==; Path=/; Secure
x-content-type-options: nosniff
x-dns-prefetch-control: off
x-frame-options: DENY
x-xss-protection: 1; mode=block
date: Tue, 17 Nov 2020 08:42:10 GMT
content-type: text/html; charset=utf-8
set-cookie:
1e2670d92730b515ce3a1bb65da45062=9b714eb87e93cf34853e87a92d6894be;
path=/; HttpOnly; Secure; SameSite=None
cache-control: private
```

d. 运行以下命令并查看输出，验证您可以在端口 443 上访问每个集群应用程序：

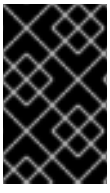
```
$ curl https://console-openshift-console.apps.<cluster_name>.<base_domain> -I -
L --insecure
```

如果配置正确，命令的输出会显示以下响应：

```
HTTP/1.1 200 OK
referrer-policy: strict-origin-when-cross-origin
set-cookie: csrf-
token=UlYWOyQ62LWjw2h003xtYSKlh1a0Py2hhctw0WmV2YEdhJjFyQwWcGBsja2
61dGLgaYO0nxzVErhiXt6QepA7g==; Path=/; Secure; SameSite=Lax
x-content-type-options: nosniff
x-dns-prefetch-control: off
x-frame-options: DENY
x-xss-protection: 1; mode=block
date: Wed, 04 Oct 2023 16:29:38 GMT
content-type: text/html; charset=utf-8
set-cookie:
1e2670d92730b515ce3a1bb65da45062=1bf5e9573c9a2760c964ed1659cc1673;
path=/; HttpOnly; Secure; SameSite=None
cache-control: private
```

23.2.6.8. 部署集群

您可以在兼容云平台上安装 OpenShift Container Platform。

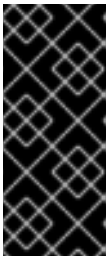


重要

在初始安装过程中，您只能运行安装程序的 create cluster 命令一次。

先决条件

- 您有 OpenShift Container Platform 安装程序和集群的 pull secret。
- 已确认主机上的云供应商帐户具有部署集群的正确权限。权限不正确的帐户会导致安装过程失败，并显示包括缺失权限的错误消息。
- 可选：在创建集群时，配置外部负载均衡器来代替默认负载均衡器。



重要

您不需要为安装程序指定 API 和 Ingress 静态地址。如果选择此配置，则必须采取额外的操作来定义接受每个引用的 vSphere 子网的 IP 地址的网络目标。请参阅"配置用户管理的负载均衡器"部分。

流程

- 进入包含安装程序的目录并初始化集群部署：

```
$ ./openshift-install create cluster --dir <installation_directory> \ 1
  --log-level=info 2
```

1. 对于 <installation_directory>，请指定自定义 ./install-config.yaml 文件的位置。
2. 要查看不同的安装详情，请指定 warn、debug 或 error，而不是 info。

验证

当集群部署成功完成时：

- 终端会显示用于访问集群的说明，包括指向 Web 控制台和 kubeadmin 用户的凭证的链接。
- 凭证信息还会输出到 <installation_directory>/.openshift_install.log.

重要

不要删除安装程序或安装程序所创建的文件。需要这两者才能删除集群。

输出示例

```
...
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'
INFO Access the OpenShift web-console here: https://console-openshift-
console.apps.mycluster.example.com
INFO Login to the console with user: "kubeadmin", and password: "password"
INFO Time elapsed: 36m22s
```

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 node-bootstrapper 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。

- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

23.2.6.9. 使用 CLI 登录集群

您可以通过导出集群 kubeconfig 文件，以默认系统用户身份登录集群。kubeconfig 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。

- 已安装 oc CLI。

流程

1. 导出 kubeadmin 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 <installation_directory>，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 oc 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

23.2.6.10. 禁用默认的 OperatorHub 目录源

在 OpenShift Container Platform 安装过程中，默认为 OperatorHub 配置由红帽和社区项目提供的源内容的 operator 目录。在受限网络环境中，必须以集群管理员身份禁用默认目录。

流程

- 通过在 OperatorHub 对象中添加 disableAllDefaultSources: true 来 禁用默认目录的源：

```
$ oc patch OperatorHub cluster --type json \
    -p '[{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]'
```

提示

或者，您可以使用 Web 控制台管理目录源。在 Administration → Cluster Settings → Configuration → OperatorHub 页面中，点 Sources 选项卡，您可以在其中创建、更新、删除、禁用和启用单独的源。

23.2.6.11. 创建 registry 存储

安装集群后，必须为 Registry Operator 创建存储。

23.2.6.11.1. 安装过程中删除的镜像 registry

在不提供可共享对象存储的平台上，OpenShift Image Registry Operator bootstraps 本身为 Removed。这允许 openshift-installer 在这些平台类型上完成安装。

安装后，您必须编辑 Image Registry Operator 配置，将 managementState 从 Removed 切换到 Managed。完成此操作后，您必须配置存储。

23.2.6.11.2. 镜像 registry 存储配置

对于不提供默认存储的平台，Image Registry Operator 最初不可用。安装后，您必须将 registry 配置为使用存储，以便 Registry Operator 可用。

显示配置生产集群所需的持久性卷的说明。如果适用，显示有关将空目录配置为存储位置的说明，这仅适用于非生产集群。

提供了在升级过程中使用 Recreate rollout 策略来允许镜像 registry 使用块存储类型的说明。

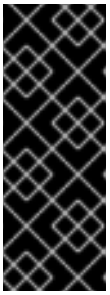
23.2.6.11.2.1. 为 VMware vSphere 配置 registry 存储

作为集群管理员，在安装后需要配置 registry 来使用存储。

先决条件

- 集群管理员权限。

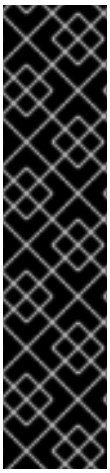
- VMware vSphere 上有一个集群。
- 为集群置备的持久性存储，如 Red Hat OpenShift Data Foundation。



重要

当您只有一个副本时，OpenShift Container Platform 支持对镜像 registry 存储的 ReadWriteOnce 访问。ReadWriteOnce 访问还要求 registry 使用 Recreate rollout 策略。要部署支持高可用性的镜像 registry，需要两个或多个副本，ReadWriteMany 访问。

- 必须具有"100Gi"容量.



重要

测试显示在 RHEL 中使用 NFS 服务器作为核心服务的存储后端的问题。这包括 OpenShift Container Registry 和 Quay，Prometheus 用于监控存储，以及 Elasticsearch 用于日志存储。因此，不建议使用 RHEL NFS 作为 PV 后端用于核心服务。

市场上的其他 NFS 实现可能没有这些问题。如需了解更多与此问题相关的信息，请联络相关的 NFS 厂商。

流程

1. 要将 registry 配置为使用存储，修改 configs.imageregistry/cluster 资源中的 spec.storage.pvc。

注意

使用共享存储时，请查看您的安全设置以防止外部访问。

2. 验证您没有 registry pod:

```
$ oc get pod -n openshift-image-registry -l docker-registry=default
```

输出示例

```
No resourses found in openshift-image-registry namespace
```

注意

如果您的输出中有一个 registry pod，则不需要继续这个过程。

3. 检查 registry 配置：

```
$ oc edit configs.imageregistry.operator.openshift.io
```

输出示例

```
storage:
  pvc:
    claim: 1
```

1

将 claim 字段留空以允许自动创建 image-registry-storage 持久性卷声明(PVC)。PVC 基于默认存储类生成。但请注意，默认存储类可能会提供 ReadWriteOnce (RWO) 卷，如 RADOS 块设备(RBD)，这可能会在复制到多个副本时导致问题。

4. 检查 clusteroperator 状态：

```
$ oc get clusteroperator image-registry
```

输出示例

```
-
```

```
NAME              VERSION                              AVAILABLE   PROGRESSING   DEGRADED
SINCE   MESSAGE
image-registry    4.7                                  True        False         False     6h50m
```

23.2.6.12. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控

23.2.6.13. 后续步骤

- 自定义集群。
- 如果需要，您可以选择不使用远程健康报告。
- 如果需要，请参阅注册断开连接的集群。
- 设置 registry 并配置 registry 存储。

23.3. 用户置备的基础架构

23.3.1. 用户置备的基础架构的 vSphere 安装要求

在您置备的基础架构上开始安装前，请确保您的 vSphere 环境满足以下安装要求。

23.3.1.1. VMware vSphere 基础架构要求

您必须在满足您使用的组件要求的 VMware vSphere 实例之一上安装 OpenShift Container Platform 集群：

- 版本 7.0 更新 2 或更高版本
- 版本 8.0 更新 1 或更高版本

这两个版本都支持 Container Storage Interface (CSI) 迁移，它在 OpenShift Container Platform 4.16 中默认启用。

您可以在内部或 VMware Cloud 验证的供应商中托管 VMware vSphere 基础架构，以满足下表中概述的要求：

表 23.18. vSphere 虚拟环境的版本要求

虚拟环境产品	所需的版本
VMware 虚拟硬件	15 或更高版本
vSphere ESXi 主机	7.0 更新 2 或更高版本; 8.0 更新 1 或更高版本
vCenter 主机	7.0 更新 2 或更高版本; 8.0 更新 1 或更高版本

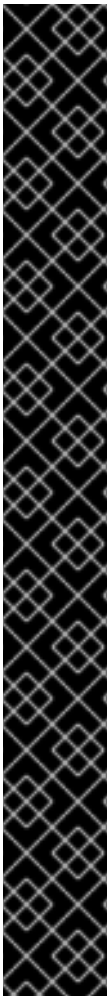
重要

您必须确保在安装 OpenShift Container Platform 前同步 ESXi 主机上的时间。请参阅 VMware 文档中的 编辑主机时间配置。

表 23.19. VMware 组件支持的最低 vSphere 版本

组件	最低支持版本	描述
虚拟机监控程序（Hypervisor）	vSphere 7.0 更新 2 （或更新版本）或 vSphere 8.0 更新 1 （或更新版本）带有虚拟硬件版本 15	此 hypervisor 版本是 Red Hat Enterprise Linux CoreOS (RHCOS) 支持的最低版本。有关与 RHCOS 兼容的 Red Hat Enterprise Linux (RHEL) 最新版本中支持的硬件的更多信息，请参阅红帽客户门户网站中的硬件。

组件	最低支持版本	描述
可选： Networking(NSX-T)	vSphere 7.0 更新 2 或更高版本; vSphere 8.0 更新 1 或更高版本	OpenShift Container Platform 至少需要 vSphere 7.0 Update 2 或 vSphere 8.0 Update 1。有关 NSX 和 OpenShift Container Platform 兼容性的更多信息，请参阅 VMware 的 NSX 容器插件文档中的发行注记部分。
CPU 微架构	x86-64-v2 或更高版本	OpenShift 4.13 及更高版本基于 RHEL 9.2 主机操作系统，这提高了 x86-64-v2 的微架构要求。请参阅 RHEL Microarchitecture 要求文档。您可以按照这个 KCS 文章中介绍的步骤验证兼容性。



重要

为确保在 Oracle® Cloud Infrastructure (OCI) 和 Oracle® Cloud VMware Solution (OCVS) 服务上运行的集群工作负载的最佳性能条件，请确保块卷的卷性能单元 (VPU) 为您的工作负载的大小。

以下列表提供了一些有关选择特定性能需要 VPU 的指导信息：

- 测试或概念验证环境：100 GB，20 到 30 个 VPU。
- 基础生产环境：500 GB 和 60 个 VPU.
- 高度使用生产环境：超过 500 GB，100 个或更多 VPU。

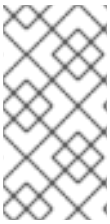
考虑分配额外的 VPU，以便为更新和扩展活动提供足够的容量。请参阅块卷性能级别 (Oracle 文档)。

23.3.1.2. VMware vSphere CSI Driver Operator 要求

要安装 vSphere Container Storage Interface (CSI) Driver Operator，必须满足以下要求：

- VMware vSphere 版本： 7.0 更新 2 或更高版本； 8.0 更新 1 或更高版本
- vCenter 版本： 7.0 更新 2 或更高版本； 8.0 更新 1 或更高版本
- 硬件版本 15 或更高版本的虚拟机
- 集群中还没有安装第三方 vSphere CSI 驱动程序

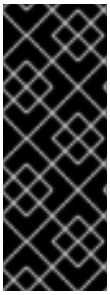
如果集群中存在第三方 vSphere CSI 驱动程序，OpenShift Container Platform 不会覆盖它。存在第三方 vSphere CSI 驱动程序可防止 OpenShift Container Platform 更新到 OpenShift Container Platform 4.13 或更高版本。



注意

只有在安装清单中使用 platform: vsphere 部署的集群中才支持 VMware vSphere CSI Driver Operator。

您可以为 Container Storage Interface (CSI) 驱动程序、vSphere CSI Driver Operator 和 vSphere Problem Detector Operator 创建自定义角色。自定义角色可以包含为每个 vSphere 对象分配最小权限集的权限集。这意味着 CSI 驱动程序、vSphere CSI Driver Operator 和 vSphere Problem Detector Operator 可以建立与这些对象的基本交互。



重要

在 vCenter 中安装 OpenShift Container Platform 集群会根据完整权限列表进行测试，如 "Required vCenter account privileges" 部分所述。通过遵循完整的特权列表，您可以减少创建具有一组受限权限的自定义角色时可能会出现意外和不支持的行为的可能性。

其他资源

- 要删除第三方 vSphere CSI 驱动程序，请参阅 删除第三方 vSphere CSI 驱动程序。
- 要为您的 vSphere 节点更新硬件版本，请参阅在 vSphere 中运行的节点上更新硬件。
- 存储组件的最低权限

23.3.1.3. 具有用户置备基础架构的集群的要求

对于包含用户置备的基础架构的集群，您必须部署所有所需的机器。

本节论述了在用户置备的基础架构上部署 OpenShift Container Platform 的要求。

23.3.1.3.1. vCenter 要求

在使用您提供的基础架构的 vCenter 上安装 OpenShift Container Platform 集群前，您必须准备您的环境。

所需的 vCenter 帐户权限

要在 vCenter 中安装 OpenShift Container Platform 集群，您的 vSphere 帐户必须包含读取和创建所需资源的权限。使用具有全局管理特权的帐户是访问所有所需权限的最简单方法。

例 23.8. 在 vSphere API 中安装所需的角色和权限

适用于角色的 vSphere 对象	必要时	vSphere API 中所需的权限
vSphere vCenter	Always	**Cns.Searchable** **InventoryService.Tagging.AttachTag** **InventoryService.Tagging.CreateCategory** **InventoryService.Tagging.CreateTag** **InventoryService.Tagging.DeleteCategory** **InventoryService.Tagging.DeleteTag** **InventoryService.Tagging.EditCategory** **InventoryService.Tagging.EditTag** **Sessions.ValidateSession** **StorageProfile.Update** **StorageProfile.View**
vSphere vCenter 集群	如果在集群 root 中创建虚拟机	**Host.Config.Storage** **Resource.AssignVMToPool** **VApp.AssignResourcePool** **VApp.Import** **VirtualMachine.Config.AddNewDisk**
vSphere vCenter 资源池	如果提供了现有的资源池	**Resource.AssignVMToPool** **VApp.AssignResourcePool** **VApp.Import** **VirtualMachine.Config.AddNewDisk**
vSphere Datastore	Always	**Datastore.AllocateSpace** **Datastore.Browse** **Datastore.FileManagement** **InventoryService.Tagging.ObjectAttachable**
vSphere 端口组	Always	**Network.Assign**
虚拟机文件夹	Always	**InventoryService.Tagging.ObjectAttachable** **Resource.AssignVMToPool** **VApp.Import**

适用于角色的 vSphere 对象	必要时	vSphere API 中所需的权限
		VirtualMachine.Config.AddExistingDisk VirtualMachine.Config.AddNewDisk VirtualMachine.Config.AddRemoveDevice VirtualMachine.Config.AdvancedConfig VirtualMachine.Config.Annotation VirtualMachine.Config.CPUCount VirtualMachine.Config.DiskExtend VirtualMachine.Config.DiskLease VirtualMachine.Config.EditDevice VirtualMachine.Config.Memory VirtualMachine.Config.RemoveDisk VirtualMachine.Config.Rename Host.Config.Storage VirtualMachine.Config.ResetGuestInfo VirtualMachine.Config.Resource VirtualMachine.Config.Settings VirtualMachine.Config.UpgradeVirtualHardware VirtualMachine.Interact.GuestControl VirtualMachine.Interact.PowerOff VirtualMachine.Interact.PowerOn VirtualMachine.Interact.Reset VirtualMachine.Inventory.Create VirtualMachine.Inventory.CreateFromExisting VirtualMachine.Inventory.Delete VirtualMachine.Provisioning.Clone VirtualMachine.Provisioning.MarkAsTemplate VirtualMachine.Provisioning.DeployTemplate
vSphere vCenter Datacenter	如果安装程序创建虚拟机文件夹对于用户置备的基础架构，如果	InventoryService.Tagging.ObjectAttachable

适用于角色的 vSphere 对象	必要时	vSphere API 中所需的权限
	集群没有使用 Machine API，则 **VirtualMachine.Inventory.Create** 和 **VirtualMachine.Inventory.Delete** 权限是可选的。请参阅"机器 API 的最小权限"表。	**Resource.AssignVMToPool** **VirtualMachine.Config.AddExistingDisk** **VirtualMachine.Config.AddNewDisk** **VirtualMachine.Config.AddRemoveDevice** **VirtualMachine.Config.AdvancedConfig** **VirtualMachine.Config.Annotation** **VirtualMachine.Config.CPUCount** **VirtualMachine.Config.DiskExtend** **VirtualMachine.Config.DiskLease** **VirtualMachine.Config.EditDevice** **VirtualMachine.Config.Memory** **VirtualMachine.Config.RemoveDisk** **VirtualMachine.Config.Rename** **VirtualMachine.Config.ResetGuestInfo** **VirtualMachine.Config.Resource** **VirtualMachine.Config.Settings** **VirtualMachine.Config.UpgradeVirtualHardware** **VirtualMachine.Interact.GuestControl** **VirtualMachine.Interact.PowerOff** **VirtualMachine.Interact.PowerOn** **VirtualMachine.Interact.Reset** **VirtualMachine.Inventory.Create** **VirtualMachine.Inventory.CreateFromExisting** **VirtualMachine.Inventory.Delete** **VirtualMachine.Provisioning.Clone** **VirtualMachine.Provisioning.DeployTemplate** **VirtualMachine.Provisioning.MarkAsTemplate** **Folder.Create** **Folder.Delete**

例 23.9. 在 vCenter 图形用户界面 (GUI) 中安装所需的角色和权限

适用于角色的 vSphere 对象	必要时	vCenter GUI 中所需的权限
vSphere vCenter	Always	**Cns.Searchable "vSphere Tagging"."Assign or Unassign vSphere Tag" "vSphere Tagging"."Create vSphere Tag Category" "vSphere Tagging"."Create vSphere Tag" vSphere Tagging"."Delete vSphere Tag Category" "vSphere Tagging"."Delete vSphere Tag" "vSphere Tagging"."Edit vSphere Tag Category" "vSphere Tagging"."Edit vSphere Tag" Sessions."Validate session" "Profile-driven storage"."Profile-driven storage update" "Profile-driven storage"."Profile-driven storage view"**
vSphere vCenter 集群	如果在集群 root 中创建虚拟机	**Host.Configuration."Storage partition configuration" Resource."Assign virtual machine to resource pool" VApp."Assign resource pool" VApp.Import "Virtual machine"."Change Configuration"."Add new disk"**
vSphere vCenter 资源池	如果提供了现有的资源池	**Host.Configuration."Storage partition configuration" Resource."Assign virtual machine to resource pool" VApp."Assign resource pool" VApp.Import "Virtual machine"."Change Configuration"."Add new disk"**

适用于角色的 vSphere 对象	必要时	vCenter GUI 中所需的权限
vSphere Datastore	Always	**Datastore."Allocate space" Datastore."Browse datastore" Datastore."Low level file operations" "vSphere Tagging"."Assign or Unassign vSphere Tag on Object"**
vSphere 端口组	Always	**Network."Assign network"**
虚拟机文件夹	Always	**"vSphere Tagging"."Assign or Unassign vSphere Tag on Object" Resource."Assign virtual machine to resource pool" VApp.Import "Virtual machine"."Change Configuration"."Add existing disk" "Virtual machine"."Change Configuration"."Add new disk" "Virtual machine"."Change Configuration"."Add or remove device" "Virtual machine"."Change Configuration"."Advanced configuration" "Virtual machine"."Change Configuration"."Set annotation" "Virtual machine"."Change Configuration"."Change CPU count" "Virtual machine"."Change Configuration"."Extend virtual disk" "Virtual machine"."Change Configuration"."Acquire disk lease" "Virtual machine"."Change Configuration"."Modify device settings" "Virtual machine"."Change Configuration"."Change Memory" "Virtual machine"."Change Configuration"."Remove disk" "Virtual machine"."Change Configuration".Rename "Virtual machine"."Change Configuration"."Reset guest information"**

适用于角色的 vSphere 对象	必要时	vCenter GUI 中所需的权限
		"Virtual machine"."Change Configuration"."Change resource" **"Virtual machine"."Change Configuration"."Change Settings"** **"Virtual machine"."Change Configuration"."Upgrade virtual machine compatibility"** **"Virtual machine".Interaction."Guest operating system management by VIX API"** **"Virtual machine".Interaction."Power off"** **"Virtual machine".Interaction."Power on"** **"Virtual machine".Interaction.Reset** **"Virtual machine"."Edit Inventory"."Create new"** **"Virtual machine"."Edit Inventory"."Create from existing"** **"Virtual machine"."Edit Inventory"."Remove"** **"Virtual machine".Provisioning."Clone virtual machine"** **"Virtual machine".Provisioning."Mark as template"** **"Virtual machine".Provisioning."Deploy template"**
vSphere vCenter Datacenter	如果安装程序创建虚拟机文件夹对于用户置备的基础架构，如果集群没有使用 Machine API，则 **VirtualMachine.Inventory.Create** 和 **VirtualMachine.Inventory.Delete** 权限是可选的。	**"vSphere Tagging"."Assign or Unassign vSphere Tag on Object"** **Resource."Assign virtual machine to resource pool"** **VApp.Import** **"Virtual machine"."Change Configuration"."Add existing disk"** **"Virtual machine"."Change Configuration"."Add new disk"** **"Virtual machine"."Change Configuration"."Add or remove device"** **"Virtual machine"."Change Configuration"."Advanced configuration"** **"Virtual machine"."Change Configuration"."Set annotation"**

适用于角色的 vSphere 对象	必要时	vCenter GUI 中所需的权限
		"Virtual machine"."Change Configuration"."Change CPU count" "Virtual machine"."Change Configuration"."Extend virtual disk" "Virtual machine"."Change Configuration"."Acquire disk lease" "Virtual machine"."Change Configuration"."Modify device settings" "Virtual machine"."Change Configuration"."Change Memory" "Virtual machine"."Change Configuration"."Remove disk" "Virtual machine"."Change Configuration".Rename "Virtual machine"."Change Configuration"."Reset guest information" "Virtual machine"."Change Configuration"."Change resource" "Virtual machine"."Change Configuration"."Change Settings" "Virtual machine"."Change Configuration"."Upgrade virtual machine compatibility" "Virtual machine".Interaction."Guest operating system management by VIX API" "Virtual machine".Interaction."Power off" "Virtual machine".Interaction."Power on" "Virtual machine".Interaction.Reset "Virtual machine"."Edit Inventory"."Create new" "Virtual machine"."Edit Inventory"."Create from existing" "Virtual machine"."Edit Inventory"."Remove" "Virtual machine".Provisioning."Clone virtual machine" "Virtual machine".Provisioning."Deploy template"

适用于角色的 vSphere 对象	必要时	vCenter GUI 中所需的权限
		"Virtual machine".Provisioning."Mark as template" Folder."Create folder" Folder."Delete folder"

此外，用户需要一些 ReadOnly 权限，一些角色需要相应的权限来将权限代理到子对象。这些设置会因您是否将集群安装到现有文件夹而有所不同。

例 23.10. 所需的权限和传播设置

vSphere object	必要时	传播到子对象	所需的权限
vSphere vCenter	Always	False	列出所需的权限
vSphere vCenter Datacenter	现有文件夹	False	只读 权限
	安装程序创建文件夹	True	列出所需的权限
vSphere vCenter 集群	现有资源池	False	只读 权限
	集群 root 中的虚拟机	True	列出所需的权限
vSphere vCenter 数据存储	Always	False	列出所需的权限
vSphere Switch	Always	False	只读 权限
vSphere 端口组	Always	False	列出所需的权限
vSphere vCenter 虚拟机文件夹	现有文件夹	True	列出所需的权限
vSphere vCenter 资源池	现有资源池	True	列出所需的权限

有关只使用所需权限创建帐户的更多信息，请参阅 vSphere 文档中的 vSphere 权限和用户管理任务。

最低所需的 vCenter 帐户权限

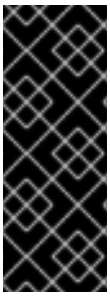
创建自定义角色并为其分配特权后，您可以通过选择特定的 vSphere 对象来创建权限，然后为每个对象将自定义角色分配给用户或组。

在为 vSphere 对象创建权限或请求创建权限前，请确定将什么最小权限应用到 vSphere 对象。通过执行此任务，您可以确保 vSphere 对象和 OpenShift Container Platform 架构之间存在基本交互。

重要

如果您创建一个自定义角色，且没有为其分配特权，vSphere 服务器默认会为自定义角色分配一个 Read Only 角色。请注意，对于云供应商 API，自定义角色只需要继承 Read Only 角色的权限。

当具有全局管理特权的帐户不满足您的需要时，请考虑创建自定义角色。



重要

不支持使用所需权限配置的帐户。在 vCenter 中安装 OpenShift Container Platform 集群会根据完整权限列表进行测试，如 "Required vCenter account privileges" 部分所述。通过遵循完整的特权列表，您可以降低创建具有受限特权的自定义角色时可能会出现意外行为的可能性。

下表列出了与特定 OpenShift Container Platform 架构交互的 vSphere 对象的最低权限。

例 23.11. 安装后管理组件的最小权限

适用于角色的 vSphere 对象	必要时	所需的权限
vSphere vCenter	Always	Cns.Searchable InventoryService.Tagging.AttachTag InventoryService.Tagging.CreateCategory InventoryService.Tagging.CreateTag InventoryService.Tagging.DeleteCategory InventoryService.Tagging.DeleteTag InventoryService.Tagging.EditCategory InventoryService.Tagging.EditTag Sessions.ValidateSession StorageProfile.Update StorageProfile.View

适用于角色的 vSphere 对象	必要时	所需的权限
vSphere vCenter 集群	如果要在集群 root 中创建虚拟机	**Host.Config.Storage** **Resource.AssignVMT** **oPool**
vSphere vCenter 资源池	如果您在 **install-config.yaml** 文件中提供现有的资源池	**Host.Config.Storage**
vSphere Datastore	Always	**Datastore.AllocateSpa** **ce** **Datastore.Browse** **Datastore.FileManage** **ment** **InventoryService.Tag** **ging.ObjectAttachable**
vSphere 端口组	Always	**Network.Assign**
虚拟机文件夹	Always	**VirtualMachine.Config** **.AddExistingDisk** **VirtualMachine.Config** **.AddRemoveDevice** **VirtualMachine.Config** **.AdvancedConfig** **VirtualMachine.Config** **.Annotation** **VirtualMachine.Config** **.CPUCount** **VirtualMachine.Config** **.DiskExtend** **VirtualMachine.Config** **.Memory** **VirtualMachine.Config** **.Settings** **VirtualMachine.Interac** **t.PowerOff** **VirtualMachine.Interac** **t.PowerOn** **VirtualMachine.Invent** **ory.CreateFromExisti** **ng** **VirtualMachine.Invent** **ory.Delete** **VirtualMachine.Provisi** **oning.Clone** **VirtualMachine.Provisi** **oning.DeployTemplat** **e**

适用于角色的 vSphere 对象	必要时	所需的权限
vSphere vCenter Datacenter	如果安装程序创建虚拟机文件夹对于用户置备的基础架构，如果集群没有使用 Machine API，则 **VirtualMachine.Inventory.Create** 和 **VirtualMachine.Inventory.Delete** 权限是可选的。如果您的集群使用 Machine API，并且要为 API 设置最小权限集，请参阅 Machine API 的"最小权限"表。	**Resource.AssignVMToPool VirtualMachine.Config.AddExistingDisk VirtualMachine.Config.AddRemoveDevice VirtualMachine.Interact.PowerOff VirtualMachine.Interact.PowerOn VirtualMachine.Provisioning.DeployTemplate**

例 23.12. 存储组件的最低权限

适用于角色的 vSphere 对象	必要时	所需的权限
vSphere vCenter	Always	**Cns.Searchable InventoryService.Tagging.CreateCategory InventoryService.Tagging.CreateTag InventoryService.Tagging.EditCategory InventoryService.Tagging.EditTag StorageProfile.Update StorageProfile.View**
vSphere vCenter 集群	如果要在集群 root 中创建虚拟机	**Host.Config.Storage**
vSphere vCenter 资源池	如果您在 **install-config.yaml** 文件中提供现有的资源池	**Host.Config.Storage**
vSphere Datastore	Always	**Datastore.Browse Datastore.FileManagement InventoryService.Tagging.ObjectAttachable**
vSphere 端口组	Always	**只读**
虚拟机文件夹	Always	**VirtualMachine.Config.AddExistingDisk VirtualMachine.Config.AddRemoveDevice**

适用于角色的 vSphere 对象	必要时	所需的权限
vSphere vCenter Datacenter	如果安装程序创建虚拟机文件夹对于用户置备的基础架构，如果集群没有使用 Machine API，则 **VirtualMachine.Inventory.Create** 和 **VirtualMachine.Inventory.Delete** 权限是可选的。如果您的集群使用 Machine API，并且要为 API 设置最小权限集，请参阅 Machine API 的"最小权限"表。	**VirtualMachine.Config.AddExistingDisk** **VirtualMachine.Config.AddRemoveDevice**

例 23.13. Machine API 的最低权限

适用于角色的 vSphere 对象	必要时	所需的权限
vSphere vCenter	Always	**InventoryService.Tagging.AttachTag** **InventoryService.Tagging.CreateCategory** **InventoryService.Tagging.CreateTag** **InventoryService.Tagging.DeleteCategory** **InventoryService.Tagging.DeleteTag** **InventoryService.Tagging.EditCategory** **InventoryService.Tagging.EditTag** **Sessions.ValidateSession** **StorageProfile.Update** **StorageProfile.View**
vSphere vCenter 集群	如果要在集群 root 中创建虚拟机	**Resource.AssignVMToPool**
vSphere vCenter 资源池	如果您在 **install-config.yaml** 文件中提供现有的资源池	只读
vSphere Datastore	Always	**Datastore.AllocateSpace** **Datastore.Browse**
vSphere 端口组	Always	**Network.Assign**

适用于角色的 vSphere 对象	必要时	所需的权限
虚拟机文件夹	Always	**VirtualMachine.Config.AddRemoveDevice** **VirtualMachine.Config.AdvancedConfig** **VirtualMachine.Config.Annotation** **VirtualMachine.Config.CPUCount** **VirtualMachine.Config.DiskExtend** **VirtualMachine.Config.Memory** **VirtualMachine.Config.Settings** **VirtualMachine.Interact.PowerOff** **VirtualMachine.Interact.PowerOn** **VirtualMachine.Inventory.CreateFromExisting** **VirtualMachine.Inventory.Delete** **VirtualMachine.Provisioning.Clone** **VirtualMachine.Provisioning.DeployTemplate**
vSphere vCenter Datacenter	如果安装程序创建虚拟机文件夹对于用户置备的基础架构，如果集群没有使用 Machine API，则 **VirtualMachine.Inventory.Create** 和 **VirtualMachine.Inventory.Delete** 权限是可选的。	**Resource.AssignVMToPool** **VirtualMachine.Interact.PowerOff** **VirtualMachine.Interact.PowerOn** **VirtualMachine.Provisioning.DeployTemplate**

将 OpenShift Container Platform 与 vMotion 搭配使用

如果要在 vSphere 环境中使用 vMotion，请在安装 OpenShift Container Platform 集群前考虑以下内容。

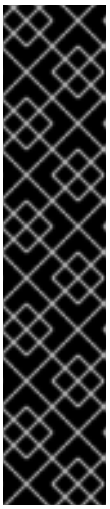
- **OpenShift Container Platform 通常支持 compute-only vMotion，其中*通常*意味着您满足所有 VMware 最佳实践进行 vMotion。**

为了帮助确保计算和 control plane 节点的正常运行时间，请确保遵循 VMware 最佳实践进行 vMotion，并使用 VMware 反关联性规则提高 OpenShift Container Platform 在维护或硬件问题期间的可用性。

有关 vMotion 和 anti-affinity 规则的更多信息，请参阅 VMware vSphere 文档以了解 vMotion 网络要求和虚拟机反关联性规则。

- 使用 Storage vMotion 可能会导致问题且不受支持。如果您在 pod 中使用 vSphere 卷，请手动或通过 Storage vMotion 在数据存储间迁移虚拟机，这会导致 OpenShift Container Platform 持久性卷(PV)对象中的无效引用，这可能会导致数据丢失。

- OpenShift Container Platform 不支持在数据存储间有选择地迁移 VMDK，使用数据存储集群进行虚拟机置备或动态或静态置备 PV，或使用作为数据存储集群一部分的数据存储来动态或静态置备 PV。



重要

您可以指定数据存储集群中存在的任何数据存储路径。默认情况下，使用 Storage vMotion 的存储分布式资源调度程序(SDRS)会自动为数据存储集群启用。红帽不支持 Storage vMotion，因此您必须禁用 Storage DRS 以避免 OpenShift Container Platform 集群的数据丢失问题。

如果需要在多个数据存储间指定虚拟机，请使用 数据存储 对象在集群 install-config.yaml 配置文件中指定故障域。如需更多信息，请参阅"VMware vSphere 区域和区启用"。

集群资源

当您部署使用您提供的基础架构的 OpenShift Container Platform 集群时，您必须在 vCenter 实例中创建以下资源：

- 1 个文件夹

- 1 标签类别

- 1 标签

- 虚拟机：
  - 1 个模板
  - 1 个临时 bootstrap 节点
  - 3 个 control plane 节点
  - 3 个计算机器

虽然这些资源使用 856 GB 存储，但 bootstrap 节点会在集群安装过程中销毁。使用标准集群至少需要 800 GB 存储。

如果部署更多计算机器，OpenShift Container Platform 集群将使用更多存储。

集群限制

可用资源因集群而异。vCenter 中可能的集群数量主要受可用存储空间以及对所需资源数量的限制。确保考虑集群创建的 vCenter 资源的限制和部署集群所需的资源，如 IP 地址和网络。

网络要求

对网络使用动态主机配置协议(DHCP)，并确保 DHCP 服务器配置为为集群机器提供持久的 IP 地址。

注意

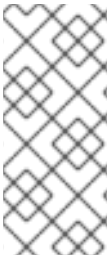
如果要使用静态 IP 地址置备节点，则不需要将 DHCP 用于网络。

将默认网关配置为使用 DHCP 服务器。所有节点必须位于同一 VLAN 中。您不能将第二 VLAN 用作第 2 天操作来缩放集群。

您必须为网络使用动态主机配置协议 (DHCP)，并确保 DHCP 服务器被配置为为集群机器提供持久的 IP 地址。在 DHCP 租期中，您必须将 DHCP 配置为使用默认网关。所有节点必须位于同一 VLAN 中。您不能将第二 VLAN 用作第 2 天操作来缩放集群。

如果要安装到受限环境中，受限网络中的虚拟机必须有权访问 vCenter，以便它可以置备和管理节点、持久性卷声明 (PVC) 和其他资源。

另外，在安装 OpenShift Container Platform 集群前，您必须创建以下网络资源：



注意

建议集群中的每个 OpenShift Container Platform 节点都必须有权访问可通过 DHCP 发现的网络时间协议(NTP)服务器。没有 NTP 服务器即可安装。但是，异步服务器时钟将导致错误，NTP 服务器会阻止。

DNS 记录

您必须在适当的 DNS 服务器中为托管 OpenShift Container Platform 集群的 vCenter 实例创建两个静态 IP 地址的 DNS 记录。在每个记录中，<cluster_name> 是集群名称，<base_domain> 是您 在安装集群时指定的集群基域。完整的 DNS 记录采用以下形式：<component>.<cluster_name>.<base_domain>.。

表 23.20. 所需的 DNS 记录

组件	记录	描述
API VIP	api.<cluster_name>.<base_domain>.	此 DNS A/AAAA 或 CNAME (Canonical Name) 记录必须指向 control plane 机器的负载均衡器。此记录必须由集群外的客户端和集群中的所有节点解析。
Ingress VIP	*.apps.<cluster_name>.<base_domain>.	通配符 DNS A/AAAA 或 CNAME 记录，指向以运行入口路由器 Pod 的机器（默认为 worker 节点）为目标的负载均衡器。此记录必须由集群外的客户端和集群中的所有节点解析。

其他资源

- 在 vSphere 上创建计算机器设置

23.3.1.3.2. 集群安装所需的机器

最小的 OpenShift Container Platform 集群需要以下主机：

表 23.21. 最低所需的主机

主机	描述
一个临时 bootstrap 机器	集群需要 bootstrap 机器在三台 control plane 机器上部署 OpenShift Container Platform 集群。您可在安装集群后删除 bootstrap 机器。
三台 control plane 机器	control plane 机器运行组成 control plane 的 Kubernetes 和 OpenShift Container Platform 服务。
至少两台计算机器，也称为 worker 机器。	OpenShift Container Platform 用户请求的工作负载在计算机器上运行。

重要

要保持集群的高可用性，请将独立的物理主机用于这些集群机器。

bootstrap 和 control plane 机器必须使用 Red Hat Enterprise Linux CoreOS(RHCOS)作为操作系统。但是，计算机器可以在 Red Hat Enterprise Linux CoreOS(RHCOS)、Red Hat Enterprise Linux(RHEL) 8.6 和更高的版本。

请注意，RHCOS 基于 Red Hat Enterprise Linux(RHEL) 9.2，并继承其所有硬件认证和要求。查看红帽企业 Linux 技术功能和限制。

23.3.1.3.3. 集群安装的最低资源要求

每台集群机器都必须满足以下最低要求：

表 23.22. 最低资源要求

机器	操作系统	vCPU	虚拟内存	Storage	输入/输出每秒(IOPS)[1]
bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane（控制平面）	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS、RHEL 8.6 及更新版本 [2]	2	8 GB	100 GB	300

1. OpenShift Container Platform 和 Kubernetes 对磁盘性能非常敏感，建议使用更快的存储速度，特别是 control plane 节点上需要 10 ms p99 fsync 持续时间的 etcd。请注意，在许多云平台上，存储大小和 IOPS 可一起扩展，因此您可能需要过度分配存储卷来获取足够的性能。

2. 与所有用户置备的安装一样，如果您选择在集群中使用 RHEL 计算机器，则负责所有操作系统生命周期管理和维护，包括执行系统更新、应用补丁和完成所有其他必要的任务。RHEL 7 计算机器的使用已弃用，并已在 OpenShift Container Platform 4.10 及更新的版本中删除。

注意

从 OpenShift Container Platform 版本 4.13 开始，RHCOS 基于 RHEL 版本 9.2，它更新了微架构要求。以下列表包含每个架构需要的最小指令集架构 (ISA)：

- x86-64 体系结构需要 x86-64-v2 ISA
- ARM64 架构需要 ARMv8.0-A ISA
- IBM Power 架构需要 Power 9 ISA
- s390x 架构需要 z14 ISA

如需更多信息，请参阅 RHEL 架构。

如果平台的实例类型满足集群机器的最低要求，则 OpenShift Container Platform 支持使用它。

其他资源

- 优化存储

23.3.1.3.4. 加密虚拟机的要求

您可以通过满足以下要求，在安装 OpenShift Container Platform 4.16 前对虚拟机进行加密。

- 您已在 vSphere 中配置了标准密钥供应商。如需更多信息，请参阅在 vCenter 服务器中添加 KMS。

重要

不支持 vCenter 中的原生密钥供应商。如需更多信息，请参阅 vSphere 原生密钥提供程序概述。

- 您已在托管集群的所有 ESXi 主机上启用了主机加密模式。如需更多信息，请参阅启用主机加密模式。

- 您有一个启用了所有加密权限的 vSphere 帐户。如需更多信息，请参阅 Cryptographic Operations Privileges。

当您在标题为"安装 RHCOS 并启动 OpenShift Container Platform bootstrap 过程"的部分中部署 OVF 模板时，在为 OVF 模板选择存储时选择 "Encrypt this virtual virtual machine"。完成集群安装后，创建一个使用您用来加密虚拟机的加密存储策略的存储类。

其他资源

- 创建加密的存储类

23.3.1.3.5. 证书签名请求管理

在使用您置备的基础架构时，集群只能有限地访问自动机器管理，因此您必须提供一种在安装后批准集群证书签名请求 (CSR) 的机制。kube-controller-manager 只能批准 kubelet 客户端 CSR。machine-approver 无法保证使用 kubelet 凭证请求的提供证书的有效性，因为它不能确认是正确的机器发出了该请求。您必须决定并实施一种方法，以验证 kubelet 提供证书请求的有效性并进行批准。

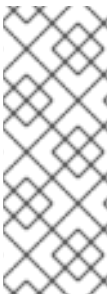
23.3.1.3.6. 用户置备的基础架构对网络的要求

所有 Red Hat Enterprise Linux CoreOS(RHCOS)机器都需要在启动时在 initramfs 中配置联网，以获取它们的 Ignition 配置文件。

在初次启动过程中，机器需要 IP 地址配置，该配置通过 DHCP 服务器或静态设置，提供所需的引导选项。建立网络连接后，机器会从 HTTP 或 HTTPS 服务器下载 Ignition 配置文件。然后，Ignition 配置

文件用于设置每台机器的确切状态。Machine Config Operator 在安装后完成对机器的更多更改，如应用新证书或密钥。

建议使用 DHCP 服务器对集群机器进行长期管理。确保 DHCP 服务器已配置为向集群机器提供持久的 IP 地址、DNS 服务器信息和主机名。



注意

如果用户置备的基础架构没有 DHCP 服务，您可以在 RHCOS 安装时向节点提供 IP 网络配置和 DNS 服务器地址。如果要从 ISO 镜像安装，这些参数可作为引导参数传递。如需有关静态 IP 置备和高级网络选项的更多信息，请参阅 *安装 RHCOS 并启动 OpenShift Container Platform bootstrap 过程* 部分。

Kubernetes API 服务器必须能够解析集群机器的节点名称。如果 API 服务器和 worker 节点位于不同的区域中，您可以配置默认 DNS 搜索区域，以允许 API 服务器解析节点名称。另一种支持的方法是始终通过节点对象和所有 DNS 请求中的完全限定域名引用主机。

23.3.1.3.6.1. 通过 DHCP 设置集群节点主机名

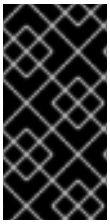
在 Red Hat Enterprise Linux CoreOS(RHCOS)机器上，主机名是通过 NetworkManager 设置的。默认情况下，机器通过 DHCP 获取其主机名。如果主机名不是由 DHCP 提供，请通过内核参数或者其它方法进行静态设置，请通过反向 DNS 查找获取。反向 DNS 查找在网络初始化后进行，可能需要一些时间来解决。其他系统服务可以在此之前启动，并将主机名检测为 localhost 或类似的内容。您可以使用 DHCP 为每个集群节点提供主机名来避免这种情况。

另外，通过 DHCP 设置主机名可以绕过实施 DNS split-horizon 的环境中的手动 DNS 记录名称配置错误。

23.3.1.3.6.2. 网络连接要求

您必须配置机器之间的网络连接，以允许 OpenShift Container Platform 集群组件进行通信。每台机器都必须能够解析集群中所有其他机器的主机名。

本节详细介绍了所需的端口。



重要

在连接的 OpenShift Container Platform 环境中，所有节点都需要访问互联网才能为平台容器拉取镜像，并向红帽提供遥测数据。

表 23.23. 用于全机器到所有机器通信的端口

协议	port	描述
ICMP	N/A	网络可访问性测试
TCP	**1936**	指标
	9000-**9999**	主机级别的服务，包括端口 9 **100**-**9101** 上的节点导出器，以及端口 **9099** 上的 Cluster Version Operator。
	10250-**10259**	Kubernetes 保留的默认端口
UDP	**4789**	VXLAN
	6081	Geneve
	9000-**9999**	主机级别的服务，包括端口 **9100**到**9101** 上的节点导出器。
	500	IPsec IKE 数据包
	4500	IPsec NAT-T 数据包
	123	UDP 端口 **123** 上的网络时间协议 (NTP) 如果配置了外部 NTP 时间服务器，需要打开 UDP 端口 **123**。
TCP/UDP	**30000**-**32767**	Kubernetes 节点端口
ESP	N/A	IPsec Encapsulating Security Payload(ESP)

表 23.24. 用于所有机器控制平面通信的端口

协议	port	描述
TCP	**6443**	Kubernetes API

表 23.25. control plane 机器用于 control plane 机器通信的端口

协议	port	描述
TCP	**2379-2380**	etcd 服务器和对等端口

以太网适配器硬件地址要求

当为集群置备虚拟机时，为每个虚拟机配置的以太网接口必须使用 VMware 机构唯一识别符(OUI)分配范围中的 MAC 地址：

- **00:05:69:00:00:00 到 00:05:69:FF:FF:FF**
- **00:0c:29:00:00:00 到 00:0c:29:FF:FF:FF**
- **00:1C:14:00:00:00 到 00:1c:14:FF:FF:FF**
- **00:50:56:00:00:00 到 00:50:56:3F:FF:FF**

如果使用 VMware OUI 以外的 MAC 地址，集群安装将无法成功。

用户置备的基础架构的 NTP 配置

OpenShift Container Platform 集群被配置为默认使用公共网络时间协议(NTP)服务器。如果要使用本地企业 NTP 服务器，或者集群部署在断开连接的网络中，您可以将集群配置为使用特定的时间服务器。如需更多信息，*请参阅配置 chrony 时间服务* 的文档。

如果 DHCP 服务器提供 NTP 服务器信息，Red Hat Enterprise Linux CoreOS(RHCOS)机器上的 chrony 时间服务会读取信息，并可以把时钟与 NTP 服务器同步。

其他资源

- **配置 chrony 时间服务**

23.3.1.3.7. 用户置备的 DNS 要求

在 OpenShift Container Platform 部署中，以下组件需要 DNS 名称解析：

- The Kubernetes API
- OpenShift Container Platform 应用程序通配符
- bootstrap、control plane 和计算机器

Kubernetes API、bootstrap 机器、control plane 机器和计算机器也需要反向 DNS 解析。

DNS A/AAAA 或 CNAME 记录用于名称解析，PTR 记录用于反向名称解析。反向记录很重要，因为 Red Hat Enterprise Linux CoreOS(RHCOS)使用反向记录为所有节点设置主机名，除非 DHCP 提供主机名。另外，反向记录用于生成 OpenShift Container Platform 需要操作的证书签名请求(CSR)。

注意

建议使用 DHCP 服务器为每个群集节点提供主机名。如需更多信息，*请参阅用户置备的基础架构部分的 DHCP 建议*。

用户置备的 OpenShift Container Platform 集群需要以下 DNS 记录，这些记录必须在安装前就位。在每个记录中，<cluster_name> 是集群名称，<base_domain> 是您在 install-config.yaml 文件中指定的基域。完整的 DNS 记录采用以下形式： <component>.<cluster_name>.<base_domain>.。

表 23.26. 所需的 DNS 记录

组件	记录	描述
Kubernetes API	**api.<cluster_name>.<base_domain>.**	DNS A/AAAA 或 CNAME 记录，以及用于标识 API 负载均衡器的 DNS PTR 记录。这些记录必须由集群外的客户端和集群中的所有节点解析。
	api-int.<cluster_name>.<base_domain>.	DNS A/AAAA 或 CNAME 记录，以及用于内部标识 API 负载均衡器的 DNS PTR 记录。这些记录必须可以从集群中的所有节点解析。  **重要** API 服务器必须能够根据 Kubernetes 中记录的主机名解析 worker 节点。如果 API 服务器无法解析节点名称，则代理的 API 调用会失败，且您无法从 pod 检索日志。

组件	记录	描述
Routes	***.apps.<cluster_name>. <base_domain>.**	通配符 DNS A/AAAA 或 CNAME 记录，指向应用程序入口负载均衡器。应用程序入口负载均衡器以运行 Ingress Controller Pod 的机器为目标。默认情况下，Ingress Controller Pod 在计算机器上运行。这些记录必须由集群外的客户端和集群中的所有节点解析。 例如，console **-openshift-console.apps. <cluster_name>.<base_domain>** 用作到 OpenShift Container Platform 控制台的通配符路由。
bootstrap 机器	**bootstrap.<cluster_name>. <base_domain>.**	DNS A/AAAA 或 CNAME 记录，以及用于标识 bootstrap 机器的 DNS PTR 记录。这些记录必须由集群中的节点解析。
control plane 机器	**<control_plane><n>. <cluster_name>. <base_domain>.**	DNS A/AAAA 或 CNAME 记录，以识别 control plane 节点的每台机器。这些记录必须由集群中的节点解析。
计算机器	**<compute><n>. <cluster_name>. <base_domain>.**	DNS A/AAAA 或 CNAME 记录，用于识别 worker 节点的每台机器。这些记录必须由集群中的节点解析。

注意

在 OpenShift Container Platform 4.4 及更新的版本中，您不需要在 DNS 配置中指定 etcd 主机和 SRV 记录。

提示

您可以使用 dig 命令验证名称和反向名称解析。如需了解详细的 *验证步骤，请参阅为用户置备的基础架构验证 DNS 解析* 一节。

23.3.1.3.7.1. 用户置备的集群的 DNS 配置示例

本节提供 A 和 PTR 记录配置示例，它们满足了在用户置备的基础架构上部署 OpenShift Container Platform 的 DNS 要求。样本不是为选择一个 DNS 解决方案提供建议。

在这个示例中，集群名称为 ocp4，基域是 example.com。

用户置备的集群的 DNS A 记录配置示例

以下示例是 BIND 区域文件，其中显示了用户置备的集群中名称解析的 A 记录示例。

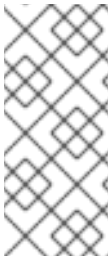
例 23.14. DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
  2019070700 ; serial
  3H  ; refresh (3 hours)
  30M  ; retry (30 minutes)
  2W  ; expiry (2 weeks)
  1W )  ; minimum (1 week)
 IN NS ns1.example.com.
 IN MX 10 smtp.example.com.
;
;
ns1.example.com.  IN A 192.168.1.5
smtp.example.com.  IN A 192.168.1.5
;
helper.example.com.  IN A 192.168.1.5
helper.ocp4.example.com. IN A 192.168.1.5
;
api.ocp4.example.com.  IN A 192.168.1.5 1
api-int.ocp4.example.com. IN A 192.168.1.5 2
;
*.apps.ocp4.example.com. IN A 192.168.1.5 3
;
bootstrap.ocp4.example.com. IN A 192.168.1.96 4
;
control-plane0.ocp4.example.com. IN A 192.168.1.97 5
control-plane1.ocp4.example.com. IN A 192.168.1.98 6
control-plane2.ocp4.example.com. IN A 192.168.1.99 7
;
compute0.ocp4.example.com. IN A 192.168.1.11 8
compute1.ocp4.example.com. IN A 192.168.1.7 9
;
;EOF
```

1 为 Kubernetes API 提供名称解析。记录引用 API 负载均衡器的 IP 地址。

2 为 Kubernetes API 提供名称解析。记录引用 API 负载均衡器的 IP 地址，用于内部集群通信。

3 为通配符路由提供名称解析。记录引用应用程序入口负载均衡器的 IP 地址。应用程序入口负载均衡器以运行 Ingress Controller Pod 的机器为目标。默认情况下，Ingress Controller Pod 在计算机器上运行。



注意

在这个示例中，将相同的负载均衡器用于 Kubernetes API 和应用入口流量。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。

4

为 bootstrap 机器提供名称解析。

5 6 7

为 control plane 机器提供名称解析。

8 9

为计算机器提供名称解析。

用户置备的集群的 DNS PTR 记录配置示例

以下示例 BIND 区域文件显示了用户置备的集群中反向名称解析的 PTR 记录示例。

例 23.15. 反向记录的 DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
  2019070700 ; serial
  3H  ; refresh (3 hours)
  30M  ; retry (30 minutes)
  2W  ; expiry (2 weeks)
  1W ) ; minimum (1 week)
 IN NS ns1.example.com.
;
5.1.168.192.in-addr.arpa. IN PTR api.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. IN PTR api-int.ocp4.example.com. 2
;
96.1.168.192.in-addr.arpa. IN PTR bootstrap.ocp4.example.com. 3
;
97.1.168.192.in-addr.arpa. IN PTR control-plane0.ocp4.example.com. 4
98.1.168.192.in-addr.arpa. IN PTR control-plane1.ocp4.example.com. 5
99.1.168.192.in-addr.arpa. IN PTR control-plane2.ocp4.example.com. 6
;
11.1.168.192.in-addr.arpa. IN PTR compute0.ocp4.example.com. 7
```

```
7.1.168.192.in-addr.arpa. IN PTR compute1.ocp4.example.com. 8
;
;EOF
```

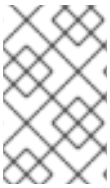
1 为 Kubernetes API 提供反向 DNS 解析。PTR 记录引用 API 负载均衡器的记录名称。

2 为 Kubernetes API 提供反向 DNS 解析。PTR 记录引用 API 负载均衡器的记录名称，用于内部集群通信。

3 为 bootstrap 机器提供反向 DNS 解析。

4 5 6 为 control plane 机器提供反向 DNS 解析。

7 8 为计算机器提供反向 DNS 解析。

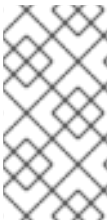


注意

OpenShift Container Platform 应用程序通配符不需要 PTR 记录。

23.3.1.3.8. 用户置备的基础架构的负载均衡要求

在安装 OpenShift Container Platform 前，您必须置备 API 和应用程序入口负载均衡基础架构。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。



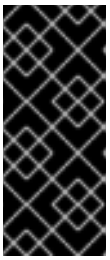
注意

如果要使用 Red Hat Enterprise Linux (RHEL) 实例部署 API 和应用程序入口负载均衡器，您必须单独购买 RHEL 订阅。

负载平衡基础架构必须满足以下要求：

1. API 负载均衡器 ：提供一个通用端点，供用户（包括人工和机器）与平台交互和配置。配置以下条件：

   - 仅第 4 层负载均衡.这可被称为 Raw TCP 或 SSL Passthrough 模式。
   - 无状态负载平衡算法。这些选项根据负载均衡器的实施而有所不同。

   重要

   不要为 API 负载均衡器配置会话持久性。为 Kubernetes API 服务器配置会话持久性可能会导致出现过量 OpenShift Container Platform 集群应用程序流量，以及过量的在集群中运行的 Kubernetes API。

   在负载均衡器的前端和后端配置以下端口：

   表 23.27. API 负载均衡器

port	后端机器（池成员）	internal	外部	描述
6443	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。您必须为 API 服务器健康检查探测配置 /readyz 端点。	X	X	Kubernetes API 服务器
22623	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。	X		机器配置服务器

   注意

   负载均衡器必须配置为，从 API 服务器关闭 /readyz 端点到从池中移除 API 服务器实例时最多需要 30 秒。在 /readyz 返回错误或健康后的时间范围内，端点必须被删除或添加。每 5 秒或 10 秒探测一次，有两个成功请求处于健康状态，三个成为不健康的请求是经过良好测试的值。

2.

应用程序入口负载均衡器 ：为应用程序流量从集群外部流提供入口点。OpenShift Container Platform 集群需要正确配置入口路由器。

配置以下条件：

- 仅第 4 层负载均衡.这可被称为 Raw TCP 或 SSL Passthrough 模式。
- 建议根据可用选项以及平台上托管的应用程序类型，使用基于连接的或基于会话的持久性。

提示

如果应用程序入口负载均衡器可以看到客户端的真实 IP 地址，启用基于 IP 的会话持久性可以提高使用端到端 TLS 加密的应用程序的性能。

在负载均衡器的前端和后端配置以下端口：

表 23.28. 应用程序入口负载均衡器

port	后端机器（池成员）	internal	外部	描述
443	默认情况下，运行 Ingress Controller Pod、计算或 worker 的机器。	X	X	HTTPS 流量
80	默认情况下，运行 Ingress Controller Pod、计算或 worker 的机器。	X	X	HTTP 流量

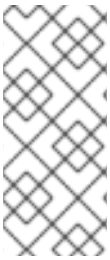
注意

如果要部署一个带有零计算节点的三节点集群，Ingress Controller Pod 在 control plane 节点上运行。在三节点集群部署中，您必须配置应用程序入口负载均衡器，将 HTTP 和 HTTPS 流量路由到 control plane 节点。

23.3.1.3.8.1. 用户置备的集群的负载均衡器配置示例

本节提供了一个满足用户置备集群的负载均衡要求的 API 和应用程序入口负载均衡器配置示例。示例是 HAProxy 负载均衡器的 /etc/haproxy/haproxy.cfg 配置。这个示例不是为选择一个负载平衡解决方案提供建议。

在这个示例中，将相同的负载均衡器用于 Kubernetes API 和应用入口流量。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。



注意

如果您使用 HAProxy 作为负载均衡器，并且 SELinux 设置为 enforcing，您必须通过运行 setsebool -P haproxy_connect_any=1 来确保 HAProxy 服务可以绑定到配置的 TCP 端口。

例 23.16. API 和应用程序入口负载均衡器配置示例

```
global
  log         127.0.0.1 local2
  pidfile     /var/run/haproxy.pid
  maxconn     4000
  daemon
defaults
  mode                    http
  log                     global
  option                  dontlognull
  option http-server-close
  option                  redispatch
  retries                 3
  timeout http-request    10s
  timeout queue           1m
  timeout connect         10s
  timeout client          1m
  timeout server          1m
  timeout http-keep-alive 10s
  timeout check           10s
  maxconn                 3000
listen api-server-6443 1
  bind *:6443
  mode tcp
  option  httpchk GET /readyz HTTP/1.0
  option  log-health-checks
  balance roundrobin
  server bootstrap bootstrap.ocp4.example.com:6443 verify none check check-ssl inter 10s
fall 2 rise 3 backup 2
  server master0 master0.ocp4.example.com:6443 weight 1 verify none check check-ssl
inter 10s fall 2 rise 3
  server master1 master1.ocp4.example.com:6443 weight 1 verify none check check-ssl
inter 10s fall 2 rise 3
  server master2 master2.ocp4.example.com:6443 weight 1 verify none check check-ssl
inter 10s fall 2 rise 3
listen machine-config-server-22623 3
  bind *:22623
  mode tcp
  server bootstrap bootstrap.ocp4.example.com:22623 check inter 1s backup 4
  server master0 master0.ocp4.example.com:22623 check inter 1s
```

```
  server master1 master1.ocp4.example.com:22623 check inter 1s
  server master2 master2.ocp4.example.com:22623 check inter 1s
listen ingress-router-443 5
  bind *:443
  mode tcp
  balance source
  server compute0 compute0.ocp4.example.com:443 check inter 1s
  server compute1 compute1.ocp4.example.com:443 check inter 1s
listen ingress-router-80 6
  bind *:80
  mode tcp
  balance source
  server compute0 compute0.ocp4.example.com:80 check inter 1s
  server compute1 compute1.ocp4.example.com:80 check inter 1s
```

1

端口 6443 处理 Kubernetes API 流量并指向 control plane 机器。

2 4

bootstrap 条目必须在 OpenShift Container Platform 集群安装前就位，且必须在 bootstrap 过程完成后删除它们。

3

端口 22623 处理机器配置服务器流量并指向 control plane 机器。

5

端口 443 处理 HTTPS 流量，并指向运行 Ingress Controller pod 的机器。默认情况下，Ingress Controller Pod 在计算机器上运行。

6

端口 80 处理 HTTP 流量，并指向运行 Ingress Controller pod 的机器。默认情况下，Ingress Controller Pod 在计算机器上运行。

注意

如果要部署一个带有零计算节点的三节点集群，Ingress Controller Pod 在 control plane 节点上运行。在三节点集群部署中，您必须配置应用程序入口负载均衡器，将 HTTP 和 HTTPS 流量路由到 control plane 节点。

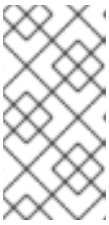
提示

如果您使用 HAProxy 作为负载均衡器，您可以通过在 HAProxy 节点上运行 netstat -nltupe 来检查 haproxy 进程是否在侦听端口 6443、22623、443 和 80。

23.3.2. 准备使用用户置备的基础架构安装集群

您可以通过完成以下步骤，准备在 vSphere 上安装 OpenShift Container Platform 集群：

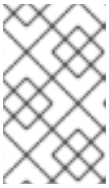
- 下载安装程序。



注意

如果您要在断开连接的环境中安装，您可以从镜像内容中提取安装程序。如需更多信息，请参阅为断开连接的安装镜像镜像。

- 安装 OpenShift CLI (oc)。



注意

如果要在断开连接的环境中安装，请将 oc 安装到镜像主机上。

- 生成 SSH 密钥对。您可以在部署后使用此密钥对在 OpenShift Container Platform 集群的节点上进行身份验证。

- 准备用户置备的基础架构。

- 验证 DNS 解析。

23.3.2.1. 获取安装程序

在安装 OpenShift Container Platform 前，将安装文件下载到您用于安装的主机上。

先决条件

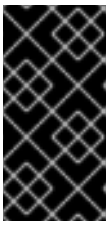
- 您有一台运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB。

流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐户，请使用您的凭证登录。如果没有，请创建一个帐户。

2. 选择您的基础架构供应商。

3. 进入到安装类型的页面，下载与您的主机操作系统和架构对应的安装程序，并将该文件放在您要存储安装配置文件的目录中。



重要

安装程序会在用来安装集群的计算机上创建几个文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。

重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，请为特定云供应商完成 OpenShift Container Platform 卸载流程。

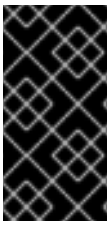
4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret。此 pull secret 允许您与所含授权机构提供的服务进行身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

23.3.2.2. 安装 OpenShift CLI

您可以安装 OpenShift CLI(oc)来使用命令行界面与 OpenShift Container Platform 进行交互。您可以在 Linux、Windows 或 macOS 上安装 oc。



重要

如果安装了旧版本的 oc，则可能无法使用 OpenShift Container Platform 4.16 中的所有命令。下载并安装新版本的 oc。

在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI(oc)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 产品变体 下拉列表中选择架构。
3. 从 版本 下拉列表中选择适当的版本。
4. 点 OpenShift v4.16 Linux Client 条目旁的 Download Now 来保存文件。
5. 解包存档：

```
$ tar xvf <file>
```

6. 将 oc 二进制文件放到 PATH 中的目录中。

   要查看您的 PATH，请执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 oc 命令：

```
$ oc <command>
```

在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI(oc)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 版本 下拉列表中选择适当的版本。
3. 点 OpenShift v4.16 Windows Client 条目旁的 Download Now 来保存文件。
4. 使用 ZIP 程序解压存档。
5. 将 oc 二进制文件移到 PATH 中的目录中。

   要查看您的 PATH，请打开命令提示并执行以下命令：

   ```
   C:\> path
   ```

验证

- 安装 OpenShift CLI 后，可以使用 oc 命令：

  ```
  C:\> oc <command>
  ```

在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI(oc)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 版本 下拉列表中选择适当的版本。

3. 点 OpenShift v4.16 macOS Client 条目旁的 Download Now 来保存文件。

注意

对于 macOS arm64，请选择 OpenShift v4.16 macOS arm64 Client 条目。

4. 解包和解压存档。

5. 将 oc 二进制文件移到 PATH 的目录中。

要查看您的 PATH，请打开终端并执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 oc 命令：

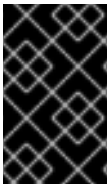
```
$ oc <command>
```

23.3.2.3. 为集群节点 SSH 访问生成密钥对

在 OpenShift Container Platform 安装过程中，您可以为安装程序提供 SSH 公钥。密钥通过它们的 Ignition 配置文件传递给 Red Hat Enterprise Linux CoreOS(RHCOS)节点，用于验证对节点的 SSH 访问。密钥添加到每个节点上 core 用户的 ~/.ssh/authorized_keys 列表中，这将启用免密码身份验证。

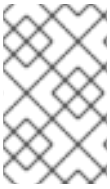
将密钥传递给节点后，您可以使用密钥对作为用户 核心 通过 SSH 连接到 RHCOS 节点。若要通过 SSH 访问节点，必须由 SSH 为您的本地用户管理私钥身份。

如果要通过 SSH 连接到集群节点来执行安装调试或灾难恢复，则必须在安装过程中提供 SSH 公钥。./openshift-install gather 命令还需要在集群节点上设置 SSH 公钥。



重要

不要在生产环境中跳过这个过程，在生产环境中需要灾难恢复和调试。



注意

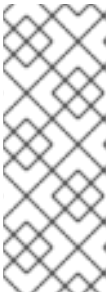
您必须使用本地密钥，而不是使用特定平台方法配置 的密钥，如 AWS 密钥对。

流程

1. 如果您在本地计算机上没有可用于在集群节点上进行身份验证的现有 SSH 密钥对，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 ~/.ssh/id_ed25519。如果您已有密钥对，请确保您的公钥位于 ~/.ssh 目录中。



注意

如果您计划在 x86_64、ppc64le 和 s390x 架构上安装使用 RHEL 加密库（这些加密库已提交给 NIST 用于 FIPS 140-2/140-3 验证）的 OpenShift Container Platform 集群，则不要创建使用 ed25519 算法的密钥。相反，创建一个使用 rsa 或 ecdsa 算法的密钥。

2. 查看公共 SSH 密钥：

```
$ cat <path>/<file_name>.pub
```

例如，运行以下命令来查看 ~/.ssh/id_ed25519.pub 公钥：

```
$ cat ~/.ssh/id_ed25519.pub
```

3. 将 SSH 私钥身份添加到本地用户的 SSH 代理（如果尚未添加）。在集群节点上，或者要使用 ./openshift-install gather 命令，需要对该密钥进行 SSH 代理管理，才能在集群节点上进行

免密码 SSH 身份验证。

注意

在某些发行版中，自动管理默认 SSH 私钥身份，如 ~/.ssh/id_rsa 和 ~/.ssh/id_dsa。

a. 如果 ssh-agent 进程尚未为您的本地用户运行，请将其作为后台任务启动：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```

注意

如果集群处于 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

4. 将 SSH 私钥添加到 ssh-agent ：

```
$ ssh-add <path>/<file_name> 1
```

1 指定 SSH 私钥的路径和文件名，如 ~/.ssh/id_ed25519.pub

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

后续步骤

- 安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。如果在您置备的基础架构上安装集群，则必须为安装程序提供密钥。

23.3.2.4. 准备用户置备的基础架构

在用户置备的基础架构上安装 OpenShift Container Platform 之前，您必须准备底层基础架构。

本节详细介绍了设置集群基础架构以准备 OpenShift Container Platform 安装所需的高级别步骤。这包括为您的集群节点配置 IP 网络和网络连接，通过防火墙启用所需的端口，以及设置所需的 DNS 和负载均衡基础架构。

准备后，集群基础架构必须满足 *带有用户置备的基础架构部分的集群要求*。

先决条件

- 您已参阅 OpenShift Container Platform 4.x Tested Integrations 页面。
- 您已查看了 *具有用户置备基础架构的集群要求部分中详述的基础架构* 要求。

流程

1. 如果您使用 DHCP 向集群节点提供 IP 网络配置，请配置 DHCP 服务。
   a. 将节点的持久 IP 地址添加到您的 DHCP 服务器配置。在您的配置中，将相关网络接口的 MAC 地址与每个节点的预期 IP 地址匹配。
   b. 当您使用 DHCP 为集群机器配置 IP 寻址时，机器还通过 DHCP 获取 DNS 服务器信息。定义集群节点通过 DHCP 服务器配置使用的持久性 DNS 服务器地址。

注意

如果没有使用 DHCP 服务，则必须在 RHCOS 安装时为节点提供 IP 网络配置和 DNS 服务器地址。如果要从 ISO 镜像安装，这些参数可作为引导参数传递。如需有关静态 IP 置备和高级网络选项的更多信息，请参阅 *安装 RHCOS 并启动 OpenShift Container Platform bootstrap 过程* 部分。

c.

在 DHCP 服务器配置中定义集群节点的主机名。有关 *主机名注意事项的详情，请参阅通过 DHCP 设置集群节点* 主机名部分。

注意

如果没有使用 DHCP 服务，集群节点可以通过反向 DNS 查找来获取其主机名。

2.

确保您的网络基础架构提供集群组件之间所需的网络连接。有关 *要求的详情，请参阅用户置备的基础架构* 的网络要求部分。

3.

将防火墙配置为启用 OpenShift Container Platform 集群组件进行通信所需的端口。*如需有关所需端口的详细信息，请参阅用户置备的基础架构* 部分的网络要求。

重要

默认情况下，OpenShift Container Platform 集群可以访问端口 1936，因为每个 control plane 节点都需要访问此端口。

避免使用 Ingress 负载均衡器公开此端口，因为这样做可能会导致公开敏感信息，如统计信息和指标（与 Ingress Controller 相关的统计信息和指标）。

4.

为集群设置所需的 DNS 基础架构。

a.

为 Kubernetes API、应用程序通配符、bootstrap 机器、control plane 机器和计算机器配置 DNS 名称解析。

b.

为 Kubernetes API、bootstrap 机器、control plane 机器和计算机器配置反向 DNS

解析。

如需有关 *OpenShift Container Platform DNS 要求的更多信息，请参阅用户置备* DNS 要求部分。

5. 验证您的 DNS 配置。

a. 从安装节点，针对 Kubernetes API 的记录名称、通配符路由和集群节点运行 DNS 查找。验证响应中的 IP 地址是否与正确的组件对应。

b. 从安装节点，针对负载均衡器和集群节点的 IP 地址运行反向 DNS 查找。验证响应中的记录名称是否与正确的组件对应。

有关详细的 *DNS 验证步骤，请参阅用户置备的基础架构* 验证 DNS 解析部分。

6. 置备所需的 API 和应用程序入口负载平衡基础架构。有关 *要求的更多信息，请参阅用户置备的基础架构的负载平衡* 要求部分。

注意

某些负载平衡解决方案要求在初始化负载平衡之前，对群集节点进行 DNS 名称解析。

23.3.2.5. 验证用户置备的基础架构的 DNS 解析

您可以在在用户置备的基础架构上安装 OpenShift Container Platform 前验证 DNS 配置。

重要

本节中详述的验证步骤必须在安装集群前成功。

先决条件

- 已为您的用户置备的基础架构配置了所需的 DNS 记录。

流程

1. 从安装节点，针对 Kubernetes API 的记录名称、通配符路由和集群节点运行 DNS 查找。验证响应中包含的 IP 地址是否与正确的组件对应。

   a. 对 Kubernetes API 记录名称执行查询。检查结果是否指向 API 负载均衡器的 IP 地址：

   ```
   $ dig +noall +answer @<nameserver_ip> api.<cluster_name>.<base_domain> 1
   ```

   1 将 <nameserver_ip> 替换为 nameserver 的 IP 地址，<cluster_name> 替换为您的集群名称，<base_domain> 替换为您的基本域名。

   输出示例

   ```
   api.ocp4.example.com.  604800 IN A 192.168.1.5
   ```

   b. 对 Kubernetes 内部 API 记录名称执行查询。检查结果是否指向 API 负载均衡器的 IP 地址：

   ```
   $ dig +noall +answer @<nameserver_ip> api-int.<cluster_name>.<base_domain>
   ```

   输出示例

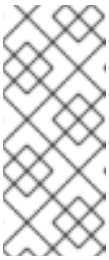
   ```
   api-int.ocp4.example.com.  604800 IN A 192.168.1.5
   ```

   c. 测试 *.apps.<cluster_name>.<base_domain> DNS 通配符查找示例。所有应用程序通配符查询都必须解析为应用程序入口负载均衡器的 IP 地址：

   ```
   $ dig +noall +answer @<nameserver_ip> random.apps.<cluster_name>.
   <base_domain>
   ```

输出示例

```
random.apps.ocp4.example.com.  604800 IN A 192.168.1.5
```



注意

在示例中，将相同的负载均衡器用于 Kubernetes API 和应用程序入口流量。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。

您可以使用另一个通配符值替换 random。例如，您可以查询到 OpenShift Container Platform 控制台的路由：

```
$ dig +noall +answer @<nameserver_ip> console-openshift-console.apps.
<cluster_name>.<base_domain>
```

输出示例

```
console-openshift-console.apps.ocp4.example.com. 604800 IN A 192.168.1.5
```

d. 针对 bootstrap DNS 记录名称运行查询。检查结果是否指向 bootstrap 节点的 IP 地址：

```
$ dig +noall +answer @<nameserver_ip> bootstrap.<cluster_name>.
<base_domain>
```

输出示例

```
bootstrap.ocp4.example.com.  604800 IN A 192.168.1.96
```

e. 使用此方法对 control plane 和计算节点的 DNS 记录名称执行查找。检查结果是否与每个节点的 IP 地址对应。

2. 从安装节点，针对负载均衡器和集群节点的 IP 地址运行反向 DNS 查找。验证响应中包含的记录名称是否与正确的组件对应。

a. 对 API 负载均衡器的 IP 地址执行反向查找。检查响应是否包含 Kubernetes API 和 Kubernetes 内部 API 的记录名称：

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.5
```

输出示例

```
5.1.168.192.in-addr.arpa. 604800 IN PTR api-int.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. 604800 IN PTR api.ocp4.example.com. 2
```

1 为 Kubernetes 内部 API 提供记录名称。

2 为 Kubernetes API 提供记录名称。

注意

OpenShift Container Platform 应用程序通配符不需要 PTR 记录。针对应用程序入口负载均衡器的 IP 地址解析反向 DNS 解析不需要验证步骤。

b. 对 bootstrap 节点的 IP 地址执行反向查找。检查结果是否指向 bootstrap 节点的 DNS 记录名称：

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.96
```

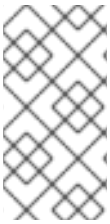
输出示例

```
96.1.168.192.in-addr.arpa. 604800 IN PTR bootstrap.ocp4.example.com.
```

c. 使用此方法对 control plane 和计算节点的 IP 地址执行反向查找。检查结果是否与每个节点的 DNS 记录名称对应。

23.3.3. 使用用户置备的基础架构在 vSphere 上安装集群

在 OpenShift Container Platform 版本 4.16 中，您可以在您置备的 VMware vSphere 基础架构上安装集群。



注意

OpenShift Container Platform 支持将集群部署到单个 VMware vCenter 中。不支持在多个 vCenter 上使用机器/机器集部署集群。

重要

执行用户置备的基础架构安装的步骤仅作为示例。使用您提供的基础架构安装集群需要了解 vSphere 平台和 OpenShift Container Platform 的安装过程。使用用户置备的基础架构安装说明作为指南；您可以通过其他方法创建所需的资源。

23.3.3.1. 先决条件

- 您已完成了 使用用户置备的基础架构准备安装集群 中的任务。
- 您检查了 VMware 平台许可证。红帽不会对 VMware 许可证产生任何限制，但有些 VMware 基础架构组件需要许可。
- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。

- 您可以阅读有关 选择集群安装方法的文档，并为用户准备它。

- 已为集群置备了 持久性存储。要部署私有镜像 registry，您的存储必须提供 ReadWriteMany 访问模式。

- 完成安装要求您在 vSphere 主机上上传 Red Hat Enterprise Linux CoreOS(RHCOS)OVA。完成此过程的机器需要访问 vCenter 和 ESXi 主机上的端口 443。您确认可以访问端口 443。

- 如果您使用防火墙，您与管理员确认可以访问端口 443。control plane 节点必须能够通过端口 443 访问 vCenter 和 ESXi 主机，才能成功安装。

- 如果使用防火墙，则会 将其配置为允许集群需要访问的站点。

注意

如果要配置代理，请务必查看此站点列表。

23.3.3.2. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。

- 访问 Quay.io，以获取安装集群所需的软件包。

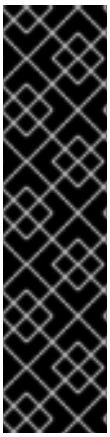
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类型的基础架构上执行受限网络安装。在此过程中，您可以下载所需的内容，并使用它为镜像 registry 填充安装软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群前，您要更新镜像 registry 的内容。

23.3.3.3. VMware vSphere 区域和区启用

您可以将 OpenShift Container Platform 集群部署到在单个 VMware vCenter 中运行的多个 vSphere 数据中心。每个数据中心都可以运行多个集群。此配置降低了导致集群失败的硬件故障或网络中断的风险。要启用区域和区域，您必须为 OpenShift Container Platform 集群定义多个故障域。



重要

VMware vSphere 区域和区启用功能需要 vSphere Container Storage Interface (CSI) 驱动程序作为集群中的默认存储驱动程序。因此，这个功能只在新安装的集群中可用。

对于从上一版本升级的集群，您必须为集群启用 CSI 自动迁移。然后，您可以为升级的集群配置多个区域和区域。

默认安装配置将集群部署到单个 vSphere 数据中心。如果要将集群部署到多个 vSphere 数据中心，您必须创建一个启用地区和区功能的安装配置文件。

默认 install-config.yaml 文件包含 vcenters 和 failureDomains 字段，您可以在其中为 OpenShift Container Platform 集群指定多个 vSphere 数据中心和集群。如果要在由单个数据中心组成的 vSphere 环境中安装 OpenShift Container Platform 集群，您可以将这些字段留空。

以下列表描述了为集群定义区和区域相关的术语：

- 故障域：建立地区和区域之间的关系。您可以使用 vCenter 对象（如 datastore 对象）定义故障域。故障域定义 OpenShift Container Platform 集群节点的 vCenter 位置。
- Region ：指定 vCenter 数据中心。您可以使用 openshift-region 标签类别中的标签来定义区域。

- Zone：指定一个 vCenter 集群。您可以使用 openshift-zone 标签类别中的标签来定义区。

注意

如果您计划在 install-config.yaml 文件中指定多个故障域，则必须在创建配置文件前创建标签类别、区域标签和区域标签。

您必须为每个代表一个区域的 vCenter 数据中心创建一个 vCenter 标签。另外，您必须为比数据中心（代表一个区）中运行的每个集群创建一个 vCenter 标签。创建标签后，您必须将每个标签附加到对应的数据中心和集群。

下表概述了在单个 VMware vCenter 中运行的多个 vSphere 数据中心的区域、区域和标签之间的关系示例。

数据中心（区域）	集群（区）	Tags
us-east	us-east-1	us-east-1a
		us-east-1b
	us-east-2	us-east-2a
		us-east-2b
us-west	us-west-1	us-west-1a
		us-west-1b
	us-west-2	us-west-2a
		us-west-2b

其他资源

- 其他 VMware vSphere 配置参数
- 弃用的 VMware vSphere 配置参数

- vSphere 自动迁移
- VMware vSphere CSI Driver Operator

23.3.3.4. 手动创建安装配置文件

安装集群要求您手动创建安装配置文件。

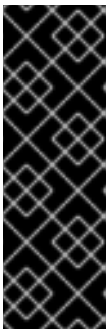
先决条件

- 您在本地机器上有一个 SSH 公钥来提供给安装程序。该密钥将用于在集群节点上进行 SSH 身份验证，以进行调试和灾难恢复。
- 已获取 OpenShift Container Platform 安装程序和集群的 pull secret。

流程

1. 创建一个安装目录来存储所需的安装资产：

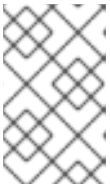
```
$ mkdir <installation_directory>
```



重要

您必须创建一个目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

2. 自定义提供的 install-config.yaml 文件模板示例，并将其保存在 <installation_directory> 中。



注意

此配置文件必须命名为 install-config.yaml。

3. 如果要安装三节点集群，请通过将 compute.replicas 参数设置为 0 来修改 install-config.yaml 文件。这样可确保集群的 control plane 可以调度。如需更多信息，请参阅"在 vSphere 上安装三节点集群"。

4. 备份 install-config.yaml 文件，以便您可以使用它安装多个集群。

重要

install-config.yaml 文件会在安装过程的下一步中使用。现在必须备份它。

其他资源

- 安装配置参数

23.3.3.4.1. VMware vSphere 的 install-config.yaml 文件示例

您可以自定义 install-config.yaml 文件，以指定有关 OpenShift Container Platform 集群平台的更多详情，或修改所需参数的值。

```
additionalTrustBundlePolicy: Proxyonly
apiVersion: v1
baseDomain: example.com (1)
compute: (2)
- architecture: amd64
  name: <worker_node>
  platform: {}
  replicas: 0 (3)
controlPlane: (4)
  architecture: amd64
  name: <parent_node>
  platform: {}
  replicas: 3 (5)
metadata:
  creationTimestamp: null
  name: test (6)
networking:
---
platform:
  vsphere:
    failureDomains: (7)
    - name: <failure_domain_name>
      region: <default_region_name>
      server: <fully_qualified_domain_name>
      topology:
```

```
      computeCluster: "/<datacenter>/host/<cluster>"
      datacenter: <datacenter> 8
      datastore: "/<datacenter>/datastore/<datastore>" 9
      networks:
      - <VM_Network_name>
      resourcePool: "/<datacenter>/host/<cluster>/Resources/<resourcePool>" 10
      folder: "/<datacenter_name>/vm/<folder_name>/<subfolder_name>" 11
    zone: <default_zone_name>
  vcenters:
  - datacenters:
    - <datacenter>
    password: <password> 12
    port: 443
    server: <fully_qualified_domain_name> 13
    user: administrator@vsphere.local
  diskType: thin 14
fips: false 15
pullSecret: '{"auths": ...}' 16
sshKey: 'ssh-ed25519 AAAA...' 17
```

1

集群的基域。所有 DNS 记录都必须是这个基域的子域，并包含集群名称。

2 4

controlPlane 部分是一个单个映射，但 compute 部分是一系列映射。为满足不同数据结构的要求，compute 部分的第一行必须以连字符 - 开头，controlPlane 部分 的第一行则不以连字符开头。两个部分都定义单个机器池，因此只使用一个 control plane。OpenShift Container Platform 不支持定义多个计算池。

3

replicas 参数的值必须设置为 0。此参数控制集群为您创建和管理的 worker 数量，在使用用户置备的基础架构时集群不会执行这些功能。在完成 OpenShift Container Platform 安装前，您必须手动为集群部署 worker 机器。

5

您添加到集群的 control plane 机器数量。由于集群使用此值作为集群中的 etcd 端点数量，所以该值必须与您部署的 control plane 机器数量匹配。

6

您在 DNS 记录中指定的集群名称。

7

建立地区和区域之间的关系。您可以使用 vCenter 对象（如 datastore 对象）定义故障域。故障域定义 OpenShift Container Platform 集群节点的 vCenter 位置。

8

vSphere 数据中心.

9

保存虚拟机文件、模板和 ISO 镜像的 vSphere 数据存储路径。

重要

您可以指定数据存储集群中存在的任何数据存储路径。默认情况下，Storage vMotion 会自动为数据存储集群启用。红帽不支持 Storage vMotion，因此您必须禁用 Storage vMotion 以避免 OpenShift Container Platform 集群的数据丢失问题。

如果需要在多个数据存储间指定虚拟机，请使用 数据存储 对象在集群 install-config.yaml 配置文件中指定故障域。如需更多信息，请参阅"VMware vSphere 区域和区启用"。

10

可选：对于安装程序置备的基础架构，安装程序创建虚拟机的现有资源池的绝对路径，例如 /<datacenter_name>/host/<cluster_name>/Resources/<resource_pool_name>/<optional_nested_resource_pool_name>。如果没有指定值，则会在集群 /example_datacenter/host/example_cluster/Resources 根中安装资源。

11

可选：对于安装程序置备的基础架构，安装程序创建虚拟机的现有文件夹的绝对路径，如 /<datacenter_name>/vm/<folder_name>/<subfolder_name>。如果没有提供这个值，安装程序会在数据中心虚拟机文件夹中创建一个顶层文件夹，其名称为基础架构 ID。如果您为集群提供基础架构，且您不想使用默认的 StorageClass 对象（名为 thin），您可以从 install-config.yaml 文件中省略 folder 参数。

12

与 vSphere 用户关联的密码。

13

vCenter 服务器的完全限定主机名或 IP 地址。

重要

Cloud Controller Manager Operator 在提供的主机名或 IP 地址上执行连接检查。确保为可访问的 vCenter 服务器指定主机名或 IP 地址。如果您向不存在的 vCenter 服务器提供元数据，集群安装会在 bootstrap 阶段失败。

14

vSphere 磁盘置备方法。

15

是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 Red Hat Enterprise Linux (RHEL) 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅在 FIPS 模式中安装该系统。

当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS)时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。

16

从 OpenShift Cluster Manager 获取的 pull secret。此 pull secret 允许您与所含授权机构提供的服务进行身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

17

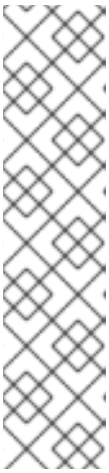
Red Hat Enterprise Linux CoreOS(RHCOS)中 core 用户的默认 SSH 密钥的公钥部分。

23.3.3.4.2. 在安装过程中配置集群范围的代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 install-config.yaml 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 install-config.yaml 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 Proxy 对象的 spec.noProxy 字段中添加站点来绕过代理。



注意

Proxy 对象 status.noProxy 字段使用安装配置中的 networking.machineNetwork[].cidr、networking.clusterNetwork[].cidr 和 networking.serviceNetwork[] 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，Proxy 对象 status.noProxy 字段也会使用实例元数据端点填充(169.254.169.254)。

流程

1. 编辑 install-config.yaml 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

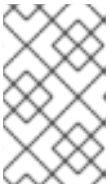
1. 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 http。
2. 用于创建集群外 HTTPS 连接的代理 URL。
3.

4

如果提供，安装程序会在 openshift-config 命名空间中生成名为 user-ca-bundle 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 trusted-ca-bundle 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， Proxy 对象的 trustedCA 字段中也会引用此配置映射。additionalTrustBundle 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

5

可选：决定 Proxy 对象的配置以引用 trustedCA 字段中 user-ca-bundle 配置映射的策略。允许的值是 Proxyonly 和 Always。仅在配置了 http/https 代理时，使用 Proxyonly 引用 user-ca-bundle 配置映射。使用 Always 始终引用 user-ca-bundle 配置映射。默认值为 Proxyonly。



注意

安装程序不支持代理的 readinessEndpoints 字段。

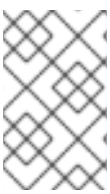
注意

如果安装程序超时，重启并使用安装程序的 wait-for 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. 保存该文件并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 cluster 的集群范围代理，该代理使用提供的 install-config.yaml 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 cluster Proxy 对象，但它会有一个空 spec。



注意

只支持名为 cluster 的 Proxy 对象，且无法创建额外的代理。

23.3.3.4.3. 为 VMware vCenter 配置区域和区域

您可以修改默认安装配置文件，以便您可以将 OpenShift Container Platform 集群部署到在单个

VMware vCenter 中运行的多个 vSphere 数据中心。

之前版本的 OpenShift Container Platform 的默认 install-config.yaml 文件配置已弃用。您可以继续使用已弃用的默认配置，但 openshift-installer 会提示您显示在配置文件中已弃用字段的警告信息。

重要

这个示例使用 govc 命令。govc 命令是 VMware 提供的开源命令；它不是红帽提供的。红帽支持团队不维护 govc 命令。有关下载和安装 govc 的说明，请参阅 VMware 文档网站

先决条件

- 您有一个现有的 install-config.yaml 安装配置文件。

重要

您必须为 OpenShift Container Platform 集群指定一个故障域，以便您可以为 VMware vCenter 服务器置备数据中心对象。如果您需要在不同的数据中心、集群、数据存储和其他组件中置备虚拟机节点，请考虑指定多个故障域。要启用区域和区域，您必须为 OpenShift Container Platform 集群定义多个故障域。

流程

1. 输入以下 govc 命令行工具命令，以创建 openshift-region 和 openshift-zone vCenter 标签类别：

重要

如果为 openshift-region 和 openshift-zone vCenter 标签类别指定不同的名称，OpenShift Container Platform 集群的安装会失败。

```
$ govc tags.category.create -d "OpenShift region" openshift-region
```

```
$ govc tags.category.create -d "OpenShift zone" openshift-zone
```

2. 要为您要部署集群的每个区域 vSphere 数据中心创建一个 region 标签，请在终端中输入以下命令：

```
$ govc tags.create -c <region_tag_category> <region_tag>
```

3. 要为您要部署集群的每个 vSphere 集群创建一个区标签，请输入以下命令：

```
$ govc tags.create -c <zone_tag_category> <zone_tag>
```

4. 输入以下命令将区域标签附加到每个 vCenter 数据中心对象：

```
$ govc tags.attach -c <region_tag_category> <region_tag_1> /<datacenter_1>
```

5. 输入以下命令将区标签附加到每个 vCenter 数据中心对象：

```
$ govc tags.attach -c <zone_tag_category> <zone_tag_1> /<datacenter_1>/host/vcs-mdcnc-workload-1
```

6. 进入包含安装程序的目录，并根据您选择的安装要求初始化集群部署。

在 vSphere 数据中心中定义的多个数据中心的 install-config.yaml 文件示例

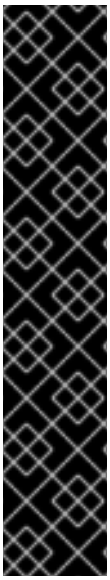
```
---
compute:
---
  vsphere:
    zones:
      - "<machine_pool_zone_1>"
      - "<machine_pool_zone_2>"
---
controlPlane:
---
vsphere:
    zones:
      - "<machine_pool_zone_1>"
      - "<machine_pool_zone_2>"
---
platform:
  vsphere:
    vcenters:
---
    datacenters:
      - <datacenter1_name>
      - <datacenter2_name>
    failureDomains:
    - name: <machine_pool_zone_1>
```

```
      region: <region_tag_1>
      zone: <zone_tag_1>
      server: <fully_qualified_domain_name>
      topology:
        datacenter: <datacenter1>
        computeCluster: "/<datacenter1>/host/<cluster1>"
        networks:
        - <VM_Network1_name>
        datastore: "/<datacenter1>/datastore/<datastore1>"
        resourcePool: "/<datacenter1>/host/<cluster1>/Resources/<resourcePool1>"
        folder: "/<datacenter1>/vm/<folder1>"
    - name: <machine_pool_zone_2>
      region: <region_tag_2>
      zone: <zone_tag_2>
      server: <fully_qualified_domain_name>
      topology:
        datacenter: <datacenter2>
        computeCluster: "/<datacenter2>/host/<cluster2>"
        networks:
        - <VM_Network2_name>
        datastore: "/<datacenter2>/datastore/<datastore2>"
        resourcePool: "/<datacenter2>/host/<cluster2>/Resources/<resourcePool2>"
        folder: "/<datacenter2>/vm/<folder2>"
---
```

23.3.3.5. 创建 Kubernetes 清单和 Ignition 配置文件

由于您必须修改一些集群定义文件并手动启动集群机器，因此您必须生成 Kubernetes 清单和 Ignition 配置文件来配置机器。

安装配置文件转换为 Kubernetes 清单。清单嵌套到 Ignition 配置文件中，稍后用于配置集群机器。



重要

- OpenShift Container Platform 安装程序生成的 Ignition 配置文件包含 24 小时后过期的证书，然后在该时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 node-bootstrapper 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。

- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

先决条件

- 已获得 OpenShift Container Platform 安装程序。
- 已创建 install-config.yaml 安装配置文件。

流程

1. 进入包含 OpenShift Container Platform 安装程序的目录，并为集群生成 Kubernetes 清单：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 对于 <installation_directory>，请指定包含您创建的 install-config.yaml 文件的安装目录。

2. 删除定义 control plane 机器的 Kubernetes 清单文件、计算机器集和 control plane 机器集：

```
$ rm -f openshift/99_openshift-cluster-api_master-machines-*.yaml
openshift/99_openshift-cluster-api_worker-machineset-*.yaml openshift/99_openshift-
machine-api_master-control-plane-machine-set.yaml
```

由于您要自行创建和管理这些资源，因此不必初始化这些资源。

- 您可以使用机器 API 来保留计算机器集文件来创建计算机器，但您必须更新对它们的引用以匹配您的环境。

> **警告**
>
> 如果您要安装一个三节点集群，请跳过以下步骤，以便可以调度 control plane 节点。

> **重要**
>
> 当您将 control plane 节点从默认的不可调度配置为可以调度时，需要额外的订阅。这是因为 control plane 节点变为计算节点。

3. 检查 <installation_directory>/manifests/cluster-scheduler-02-config.yml Kubernetes 清单文件中的 mastersSchedulable 参数是否已设置为 false。此设置可防止在 control plane 机器上调度 pod：

   a. 打开 <installation_directory>/manifests/cluster-scheduler-02-config.yml 文件。

   b. 找到 mastersSchedulable 参数，并确保它被设置为 false。

   c. 保存并退出 文件。

4. 要创建 Ignition 配置文件，请从包含安装程序的目录运行以下命令：

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

1 对于 <installation_directory>，请指定相同的安装目录。

为安装目录中的 bootstrap、control plane 和计算节点创建 Ignition 配置文

件。kubeadmin-password 和 kubeconfig 文件在 ./<installation_directory>/auth 目录中创建：

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

23.3.3.6. 提取基础架构名称

Ignition 配置文件包含一个唯一集群标识符，您可以使用它在 VMware vSphere 中唯一地标识您的集群。如果计划使用集群标识符作为虚拟机文件夹的名称，则必须提取它。

先决条件

- 已获取 OpenShift Container Platform 安装程序和集群的 pull secret。
- 已为集群生成 Ignition 配置文件。
- 已安装 jq 软件包。

流程

- 要从 Ignition 配置文件元数据中提取和查看基础架构名称，请运行以下命令：

```
$ jq -r .infraID <installation_directory>/metadata.json 1
```

1 对于 <installation_directory>，请指定安装文件保存到的目录的路径。

输出示例

```
openshift-vw9j6 1
```

1 此命令的输出是您的集群名称和随机字符串。

23.3.3.7. 安装 RHCOS 并启动 OpenShift Container Platform bootstrap 过程

要在 VMware vSphere 上的用户置备的基础架构上安装 OpenShift Container Platform，您必须在 vSphere 主机上安装 Red Hat Enterprise Linux CoreOS(RHCOS)。安装 RHCOS 时，您必须为您要安装的机器类型提供 OpenShift Container Platform 安装程序生成的 Ignition 配置文件。如果您配置了适当的网络、DNS 和负载均衡基础架构，OpenShift Container Platform bootstrap 过程会在 RHCOS 机器重启后自动启动。

先决条件

- 已获取集群的 Ignition 配置文件。
- 具有 HTTP 服务器的访问权限，以便您可从计算机进行访问，并且您创建的机器也可访问此服务器。
- 您已创建了 vSphere 集群。

流程

1. 将名为 <installation_directory>/bootstrap.ign 的 bootstrap Ignition 配置文件上传到 HTTP 服务器。注意此文件的 URL。
2. 将 bootstrap 节点的以下辅助 Ignition 配置文件保存到计算机中，存为 <installation_directory>/merge-bootstrap.ign ：

```
{
  "ignition": {
    "config": {
      "merge": [
        {
          "source": "<bootstrap_ignition_config_url>", 1
          "verification": {}
        }
```

```
      ]
     },
     "timeouts": {},
     "version": "3.2.0"
    },
    "networkd": {},
    "passwd": {},
    "storage": {},
    "systemd": {}
}
```

1

指定您托管的 bootstrap Ignition 配置文件的 URL。

为 bootstrap 机器创建虚拟机(VM)时，您要使用此 Ignition 配置文件。

3. 找到安装程序创建的以下 Ignition 配置文件：

- <installation_directory>/master.ign
- <installation_directory>/worker.ign
- <installation_directory>/merge-bootstrap.ign

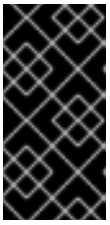
4. 将 Ignition 配置文件转换为 Base64 编码。在此流程中，您必须将这些文件添加到虚拟机中的额外配置参数 guestinfo.ignition.config.data 中。

例如，如果使用 Linux 操作系统，您可以使用 base64 命令对文件进行编码。

```
$ base64 -w0 <installation_directory>/master.ign > <installation_directory>/master.64
```

```
$ base64 -w0 <installation_directory>/worker.ign > <installation_directory>/worker.64
```

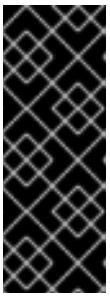
```
$ base64 -w0 <installation_directory>/merge-bootstrap.ign >
<installation_directory>/merge-bootstrap.64
```



重要

如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

5. 获取 RHCOS OVA 镜像。镜像位于 RHCOS 镜像镜像页面。



重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每个发行版本而改变。您必须下载最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。如果可用，请使用与 OpenShift Container Platform 版本匹配的镜像版本。

文件名包含 OpenShift Container Platform 版本号，格式为 rhcos-vmware.<architecture>.ova。

6. 在 vSphere 客户端中，在数据中心中创建一个文件夹来存储虚拟机。

   a. 单击 VMs and Templates 视图。

   b. 右键单击您的数据中心的名称。

   c. 点击 New Folder → New VM and Template Folder。

   d. 在显示的窗口中，输入文件夹名称。如果您没有在 install-config.yaml 文件中指定现有文件夹，请创建一个名称与基础架构 ID 相同的文件夹。您可以使用这个文件夹名称，因此 vCenter 会在适当的位置为 Workspace 配置动态置备存储。

7. 在 vSphere 客户端中，为 OVA 镜像创建一个模板，然后根据需要克隆模板。

注意

在以下步骤中，您将创建模板，然后克隆所有集群机器的模板。然后，您在置备虚拟机时为该克隆的机器类型提供 Ignition 配置文件的位置。

a. 在 Hosts and Clusters 选项卡中，右键点击您的集群名称并选择 Deploy OVF Template。

b. 在 Select an OVF 选项卡中，指定您下载的 RHCOS OVA 文件的名称。

c. 在 Select a name and folder 选项卡中，为您的模板设置 虚拟机名称，如 Template-RHCOS。点击 vSphere 集群的名称并选择您在上一步中创建的文件夹。

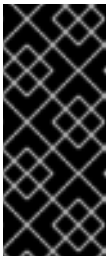
d. 在 Select a compute resource 选项卡中，点击 vSphere 集群的名称。

e. 在 Select storage 选项卡中，配置虚拟机的存储选项。

- 根据您的存储首选项，选择 Thin Provision 或 Thick Provision。
- 选择您在 install-config.yaml 文件中指定的数据存储。
- 如果要加密虚拟机，请选择 Encrypt this virtual machine。如需更多信息，请参阅标题为"加密虚拟机的要求"的部分。

f. 在 Select network 选项卡中，指定您为集群配置的网络（如果可用）。

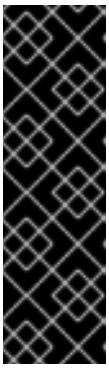
g. 在创建 OVF 模板时，不要在 Customize template 选项卡上指定值，也不会进一步配置模板。



重要

不要启动原始虚拟机模板。VM 模板必须保持关闭，必须为新的 RHCOS 机器克隆。启动虚拟机模板会将虚拟机模板配置为平台上的虚拟机，这样可防止它被用作计算机器集可应用配置的模板。

8. 可选：如果需要，更新 VM 模板中配置的虚拟硬件版本。如需更多信息，请参阅 VMware 文档中的将虚拟机升级到最新硬件版本。



重要

如有必要，建议您在从虚拟机创建虚拟机前将虚拟机模板的硬件版本更新为版本 15。在 vSphere 上运行的集群节点使用硬件版本 13 现已弃用。如果您导入的模板默认为硬件版本 13，您必须在将 VM 模板升级到硬件版本 15 前确保 ESXi 主机为 6.7U3 或更高版本。如果您的 vSphere 版本小于 6.7U3，您可以跳过此升级步骤；但是，计划将来的 OpenShift Container Platform 版本删除对小于 6.7U3 的硬件版本 13 和 vSphere 版本的支持。

9. 部署模板后，为集群中的机器部署虚拟机。

   a. 右键点击模板名称，再点击 Clone → Clone to Virtual Machine。

   b. 在 Select a name and folder 选项卡中，指定虚拟机的名称。您可以在名称中包含机器类型，如 control-plane-0 或 compute-1。

注意

确保 vSphere 安装中的所有虚拟机名称都是唯一的。

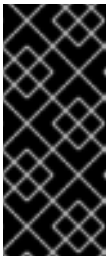
   c. 在 Select a name and folder 选项卡中，选择您为集群创建的文件夹名称。

   d. 在 Select a compute resource 选项卡中，选择数据中心中的主机名称。

   e. 在 Select clone options 选项卡中，选择 Customize this virtual machine's hardware。

f.

在 Customize hardware 选项卡上，点 Advanced Parameters。



重要

以下配置建议仅用于演示目的。作为集群管理员，您必须根据集群上的资源需求来配置资源。为了更好地管理集群资源，请考虑从集群的 root 资源池创建资源池。

- 可选：覆盖 vSphere 中的默认 DHCP 网络。启用静态 IP 网络：
  - 设置静态 IP 配置：

示例命令

```
$ export IPCFG="ip=<ip>::<gateway>:<netmask>:<hostname>:
<iface>:none nameserver=srv1 [nameserver=srv2 [nameserver=srv3 [...]]]"
```

示例命令

```
$ export IPCFG="ip=192.168.100.101::192.168.100.254:255.255.255.0:::none
nameserver=8.8.8.8"
```

  - 在从 vSphere 中的 OVA 引导虚拟机前，设置 guestinfo.afterburn.initrd.network-kargs 属性：

示例命令

```
$ govc vm.change -vm "<vm_name>" -e
"guestinfo.afterburn.initrd.network-kargs=${IPCFG}"
```

- 通过在 Attribute 和 Values 字段中指定数据来添加以下配置参数名称和值。确保为您创建的每个参数选择 Add 按钮。
  - guestinfo.ignition.config.data ：找到您在此流程中创建的 base-64 编码文件，并粘贴此机器类型的 base64 编码 Ignition 配置文件的内容。
  - guestinfo.ignition.config.data.encoding ：指定 base64。
  - disk.EnableUUID ：指定 TRUE。
  - stealclock.enable ：如果没有定义此参数，请添加它并指定 TRUE。
  - 从集群的 root 资源池创建子资源池。执行此子资源池中的资源分配。

g. 在 Customize hardware 选项卡的 Virtual Hardware 面板中，根据需要修改指定的值。确保 RAM、CPU 和磁盘存储的数量满足机器类型的最低要求。

h. 完成剩余的配置步骤。点 Finish 按钮，您已完成克隆操作。

i. 在 Virtual Machines 选项卡中，右键点您的虚拟机，然后选择 Power → Power On。

j. 检查控制台输出，以验证 Ignition 是否运行。

示例命令

```
Ignition: ran on 2022/03/14 14:48:33 UTC (this boot)
Ignition: user-provided config was applied
```

后续步骤

- 对每台机器执行前面的步骤，为集群创建其余机器。

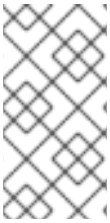
重要

此时您必须创建 bootstrap 和 control plane 机器。由于计算机器上已默认部署了一些 Pod，因此还要在安装集群前至少创建两台计算机器。

23.3.3.8. 将更多计算机器添加到 vSphere 中的集群

您可以将更多计算机器添加到 VMware vSphere 上的用户置备的 OpenShift Container Platform 集群中。

在 OpenShift Container Platform 集群中部署 vSphere 模板后，您可以为该集群中的机器部署虚拟机(VM)。



注意

如果您要安装三节点集群，请跳过这一步。三节点集群包含三个 control plane 机器，它们也可以充当计算机器。

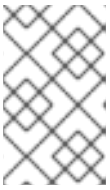
先决条件

- 获取计算机器的 base64 编码 Ignition 文件。
- 您可以访问您为集群创建的 vSphere 模板。

流程

1. 右键点击模板的名称，再点击 Clone → Clone to Virtual Machine。
2. 在 Select a name and folder 选项卡中，指定虚拟机的名称。您可以在名称中包含机器类

型，如 compute-1。



注意

确保 vSphere 安装中的所有虚拟机名称都是唯一的。

3. 在 Select a name and folder 选项卡中，选择您为集群创建的文件夹名称。

4. 在 Select a compute resource 选项卡中，选择数据中心中的主机名称。

5. 在 Select storage 选项卡中，为您的配置和磁盘文件选择存储。

6. 在 Select clone options 选项卡中，选择 Customize this virtual machine's hardware。

7. 在 Customize hardware 选项卡上，点 Advanced Parameters。

- 通过在 Attribute 和 Values 字段中指定数据来添加以下配置参数名称和值。确保为您创建的每个参数选择 Add 按钮。
  - guestinfo.ignition.config.data ：粘贴此机器类型的 base64 编码计算 Ignition 配置文件的内容。
  - guestinfo.ignition.config.data.encoding ：指定 base64。
  - disk.EnableUUID ：指定 TRUE。

8. 在 Customize hardware 选项卡的 Virtual Hardware 面板中，根据需要修改指定的值。确保 RAM、CPU 和磁盘存储的数量满足机器类型的最低要求。如果存在多个网络，请选择 Add New Device > Network Adapter，然后在 New Network 菜单项提供的字段中输入您的网络信息。

9. 完成剩余的配置步骤。点 Finish 按钮，您已完成克隆操作。

10. 在 Virtual Machines 选项卡中，右键点您的虚拟机，然后选择 Power → Power On。

后续步骤

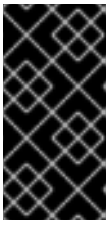
- 继续为集群创建更多计算机器。

23.3.3.9. 磁盘分区

在大多数情况下，数据分区最初是由安装 RHCOS 而不是安装另一个操作系统来创建的。在这种情况下，OpenShift Container Platform 安装程序被允许配置磁盘分区。

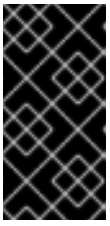
但是，在安装 OpenShift Container Platform 节点时，在两种情况下您可能需要覆盖默认分区：

- 创建单独的分区：要在空磁盘上进行 greenfield 安装，您可能需要在分区中添加单独的存储。这正式支持生成 /var 或 /var 的子目录，如 /var/lib/etcd（独立分区），但不支持两者。



重要

对于大于 100GB 的磁盘大小，特别是磁盘大小大于 1TB，请创建一个独立的 /var 分区。如需更多信息，请参阅"创建独立 /var 分区"和 红帽知识库文章。



重要

Kubernetes 仅支持两个文件系统分区。如果您在原始配置中添加多个分区，Kubernetes 无法监控所有这些分区。

- 保留现有分区：对于 brownfield 安装，您要在现有节点上重新安装 OpenShift Container Platform，并希望保留从之前的操作系统中安装的数据分区，对于 coreos-installer 来说，引导选项和选项都允许您保留现有数据分区。

创建独立 /var 分区

通常，OpenShift Container Platform 的磁盘分区应该保留给安装程序。然而，在有些情况下您可能需要在文件系统的一部分中创建独立分区。

OpenShift Container Platform 支持添加单个分区来将存储附加到 /var 分区或 /var 的子目录中。例如：

- /var/lib/containers ：保存随着系统中添加更多镜像和容器而增长的容器相关内容。
- /var/lib/etcd ：保存您可能希望独立保留的数据，比如 etcd 存储的性能优化。
- /var ：保存您可能希望独立保留的数据，以满足审计等目的。

重要

对于大于 100GB 的磁盘大小，特别是磁盘大小大于 1TB，请创建一个独立的 /var 分区。

通过单独存储 /var 目录的内容，可以更轻松地根据需要为区域扩展存储，并在以后重新安装 OpenShift Container Platform，并保持该数据的完整性。使用这个方法，您不必再次拉取所有容器，在更新系统时也不必复制大量日志文件。

因为 /var 在进行一个全新的 Red Hat Enterprise Linux CoreOS（RHCOS）安装前必需存在，所以这个流程会在 OpenShift Container Platform 安装过程的 openshift-install 准备阶段插入一个创建的机器配置清单的机器配置来设置独立的 /var 分区。

流程

1. 创建存放 OpenShift Container Platform 安装文件的目录：

```
$ mkdir $HOME/clusterconfig
```

2. 运行 openshift-install，以在 manifest 和 openshift 子目录中创建一组文件。在系统提示时回答系统问题：

```
$ openshift-install create manifests --dir $HOME/clusterconfig
? SSH Public Key ...
$ ls $HOME/clusterconfig/openshift/
99_kubeadmin-password-secret.yaml
99_openshift-cluster-api_master-machines-0.yaml
```

```
99_openshift-cluster-api_master-machines-1.yaml
99_openshift-cluster-api_master-machines-2.yaml
...
```

3. 创建用于配置额外分区的 Butane 配置。例如，将文件命名为 $HOME/clusterconfig/98-var-partition.bu，将磁盘设备名称改为 worker 系统上存储设备的名称，并根据情况设置存储大小。这个示例将 /var 目录放在一个单独的分区中：

```
variant: openshift
version: 4.16.0
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
storage:
  disks:
  - device: /dev/disk/by-id/<device_name> 1
    partitions:
    - label: var
      start_mib: <partition_start_offset> 2
      size_mib: <partition_size> 3
      number: 5
  filesystems:
    - device: /dev/disk/by-partlabel/var
      path: /var
      format: xfs
      mount_options: [defaults, prjquota] 4
      with_mount_unit: true
```

1. 要分区的磁盘的存储设备名称。

2. 当在引导磁盘中添加数据分区时，推荐最少使用 25000 MB。root 文件系统会自动调整大小以填充所有可用空间（最多到指定的偏移值）。如果没有指定值，或者指定的值小于推荐的最小值，则生成的 root 文件系统会太小，而在以后进行的 RHCOS 重新安装可能会覆盖数据分区的开始部分。

3. 以兆字节为单位的数据分区大小。

4. 对于用于容器存储的文件系统，必须启用 prjquota 挂载选项。

注意

当创建单独的 /var 分区时，如果不同的实例类型没有相同的设备名称，则无法为 worker 节点使用不同的实例类型。

4. 从 Butane 配置创建一个清单，并将它保存到 clusterconfig/openshift 目录中。例如，运行以下命令：

```
$ butane $HOME/clusterconfig/98-var-partition.bu -o
$HOME/clusterconfig/openshift/98-var-partition.yaml
```

5. 再次运行 openshift-install，从 manifest 和 openshift 子目录中的一组文件创建 Ignition 配置：

```
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
auth  bootstrap.ign  master.ign  metadata.json  worker.ign
```

现在，您可以使用 Ignition 配置文件作为 vSphere 安装程序的输入来安装 Red Hat Enterprise Linux CoreOS(RHCOS)系统。

23.3.3.10. 等待 bootstrap 过程完成

OpenShift Container Platform bootstrap 过程在集群节点首次引导到安装到磁盘的持久 RHCOS 环境后开始。通过 Ignition 配置文件提供的配置信息用于初始化 bootstrap 过程并在机器上安装 OpenShift Container Platform。您必须等待 bootstrap 过程完成。

先决条件

- 已为集群创建 Ignition 配置文件。
- 您已配置了适当的网络、DNS 和负载平衡基础架构。
- 已获得安装程序，并为集群生成 Ignition 配置文件。
- 已在集群机器上安装 RHCOS，并提供 OpenShift Container Platform 安装程序生成的 Ignition 配置文件。

- 您的机器可以直接访问互联网，或者有 HTTP 或 HTTPS 代理可用。

流程

1. 监控 bootstrap 过程：

```
$ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \ 1
  --log-level=info 2
```

1 对于 <installation_directory>，请指定安装文件保存到的目录的路径。

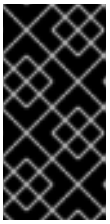
2 要查看不同的安装详情，请指定 warn、debug 或 error，而不是 info。

输出示例

```
INFO Waiting up to 30m0s for the Kubernetes API at
https://api.test.example.com:6443...
INFO API v1.29.4 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
```

当 Kubernetes API 服务器提示已在 control plane 机器上引导它时，该命令会成功。

2. bootstrap 过程完成后，从负载均衡器中删除 bootstrap 机器。



重要

此时您必须从负载均衡器中删除 bootstrap 机器。您还可以删除或重新格式化 bootstrap 机器本身。

23.3.3.11. 使用 CLI 登录集群

您可以通过导出集群 kubeconfig 文件，以默认系统用户身份登录集群。kubeconfig 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。
- 已安装 oc CLI。

流程

1. 导出 kubeadmin 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 <installation_directory>，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 oc 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

23.3.3.12. 批准机器的证书签名请求

当您将机器添加到集群时，会为您添加的每台机器生成两个待处理证书签名请求(CSR)。您必须确认这些 CSR 已获得批准，或根据需要自行批准。必须首先批准客户端请求，然后批准服务器请求。

先决条件

- 您已将机器添加到集群中。

流程

1. 确认集群可以识别这些机器：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0  Ready    master  63m  v1.29.4
master-1  Ready    master  63m  v1.29.4
master-2  Ready    master  64m  v1.29.4
```

输出中列出了您创建的所有机器。

注意

在有些 CSR 被批准前，前面的输出可能不包括计算节点（也称为 worker 节点）。

2. 检查待处理的 CSR，并确保添加到集群中的每台机器都有 Pending 或 Approved 状态的客户端请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE    REQUESTOR                                                   CONDITION
csr-8b2br   15m    system:serviceaccount:openshift-machine-config-operator:node-
bootstrapper   Pending
```

```
csr-8vnps   15m     system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
...
```

在本例中，两台机器加入集群。您可能会在列表中看到更多已批准的 CSR。

3. 如果 CSR 没有获得批准，在您添加的机器的所有待处理 CSR 都处于 Pending 状态后，请批准集群机器的 CSR：

注意

由于 CSR 会自动轮转，因此请在将机器添加到集群后一小时内批准您的 CSR。如果没有在一小时内批准它们，证书将会轮转，每个节点会存在多个证书。您必须批准所有这些证书。批准客户端 CSR 后，Kubelet 为服务证书创建一个二级 CSR，这需要手动批准。然后，如果 Kubelet 请求具有相同参数的新证书，则后续提供证书续订请求由 machine-approver 自动批准。

注意

对于在未启用机器 API 的平台上运行的集群，如裸机和其他用户置备的基础架构，您必须实施一种方法来自动批准 kubelet 提供证书请求(CSR)。如果没有批准请求，则 oc exec、ocrsh 和 oc logs 命令将无法成功，因为 API 服务器连接到 kubelet 时需要服务证书。与 Kubelet 端点联系的任何操作都需要此证书批准。该方法必须监视新的 CSR，确认 CSR 由 system: node 或 system:admin 组中的 node-bootstrapper 服务帐户提交，并确认节点的身份。

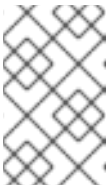
- 要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1. <csr_name> 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}
{{"\n"}}{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```



注意

在有些 CSR 被批准前，一些 Operator 可能无法使用。

4. 现在，您的客户端请求已被批准，您必须查看添加到集群中的每台机器的服务器请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE     REQUESTOR                                                   CONDITION
csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal       Pending
csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal       Pending
...
```

5. 如果剩余的 CSR 没有被批准，且处于 Pending 状态，请批准集群机器的 CSR：

- 要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 <csr_name> 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}
{{"\n"}}{{end}}{{end}}' | xargs oc adm certificate approve
```

6. 批准所有客户端和服务器 CSR 后，机器将 处于 Ready 状态。运行以下命令验证：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0  Ready    master  73m  v1.29.4
master-1  Ready    master  73m  v1.29.4
master-2  Ready    master  74m  v1.29.4
worker-0  Ready    worker  11m  v1.29.4
worker-1  Ready    worker  11m  v1.29.4
```

注意

批准服务器 CSR 后可能需要几分钟时间让机器过渡到 Ready 状态。

其他信息

- 如需有关 CSR 的更多信息，请参阅 证书签名请求。

23.3.3.13. 初始 Operator 配置

在 control plane 初始化后，您必须立即配置一些 Operator，以便它们都可用。

先决条件

- 您的 control plane 已初始化。

流程

1. 观察集群组件上线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION   AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.16.0    True        False         False      19m
baremetal                                  4.16.0    True        False         False      37m
cloud-credential                           4.16.0    True        False         False      40m
cluster-autoscaler                         4.16.0    True        False         False      37m
config-operator                            4.16.0    True        False         False      38m
console                                    4.16.0    True        False         False      26m
csi-snapshot-controller                    4.16.0    True        False         False      37m
dns                                        4.16.0    True        False         False      37m
etcd                                       4.16.0    True        False         False      36m
image-registry                             4.16.0    True        False         False      31m
ingress                                    4.16.0    True        False         False      30m
insights                                   4.16.0    True        False         False      31m
kube-apiserver                             4.16.0    True        False         False      26m
kube-controller-manager                    4.16.0    True        False         False      36m
kube-scheduler                             4.16.0    True        False         False      36m
kube-storage-version-migrator              4.16.0    True        False         False      37m
machine-api                                4.16.0    True        False         False      29m
machine-approver                           4.16.0    True        False         False      37m
machine-config                             4.16.0    True        False         False      36m
marketplace                                4.16.0    True        False         False      37m
monitoring                                 4.16.0    True        False         False      29m
network                                    4.16.0    True        False         False      38m
node-tuning                                4.16.0    True        False         False      37m
openshift-apiserver                        4.16.0    True        False         False      32m
openshift-controller-manager               4.16.0    True        False         False      30m
openshift-samples                          4.16.0    True        False         False      32m
operator-lifecycle-manager                 4.16.0    True        False         False      37m
operator-lifecycle-manager-catalog         4.16.0    True        False         False      37m
operator-lifecycle-manager-packageserver   4.16.0    True        False         False      32m
service-ca                                 4.16.0    True        False         False      38m
storage                                    4.16.0    True        False         False      37m
```

2. 配置不可用的 Operator。

23.3.3.13.1. 安装过程中删除的镜像 registry

在不提供可共享对象存储的平台上，OpenShift Image Registry Operator bootstraps 本身为 Removed。这允许 openshift-installer 在这些平台类型上完成安装。

安装后，您必须编辑 Image Registry Operator 配置，将 managementState 从 Removed 切换到

Managed。完成此操作后，您必须配置存储。

23.3.3.13.2. 镜像 registry 存储配置

对于不提供默认存储的平台，Image Registry Operator 最初不可用。安装后，您必须将 registry 配置为使用存储，以便 Registry Operator 可用。

显示配置生产集群所需的持久性卷的说明。如果适用，显示有关将空目录配置为存储位置的说明，这仅适用于非生产集群。

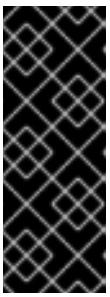
提供了在升级过程中使用 Recreate rollout 策略来允许镜像 registry 使用块存储类型的说明。

23.3.3.13.2.1. 为 VMware vSphere 配置 registry 存储

作为集群管理员，在安装后需要配置 registry 来使用存储。

先决条件

- 集群管理员权限。
- VMware vSphere 上有一个集群。
- 为集群置备的持久性存储，如 Red Hat OpenShift Data Foundation。



重要

当您只有一个副本时，OpenShift Container Platform 支持对镜像 registry 存储的 ReadWriteOnce 访问。ReadWriteOnce 访问还要求 registry 使用 Recreate rollout 策略。要部署支持高可用性的镜像 registry，需要两个或多个副本，ReadWriteMany 访问。

- 必须具有"100Gi"容量.

重要

测试显示在 RHEL 中使用 NFS 服务器作为核心服务的存储后端的问题。这包括 OpenShift Container Registry 和 Quay，Prometheus 用于监控存储，以及 Elasticsearch 用于日志存储。因此，不建议使用 RHEL NFS 作为 PV 后端用于核心服务。

市场上的其他 NFS 实现可能没有这些问题。如需了解更多与此问题相关的信息，请联络相关的 NFS 厂商。

流程

1. 要将 registry 配置为使用存储，修改 configs.imageregistry/cluster 资源中的 spec.storage.pvc。

注意

使用共享存储时，请查看您的安全设置以防止外部访问。

2. 验证您没有 registry pod:

```
$ oc get pod -n openshift-image-registry -l docker-registry=default
```

输出示例

```
No resourses found in openshift-image-registry namespace
```

注意

如果您的输出中有一个 registry pod，则不需要继续这个过程。

3. 检查 registry 配置：

```
$ oc edit configs.imageregistry.operator.openshift.io
```

输出示例

```
storage:
  pvc:
    claim: 1
```

1

将 claim 字段留空以允许自动创建 image-registry-storage 持久性卷声明(PVC)。PVC 基于默认存储类生成。但请注意，默认存储类可能会提供 ReadWriteOnce (RWO) 卷，如 RADOS 块设备(RBD)，这可能会在复制到多个副本时导致问题。

4. 检查 clusteroperator 状态：

```
$ oc get clusteroperator image-registry
```

输出示例

```
NAME           VERSION                             AVAILABLE   PROGRESSING   DEGRADED
SINCE   MESSAGE
image-registry   4.7                               True        False         False      6h50m
```

23.3.3.13.2.2. 在非生产集群中为镜像 registry 配置存储

您必须为 Image Registry Operator 配置存储。对于非生产集群，您可以将镜像 registry 设置为空目录。如果您这样做，重启 registry 时会丢失所有镜像。

流程

- 将镜像 registry 存储设置为空目录：

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch
'{"spec":{"storage":{"emptyDir":{}}}}'
```

警告

仅为非生产集群配置这个选项。

如果在 Image Registry Operator 初始化其组件前运行这个命令，oc patch 命令会失败并显示以下错误：

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster"
not found
```

等待几分钟，然后再次运行 命令。

23.3.3.13.2.3. 为 VMware vSphere 配置块 registry 存储

要允许镜像 registry 在作为集群管理员升级过程中使用块存储类型，如 vSphere Virtual Machine Disk(VMDK)，您可以使用 Recreate rollout 策略。

重要

支持块存储卷，但不建议在生产环境中用于镜像 registry。在块存储上配置 registry 的安装不具有高可用性，因为 registry 无法具有多个副本。

流程

1. 输入以下命令将镜像 registry 存储设置为块存储类型，对 registry 进行补丁，使其使用 Recreate rollout 策略，并只使用一个副本运行：

```
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec":
{"rolloutStrategy":"Recreate","replicas":1}}'
```

2. 为块存储设备置备 PV，并为该卷创建 PVC。请求的块卷使用 ReadWriteOnce(RWO)访问模式。

a. 创建包含以下内容的 pvc.yaml 文件以定义 VMware vSphere PersistentVolumeClaim 对象：

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: image-registry-storage (1)
  namespace: openshift-image-registry (2)
spec:
  accessModes:
  - ReadWriteOnce (3)
  resources:
    requests:
      storage: 100Gi (4)
```

(1) 代表 PersistentVolumeClaim 对象的唯一名称。

(2) PersistentVolumeClaim 对象的命名空间，即 openshift-image-registry。

(3) 持久性卷声明的访问模式。使用 ReadWriteOnce 时，单个节点可以通过读写权限挂载该卷。

(4) 持久性卷声明的大小。

b. 输入以下命令从文件创建 PersistentVolumeClaim 对象：

```
$ oc create -f pvc.yaml -n openshift-image-registry
```

3. 输入以下命令编辑 registry 配置，使其引用正确的 PVC：

```
$ oc edit config.imageregistry.operator.openshift.io -o yaml
```

输出示例

```
storage:
  pvc:
    claim: 1
```

1

通过创建自定义 PVC，您可以将 claim 字段留空，以便默认自动创建 image-registry-storage PVC。

有关配置 registry 存储以便引用正确的 PVC 的说明，请参阅 为 vSphere 配置 registry。

23.3.3.14. 在用户置备的基础架构上完成安装

完成 Operator 配置后，可以在您提供的基础架构上完成集群安装。

先决条件

- 您的 control plane 已初始化。
- 已完成初始 Operator 配置。

流程

1. 使用以下命令确认所有集群组件都在线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION   AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.16.0    True        False         False      19m
baremetal                                  4.16.0    True        False         False      37m
cloud-credential                           4.16.0    True        False         False      40m
cluster-autoscaler                         4.16.0    True        False         False      37m
config-operator                            4.16.0    True        False         False      38m
```

```
console                                    4.16.0    True        False        False      26m
csi-snapshot-controller                    4.16.0    True        False        False      37m
dns                                        4.16.0    True        False        False      37m
etcd                                       4.16.0    True        False        False      36m
image-registry                             4.16.0    True        False        False      31m
ingress                                    4.16.0    True        False        False      30m
insights                                   4.16.0    True        False        False      31m
kube-apiserver                             4.16.0    True        False        False      26m
kube-controller-manager                    4.16.0    True        False        False      36m
kube-scheduler                             4.16.0    True        False        False      36m
kube-storage-version-migrator              4.16.0    True        False        False      37m
machine-api                                4.16.0    True        False        False      29m
machine-approver                           4.16.0    True        False        False      37m
machine-config                             4.16.0    True        False        False      36m
marketplace                                4.16.0    True        False        False      37m
monitoring                                 4.16.0    True        False        False      29m
network                                    4.16.0    True        False        False      38m
node-tuning                                4.16.0    True        False        False      37m
openshift-apiserver                        4.16.0    True        False        False      32m
openshift-controller-manager               4.16.0    True        False        False      30m
openshift-samples                          4.16.0    True        False        False      32m
operator-lifecycle-manager                 4.16.0    True        False        False      37m
operator-lifecycle-manager-catalog         4.16.0    True        False        False      37m
operator-lifecycle-manager-packageserver   4.16.0    True        False        False      32m
service-ca                                 4.16.0    True        False        False      38m
storage                                    4.16.0    True        False        False      37m
```

另外，当所有集群都可用时，以下命令会通知您。它还检索并显示凭证：

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

1

对于 <installation_directory>，请指定安装文件保存到的目录的路径。

输出示例

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

Cluster Version Operator 完成从 Kubernetes API 服务器部署 OpenShift Container

Platform 集群时，该命令会成功。

重要

- 安装程序生成的 Ignition 配置文件包含 24 小时后过期的证书，然后在该时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 node-bootstrapper 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。

- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

2. 确认 Kubernetes API 服务器正在与 pod 通信。

   a. 要查看所有 pod 的列表，请使用以下命令：

```
$ oc get pods --all-namespaces
```

输出示例

```
NAMESPACE                  NAME                                        READY   STATUS
RESTARTS   AGE
openshift-apiserver-operator    openshift-apiserver-operator-85cb746d55-zqhs8
1/1    Running    1        9m
openshift-apiserver        apiserver-67b9g                             1/1     Running    0
3m
openshift-apiserver        apiserver-ljcmx                             1/1     Running    0
1m
openshift-apiserver        apiserver-z25h4                             1/1     Running    0
2m
openshift-authentication-operator authentication-operator-69d5d8bf84-vh2n8
1/1    Running    0        5m
...
```

.

b. 使用以下命令，查看上一命令的输出中所列 pod 的日志：

```
$ oc logs <pod_name> -n <namespace> 1
```

1 指定 pod 名称和命名空间，如上一命令的输出中所示。

如果 pod 日志显示，Kubernetes API 服务器可以与集群机器通信。

3. 对于使用光纤通道协议(FCP)的安装，还需要额外的步骤才能启用多路径。不要在安装过程中启用多路径。

如需更多信息，请参阅 *安装后机器配置任务* 文档中的"使用 RHCOS 上使用内核参数启用多路径"。

您可以按照将计算机器 添加到 vSphere 的内容在集群安装后添加额外的计算机器。

23.3.3.15. 为 control plane 节点配置 vSphere DRS 反关联性规则

可将 vSphere 分布式资源调度程序 DRS) 关联性规则配置为支持 OpenShift Container Platform Control Plane 节点的高可用性。反关联性规则确保 OpenShift Container Platform Control Plane 节点的 vSphere 虚拟机没有调度到同一 vSphere 主机。

重要

- 以下信息只适用于计算 DRS，不适用于存储 DRS。
- govc 命令是 VMware 提供的开源命令；它不是红帽提供的。红帽不支持 govc 命令。
- 有关下载和安装 govc 的说明，请参阅 VMware 文档网站。

运行以下命令来创建反关联性规则：

示例命令

```
$ govc cluster.rule.create \
  -name openshift4-control-plane-group \
  -dc MyDatacenter -cluster MyCluster \
  -enable \
  -anti-affinity master-0 master-1 master-2
```

创建规则后，您的 control plane 节点由 vSphere 自动迁移，以便它们不会在同一主机上运行。当 vSphere 协调新规则时，这可能需要一些时间。以下流程中会显示成功的命令完成。

注意

迁移会自动进行，并可能导致 OpenShift API 中断或延迟，直到迁移完成为止。

当 control plane 虚拟机名称发生变化或迁移到新的 vSphere 集群时，需要手动更新 vSphere DRS 反关联性规则。

流程

1. 运行以下命令来删除任何现有的 DRS 反关联性规则：

```
$ govc cluster.rule.remove \
  -name openshift4-control-plane-group \
  -dc MyDatacenter -cluster MyCluster
```

输出示例

```
[13-10-22 09:33:24] Reconfigure /MyDatacenter/host/MyCluster...OK
```

2. 运行以下命令，使用更新的名称再次创建规则：

```
$ govc cluster.rule.create \
  -name openshift4-control-plane-group \
  -dc MyDatacenter -cluster MyOtherCluster \
  -enable \
  -anti-affinity master-0 master-1 master-2
```

23.3.3.16. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于 远程健康监控

23.3.3.17. 后续步骤

- 自定义集群。
- 如果需要，您可以选择 不使用远程健康报告。
- 设置 registry 并配置 registry 存储。
- 可选： 查看 vSphere 问题检测器 Operator 中的事件，以确定集群是否有权限或存储配置问题。
- 可选：如果您创建了加密的虚拟机，请创建一个加密的存储类。

23.3.4. 使用自定义网络在 vSphere 上安装集群

在 OpenShift Container Platform 版本 4.16 中，您可以使用自定义的网络配置选项在 VMware vSphere 环境中安装集群。通过自定义网络配置，您的集群可以与环境中现有的 IP 地址分配共存，并与现有的 MTU 和 VXLAN 配置集成。

注意

OpenShift Container Platform 支持将集群部署到单个 VMware vCenter 中。不支持在多个 vCenter 上使用机器/机器集部署集群。

您必须在安装过程中设置大多数网络配置参数，且您只能在正在运行的集群中修改 kubeProxy 配置参数。

重要

执行用户置备的基础架构安装的步骤仅作为示例。使用您提供的基础架构安装集群需要了解 vSphere 平台和 OpenShift Container Platform 的安装过程。使用用户置备的基础架构安装说明作为指南；您可以通过其他方法创建所需的资源。

23.3.4.1. 先决条件

- 您已完成了 使用用户置备的基础架构准备安装集群 中的任务。
- 您检查了 VMware 平台许可证。红帽不会对 VMware 许可证产生任何限制，但有些 VMware 基础架构组件需要许可。
- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读有关 选择集群安装方法的文档，并为用户准备它。
- 完成安装要求您在 vSphere 主机上上传 Red Hat Enterprise Linux CoreOS(RHCOS)OVA。完成此过程的机器需要访问 vCenter 和 ESXi 主机上的端口 443。验证是否可以访问端口 443。
- 如果您使用防火墙，您与管理员确认可以访问端口 443。control plane 节点必须能够通过端口 443 访问 vCenter 和 ESXi 主机，才能成功安装。

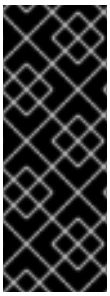
- 如果使用防火墙，则会 将其配置为允许集群需要访问的站点。

23.3.4.2. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

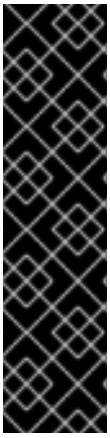


重要

如果您的集群无法直接访问互联网，则可以在置备的某些类型的基础架构上执行受限网络安装。在此过程中，您可以下载所需的内容，并使用它为镜像 registry 填充安装软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群前，您要更新镜像 registry 的内容。

23.3.4.3. VMware vSphere 区域和区启用

您可以将 OpenShift Container Platform 集群部署到在单个 VMware vCenter 中运行的多个 vSphere 数据中心。每个数据中心都可以运行多个集群。此配置降低了导致集群失败的硬件故障或网络中断的风险。要启用区域和区域，您必须为 OpenShift Container Platform 集群定义多个故障域。



重要

VMware vSphere 区域和区启用功能需要 vSphere Container Storage Interface (CSI) 驱动程序作为集群中的默认存储驱动程序。因此，这个功能只在新安装的集群中可用。

对于从上一版本升级的集群，您必须为集群启用 CSI 自动迁移。然后，您可以为升级的集群配置多个区域和区域。

默认安装配置将集群部署到单个 vSphere 数据中心。如果要将集群部署到多个 vSphere 数据中心，您必须创建一个启用地区和区功能的安装配置文件。

默认 install-config.yaml 文件包含 vcenters 和 failureDomains 字段，您可以在其中为 OpenShift Container Platform 集群指定多个 vSphere 数据中心和集群。如果要在由单个数据中心组成的 vSphere 环境中安装 OpenShift Container Platform 集群，您可以将这些字段留空。

以下列表描述了为集群定义区和区域相关的术语：

- 故障域：建立地区和区域之间的关系。您可以使用 vCenter 对象（如 datastore 对象）定义故障域。故障域定义 OpenShift Container Platform 集群节点的 vCenter 位置。
- Region ：指定 vCenter 数据中心。您可以使用 openshift-region 标签类别中的标签来定义区域。
- Zone：指定一个 vCenter 集群。您可以使用 openshift-zone 标签类别中的标签来定义区。

注意

如果您计划在 install-config.yaml 文件中指定多个故障域，则必须在创建配置文件前创建标签类别、区域标签和区域标签。

您必须为每个代表一个区域的 vCenter 数据中心创建一个 vCenter 标签。另外，您必须为比数据中心（代表一个区）中运行的每个集群创建一个 vCenter 标签。创建标签后，您必须将每个标签附加到对应的数据中心和集群。

下表概述了在单个 VMware vCenter 中运行的多个 vSphere 数据中心的区域、区域和标签之间的关系示例。

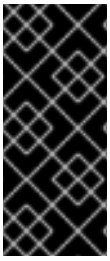
数据中心（区域）	集群（区）	Tags
us-east	us-east-1	us-east-1a
		us-east-1b
	us-east-2	us-east-2a
		us-east-2b
us-west	us-west-1	us-west-1a
		us-west-1b
	us-west-2	us-west-2a
		us-west-2b

其他资源

- 其他 VMware vSphere 配置参数
- 弃用的 VMware vSphere 配置参数
- vSphere 自动迁移
- VMware vSphere CSI Driver Operator

23.3.4.4. 手动创建安装配置文件

安装集群要求您手动创建安装配置文件。



重要

Cloud Controller Manager Operator 在提供的主机名或 IP 地址上执行连接检查。确保为可访问的 vCenter 服务器指定主机名或 IP 地址。如果您向不存在的 vCenter 服务器提供元数据，集群安装会在 bootstrap 阶段失败。

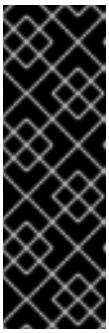
先决条件

- 您在本地机器上有一个 SSH 公钥来提供给安装程序。该密钥将用于在集群节点上进行 SSH 身份验证，以进行调试和灾难恢复。
- 已获取 OpenShift Container Platform 安装程序和集群的 pull secret。

流程

1. 创建一个安装目录来存储所需的安装资产：

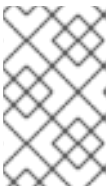
```
$ mkdir <installation_directory>
```



重要

您必须创建一个目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

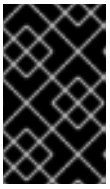
2. 自定义提供的 install-config.yaml 文件模板示例，并将其保存在 <installation_directory> 中。



注意

此配置文件必须命名为 install-config.yaml。

3. 备份 install-config.yaml 文件，以便您可以使用它安装多个集群。



重要

install-config.yaml 文件会在安装过程的下一步中使用。现在必须备份它。

其他资源

- 安装配置参数

23.3.4.4.1. VMware vSphere 的 install-config.yaml 文件示例

您可以自定义 install-config.yaml 文件，以指定有关 OpenShift Container Platform 集群平台的更多详情，或修改所需参数的值。

```
additionalTrustBundlePolicy: Proxyonly
apiVersion: v1
baseDomain: example.com (1)
compute: (2)
- architecture: amd64
  name: <worker_node>
  platform: {}
  replicas: 0 (3)
controlPlane: (4)
  architecture: amd64
  name: <parent_node>
  platform: {}
  replicas: 3 (5)
metadata:
  creationTimestamp: null
  name: test (6)
networking:
---
platform:
  vsphere:
    failureDomains: (7)
    - name: <failure_domain_name>
      region: <default_region_name>
      server: <fully_qualified_domain_name>
      topology:
        computeCluster: "/<datacenter>/host/<cluster>"
        datacenter: <datacenter> (8)
        datastore: "/<datacenter>/datastore/<datastore>" (9)
        networks:
        - <VM_Network_name>
        resourcePool: "/<datacenter>/host/<cluster>/Resources/<resourcePool>" (10)
        folder: "/<datacenter_name>/vm/<folder_name>/<subfolder_name>" (11)
      zone: <default_zone_name>
    vcenters:
    - datacenters:
```

```
    - <datacenter>
    password: <password> 12
    port: 443
    server: <fully_qualified_domain_name> 13
    user: administrator@vsphere.local
  diskType: thin 14
fips: false 15
pullSecret: '{"auths": ...}' 16
sshKey: 'ssh-ed25519 AAAA...' 17
```

1

集群的基域。所有 DNS 记录都必须是这个基域的子域，并包含集群名称。

2 4

controlPlane 部分是一个单个映射，但 compute 部分是一系列映射。为满足不同数据结构的要求，compute 部分的第一行必须以连字符 - 开头，controlPlane 部分 的第一行则不以连字符开头。两个部分都定义单个机器池，因此只使用一个 control plane。OpenShift Container Platform 不支持定义多个计算池。

3

replicas 参数的值必须设置为 0。此参数控制集群为您创建和管理的 worker 数量，在使用用户置备的基础架构时集群不会执行这些功能。在完成 OpenShift Container Platform 安装前，您必须手动为集群部署 worker 机器。

5

您添加到集群的 control plane 机器数量。由于集群使用此值作为集群中的 etcd 端点数量，所以该值必须与您部署的 control plane 机器数量匹配。

6

您在 DNS 记录中指定的集群名称。

7

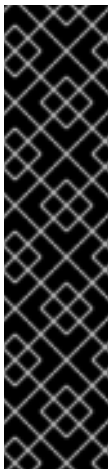
建立地区和区域之间的关系。您可以使用 vCenter 对象（如 datastore 对象）定义故障域。故障域定义 OpenShift Container Platform 集群节点的 vCenter 位置。

8

vSphere 数据中心.

9

保存虚拟机文件、模板和 ISO 镜像的 vSphere 数据存储路径。



重要

您可以指定数据存储集群中存在的任何数据存储路径。默认情况下，Storage vMotion 会自动为数据存储集群启用。红帽不支持 Storage vMotion，因此您必须禁用 Storage vMotion 以避免 OpenShift Container Platform 集群的数据丢失问题。

如果需要在多个数据存储间指定虚拟机，请使用 数据存储 对象在集群 install-config.yaml 配置文件中指定故障域。如需更多信息，请参阅"VMware vSphere 区域和区启用"。

10

可选：对于安装程序置备的基础架构，安装程序创建虚拟机的现有资源池的绝对路径，例如 /<datacenter_name>/host/<cluster_name>/Resources/<resource_pool_name>/<optional_nested_resource_pool_name>。如果没有指定值，则会在集群 /example_datacenter/host/example_cluster/Resources 根中安装资源。

11

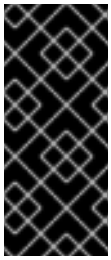
可选：对于安装程序置备的基础架构，安装程序创建虚拟机的现有文件夹的绝对路径，如 /<datacenter_name>/vm/<folder_name>/<subfolder_name>。如果没有提供这个值，安装程序会在数据中心虚拟机文件夹中创建一个顶层文件夹，其名称为基础架构 ID。如果您为集群提供基础架构，且您不想使用默认的 StorageClass 对象（名为 thin），您可以从 install-config.yaml 文件中省略 folder 参数。

12

与 vSphere 用户关联的密码。

13

vCenter 服务器的完全限定主机名或 IP 地址。



重要

Cloud Controller Manager Operator 在提供的主机名或 IP 地址上执行连接检查。确保为可访问的 vCenter 服务器指定主机名或 IP 地址。如果您向不存在的 vCenter 服务器提供元数据，集群安装会在 bootstrap 阶段失败。

14

vSphere 磁盘置备方法。

15

重要

要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 Red Hat Enterprise Linux (RHEL) 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅在 FIPS 模式中安装该系统。

当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS)时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。

16

从 OpenShift Cluster Manager 获取的 pull secret。此 pull secret 允许您与所含授权机构提供的服务进行身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

17

Red Hat Enterprise Linux CoreOS(RHCOS)中 core 用户的默认 SSH 密钥的公钥部分。

23.3.4.4.2. 在安装过程中配置集群范围的代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 install-config.yaml 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 install-config.yaml 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 Proxy 对象的 spec.noProxy 字段中添加站点来绕过代理。

注意

Proxy 对象 status.noProxy 字段使用安装配置中的 networking.machineNetwork[].cidr、networking.clusterNetwork[].cidr 和 networking.serviceNetwork[] 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，Proxy 对象 status.noProxy 字段也会使用实例元数据端点填充(169.254.169.254)。

流程

1. 编辑 install-config.yaml 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 http。

2. 用于创建集群外 HTTPS 连接的代理 URL。

3. 要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 . 以仅匹配子域。例如，.y.com 匹配 x.y.com，但不匹配 y.com。使用 * 绕过所有目的地的代理。您必须包含 vCenter 的 IP 地址以及用于其机器的 IP 范围。

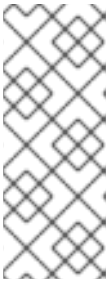
4. 如果提供，安装程序会在 openshift-config 命名空间中生成名为 user-ca-bundle 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 trusted-ca-bundle 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， Proxy 对象的 trustedCA 字段中

也会引用此配置映射。additionalTrustBundle 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

5 可选：决定 Proxy 对象的配置以引用 trustedCA 字段中 user-ca-bundle 配置映射的策略。允许的值是 Proxyonly 和 Always。仅在配置了 http/https 代理时，使用 Proxyonly 引用 user-ca-bundle 配置映射。使用 Always 始终引用 user-ca-bundle 配置映射。默认值为 Proxyonly。

注意

安装程序不支持代理的 readinessEndpoints 字段。



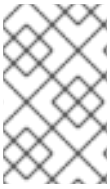
注意

如果安装程序超时，重启并使用安装程序的 wait-for 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. 保存该文件并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 cluster 的集群范围代理，该代理 使用 提供的 install-config.yaml 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 cluster Proxy 对象，但它会有一个空 spec。



注意

只支持名为 cluster 的 Proxy 对象，且无法创建额外的代理。

23.3.4.4.3. 为 VMware vCenter 配置区域和区域

您可以修改默认安装配置文件，以便您可以将 OpenShift Container Platform 集群部署到在单个 VMware vCenter 中运行的多个 vSphere 数据中心。

之前版本的 OpenShift Container Platform 的默认 install-config.yaml 文件配置已弃用。您可以继续使用已弃用的默认配置，但 openshift-installer 会提示您显示在配置文件中已弃用字段的警告信息。

重要

这个示例使用 govc 命令。govc 命令是 VMware 提供的开源命令；它不是红帽提供的。红帽支持团队不维护 govc 命令。有关下载和安装 govc 的说明，请参阅 VMware 文档网站

先决条件

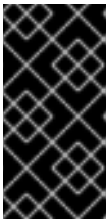
- 您有一个现有的 install-config.yaml 安装配置文件。

重要

您必须为 OpenShift Container Platform 集群指定一个故障域，以便您可以为 VMware vCenter 服务器置备数据中心对象。如果您需要在不同的数据中心、集群、数据存储和其他组件中置备虚拟机节点，请考虑指定多个故障域。要启用区域和区域，您必须为 OpenShift Container Platform 集群定义多个故障域。

流程

1. 输入以下 govc 命令行工具命令，以创建 openshift-region 和 openshift-zone vCenter 标签类别：



重要

如果为 openshift-region 和 openshift-zone vCenter 标签类别指定不同的名称，OpenShift Container Platform 集群的安装会失败。

```
$ govc tags.category.create -d "OpenShift region" openshift-region
```

```
$ govc tags.category.create -d "OpenShift zone" openshift-zone
```

2. 要为您要部署集群的每个区域 vSphere 数据中心创建一个 region 标签，请在终端中输入以下命令：

```
$ govc tags.create -c <region_tag_category> <region_tag>
```

3. 要为您要部署集群的每个 vSphere 集群创建一个区标签，请输入以下命令：

```
$ govc tags.create -c <zone_tag_category> <zone_tag>
```

4. 输入以下命令将区域标签附加到每个 vCenter 数据中心对象：

```
$ govc tags.attach -c <region_tag_category> <region_tag_1> /<datacenter_1>
```

5. 输入以下命令将区标签附加到每个 vCenter 数据中心对象：

```
$ govc tags.attach -c <zone_tag_category> <zone_tag_1> /<datacenter_1>/host/vcs-mdcnc-workload-1
```

6. 进入包含安装程序的目录，并根据您选择的安装要求初始化集群部署。

在 vSphere 数据中心中定义的多个数据中心的 install-config.yaml 文件示例

```
---
compute:
---
  vsphere:
    zones:
      - "<machine_pool_zone_1>"
      - "<machine_pool_zone_2>"
---
controlPlane:
---
vsphere:
    zones:
      - "<machine_pool_zone_1>"
      - "<machine_pool_zone_2>"
---
platform:
  vsphere:
    vcenters:
---
    datacenters:
      - <datacenter1_name>
      - <datacenter2_name>
    failureDomains:
    - name: <machine_pool_zone_1>
      region: <region_tag_1>
      zone: <zone_tag_1>
      server: <fully_qualified_domain_name>
      topology:
        datacenter: <datacenter1>
        computeCluster: "/<datacenter1>/host/<cluster1>"
        networks:
        - <VM_Network1_name>
```

```
      datastore: "/<datacenter1>/datastore/<datastore1>"
      resourcePool: "/<datacenter1>/host/<cluster1>/Resources/<resourcePool1>"
      folder: "/<datacenter1>/vm/<folder1>"
    - name: <machine_pool_zone_2>
      region: <region_tag_2>
      zone: <zone_tag_2>
      server: <fully_qualified_domain_name>
      topology:
        datacenter: <datacenter2>
        computeCluster: "/<datacenter2>/host/<cluster2>"
        networks:
        - <VM_Network2_name>
        datastore: "/<datacenter2>/datastore/<datastore2>"
        resourcePool: "/<datacenter2>/host/<cluster2>/Resources/<resourcePool2>"
        folder: "/<datacenter2>/vm/<folder2>"
---
```

23.3.4.5. 网络配置阶段

OpenShift Container Platform 安装前有两个阶段，您可以在其中自定义网络配置。

第 1 阶段

在创建清单文件前，您可以自定义 **install-config.yaml** 文件中的以下与网络相关的字段：

- networking.networkType
- networking.clusterNetwork
- networking.serviceNetwork
- networking.machineNetwork

有关这些字段的更多信息，请参阅 *安装配置参数*。

注意

将 networking.machineNetwork 设置为与首选 NIC 所在的 CIDR 匹配。

重要

CIDR 范围 172.17.0.0/16 由 libVirt 保留。对于集群中的任何网络，您无法使用此范围或与这个范围重叠的范围。

第 2 阶段

运行 openshift-install create 清单创建 清单文件后，您可以只使用您要修改的字段定义自定义 Cluster Network Operator 清单。您可以使用 清单指定高级网络配置。

您不能覆盖在 stage 2 阶段 1 中在 install-config.yaml 文件中指定的值。但是，您可以在第 2 阶段进一步自定义网络插件。

23.3.4.6. 指定高级网络配置

您可以使用网络插件的高级网络配置将集群集成到现有网络环境中。您只能在安装集群前指定高级网络配置。

重要

不支持通过修改安装程序创建的 OpenShift Container Platform 清单文件来自定义网络配置。支持应用您创建的清单文件，如以下流程中所示。

先决条件

- 您已创建 install-config.yaml 文件并完成对其所做的任何修改。

流程

1. 进入包含安装程序的目录并创建清单：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1

<installation_directory> 指定包含集群的 install-config.yaml 文件的目录名称。

2. 在 <installation_directory>/manifests/ 目录中 为高级网络配置创建一个名为 cluster-network-03-config.yml 的 stub 清单文件：

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
```

3. 在 cluster-network-03-config.yml 文件中指定集群的高级网络配置，如下例所示：

为 OVN-Kubernetes 网络供应商启用 IPsec

```
apiVersion: operator.openshift.io/v1
kind: Network
metadata:
  name: cluster
spec:
  defaultNetwork:
    ovnKubernetesConfig:
      ipsecConfig:
        mode: Full
```

4. 可选：备份 manifests/cluster-network-03-config.yml 文件。创建 Ignition 配置文件时，安装程序会使用 manifests/ 目录。

5. 删除定义 control plane 机器的 Kubernetes 清单文件和计算 machineSets：

```
$ rm -f openshift/99_openshift-cluster-api_master-machines-*.yaml
openshift/99_openshift-cluster-api_worker-machineset-*.yaml
```

由于您要自行创建和管理这些资源，因此不必初始化这些资源。

- 您可以使用机器 API 保留 MachineSet 文件以创建计算机器，但您必须更新对它们的引用以匹配您的环境。

23.3.4.7. Cluster Network Operator 配置

集群网络的配置作为 Cluster Network Operator(CNO)配置的一部分指定，并存储在名为 cluster 的自定义资源(CR)对象中。CR 指定 operator.openshift.io API 组中的 Network API 的字段。

CNO 配置在集群安装过程中从 Network.config.openshift.io API 组中的 Network API 继承以下字段：

clusterNetwork

从中分配 Pod IP 地址的 IP 地址池。

serviceNetwork

服务的 IP 地址池.

defaultNetwork.type

集群网络插件。OVNKubernetes 是安装期间唯一支持的插件。

您可以通过在名为 cluster 的 CNO 对象中设置 defaultNetwork 对象的字段来为集群指定集群网络插件配置。

23.3.4.7.1. Cluster Network Operator 配置对象

下表中描述了 Cluster Network Operator(CNO)的字段：

表 23.29. Cluster Network Operator 配置对象

字段	类型	描述
metadata.name	字符串	CNO 对象的名称。这个名称始终是 集群。

字段	类型	描述
spec.clusterNetwork	**array**	用于指定从哪些 IP 地址块分配 Pod IP 地址以及集群中每个节点的子网前缀长度的列表。例如： spec: clusterNetwork: - cidr: 10.128.0.0/19 hostPrefix: 23 - cidr: 10.128.32.0/19 hostPrefix: 23
spec.serviceNetwork	**array**	服务的 IP 地址块。OpenShift SDN 和 OVN-Kubernetes 网络插件只支持服务网络的一个 IP 地址块。例如： spec: serviceNetwork: - 172.30.0.0/14 您只能在创建清单前在 **install-config.yaml** 文件中自定义此字段。该值在清单文件中是只读的。
spec.defaultNetwork	**object**	为集群网络配置网络插件。
spec.kubeProxyConfig	**object**	此对象的字段指定 kube-proxy 配置。如果使用 OVN-Kubernetes 集群网络供应商，则 kube-proxy 配置不会起作用。

defaultNetwork 对象配置

下表列出了 defaultNetwork 对象的值：

表 23.30. defaultNetwork 对象

字段	类型	描述
type	**字符串**	**OVNKubernetes**。Red Hat OpenShift Networking 网络插件在安装过程中被选择。此值在集群安装后无法更改。 **注意** OpenShift Container Platform 默认使用 OVN-Kubernetes 网络插件。OpenShift SDN 不再作为新集群的安装选择提供。
ovnKubernetesConfig	**object**	此对象仅对 OVN-Kubernetes 网络插件有效。

配置 OVN-Kubernetes 网络插件

下表描述了 OVN-Kubernetes 网络插件的配置字段：

表 23.31. ovnKubernetesConfig object

字段	类型	描述
mtu	**integer**	Geneve（通用网络虚拟化封装）覆盖网络的最大传输单元 (MTU)。这根据主网络接口的 MTU 自动探测。您通常不需要覆盖检测到的 MTU。 如果自动探测的值不是您期望的值，请确认节点上主网络接口上的 MTU 是否正确。您不能使用这个选项更改节点上主网络接口的 MTU 值。 如果集群中不同节点需要不同的 MTU 值，则必须将此值设置为 **比** 集群中的最低 MTU 值小 100。例如，如果集群中的某些节点的 MTU 为 **9001**，而某些节点的 MTU 为 **1500**，则必须将此值设置为 **1400**。
genevePort	**integer**	用于所有 Geneve 数据包的端口。默认值为 **6081**。此值在集群安装后无法更改。
ipsecConfig	**object**	指定用于自定义 IPsec 配置的配置对象。
ipv4	**object**	为 IPv4 设置指定配置对象。
ipv6	**object**	为 IPv6 设置指定配置对象。
policyAuditConfig	**object**	指定用于自定义网络策略审计日志的配置对象。如果未设置，则使用默认的审计日志设置。
gatewayConfig	**object**	可选：指定一个配置对象来自定义如何将出口流量发送到节点网关。 **注意** 在迁移出口流量时，工作负载和服务流量会受到一定影响，直到 Cluster Network Operator (CNO) 成功推出更改。

表 23.32. ovnKubernetesConfig.ipv4 object

字段	类型	描述

字段	类型	描述
internalTransitSwitchSubnet	字符串	如果您的现有网络基础架构与 **100.88.0.0/16** IPv4 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。启用东西流量的分布式传输交换机的子网。此子网不能与 OVN-Kubernetes 或主机本身使用的任何其他子网重叠。必须足够大，以适应集群中的每个节点一个 IP 地址。 默认值为 **100.88.0.0/16**。
internalJoinSubnet	字符串	如果您的现有网络基础架构与 **100.64.0.0/16** IPv4 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。您必须确保 IP 地址范围没有与 OpenShift Container Platform 安装使用的任何其他子网重叠。IP 地址范围必须大于可添加到集群的最大节点数。例如，如果 **clusterNetwork.cidr** 值为 **10.128.0.0/14**，并且 **clusterNetwork.hostPrefix** 值为 **/23**，则最大节点数量为 **2^(23-14)=512**。 默认值为 **100.64.0.0/16**。

表 23.33. ovnKubernetesConfig.ipv6 object

字段	类型	描述
internalTransitSwitchSubnet	字符串	如果您的现有网络基础架构与 **fd97::/64** IPv6 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。启用东西流量的分布式传输交换机的子网。此子网不能与 OVN-Kubernetes 或主机本身使用的任何其他子网重叠。必须足够大，以适应集群中的每个节点一个 IP 地址。 默认值为 **fd97::/64**。
internalJoinSubnet	字符串	如果您的现有网络基础架构与 **fd98::/64** IPv6 子网重叠，您可以指定不同的 IP 地址范围供 OVN-Kubernetes 使用。您必须确保 IP 地址范围没有与 OpenShift Container Platform 安装使用的任何其他子网重叠。IP 地址范围必须大于可添加到集群的最大节点数。 默认值为 **fd98::/64**。

表 23.34. policyAuditConfig object

字段	类型	描述
rateLimit	整数	每个节点每秒生成一次的消息数量上限。默认值为每秒 **20** 条消息。
maxFileSize	整数	审计日志的最大大小，以字节为单位。默认值为 **50000000** 或 50 MB。
maxLogFiles	整数	保留的日志文件的最大数量。

字段	类型	描述
目的地	字符串	以下附加审计日志目标之一： **libc** 主机上的 journald 进程的 libc **syslog（）** 函数。 **UDP:<host>:<port>** 一个 syslog 服务器。将 **<host>:<port>** 替换为 **syslog 服务器的主机** 和端口。 **Unix:<file>** 由 **<file>** 指定的 Unix 域套接字文件。 **null** 不要将审计日志发送到任何其他目标。
syslogFacility	字符串	syslog 工具，如 as **kern**，如 RFC5424 定义。默认值为 **local0**。

表 23.35. gatewayConfig object

字段	类型	描述
routingViaHost	**布尔值**	将此字段设置为 **true**，将来自 pod 的出口流量发送到主机网络堆栈。对于依赖于在内核路由表中手动配置路由的高级别安装和应用程序，您可能需要将出口流量路由到主机网络堆栈。默认情况下，出口流量在 OVN 中进行处理以退出集群，不受内核路由表中的特殊路由的影响。默认值为 **false**。 此字段与 Open vSwitch 硬件卸载功能有交互。如果将此字段设置为 **true**，则不会获得卸载的性能优势，因为主机网络堆栈会处理出口流量。
ipForwarding	**object**	您可以使用 **Network** 资源中的 **ipForwarding** 规格来控制 OVN-Kubernetes 管理接口上所有流量的 IP 转发。指定 **Restricted** 只允许 Kubernetes 相关流量的 IP 转发。指定 **Global** 以允许转发所有 IP 流量。对于新安装，默认值为 **Restricted**。对于到 OpenShift Container Platform 4.14 或更高版本的更新，默认值为 **Global**。
ipv4	**object**	可选：指定一个对象来为主机配置内部 OVN-Kubernetes 伪装地址，以服务 IPv4 地址的流量。
ipv6	**object**	可选：指定一个对象来为主机配置内部 OVN-Kubernetes 伪装地址，以服务 IPv6 地址的流量。

表 23.36. gatewayConfig.ipv4 对象

字段	类型	描述
internalMasqueradeSubnet	**字符串**	内部使用的伪装 IPv4 地址，以启用主机服务流量。主机配置了这些 IP 地址和共享网关网桥接口。默认值为 **169.254.169.0/29**。

表 23.37. gatewayConfig.ipv6 对象

字段	类型	描述
internalMasqueradeSubnet	**字符串**	内部使用的伪装 IPv6 地址，以启用主机服务流量。主机配置了这些 IP 地址和共享网关网桥接口。默认值为 **fd69::/125**。

表 23.38. ipsecConfig 对象

字段	类型	描述
模式	**字符串**	指定 IPsec 实现的行为。必须是以下值之一： • **Disabled**: 在集群节点上不启用 IPsec。 • **External** ：对于带有外部主机的网络流量，启用 IPsec。 • **Full**: IPsec 为带有外部主机的 pod 流量和网络流量启用 IPsec。

启用 IPSec 的 OVN-Kubernetes 配置示例

```
defaultNetwork:
  type: OVNKubernetes
  ovnKubernetesConfig:
    mtu: 1400
    genevePort: 6081
      ipsecConfig:
        mode: Full
```

重要

使用 OVNKubernetes 可能会导致 IBM Power® 上的堆栈耗尽问题。

kubeProxyConfig 对象配置（仅限 OpenShiftSDN 容器网络接口）

kubeProxyConfig 对象的值在下表中定义：

表 23.39. kubeProxyConfig object

字段	类型	描述
iptablesSyncPeriod	**字符串**	**iptables** 规则的刷新周期。默认值为 **30s**。有效的后缀包括 **s**、**m** 和 **h**，具体参见 Go 时间 包 文档。 **注意** 由于 OpenShift Container Platform 4.3 及更高版本中引进了性能改进，不再需要调整 **iptablesSyncPeriod** 参数。
proxyArguments.iptables-min-sync-period	**array**	刷新 **iptables** 规则前的最短持续时间。此字段确保刷新的频率不会过于频繁。有效的后缀包括 **s**、**m** 和 **h**，具体参见 Go **time** 软件包。默认值为： kubeProxyConfig: proxyArguments: iptables-min-sync-period: - 0s

23.3.4.8. 创建 Ignition 配置文件

由于您必须手动启动集群机器，因此您必须生成 Ignition 配置文件，集群需要这些配置文件来创建其机器。

重要

- **安装程序生成的 Ignition 配置文件包含 24 小时后过期的证书，然后在该时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 node-bootstrapper 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。**
- **建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。**

先决条件

- 获取 OpenShift Container Platform 安装程序和集群的 pull secret。

流程

- 获取 Ignition 配置文件：

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

1 对于 <installation_directory>，请指定要存储安装程序创建的文件的目录名称。

重要

如果创建了 install-config.yaml 文件，请指定包含该文件的目录。否则，指定一个空目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

该目录中会生成以下文件：

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

23.3.4.9. 提取基础架构名称

Ignition 配置文件包含一个唯一集群标识符，您可以使用它在 VMware vSphere 中唯一地标识您的集群。如果计划使用集群标识符作为虚拟机文件夹的名称，则必须提取它。

先决条件

- 已获取 OpenShift Container Platform 安装程序和集群的 pull secret。
- 已为集群生成 Ignition 配置文件。
- 已安装 jq 软件包。

流程

- 要从 Ignition 配置文件元数据中提取和查看基础架构名称，请运行以下命令：

```
$ jq -r .infraID <installation_directory>/metadata.json 1
```

1 对于 <installation_directory>，请指定安装文件保存到的目录的路径。

输出示例

```
openshift-vw9j6 1
```

1 此命令的输出是您的集群名称和随机字符串。

23.3.4.10. 安装 RHCOS 并启动 OpenShift Container Platform bootstrap 过程

要在 VMware vSphere 上的用户置备的基础架构上安装 OpenShift Container Platform，您必须在 vSphere 主机上安装 Red Hat Enterprise Linux CoreOS(RHCOS)。安装 RHCOS 时，您必须为您要安装的机器类型提供 OpenShift Container Platform 安装程序生成的 Ignition 配置文件。如果您配置了适当的网络、DNS 和负载均衡基础架构，OpenShift Container Platform bootstrap 过程会在 RHCOS 机器重启后自动启动。

先决条件

- 已获取集群的 Ignition 配置文件。
- 具有 HTTP 服务器的访问权限，以便您可从计算机进行访问，并且您创建的机器也可访问此服务器。
- 您已创建了 vSphere 集群。

流程

1. 将名为 <installation_directory>/bootstrap.ign 的 bootstrap Ignition 配置文件上传到 HTTP 服务器。注意此文件的 URL。
2. 将 bootstrap 节点的以下辅助 Ignition 配置文件保存到计算机中，存为 <installation_directory>/merge-bootstrap.ign ：

```
{
  "ignition": {
    "config": {
      "merge": [
        {
          "source": "<bootstrap_ignition_config_url>", 1
          "verification": {}
        }
      ]
    },
    "timeouts": {},
    "version": "3.2.0"
  },
  "networkd": {},
  "passwd": {},
  "storage": {},
  "systemd": {}
}
```

1. 指定您托管的 bootstrap Ignition 配置文件的 URL。

为 bootstrap 机器创建虚拟机(VM)时，您要使用此 Ignition 配置文件。

3. 找到安装程序创建的以下 Ignition 配置文件：

   - <installation_directory>/master.ign
   - <installation_directory>/worker.ign
   - <installation_directory>/merge-bootstrap.ign

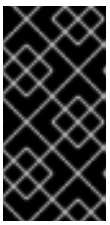
4. 将 Ignition 配置文件转换为 Base64 编码。在此流程中，您必须将这些文件添加到虚拟机中的额外配置参数 guestinfo.ignition.config.data 中。

   例如，如果使用 Linux 操作系统，您可以使用 base64 命令对文件进行编码。

```
$ base64 -w0 <installation_directory>/master.ign > <installation_directory>/master.64
```

```
$ base64 -w0 <installation_directory>/worker.ign > <installation_directory>/worker.64
```

```
$ base64 -w0 <installation_directory>/merge-bootstrap.ign >
<installation_directory>/merge-bootstrap.64
```



重要

如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

5. 获取 RHCOS OVA 镜像。镜像位于 RHCOS 镜像镜像页面。

重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每个发行版本而改变。您必须下载最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。如果可用，请使用与 OpenShift Container Platform 版本匹配的镜像版本。

文件名包含 OpenShift Container Platform 版本号，格式为 rhcos-vmware.

<architecture>.ova。

6. **在 vSphere 客户端中，在数据中心中创建一个文件夹来存储虚拟机。**

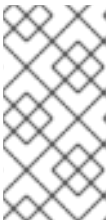
   a. 单击 **VMs and Templates** 视图。

   b. **右键单击您的数据中心的名称。**

   c. **点击 New Folder → New VM and Template Folder。**

   d. **在显示的窗口中，输入文件夹名称。如果您没有在 install-config.yaml 文件中指定现有文件夹，请创建一个名称与基础架构 ID 相同的文件夹。您可以使用这个文件夹名称，因此 vCenter 会在适当的位置为 Workspace 配置动态置备存储。**

7. **在 vSphere 客户端中，为 OVA 镜像创建一个模板，然后根据需要克隆模板。**

   注意

   在以下步骤中，您将创建模板，然后克隆所有集群机器的模板。然后，您在置备虚拟机时为该克隆的机器类型提供 Ignition 配置文件的位置。

   a. **在 Hosts and Clusters 选项卡中，右键点击您的集群名称并选择 Deploy OVF Template。**

   b. **在 Select an OVF 选项卡中，指定您下载的 RHCOS OVA 文件的名称。**

   c. **在 Select a name and folder 选项卡中，为您的模板设置 虚拟机名称，如 Template-RHCOS。点击 vSphere 集群的名称并选择您在上一步中创建的文件夹。**

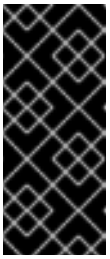
   d. **在 Select a compute resource 选项卡中，点击 vSphere 集群的名称。**

e. 在 Select storage 选项卡中，配置虚拟机的存储选项。

- 根据您的存储首选项，选择 Thin Provision 或 Thick Provision。
- 选择您在 install-config.yaml 文件中指定的数据存储。
- 如果要加密虚拟机，请选择 Encrypt this virtual machine。如需更多信息，请参阅标题为"加密虚拟机的要求"的部分。

f. 在 Select network 选项卡中，指定您为集群配置的网络（如果可用）。

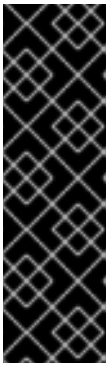
g. 在创建 OVF 模板时，不要在 Customize template 选项卡上指定值，也不会进一步配置模板。



重要

不要启动原始虚拟机模板。VM 模板必须保持关闭，必须为新的 RHCOS 机器克隆。启动虚拟机模板会将虚拟机模板配置为平台上的虚拟机，这样可防止它被用作计算机器集可应用配置的模板。

8. 可选： 如果需要，更新 VM 模板中配置的虚拟硬件版本。如需更多信息，请参阅 VMware 文档中的将虚拟机升级到最新硬件版本。



重要

如有必要，建议您在从虚拟机创建虚拟机前将虚拟机模板的硬件版本更新为版本 15。在 vSphere 上运行的集群节点使用硬件版本 13 现已弃用。如果您导入的模板默认为硬件版本 13，您必须在将 VM 模板升级到硬件版本 15 前确保 ESXi 主机为 6.7U3 或更高版本。如果您的 vSphere 版本小于 6.7U3，您可以跳过此升级步骤；但是，计划将来的 OpenShift Container Platform 版本删除对小于 6.7U3 的硬件版本 13 和 vSphere 版本的支持。

9. 部署模板后，为集群中的机器部署虚拟机。

a. 右键点击模板名称，再点击 Clone → Clone to Virtual Machine。

b. 在 Select a name and folder 选项卡中，指定虚拟机的名称。您可以在名称中包含机器类型，如 control-plane-0 或 compute-1。

注意

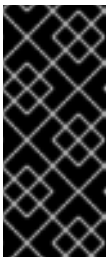
确保 vSphere 安装中的所有虚拟机名称都是唯一的。

c. 在 Select a name and folder 选项卡中，选择您为集群创建的文件夹名称。

d. 在 Select a compute resource 选项卡中，选择数据中心中的主机名称。

e. 在 Select clone options 选项卡中，选择 Customize this virtual machine's hardware。

f. 在 Customize hardware 选项卡上，点 Advanced Parameters。



重要

以下配置建议仅用于演示目的。作为集群管理员，您必须根据集群上的资源需求来配置资源。为了更好地管理集群资源，请考虑从集群的 root 资源池创建资源池。

- 可选：覆盖 vSphere 中的默认 DHCP 网络。启用静态 IP 网络：
  - 设置静态 IP 配置：

示例命令

```
$ export IPCFG="ip=<ip>::<gateway>:<netmask>:<hostname>:
<iface>:none nameserver=srv1 [nameserver=srv2 [nameserver=srv3 [...]]]"
```

示例命令

```
$ export IPCFG="ip=192.168.100.101::192.168.100.254:255.255.255.0:::none
nameserver=8.8.8.8"
```

- 在从 vSphere 中的 OVA 引导虚拟机前，设置 guestinfo.afterburn.initrd.network-kargs 属性：

示例命令

```
$ govc vm.change -vm "<vm_name>" -e
"guestinfo.afterburn.initrd.network-kargs=${IPCFG}"
```

- 通过在 Attribute 和 Values 字段中指定数据来添加以下配置参数名称和值。确保为您创建的每个参数选择 Add 按钮。
  - guestinfo.ignition.config.data ：找到您在此流程中创建的 base-64 编码文件，并粘贴此机器类型的 base64 编码 Ignition 配置文件的内容。
  - guestinfo.ignition.config.data.encoding ：指定 base64。
  - disk.EnableUUID ：指定 TRUE。
  - stealclock.enable ：如果没有定义此参数，请添加它并指定 TRUE。
  -

从集群的 root 资源池创建子资源池。执行此子资源池中的资源分配。

g. 在 Customize hardware 选项卡的 Virtual Hardware 面板中，根据需要修改指定的值。确保 RAM、CPU 和磁盘存储的数量满足机器类型的最低要求。

h. 完成剩余的配置步骤。点 Finish 按钮，您已完成克隆操作。

i. 在 Virtual Machines 选项卡中，右键点您的虚拟机，然后选择 Power → Power On。

j. 检查控制台输出，以验证 Ignition 是否运行。

示例命令

```
Ignition: ran on 2022/03/14 14:48:33 UTC (this boot)
Ignition: user-provided config was applied
```

后续步骤

- 对每台机器执行前面的步骤，为集群创建其余机器。

重要

此时您必须创建 bootstrap 和 control plane 机器。由于计算机器上已默认部署了一些 Pod，因此还要在安装集群前至少创建两台计算机器。

23.3.4.11. 将更多计算机器添加到 vSphere 中的集群

您可以将更多计算机器添加到 VMware vSphere 上的用户置备的 OpenShift Container Platform 集群中。

在 OpenShift Container Platform 集群中部署 vSphere 模板后，您可以为该集群中的机器部署虚拟机(VM)。

先决条件

- 获取计算机器的 base64 编码 Ignition 文件。
- 您可以访问您为集群创建的 vSphere 模板。

流程

1. 右键点击模板的名称，再点击 Clone → Clone to Virtual Machine。
2. 在 Select a name and folder 选项卡中，指定虚拟机的名称。您可以在名称中包含机器类型，如 compute-1。

   注意

   确保 vSphere 安装中的所有虚拟机名称都是唯一的。
3. 在 Select a name and folder 选项卡中，选择您为集群创建的文件夹名称。
4. 在 Select a compute resource 选项卡中，选择数据中心中的主机名称。
5. 在 Select storage 选项卡中，为您的配置和磁盘文件选择存储。
6. 在 Select clone options 选项卡中，选择 Customize this virtual machine's hardware。
7. 在 Customize hardware 选项卡上，点 Advanced Parameters。

   - 通过在 Attribute 和 Values 字段中指定数据来添加以下配置参数名称和值。确保为您创建的每个参数选择 Add 按钮。

     - guestinfo.ignition.config.data ：粘贴此机器类型的 base64 编码计算 Ignition

配置文件的内容。

- guestinfo.ignition.config.data.encoding ：指定 base64。
- disk.EnableUUID ：指定 TRUE。

8. 在 Customize hardware 选项卡的 Virtual Hardware 面板中，根据需要修改指定的值。确保 RAM、CPU 和磁盘存储的数量满足机器类型的最低要求。如果存在多个网络，请选择 Add New Device > Network Adapter，然后在 New Network 菜单项提供的字段中输入您的网络信息。

9. 完成剩余的配置步骤。点 Finish 按钮，您已完成克隆操作。

10. 在 Virtual Machines 选项卡中，右键点您的虚拟机，然后选择 Power → Power On。

后续步骤

- 继续为集群创建更多计算机器。

23.3.4.12. 磁盘分区

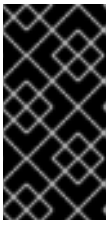
在大多数情况下，数据分区最初是由安装 RHCOS 而不是安装另一个操作系统来创建的。在这种情况下，OpenShift Container Platform 安装程序被允许配置磁盘分区。

但是，在安装 OpenShift Container Platform 节点时，在两种情况下您可能需要覆盖默认分区：

- 创建单独的分区： 要在空磁盘上进行 greenfield 安装，您可能需要在分区中添加单独的存储。这正式支持生成 /var 或 /var 的子目录， 如 /var/lib/etcd （独立分区），但不支持两者。

重要

对于大于 100GB 的磁盘大小，特别是磁盘大小大于 1TB，请创建一个独立的 /var 分区。如需更多信息，请参阅"创建独立 /var 分区"和 红帽知识库文章。



重要

Kubernetes 仅支持两个文件系统分区。如果您在原始配置中添加多个分区，Kubernetes 无法监控所有这些分区。

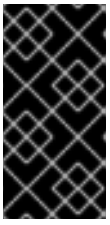
- 保留现有分区：对于 brownfield 安装，您要在现有节点上重新安装 OpenShift Container Platform，并希望保留从之前的操作系统中安装的数据分区，对于 coreos-installer 来说，引导选项和选项都允许您保留现有数据分区。

创建独立 /var 分区

通常，OpenShift Container Platform 的磁盘分区应该保留给安装程序。然而，在有些情况下您可能需要在文件系统的一部分中创建独立分区。

OpenShift Container Platform 支持添加单个分区来将存储附加到 /var 分区或 /var 的子目录中。例如：

- /var/lib/containers ：保存随着系统中添加更多镜像和容器而增长的容器相关内容。
- /var/lib/etcd ：保存您可能希望独立保留的数据，比如 etcd 存储的性能优化。
- /var ：保存您可能希望独立保留的数据，以满足审计等目的。



重要

对于大于 100GB 的磁盘大小，特别是磁盘大小大于 1TB，请创建一个独立的 /var 分区。

通过单独存储 /var 目录的内容，可以更轻松地根据需要为区域扩展存储，并在以后重新安装 OpenShift Container Platform，并保持该数据的完整性。使用这个方法，您不必再次拉取所有容器，在更新系统时也不必复制大量日志文件。

因为 /var 在进行一个全新的 Red Hat Enterprise Linux CoreOS（RHCOS）安装前必需存在，所以这个流程会在 OpenShift Container Platform 安装过程的 openshift-install 准备阶段插入一个创建的机器配置清单的机器配置来设置独立的 /var 分区。

流程

1. 创建存放 OpenShift Container Platform 安装文件的目录：

```
$ mkdir $HOME/clusterconfig
```

2. 运行 openshift-install，以在 manifest 和 openshift 子目录中创建一组文件。在系统提示时回答系统问题：

```
$ openshift-install create manifests --dir $HOME/clusterconfig
? SSH Public Key ...
$ ls $HOME/clusterconfig/openshift/
99_kubeadmin-password-secret.yaml
99_openshift-cluster-api_master-machines-0.yaml
99_openshift-cluster-api_master-machines-1.yaml
99_openshift-cluster-api_master-machines-2.yaml
...
```

3. 创建用于配置额外分区的 Butane 配置。例如，将文件命名为 $HOME/clusterconfig/98-var-partition.bu，将磁盘设备名称改为 worker 系统上存储设备的名称，并根据情况设置存储大小。这个示例将 /var 目录放在一个单独的分区中：

```
variant: openshift
version: 4.16.0
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
storage:
  disks:
  - device: /dev/disk/by-id/<device_name> 1
    partitions:
    - label: var
      start_mib: <partition_start_offset> 2
      size_mib: <partition_size> 3
      number: 5
  filesystems:
    - device: /dev/disk/by-partlabel/var
      path: /var
      format: xfs
      mount_options: [defaults, prjquota] 4
      with_mount_unit: true
```

1. 要分区的磁盘的存储设备名称。

2.

3

以兆字节为单位的数据分区大小。

4

对于用于容器存储的文件系统，必须启用 prjquota 挂载选项。

注意

当创建单独的 /var 分区时，如果不同的实例类型没有相同的设备名称，则无法为 worker 节点使用不同的实例类型。

4. 从 Butane 配置创建一个清单，并将它保存到 clusterconfig/openshift 目录中。例如，运行以下命令：

```
$ butane $HOME/clusterconfig/98-var-partition.bu -o
$HOME/clusterconfig/openshift/98-var-partition.yaml
```

5. 再次运行 openshift-install，从 manifest 和 openshift 子目录中的一组文件创建 Ignition 配置：

```
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
auth  bootstrap.ign  master.ign  metadata.json  worker.ign
```

现在，您可以使用 Ignition 配置文件作为 vSphere 安装程序的输入来安装 Red Hat Enterprise Linux CoreOS(RHCOS)系统。

23.3.4.13. 等待 bootstrap 过程完成

OpenShift Container Platform bootstrap 过程在集群节点首次引导到安装到磁盘的持久 RHCOS 环境后开始。通过 Ignition 配置文件提供的配置信息用于初始化 bootstrap 过程并在机器上安装 OpenShift Container Platform。您必须等待 bootstrap 过程完成。

先决条件

- 已为集群创建 Ignition 配置文件。
- 您已配置了适当的网络、DNS 和负载平衡基础架构。
- 已获得安装程序，并为集群生成 Ignition 配置文件。
- 已在集群机器上安装 RHCOS，并提供 OpenShift Container Platform 安装程序生成的 Ignition 配置文件。
- 您的机器可以直接访问互联网，或者有 HTTP 或 HTTPS 代理可用。

流程

1. 监控 bootstrap 过程：

```
$ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \ 1
    --log-level=info 2
```

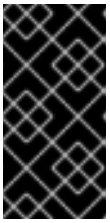
1. 对于 <installation_directory>，请指定安装文件保存到的目录的路径。
2. 要查看不同的安装详情，请指定 warn、debug 或 error，而不是 info。

输出示例

```
INFO Waiting up to 30m0s for the Kubernetes API at
https://api.test.example.com:6443...
INFO API v1.29.4 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
```

当 Kubernetes API 服务器提示已在 control plane 机器上引导它时，该命令会成功。

2. bootstrap 过程完成后，从负载均衡器中删除 bootstrap 机器。



重要

此时您必须从负载均衡器中删除 bootstrap 机器。您还可以删除或重新格式化 bootstrap 机器本身。

23.3.4.14. 使用 CLI 登录集群

您可以通过导出集群 kubeconfig 文件，以默认系统用户身份登录集群。kubeconfig 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。
- 已安装 oc CLI。

流程

1. 导出 kubeadmin 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 <installation_directory>，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 oc 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

23.3.4.15. 批准机器的证书签名请求

当您将机器添加到集群时，会为您添加的每台机器生成两个待处理证书签名请求(CSR)。您必须确认这些 CSR 已获得批准，或根据需要自行批准。必须首先批准客户端请求，然后批准服务器请求。

先决条件

- 您已将机器添加到集群中。

流程

1. 确认集群可以识别这些机器：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0  Ready    master  63m  v1.29.4
master-1  Ready    master  63m  v1.29.4
master-2  Ready    master  64m  v1.29.4
```

输出中列出了您创建的所有机器。

注意

在有些 CSR 被批准前，前面的输出可能不包括计算节点（也称为 worker 节点）。

2. 检查待处理的 CSR，并确保添加到集群中的每台机器都有 Pending 或 Approved 状态的客户端请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE    REQUESTOR                                                   CONDITION
csr-8b2br   15m    system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
csr-8vnps   15m    system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
...
```

在本例中，两台机器加入集群。您可能会在列表中看到更多已批准的 CSR。

3. 如果 CSR 没有获得批准，在您添加的机器的所有待处理 CSR 都处于 Pending 状态后，请批准集群机器的 CSR：

注意

由于 CSR 会自动轮转，因此请在将机器添加到集群后一小时内批准您的 CSR。如果没有在一小时内批准它们，证书将会轮转，每个节点会存在多个证书。您必须批准所有这些证书。批准客户端 CSR 后，Kubelet 为服务证书创建一个二级 CSR，这需要手动批准。然后，如果 Kubelet 请求具有相同参数的新证书，则后续提供证书续订请求由 machine-approver 自动批准。

注意

对于在未启用机器 API 的平台上运行的集群，如裸机和其他用户置备的基础架构，您必须实施一种方法来自动批准 kubelet 提供证书请求(CSR)。如果没有批准请求，则 oc exec、ocrsh 和 oc logs 命令将无法成功，因为 API 服务器连接到 kubelet 时需要服务证书。与 Kubelet 端点联系的任何操作都需要此证书批准。该方法必须监视新的 CSR，确认 CSR 由 system: node 或 system:admin 组中的 node-bootstrapper 服务帐户提交，并确认节点的身份。

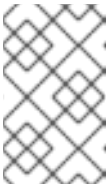
- 要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 <csr_name> 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}
{{"\n"}}{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```



注意

在有些 CSR 被批准前，一些 Operator 可能无法使用。

4. 现在，您的客户端请求已被批准，您必须查看添加到集群中的每台机器的服务器请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE     REQUESTOR                                                   CONDITION
csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal       Pending
csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal       Pending
...
```

5. 如果剩余的 CSR 没有被批准，且处于 Pending 状态，请批准集群机器的 CSR：

- 要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1

<csr_name> 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}
{{"\n"}}{{end}}{{end}}' | xargs oc adm certificate approve
```

6. 批准所有客户端和服务器 CSR 后，机器将 处于 Ready 状态。运行以下命令验证：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0  Ready    master  73m  v1.29.4
master-1  Ready    master  73m  v1.29.4
master-2  Ready    master  74m  v1.29.4
worker-0  Ready    worker  11m  v1.29.4
worker-1  Ready    worker  11m  v1.29.4
```

注意

批准服务器 CSR 后可能需要几分钟时间让机器过渡到 Ready 状态。

其他信息

- 如需有关 CSR 的更多信息，请参阅 证书签名请求。

23.3.4.15.1. 初始 Operator 配置

在 control plane 初始化后，您必须立即配置一些 Operator，以便它们都可用。

先决条件

- 您的 control plane 已初始化。

流程

1. 观察集群组件上线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION   AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.16.0    True        False         False      19m
baremetal                                  4.16.0    True        False         False      37m
cloud-credential                           4.16.0    True        False         False      40m
cluster-autoscaler                         4.16.0    True        False         False      37m
config-operator                            4.16.0    True        False         False      38m
console                                    4.16.0    True        False         False      26m
csi-snapshot-controller                    4.16.0    True        False         False      37m
dns                                        4.16.0    True        False         False      37m
etcd                                       4.16.0    True        False         False      36m
image-registry                             4.16.0    True        False         False      31m
ingress                                    4.16.0    True        False         False      30m
insights                                   4.16.0    True        False         False      31m
kube-apiserver                             4.16.0    True        False         False      26m
kube-controller-manager                    4.16.0    True        False         False      36m
kube-scheduler                             4.16.0    True        False         False      36m
kube-storage-version-migrator              4.16.0    True        False         False      37m
machine-api                                4.16.0    True        False         False      29m
machine-approver                           4.16.0    True        False         False      37m
machine-config                             4.16.0    True        False         False      36m
marketplace                                4.16.0    True        False         False      37m
monitoring                                 4.16.0    True        False         False      29m
network                                    4.16.0    True        False         False      38m
node-tuning                                4.16.0    True        False         False      37m
openshift-apiserver                        4.16.0    True        False         False      32m
openshift-controller-manager               4.16.0    True        False         False      30m
openshift-samples                          4.16.0    True        False         False      32m
operator-lifecycle-manager                 4.16.0    True        False         False      37m
operator-lifecycle-manager-catalog         4.16.0    True        False         False      37m
operator-lifecycle-manager-packageserver   4.16.0    True        False         False      32m
service-ca                                 4.16.0    True        False         False      38m
storage                                    4.16.0    True        False         False      37m
```

2. 配置不可用的 Operator。

23.3.4.15.2. 安装过程中删除的镜像 registry

在不提供可共享对象存储的平台上，OpenShift Image Registry Operator bootstraps 本身为 Removed。这允许 openshift-installer 在这些平台类型上完成安装。

安装后，您必须编辑 Image Registry Operator 配置，将 managementState 从 Removed 切换到 Managed。完成此操作后，您必须配置存储。

23.3.4.15.3. 镜像 registry 存储配置

对于不提供默认存储的平台，Image Registry Operator 最初不可用。安装后，您必须将 registry 配置为使用存储，以便 Registry Operator 可用。

显示配置生产集群所需的持久性卷的说明。如果适用，显示有关将空目录配置为存储位置的说明，这仅适用于非生产集群。

提供了在升级过程中使用 Recreate rollout 策略来允许镜像 registry 使用块存储类型的说明。

23.3.4.15.3.1. 为 VMware vSphere 配置块 registry 存储

要允许镜像 registry 在作为集群管理员升级过程中使用块存储类型，如 vSphere Virtual Machine Disk(VMDK)，您可以使用 Recreate rollout 策略。

重要

支持块存储卷，但不建议在生产环境中用于镜像 registry。在块存储上配置 registry 的安装不具有高可用性，因为 registry 无法具有多个副本。

流程

1. 输入以下命令将镜像 registry 存储设置为块存储类型，对 registry 进行补丁，使其使用 Recreate rollout 策略，并只使用一个副本运行：

```
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec":
{"rolloutStrategy":"Recreate","replicas":1}}'
```

2. 为块存储设备置备 PV，并为该卷创建 PVC。请求的块卷使用 ReadWriteOnce(RWO)访问模式。

   a. 创建包含以下内容的 pvc.yaml 文件以定义 VMware vSphere PersistentVolumeClaim 对象：

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: image-registry-storage 1
  namespace: openshift-image-registry 2
spec:
  accessModes:
  - ReadWriteOnce 3
  resources:
    requests:
      storage: 100Gi 4
```

   1 代表 PersistentVolumeClaim 对象的唯一名称。

   2 PersistentVolumeClaim 对象的命名空间，即 openshift-image-registry。

   3 持久性卷声明的访问模式。使用 ReadWriteOnce 时，单个节点可以通过读写权限挂载该卷。

   4 持久性卷声明的大小。

   b. 输入以下命令从文件创建 PersistentVolumeClaim 对象：

```
$ oc create -f pvc.yaml -n openshift-image-registry
```

3. 输入以下命令编辑 registry 配置，使其引用正确的 PVC：

```
$ oc edit config.imageregistry.operator.openshift.io -o yaml
```

输出示例

```
storage:
  pvc:
    claim: 1
```

1

通过创建自定义 PVC，您可以将 claim 字段留空，以便默认自动创建 image-registry-storage PVC。

有关配置 registry 存储以便引用正确的 PVC 的说明，请参阅 为 vSphere 配置 registry。

23.3.4.16. 在用户置备的基础架构上完成安装

完成 Operator 配置后，可以在您提供的基础架构上完成集群安装。

先决条件

- 您的 control plane 已初始化。
- 已完成初始 Operator 配置。

流程

1. 使用以下命令确认所有集群组件都在线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION   AVAILABLE   PROGRESSING   DEGRADED
SINCE
```

```
authentication                             4.16.0    True        False         False      19m
baremetal                                  4.16.0    True        False         False      37m
cloud-credential                           4.16.0    True        False         False      40m
cluster-autoscaler                         4.16.0    True        False         False      37m
config-operator                            4.16.0    True        False         False      38m
console                                    4.16.0    True        False         False      26m
csi-snapshot-controller                    4.16.0    True        False         False      37m
dns                                        4.16.0    True        False         False      37m
etcd                                       4.16.0    True        False         False      36m
image-registry                             4.16.0    True        False         False      31m
ingress                                    4.16.0    True        False         False      30m
insights                                   4.16.0    True        False         False      31m
kube-apiserver                             4.16.0    True        False         False      26m
kube-controller-manager                    4.16.0    True        False         False      36m
kube-scheduler                             4.16.0    True        False         False      36m
kube-storage-version-migrator              4.16.0    True        False         False      37m
machine-api                                4.16.0    True        False         False      29m
machine-approver                           4.16.0    True        False         False      37m
machine-config                             4.16.0    True        False         False      36m
marketplace                                4.16.0    True        False         False      37m
monitoring                                 4.16.0    True        False         False      29m
network                                    4.16.0    True        False         False      38m
node-tuning                                4.16.0    True        False         False      37m
openshift-apiserver                        4.16.0    True        False         False      32m
openshift-controller-manager               4.16.0    True        False         False      30m
openshift-samples                          4.16.0    True        False         False      32m
operator-lifecycle-manager                 4.16.0    True        False         False      37m
operator-lifecycle-manager-catalog         4.16.0    True        False         False      37m
operator-lifecycle-manager-packageserver   4.16.0    True        False         False      32m
service-ca                                 4.16.0    True        False         False      38m
storage                                    4.16.0    True        False         False      37m
```

另外，当所有集群都可用时，以下命令会通知您。它还检索并显示凭证：

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

1

对于 <installation_directory>，请指定安装文件保存到的目录的路径。

输出示例

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

Cluster Version Operator 完成从 Kubernetes API 服务器部署 OpenShift Container Platform 集群时，该命令会成功。

重要

- 安装程序生成的 Ignition 配置文件包含 24 小时后过期的证书，然后在该时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 node-bootstrapper 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。

- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

2. 确认 Kubernetes API 服务器正在与 pod 通信。

   a. 要查看所有 pod 的列表，请使用以下命令：

```
$ oc get pods --all-namespaces
```

输出示例

```
NAMESPACE                  NAME                                        READY   STATUS
RESTARTS   AGE
openshift-apiserver-operator    openshift-apiserver-operator-85cb746d55-zqhs8
1/1    Running    1        9m
openshift-apiserver        apiserver-67b9g                             1/1     Running    0
3m
openshift-apiserver        apiserver-ljcmx                             1/1     Running    0
1m
openshift-apiserver        apiserver-z25h4                             1/1     Running    0
2m
openshift-authentication-operator authentication-operator-69d5d8bf84-vh2n8
1/1    Running    0        5m
...
```

b. 使用以下命令，查看上一命令的输出中所列 pod 的日志：

```
$ oc logs <pod_name> -n <namespace> 1
```

1 指定 pod 名称和命名空间，如上一命令的输出中所示。

如果 pod 日志显示，Kubernetes API 服务器可以与集群机器通信。

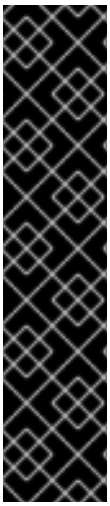
3. 对于使用光纤通道协议(FCP)的安装，还需要额外的步骤才能启用多路径。不要在安装过程中启用多路径。

如需更多信息，请参阅 *安装后机器配置任务* 文档中的"使用 RHCOS 上使用内核参数启用多路径"。

您可以按照将计算机器 添加到 vSphere 的内容在集群安装后添加额外的计算机器。

23.3.4.17. 为 control plane 节点配置 vSphere DRS 反关联性规则

可将 vSphere 分布式资源调度程序 DRS) 关联性规则配置为支持 OpenShift Container Platform Control Plane 节点的高可用性。反关联性规则确保 OpenShift Container Platform Control Plane 节点的 vSphere 虚拟机没有调度到同一 vSphere 主机。



重要

- 以下信息只适用于计算 DRS，不适用于存储 DRS。
- govc 命令是 VMware 提供的开源命令；它不是红帽提供的。红帽不支持 govc 命令。
- 有关下载和安装 govc 的说明，请参阅 VMware 文档网站。

运行以下命令来创建反关联性规则：

示例命令

```
$ govc cluster.rule.create \
  -name openshift4-control-plane-group \
  -dc MyDatacenter -cluster MyCluster \
  -enable \
  -anti-affinity master-0 master-1 master-2
```

创建规则后，您的 control plane 节点由 vSphere 自动迁移，以便它们不会在同一主机上运行。当 vSphere 协调新规则时，这可能需要一些时间。以下流程中会显示成功的命令完成。

注意

迁移会自动进行，并可能导致 OpenShift API 中断或延迟，直到迁移完成为止。

当 control plane 虚拟机名称发生变化或迁移到新的 vSphere 集群时，需要手动更新 vSphere DRS 反关联性规则。

流程

1. 运行以下命令来删除任何现有的 DRS 反关联性规则：

```
$ govc cluster.rule.remove \
  -name openshift4-control-plane-group \
  -dc MyDatacenter -cluster MyCluster
```

输出示例

```
[13-10-22 09:33:24] Reconfigure /MyDatacenter/host/MyCluster...OK
```

2. 运行以下命令，使用更新的名称再次创建规则：

```
$ govc cluster.rule.create \
  -name openshift4-control-plane-group \
  -dc MyDatacenter -cluster MyOtherCluster \
  -enable \
  -anti-affinity master-0 master-1 master-2
```

23.3.4.18. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于 远程健康监控

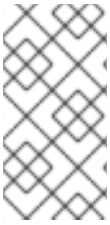
23.3.4.19. 后续步骤

- 自定义集群。
- 如果需要，您可以选择 不使用远程健康报告。
- 设置 registry 并配置 registry 存储。
- 可选：查看 vSphere 问题检测器 Operator 中的事件，以确定集群是否有权限或存储配置问题。

- 可选：如果您创建了加密的虚拟机，请创建一个加密的存储类。

23.3.5. 在使用用户置备的受限网络中的 vSphere 上安装集群

在 OpenShift Container Platform 版本 4.16 中，您可以在受限网络中置备的 VMware vSphere 基础架构上安装集群。



注意

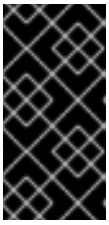
OpenShift Container Platform 支持将集群部署到单个 VMware vCenter 中。不支持在多个 vCenter 上使用机器/机器集部署集群。

重要

执行用户置备的基础架构安装的步骤仅作为示例。使用您提供的基础架构安装集群需要了解 vSphere 平台和 OpenShift Container Platform 的安装过程。使用用户置备的基础架构安装说明作为指南；您可以通过其他方法创建所需的资源。

23.3.5.1. 先决条件

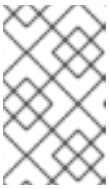
- 您已完成了 使用用户置备的基础架构准备安装集群 中的任务。
- 您检查了 VMware 平台许可证。红帽不会对 VMware 许可证产生任何限制，但有些 VMware 基础架构组件需要许可。
- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读有关 选择集群安装方法的文档，并为用户准备它。
- 您在镜像主机上创建 registry，并获取您的 OpenShift Container Platform 版本的 imageContentSources 数据。



重要

由于安装介质位于镜像主机上，因此您可以使用该计算机完成所有安装步骤。

- 已为集群置备了 持久性存储。要部署私有镜像 registry，您的存储必须提供 ReadWriteMany 访问模式。
- 完成安装要求您在 vSphere 主机上上传 Red Hat Enterprise Linux CoreOS(RHCOS)OVA。完成此过程的机器需要访问 vCenter 和 ESXi 主机上的端口 443。您确认可以访问端口 443。
- 如果您使用防火墙，您与管理员确认可以访问端口 443。control plane 节点必须能够通过端口 443 访问 vCenter 和 ESXi 主机，才能成功安装。
- 如果您使用防火墙并计划使用 Telemetry 服务，则将防火墙配置为允许集群需要访问的站点。



注意

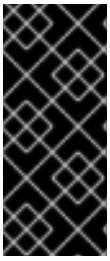
如果要配置代理，请务必查看此站点列表。

23.3.5.2. 关于在受限网络中安装

在 OpenShift Container Platform 4.16 中，可以执行不需要有效的互联网连接来获取软件组件的安装。受限网络安装可以使用安装程序置备的基础架构或用户置备的基础架构完成，具体取决于您要安装集群的云平台。

如果您选择在云平台中执行受限网络安装，您仍需要访问其云 API。有些云功能，比如 Amazon Web Service 的 Route 53 DNS 和 IAM 服务，需要访问互联网。根据您的网络，在裸机硬件、Nutanix 或 VMware vSphere 上安装可能需要较少的互联网访问。

要完成受限网络安装，您必须创建一个 registry，以镜像 OpenShift 镜像 registry 的内容并包含安装介质。您可以在镜像主机上创建此 registry，该主机可同时访问互联网和您的封闭网络，也可以使用满足您的限制条件的其他方法。



重要

由于用户置备安装配置的复杂性，在尝试使用用户置备的基础架构受限网络安装前，请考虑完成标准用户置备的基础架构安装。完成此测试安装后，您可以更轻松地隔离和排除在受限网络中安装过程中可能出现的任何问题。

23.3.5.2.1. 其他限制

受限网络中的集群有以下额外限制和限制：

- ClusterVersion 状态包含一个 Unable to retrieve available updates 错误。
- 默认情况下，您无法使用 Developer Catalog 的内容，因为您无法访问所需的镜像流标签。

23.3.5.3. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来获得用来安装集群的镜像。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。
- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

23.3.5.4. VMware vSphere 区域和区启用

您可以将 OpenShift Container Platform 集群部署到在单个 VMware vCenter 中运行的多个 vSphere 数据中心。每个数据中心都可以运行多个集群。此配置降低了导致集群失败的硬件故障或网络中断的风险。要启用区域和区域，您必须为 OpenShift Container Platform 集群定义多个故障域。

重要

VMware vSphere 区域和区启用功能需要 vSphere Container Storage Interface (CSI) 驱动程序作为集群中的默认存储驱动程序。因此，这个功能只在新安装的集群中可用。

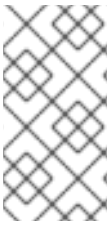
对于从上一版本升级的集群，您必须为集群启用 CSI 自动迁移。然后，您可以为升级的集群配置多个区域和区域。

默认安装配置将集群部署到单个 vSphere 数据中心。如果要将集群部署到多个 vSphere 数据中心，您必须创建一个启用地区和区功能的安装配置文件。

默认 install-config.yaml 文件包含 vcenters 和 failureDomains 字段，您可以在其中为 OpenShift Container Platform 集群指定多个 vSphere 数据中心和集群。如果要在由单个数据中心组成的 vSphere 环境中安装 OpenShift Container Platform 集群，您可以将这些字段留空。

以下列表描述了为集群定义区和区域相关的术语：

- 故障域：建立地区和区域之间的关系。您可以使用 vCenter 对象（如 datastore 对象）定义故障域。故障域定义 OpenShift Container Platform 集群节点的 vCenter 位置。
- Region ：指定 vCenter 数据中心。您可以使用 openshift-region 标签类别中的标签来定义区域。
- Zone：指定一个 vCenter 集群。您可以使用 openshift-zone 标签类别中的标签来定义区。



注意

如果您计划在 install-config.yaml 文件中指定多个故障域，则必须在创建配置文件前创建标签类别、区域标签和区域标签。

您必须为每个代表一个区域的 vCenter 数据中心创建一个 vCenter 标签。另外，您必须为比数据中心（代表一个区）中运行的每个集群创建一个 vCenter 标签。创建标签后，您必须将每个标签附加到对应的数据中心和集群。

下表概述了在单个 VMware vCenter 中运行的多个 vSphere 数据中心的区域、区域和标签之间的关系示例。

数据中心（区域）	集群（区）	Tags
us-east	us-east-1	us-east-1a
		us-east-1b
	us-east-2	us-east-2a
		us-east-2b
us-west	us-west-1	us-west-1a
		us-west-1b
	us-west-2	us-west-2a
		us-west-2b

其他资源

- 其他 VMware vSphere 配置参数
- 弃用的 VMware vSphere 配置参数
- vSphere 自动迁移
- VMware vSphere CSI Driver Operator

23.3.5.5. 手动创建安装配置文件

安装集群要求您手动创建安装配置文件。

重要

Cloud Controller Manager Operator 在提供的主机名或 IP 地址上执行连接检查。确保为可访问的 vCenter 服务器指定主机名或 IP 地址。如果您向不存在的 vCenter 服务器提供元数据，集群安装会在 bootstrap 阶段失败。

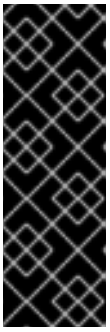
先决条件

- 您在本地机器上有一个 SSH 公钥来提供给安装程序。该密钥将用于在集群节点上进行 SSH 身份验证，以进行调试和灾难恢复。
- 已获取 OpenShift Container Platform 安装程序和集群的 pull secret。
- 获取命令输出中的 imageContentSources 部分来 镜像存储库。
- 获取您的镜像 registry 的证书内容。

流程

1. 创建一个安装目录来存储所需的安装资产：

```
$ mkdir <installation_directory>
```



重要

您必须创建一个目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

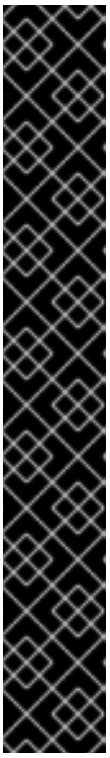
2. 自定义提供的 install-config.yaml 文件模板示例，并将其保存在 <installation_directory> 中。

注意

您必须将此配置文件命名为 **install-config.yaml**。

- 除非使用 RHCOS 默认信任的 registry，如 docker.io，否则必须在 **additionalTrustBundle** 部分中提供镜像存储库的证书内容。在大多数情况下，您必须为您的镜像提供证书。

- 您必须包含命令输出 中的 **imageContentSources** 部分才能 镜像存储库。



重要

- 在镜像过程完成后，**ImageContentSourcePolicy** 文件作为 **oc mirror** 的输出生成。

- **oc mirror** 命令会生成 **ImageContentSourcePolicy** 文件，其中包含定义 **ImageContentSourcePolicy** 所需的 **YAML**。复制此文件中的文本并将其粘贴到 **install-config.yaml** 文件中。

- 您必须运行 'oc mirror' 命令两次。第一次运行 **oc mirror** 命令时，您将获得完整的 **ImageContentSourcePolicy** 文件。第二次运行 **oc mirror** 命令时，您只获得第一次运行和第二次运行之间的差别。由于此行为，您必须始终保留这些文件的备份，以防需要将这些文件合并到一个完整的 **ImageContentSourcePolicy** 文件中。备份这两个输出文件可确保您有完整的 **ImageContentSourcePolicy** 文件。

3. 备份 **install-config.yaml** 文件，以便您可以使用它安装多个集群。

重要

install-config.yaml 文件会在安装过程的下一步中使用。现在必须备份它。

其他资源

- 安装配置参数

23.3.5.5.1. VMware vSphere 的 install-config.yaml 文件示例

您可以自定义 install-config.yaml 文件，以指定有关 OpenShift Container Platform 集群平台的更多详情，或修改所需参数的值。

```
additionalTrustBundlePolicy: Proxyonly
apiVersion: v1
baseDomain: example.com (1)
compute: (2)
- architecture: amd64
  name: <worker_node>
  platform: {}
  replicas: 0 (3)
controlPlane: (4)
  architecture: amd64
  name: <parent_node>
  platform: {}
  replicas: 3 (5)
metadata:
  creationTimestamp: null
  name: test (6)
networking:
---
platform:
  vsphere:
    failureDomains: (7)
    - name: <failure_domain_name>
      region: <default_region_name>
      server: <fully_qualified_domain_name>
      topology:
        computeCluster: "/<datacenter>/host/<cluster>"
        datacenter: <datacenter> (8)
        datastore: "/<datacenter>/datastore/<datastore>" (9)
        networks:
        - <VM_Network_name>
        resourcePool: "/<datacenter>/host/<cluster>/Resources/<resourcePool>" (10)
        folder: "/<datacenter_name>/vm/<folder_name>/<subfolder_name>" (11)
      zone: <default_zone_name>
    vcenters:
    - datacenters:
      - <datacenter>
      password: <password> (12)
      port: 443
      server: <fully_qualified_domain_name> (13)
      user: administrator@vsphere.local
    diskType: thin (14)
fips: false (15)
pullSecret: '{"auths":{"<local_registry>": {"auth": "<credentials>","email":
"you@example.com"}}}' (16)
sshKey: 'ssh-ed25519 AAAA...' (17)
additionalTrustBundle: | (18)
```

```
  -----BEGIN CERTIFICATE-----
  ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
  -----END CERTIFICATE-----
imageContentSources: 19
- mirrors:
  - <mirror_host_name>:<mirror_port>/<repo_name>/release
  source: <source_image_1>
- mirrors:
  - <mirror_host_name>:<mirror_port>/<repo_name>/release-images
  source: <source_image_2>
```

1

集群的基域。所有 DNS 记录都必须是这个基域的子域，并包含集群名称。

2 4

controlPlane 部分是一个单个映射，但 compute 部分是一系列映射。为满足不同数据结构的要求，compute 部分的第一行必须以连字符 - 开头，controlPlane 部分 的第一行则不以连字符开头。两个部分都定义单个机器池，因此只使用一个 control plane。OpenShift Container Platform 不支持定义多个计算池。

3

replicas 参数的值必须设置为 0。此参数控制集群为您创建和管理的 worker 数量，在使用用户置备的基础架构时集群不会执行这些功能。在完成 OpenShift Container Platform 安装前，您必须手动为集群部署 worker 机器。

5

您添加到集群的 control plane 机器数量。由于集群使用此值作为集群中的 etcd 端点数量，所以该值必须与您部署的 control plane 机器数量匹配。

6

您在 DNS 记录中指定的集群名称。

7

建立地区和区域之间的关系。您可以使用 vCenter 对象（如 datastore 对象）定义故障域。故障域定义 OpenShift Container Platform 集群节点的 vCenter 位置。

8

vSphere 数据中心.

9

保存虚拟机文件、模板和 ISO 镜像的 vSphere 数据存储路径。

重要

您可以指定数据存储集群中存在的任何数据存储路径。默认情况下，Storage vMotion 会自动为数据存储集群启用。红帽不支持 Storage vMotion，因此您必须禁用 Storage vMotion 以避免 OpenShift Container Platform 集群的数据丢失问题。

如果需要在多个数据存储间指定虚拟机，请使用 数据存储 对象在集群 install-config.yaml 配置文件中指定故障域。如需更多信息，请参阅"VMware vSphere 区域和区启用"。

10

可选：对于安装程序置备的基础架构，安装程序创建虚拟机的现有资源池的绝对路径，例如 /<datacenter_name>/host/<cluster_name>/Resources/<resource_pool_name>/<optional_nested_resource_pool_name>。如果没有指定值，则会在集群 /example_datacenter/host/example_cluster/Resources 根中安装资源。

11

可选：对于安装程序置备的基础架构，安装程序创建虚拟机的现有文件夹的绝对路径，如 /<datacenter_name>/vm/<folder_name>/<subfolder_name>。如果没有提供这个值，安装程序会在数据中心虚拟机文件夹中创建一个顶层文件夹，其名称为基础架构 ID。如果您为集群提供基础架构，且您不想使用默认的 StorageClass 对象（名为 thin），您可以从 install-config.yaml 文件中省略 folder 参数。

12

与 vSphere 用户关联的密码。

13

vCenter 服务器的完全限定主机名或 IP 地址。

重要

Cloud Controller Manager Operator 在提供的主机名或 IP 地址上执行连接检查。确保为可访问的 vCenter 服务器指定主机名或 IP 地址。如果您向不存在的 vCenter 服务器提供元数据，集群安装会在 bootstrap 阶段失败。

14

vSphere 磁盘置备方法。

15

是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 Red Hat Enterprise Linux (RHEL) 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅在 FIPS 模式中安装该系统。

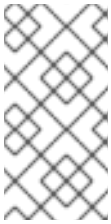
当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS)时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。

16

对于 <local_registry>，请指定 registry 域名，以及您的镜像 registry 用来提供内容的可选端口。例如 registry.example.com 或 registry.example.com:5000。对于 <credentials>，请为您的镜像 registry 指定 base64 编码的用户名和密码。

17

Red Hat Enterprise Linux CoreOS(RHCOS)中 core 用户的默认 SSH 密钥的公钥部分。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 ssh-agent 进程使用的 SSH 密钥。

18

提供用于镜像 registry 的证书文件内容。

19

提供命令输出中的 imageContentSources 部分来 镜像存储库。

23.3.5.5.2. 在安装过程中配置集群范围的代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 install-config.yaml 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 install-config.yaml 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 Proxy 对象的 spec.noProxy 字段中添加站点来绕过代理。

注意

Proxy 对象 status.noProxy 字段使用安装配置中的 networking.machineNetwork[].cidr、networking.clusterNetwork[].cidr 和 networking.serviceNetwork[] 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，Proxy 对象 status.noProxy 字段也会使用实例元数据端点填充(169.254.169.254)。

流程

1. 编辑 install-config.yaml 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 http。

2

用于创建集群外 HTTPS 连接的代理 URL。

3

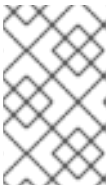
要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 . 以仅匹配子域。例如，.y.com 匹配 x.y.com，但不匹配 y.com。使用 * 绕过所有目的地的代理。您必须包含 vCenter 的 IP 地址以及用于其机器的 IP 范围。

4

如果提供，安装程序会在 openshift-config 命名空间中生成名为 user-ca-bundle 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 trusted-ca-bundle 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， Proxy 对象的 trustedCA 字段中也会引用此配置映射。additionalTrustBundle 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

5

可选：决定 Proxy 对象的配置以引用 trustedCA 字段中 user-ca-bundle 配置映射的策略。允许的值是 Proxyonly 和 Always。仅在配置了 http/https 代理时，使用 Proxyonly 引用 user-ca-bundle 配置映射。使用 Always 始终引用 user-ca-bundle 配置映射。默认值为 Proxyonly。



注意

安装程序不支持代理的 readinessEndpoints 字段。

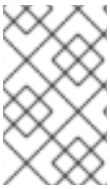
注意

如果安装程序超时，重启并使用安装程序的 wait-for 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. 保存该文件并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 cluster 的集群范围代理，该代理 使用 提供的 install-config.yaml 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 cluster Proxy 对象，但它会有一个空 spec。



注意

只支持名为 **cluster** 的 **Proxy** 对象，且无法创建额外的代理。

23.3.5.5.3. 为 VMware vCenter 配置区域和区域

您可以修改默认安装配置文件，以便您可以将 OpenShift Container Platform 集群部署到在单个 VMware vCenter 中运行的多个 vSphere 数据中心。

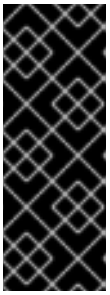
之前版本的 OpenShift Container Platform 的默认 install-config.yaml 文件配置已弃用。您可以继续使用已弃用的默认配置，但 openshift-installer 会提示您显示在配置文件中已弃用字段的警告信息。

重要

这个示例使用 govc 命令。govc 命令是 VMware 提供的开源命令；它不是红帽提供的。红帽支持团队不维护 govc 命令。有关下载和安装 govc 的说明，请参阅 VMware 文档网站

先决条件

- 您有一个现有的 install-config.yaml 安装配置文件。



重要

您必须为 OpenShift Container Platform 集群指定一个故障域，以便您可以为 VMware vCenter 服务器置备数据中心对象。如果您需要在不同的数据中心、集群、数据存储和其他组件中置备虚拟机节点，请考虑指定多个故障域。要启用区域和区域，您必须为 OpenShift Container Platform 集群定义多个故障域。

流程

1. 输入以下 govc 命令行工具命令，以创建 openshift-region 和 openshift-zone vCenter 标签类别：

重要

如果为 openshift-region 和 openshift-zone vCenter 标签类别指定不同的名称，OpenShift Container Platform 集群的安装会失败。

```
$ govc tags.category.create -d "OpenShift region" openshift-region
```

```
$ govc tags.category.create -d "OpenShift zone" openshift-zone
```

2. 要为您要部署集群的每个区域 vSphere 数据中心创建一个 region 标签，请在终端中输入以下命令：

```
$ govc tags.create -c <region_tag_category> <region_tag>
```

3. 要为您要部署集群的每个 vSphere 集群创建一个区标签，请输入以下命令：

```
$ govc tags.create -c <zone_tag_category> <zone_tag>
```

4. 输入以下命令将区域标签附加到每个 vCenter 数据中心对象：

```
$ govc tags.attach -c <region_tag_category> <region_tag_1> /<datacenter_1>
```

5. 输入以下命令将区标签附加到每个 vCenter 数据中心对象：

```
$ govc tags.attach -c <zone_tag_category> <zone_tag_1> /<datacenter_1>/host/vcs-mdcnc-workload-1
```

6. 进入包含安装程序的目录，并根据您选择的安装要求初始化集群部署。

在 vSphere 数据中心中定义的多个数据中心的 install-config.yaml 文件示例

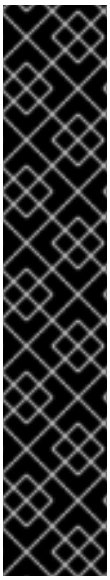
```
---
compute:
---
  vsphere:
    zones:
      - "<machine_pool_zone_1>"
      - "<machine_pool_zone_2>"
---
controlPlane:
---
vsphere:
    zones:
      - "<machine_pool_zone_1>"
      - "<machine_pool_zone_2>"
```

```
---
platform:
  vsphere:
    vcenters:
---
    datacenters:
      - <datacenter1_name>
      - <datacenter2_name>
    failureDomains:
    - name: <machine_pool_zone_1>
      region: <region_tag_1>
      zone: <zone_tag_1>
      server: <fully_qualified_domain_name>
      topology:
        datacenter: <datacenter1>
        computeCluster: "/<datacenter1>/host/<cluster1>"
        networks:
        - <VM_Network1_name>
        datastore: "/<datacenter1>/datastore/<datastore1>"
        resourcePool: "/<datacenter1>/host/<cluster1>/Resources/<resourcePool1>"
        folder: "/<datacenter1>/vm/<folder1>"
    - name: <machine_pool_zone_2>
      region: <region_tag_2>
      zone: <zone_tag_2>
      server: <fully_qualified_domain_name>
      topology:
        datacenter: <datacenter2>
        computeCluster: "/<datacenter2>/host/<cluster2>"
        networks:
        - <VM_Network2_name>
        datastore: "/<datacenter2>/datastore/<datastore2>"
        resourcePool: "/<datacenter2>/host/<cluster2>/Resources/<resourcePool2>"
        folder: "/<datacenter2>/vm/<folder2>"
---
```

23.3.5.6. 创建 Kubernetes 清单和 Ignition 配置文件

由于您必须修改一些集群定义文件并手动启动集群机器，因此您必须生成 Kubernetes 清单和 Ignition 配置文件来配置机器。

安装配置文件转换为 Kubernetes 清单。清单嵌套到 Ignition 配置文件中，稍后用于配置集群机器。



重要

- OpenShift Container Platform 安装程序生成的 Ignition 配置文件包含 24 小时后过期的证书，然后在该时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 node-bootstrapper 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。

- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

先决条件

- 已获得 OpenShift Container Platform 安装程序。对于受限网络安装，这些文件位于您的镜像主机上。

- 已创建 install-config.yaml 安装配置文件。

流程

1. 进入包含 OpenShift Container Platform 安装程序的目录，并为集群生成 Kubernetes 清单：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 对于 <installation_directory>，请指定包含您创建的 install-config.yaml 文件的安装目录。

2. 删除定义 control plane 机器的 Kubernetes 清单文件、计算机器集和 control plane 机器集：

```
$ rm -f openshift/99_openshift-cluster-api_master-machines-*.yaml
openshift/99_openshift-cluster-api_worker-machineset-*.yaml openshift/99_openshift-
machine-api_master-control-plane-machine-set.yaml
```

由于您要自行创建和管理这些资源，因此不必初始化这些资源。

- 您可以使用机器 API 来保留计算机器集文件来创建计算机器，但您必须更新对它们的引用以匹配您的环境。

3. 检查 <installation_directory>/manifests/cluster-scheduler-02-config.yml Kubernetes 清单文件中的 mastersSchedulable 参数是否已设置为 false。此设置可防止在 control plane 机器上调度 pod：

   a. 打开 <installation_directory>/manifests/cluster-scheduler-02-config.yml 文件。

   b. 找到 mastersSchedulable 参数，并确保它被设置为 false。

   c. 保存并退出 文件。

4. 要创建 Ignition 配置文件，请从包含安装程序的目录运行以下命令：

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

1 对于 <installation_directory>，请指定相同的安装目录。

为安装目录中的 bootstrap、control plane 和计算节点创建 Ignition 配置文件。kubeadmin-password 和 kubeconfig 文件在 ./<installation_directory>/auth 目录中创建：

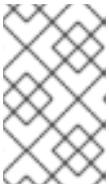
```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

23.3.5.7. 配置 chrony 时间服务

您必须通过修改 chrony .conf 文件的内容来设置 chrony 时间服务(chronyd)使用的时间服务器和相关设置，并将这些内容作为机器配置传递给节点。

流程

1. 创建一个 Butane 配置，包括 chrony.conf 文件的内容。例如，要在 worker 节点上配置 chrony，请创建一个 99-worker-chrony.bu 文件。



注意

如需有关 Butane 的信息，请参阅"使用 Butane 创建机器配置"。

```
variant: openshift
version: 4.16.0
metadata:
  name: 99-worker-chrony (1)
  labels:
    machineconfiguration.openshift.io/role: worker (2)
storage:
  files:
  - path: /etc/chrony.conf
    mode: 0644 (3)
    overwrite: true
    contents:
      inline: |
        pool 0.rhel.pool.ntp.org iburst (4)
        driftfile /var/lib/chrony/drift
        makestep 1.0 3
        rtcsync
        logdir /var/log/chrony
```

(1) (2) 在 control plane 节点上，在这两个位置中将 master 替换为 worker。

(3) 为机器配置文件的 mode 字段指定数值模式。在创建文件并应用更改后，mode 将转换为十进制值。您可以使用 oc get mc <mc-name> -o yaml 命令来检查 YAML 文件。

(4) 指定任何有效的、可访问的时间源，如 DHCP 服务器提供的源。

2. 使用 Butane 生成 MachineConfig 对象文件 99-worker-chrony.yaml，其中包含要交付至节点的配置：

```
$ butane 99-worker-chrony.bu -o 99-worker-chrony.yaml
```

3. 使用以下两种方式之一应用配置：

- 如果集群还没有运行，在生成清单文件后，将 MachineConfig 对象文件添加到 <installation_directory>/openshift 目录中，然后继续创建集群。
- 如果集群已在运行，请应用该文件：

```
$ oc apply -f ./99-worker-chrony.yaml
```

23.3.5.8. 提取基础架构名称

Ignition 配置文件包含一个唯一集群标识符，您可以使用它在 VMware vSphere 中唯一地标识您的集群。如果计划使用集群标识符作为虚拟机文件夹的名称，则必须提取它。

先决条件

- 已获取 OpenShift Container Platform 安装程序和集群的 pull secret。
- 已为集群生成 Ignition 配置文件。
- 已安装 jq 软件包。

流程

- 要从 Ignition 配置文件元数据中提取和查看基础架构名称，请运行以下命令：

```
$ jq -r .infraID <installation_directory>/metadata.json 1
```

1

对于 <installation_directory>，请指定安装文件保存到的目录的路径。

输出示例

```
openshift-vw9j6 1
```

1

此命令的输出是您的集群名称和随机字符串。

23.3.5.9. 安装 RHCOS 并启动 OpenShift Container Platform bootstrap 过程

要在 VMware vSphere 上的用户置备的基础架构上安装 OpenShift Container Platform，您必须在 vSphere 主机上安装 Red Hat Enterprise Linux CoreOS(RHCOS)。安装 RHCOS 时，您必须为您要安装的机器类型提供 OpenShift Container Platform 安装程序生成的 Ignition 配置文件。如果您配置了适当的网络、DNS 和负载均衡基础架构，OpenShift Container Platform bootstrap 过程会在 RHCOS 机器重启后自动启动。

先决条件

- 已获取集群的 Ignition 配置文件。
- 具有 HTTP 服务器的访问权限，以便您可从计算机进行访问，并且您创建的机器也可访问此服务器。
- 您已创建了 vSphere 集群。

流程

1. 将名为 <installation_directory>/bootstrap.ign 的 bootstrap Ignition 配置文件上传到 HTTP 服务器。注意此文件的 URL。
2. 将 bootstrap 节点的以下辅助 Ignition 配置文件保存到计算机中，存为

<installation_directory>/merge-bootstrap.ign ：

```
{
  "ignition": {
    "config": {
      "merge": [
        {
          "source": "<bootstrap_ignition_config_url>", 1
          "verification": {}
        }
      ]
    },
    "timeouts": {},
    "version": "3.2.0"
  },
  "networkd": {},
  "passwd": {},
  "storage": {},
  "systemd": {}
}
```

1

指定您托管的 bootstrap Ignition 配置文件的 URL。

为 bootstrap 机器创建虚拟机(VM)时，您要使用此 Ignition 配置文件。

3. 找到安装程序创建的以下 Ignition 配置文件：

- <installation_directory>/master.ign
- <installation_directory>/worker.ign
- <installation_directory>/merge-bootstrap.ign

4. 将 Ignition 配置文件转换为 Base64 编码。在此流程中，您必须将这些文件添加到虚拟机中的额外配置参数 guestinfo.ignition.config.data 中。

例如，如果使用 Linux 操作系统，您可以使用 base64 命令对文件进行编码。

```
$ base64 -w0 <installation_directory>/master.ign > <installation_directory>/master.64
```

```
$ base64 -w0 <installation_directory>/worker.ign > <installation_directory>/worker.64
```

```
$ base64 -w0 <installation_directory>/merge-bootstrap.ign >
<installation_directory>/merge-bootstrap.64
```

重要

如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

5. 获取 RHCOS OVA 镜像。镜像位于 RHCOS 镜像镜像页面。

重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每个发行版本而改变。您必须下载最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。如果可用，请使用与 OpenShift Container Platform 版本匹配的镜像版本。

文件名包含 OpenShift Container Platform 版本号，格式为 rhcos-vmware.<architecture>.ova。

6. 在 vSphere 客户端中，在数据中心中创建一个文件夹来存储虚拟机。

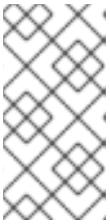
   a. 单击 VMs and Templates 视图。

   b. 右键单击您的数据中心的名称。

   c. 点击 New Folder → New VM and Template Folder。

   d. 在显示的窗口中，输入文件夹名称。如果您没有在 install-config.yaml 文件中指定现有文件夹，请创建一个名称与基础架构 ID 相同的文件夹。您可以使用这个文件夹名称，因此 vCenter 会在适当的位置为 Workspace 配置动态置备存储。

7. 在 vSphere 客户端中，为 OVA 镜像创建一个模板，然后根据需要克隆模板。



注意

在以下步骤中，您将创建模板，然后克隆所有集群机器的模板。然后，您在置备虚拟机时为该克隆的机器类型提供 Ignition 配置文件的位置。

a. 在 Hosts and Clusters 选项卡中，右键点击您的集群名称并选择 Deploy OVF Template。

b. 在 Select an OVF 选项卡中，指定您下载的 RHCOS OVA 文件的名称。

c. 在 Select a name and folder 选项卡中，为您的模板设置 虚拟机名称，如 Template-RHCOS。点击 vSphere 集群的名称并选择您在上一步中创建的文件夹。

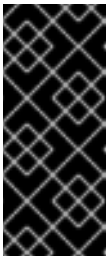
d. 在 Select a compute resource 选项卡中，点击 vSphere 集群的名称。

e. 在 Select storage 选项卡中，配置虚拟机的存储选项。

- 根据您的存储首选项，选择 Thin Provision 或 Thick Provision。
- 选择您在 install-config.yaml 文件中指定的数据存储。
- 如果要加密虚拟机，请选择 Encrypt this virtual machine。如需更多信息，请参阅标题为"加密虚拟机的要求"的部分。

f. 在 Select network 选项卡中，指定您为集群配置的网络（如果可用）。

g. 在创建 OVF 模板时，不要在 Customize template 选项卡上指定值，也不会进一步配置模板。



重要

不要启动原始虚拟机模板。VM 模板必须保持关闭，必须为新的 RHCOS 机器克隆。启动虚拟机模板会将虚拟机模板配置为平台上的虚拟机，这样可防止它被用作计算机器集可应用配置的模板。

8. 可选：如果需要，更新 VM 模板中配置的虚拟硬件版本。如需更多信息，请参阅 VMware 文档中的将虚拟机升级到最新硬件版本。

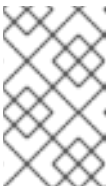
重要

如有必要，建议您在从虚拟机创建虚拟机前将虚拟机模板的硬件版本更新为版本 15。在 vSphere 上运行的集群节点使用硬件版本 13 现已弃用。如果您导入的模板默认为硬件版本 13，您必须在将 VM 模板升级到硬件版本 15 前确保 ESXi 主机为 6.7U3 或更高版本。如果您的 vSphere 版本小于 6.7U3，您可以跳过此升级步骤；但是，计划将来的 OpenShift Container Platform 版本删除对小于 6.7U3 的硬件版本 13 和 vSphere 版本的支持。

9. 部署模板后，为集群中的机器部署虚拟机。

a. 右键点击模板名称，再点击 Clone → Clone to Virtual Machine。

b. 在 Select a name and folder 选项卡中，指定虚拟机的名称。您可以在名称中包含机器类型，如 control-plane-0 或 compute-1。



注意

确保 vSphere 安装中的所有虚拟机名称都是唯一的。

c. 在 Select a name and folder 选项卡中，选择您为集群创建的文件夹名称。

d. 在 Select a compute resource 选项卡中，选择数据中心中的主机名称。

e. 在 Select clone options 选项卡中，选择 Customize this virtual machine's hardware。

f.

在 Customize hardware 选项卡上，点 Advanced Parameters。

重要

以下配置建议仅用于演示目的。作为集群管理员，您必须根据集群上的资源需求来配置资源。为了更好地管理集群资源，请考虑从集群的 root 资源池创建资源池。

- 可选：覆盖 vSphere 中的默认 DHCP 网络。启用静态 IP 网络：
  - 设置静态 IP 配置：

示例命令

```
$ export IPCFG="ip=<ip>::<gateway>:<netmask>:<hostname>:
<iface>:none nameserver=srv1 [nameserver=srv2 [nameserver=srv3 [...]]]"
```

示例命令

```
$ export IPCFG="ip=192.168.100.101::192.168.100.254:255.255.255.0:::none
nameserver=8.8.8.8"
```

  - 在从 vSphere 中的 OVA 引导虚拟机前，设置 guestinfo.afterburn.initrd.network-kargs 属性：

示例命令

```
$ govc vm.change -vm "<vm_name>" -e
"guestinfo.afterburn.initrd.network-kargs=${IPCFG}"
```

- 通过在 Attribute 和 Values 字段中指定数据来添加以下配置参数名称和值。确保为您创建的每个参数选择 Add 按钮。
  - guestinfo.ignition.config.data ：找到您在此流程中创建的 base-64 编码文件，并粘贴此机器类型的 base64 编码 Ignition 配置文件的内容。
  - guestinfo.ignition.config.data.encoding ：指定 base64。
  - disk.EnableUUID ：指定 TRUE。
  - stealclock.enable ：如果没有定义此参数，请添加它并指定 TRUE。
  - 从集群的 root 资源池创建子资源池。执行此子资源池中的资源分配。

g. 在 Customize hardware 选项卡的 Virtual Hardware 面板中，根据需要修改指定的值。确保 RAM、CPU 和磁盘存储的数量满足机器类型的最低要求。

h. 完成剩余的配置步骤。点 Finish 按钮，您已完成克隆操作。

i. 在 Virtual Machines 选项卡中，右键点您的虚拟机，然后选择 Power → Power On。

j. 检查控制台输出，以验证 Ignition 是否运行。

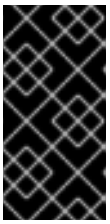
示例命令

```
Ignition: ran on 2022/03/14 14:48:33 UTC (this boot)
Ignition: user-provided config was applied
```

-

后续步骤

- 对每台机器执行前面的步骤，为集群创建其余机器。



重要

此时您必须创建 bootstrap 和 control plane 机器。由于计算机器上已默认部署了一些 Pod，因此还要在安装集群前至少创建两台计算机器。

23.3.5.10. 将更多计算机器添加到 vSphere 中的集群

您可以将更多计算机器添加到 VMware vSphere 上的用户置备的 OpenShift Container Platform 集群中。

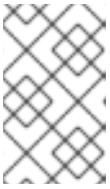
在 OpenShift Container Platform 集群中部署 vSphere 模板后，您可以为该集群中的机器部署虚拟机(VM)。

先决条件

- 获取计算机器的 base64 编码 Ignition 文件。
- 您可以访问您为集群创建的 vSphere 模板。

流程

1. 右键点击模板的名称，再点击 Clone → Clone to Virtual Machine。
2. 在 Select a name and folder 选项卡中，指定虚拟机的名称。您可以在名称中包含机器类型，如 compute-1。



注意

确保 vSphere 安装中的所有虚拟机名称都是唯一的。

3. 在 Select a name and folder 选项卡中，选择您为集群创建的文件夹名称。

4. 在 Select a compute resource 选项卡中，选择数据中心中的主机名称。

5. 在 Select storage 选项卡中，为您的配置和磁盘文件选择存储。

6. 在 Select clone options 选项卡中，选择 Customize this virtual machine's hardware。

7. 在 Customize hardware 选项卡上，点 Advanced Parameters。

   - 通过在 Attribute 和 Values 字段中指定数据来添加以下配置参数名称和值。确保为您创建的每个参数选择 Add 按钮。

     - guestinfo.ignition.config.data ：粘贴此机器类型的 base64 编码计算 Ignition 配置文件的内容。

     - guestinfo.ignition.config.data.encoding ：指定 base64。

     - disk.EnableUUID ：指定 TRUE。

8. 在 Customize hardware 选项卡的 Virtual Hardware 面板中，根据需要修改指定的值。确保 RAM、CPU 和磁盘存储的数量满足机器类型的最低要求。如果存在多个网络，请选择 Add New Device > Network Adapter，然后在 New Network 菜单项提供的字段中输入您的网络信息。

9. 完成剩余的配置步骤。点 Finish 按钮，您已完成克隆操作。

10. 在 Virtual Machines 选项卡中，右键点您的虚拟机，然后选择 Power → Power On。

后续步骤

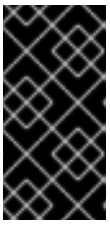
- 继续为集群创建更多计算机器。

23.3.5.11. 磁盘分区

在大多数情况下，数据分区最初是由安装 RHCOS 而不是安装另一个操作系统来创建的。在这种情况下，OpenShift Container Platform 安装程序被允许配置磁盘分区。

但是，在安装 OpenShift Container Platform 节点时，在两种情况下您可能需要覆盖默认分区：

- 创建单独的分区：要在空磁盘上进行 greenfield 安装，您可能需要在分区中添加单独的存储。这正式支持生成 /var 或 /var 的子目录，如 /var/lib/etcd（独立分区），但不支持两者。



重要

对于大于 100GB 的磁盘大小，特别是磁盘大小大于 1TB，请创建一个独立的 /var 分区。如需更多信息，请参阅"创建独立 /var 分区"和 红帽知识库文章。

重要

Kubernetes 仅支持两个文件系统分区。如果您在原始配置中添加多个分区，Kubernetes 无法监控所有这些分区。

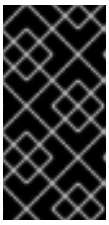
- 保留现有分区：对于 brownfield 安装，您要在现有节点上重新安装 OpenShift Container Platform，并希望保留从之前的操作系统中安装的数据分区，对于 coreos-installer 来说，引导选项和选项都允许您保留现有数据分区。

创建独立 /var 分区

通常，OpenShift Container Platform 的磁盘分区应该保留给安装程序。然而，在有些情况下您可能需要在文件系统的一部分中创建独立分区。

OpenShift Container Platform 支持添加单个分区来将存储附加到 /var 分区或 /var 的子目录中。例如：

- /var/lib/containers ：保存随着系统中添加更多镜像和容器而增长的容器相关内容。
- /var/lib/etcd ：保存您可能希望独立保留的数据，比如 etcd 存储的性能优化。
- /var ：保存您可能希望独立保留的数据，以满足审计等目的。



重要

对于大于 100GB 的磁盘大小，特别是磁盘大小大于 1TB，请创建一个独立的 /var 分区。

通过单独存储 /var 目录的内容，可以更轻松地根据需要为区域扩展存储，并在以后重新安装 OpenShift Container Platform，并保持该数据的完整性。使用这个方法，您不必再次拉取所有容器，在更新系统时也不必复制大量日志文件。

因为 /var 在进行一个全新的 Red Hat Enterprise Linux CoreOS（RHCOS）安装前必需存在，所以这个流程会在 OpenShift Container Platform 安装过程的 openshift-install 准备阶段插入一个创建的机器配置清单的机器配置来设置独立的 /var 分区。

流程

1. 创建存放 OpenShift Container Platform 安装文件的目录：

```
$ mkdir $HOME/clusterconfig
```

2. 运行 openshift-install，以在 manifest 和 openshift 子目录中创建一组文件。在系统提示时回答系统问题：

```
$ openshift-install create manifests --dir $HOME/clusterconfig
? SSH Public Key ...
$ ls $HOME/clusterconfig/openshift/
99_kubeadmin-password-secret.yaml
99_openshift-cluster-api_master-machines-0.yaml
99_openshift-cluster-api_master-machines-1.yaml
99_openshift-cluster-api_master-machines-2.yaml
...
```

3. 创建用于配置额外分区的 Butane 配置。例如，将文件命名为 $HOME/clusterconfig/98-var-partition.bu，将磁盘设备名称改为 worker 系统上存储设备的名称，并根据情况设置存储大小。这个示例将 /var 目录放在一个单独的分区中：

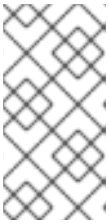
```
variant: openshift
version: 4.16.0
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
storage:
  disks:
  - device: /dev/disk/by-id/<device_name> 1
    partitions:
    - label: var
      start_mib: <partition_start_offset> 2
      size_mib: <partition_size> 3
      number: 5
  filesystems:
    - device: /dev/disk/by-partlabel/var
      path: /var
      format: xfs
      mount_options: [defaults, prjquota] 4
      with_mount_unit: true
```

1 要分区的磁盘的存储设备名称。

2 当在引导磁盘中添加数据分区时，推荐最少使用 25000 MB。root 文件系统会自动调整大小以填充所有可用空间（最多到指定的偏移值）。如果没有指定值，或者指定的值小于推荐的最小值，则生成的 root 文件系统会太小，而在以后进行的 RHCOS 重新安装可能会覆盖数据分区的开始部分。

3 以兆字节为单位的数据分区大小。

4 对于用于容器存储的文件系统，必须启用 prjquota 挂载选项。



注意

当创建单独的 /var 分区时，如果不同的实例类型没有相同的设备名称，则无法为 worker 节点使用不同的实例类型。

4. 从 Butane 配置创建一个清单，并将它保存到 clusterconfig/openshift 目录中。例如，运行以下命令：

```
$ butane $HOME/clusterconfig/98-var-partition.bu -o
$HOME/clusterconfig/openshift/98-var-partition.yaml
```

5. 再次运行 openshift-install，从 manifest 和 openshift 子目录中的一组文件创建 Ignition 配置：

```
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
auth  bootstrap.ign  master.ign  metadata.json  worker.ign
```

现在，您可以使用 Ignition 配置文件作为 vSphere 安装程序的输入来安装 Red Hat Enterprise Linux CoreOS(RHCOS)系统。

23.3.5.12. 等待 bootstrap 过程完成

OpenShift Container Platform bootstrap 过程在集群节点首次引导到安装到磁盘的持久 RHCOS 环境后开始。通过 Ignition 配置文件提供的配置信息用于初始化 bootstrap 过程并在机器上安装 OpenShift Container Platform。您必须等待 bootstrap 过程完成。

先决条件

- 已为集群创建 Ignition 配置文件。
- 您已配置了适当的网络、DNS 和负载平衡基础架构。
- 已获得安装程序，并为集群生成 Ignition 配置文件。
- 已在集群机器上安装 RHCOS，并提供 OpenShift Container Platform 安装程序生成的 Ignition 配置文件。

流程

1. 监控 bootstrap 过程：

```
$ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \ 1
    --log-level=info 2
```

1 对于 <installation_directory>，请指定安装文件保存到的目录的路径。

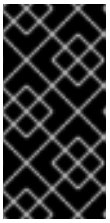
2 要查看不同的安装详情，请指定 warn、debug 或 error，而不是 info。

输出示例

```
INFO Waiting up to 30m0s for the Kubernetes API at
https://api.test.example.com:6443...
INFO API v1.29.4 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
```

当 Kubernetes API 服务器提示已在 control plane 机器上引导它时，该命令会成功。

2. bootstrap 过程完成后，从负载均衡器中删除 bootstrap 机器。



重要

此时您必须从负载均衡器中删除 bootstrap 机器。您还可以删除或重新格式化 bootstrap 机器本身。

23.3.5.13. 使用 CLI 登录集群

您可以通过导出集群 kubeconfig 文件，以默认系统用户身份登录集群。kubeconfig 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift

Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。
- 已安装 oc CLI。

流程

1. 导出 kubeadmin 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 <installation_directory>，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 oc 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

23.3.5.14. 批准机器的证书签名请求

当您将机器添加到集群时，会为您添加的每台机器生成两个待处理证书签名请求(CSR)。您必须确认这些 CSR 已获得批准，或根据需要自行批准。必须首先批准客户端请求，然后批准服务器请求。

先决条件

- 您已将机器添加到集群中。

流程

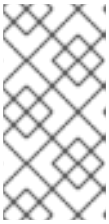
1. 确认集群可以识别这些机器：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0  Ready    master  63m  v1.29.4
master-1  Ready    master  63m  v1.29.4
master-2  Ready    master  64m  v1.29.4
```

输出中列出了您创建的所有机器。



注意

在有些 CSR 被批准前，前面的输出可能不包括计算节点（也称为 worker 节点）。

2. 检查待处理的 CSR，并确保添加到集群中的每台机器都有 Pending 或 Approved 状态的客户端请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE    REQUESTOR                                                   CONDITION
csr-8b2br   15m    system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
csr-8vnps   15m    system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
...
```

在本例中，两台机器加入集群。您可能会在列表中看到更多已批准的 CSR。

3. 如果 CSR 没有获得批准，在您添加的机器的所有待处理 CSR 都处于 Pending 状态后，请批准集群机器的 CSR：

注意

由于 CSR 会自动轮转，因此请在将机器添加到集群后一小时内批准您的 CSR。如果没有在一小时内批准它们，证书将会轮转，每个节点会存在多个证书。您必须批准所有这些证书。批准客户端 CSR 后，Kubelet 为服务证书创建一个二级 CSR，这需要手动批准。然后，如果 Kubelet 请求具有相同参数的新证书，则后续提供证书续订请求由 machine-approver 自动批准。

注意

对于在未启用机器 API 的平台上运行的集群，如裸机和其他用户置备的基础架构，您必须实施一种方法来自动批准 kubelet 提供证书请求(CSR)。如果没有批准请求，则 oc exec、ocrsh 和 oc logs 命令将无法成功，因为 API 服务器连接到 kubelet 时需要服务证书。与 Kubelet 端点联系的任何操作都需要此证书批准。该方法必须监视新的 CSR，确认 CSR 由 system: node 或 system:admin 组中的 node-bootstrapper 服务帐户提交，并确认节点的身份。

- 要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 <csr_name> 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}
{{"\n"}}{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```

注意

在有些 CSR 被批准前，一些 Operator 可能无法使用。

4. 现在，您的客户端请求已被批准，您必须查看添加到集群中的每台机器的服务器请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE     REQUESTOR                                                   CONDITION
csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal       Pending
csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal       Pending
...
```

5. 如果剩余的 CSR 没有被批准，且处于 Pending 状态，请批准集群机器的 CSR：

- 要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 <csr_name> 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}
{{"\n"}}{{end}}{{end}}' | xargs oc adm certificate approve
```

6. 批准所有客户端和服务器 CSR 后，机器将 处于 Ready 状态。运行以下命令验证：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0  Ready    master  73m  v1.29.4
master-1  Ready    master  73m  v1.29.4
master-2  Ready    master  74m  v1.29.4
worker-0  Ready    worker  11m  v1.29.4
worker-1  Ready    worker  11m  v1.29.4
```

注意

批准服务器 CSR 后可能需要几分钟时间让机器过渡到 Ready 状态。

其他信息

- 如需有关 CSR 的更多信息，请参阅 证书签名请求。

23.3.5.15. 初始 Operator 配置

在 control plane 初始化后，您必须立即配置一些 Operator，以便它们都可用。

先决条件

- 您的 control plane 已初始化。

流程

1. 观察集群组件上线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION   AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.16.0    True        False         False      19m
baremetal                                  4.16.0    True        False         False      37m
cloud-credential                           4.16.0    True        False         False      40m
cluster-autoscaler                         4.16.0    True        False         False      37m
config-operator                            4.16.0    True        False         False      38m
console                                    4.16.0    True        False         False      26m
csi-snapshot-controller                    4.16.0    True        False         False      37m
dns                                        4.16.0    True        False         False      37m
etcd                                       4.16.0    True        False         False      36m
image-registry                             4.16.0    True        False         False      31m
ingress                                    4.16.0    True        False         False      30m
insights                                   4.16.0    True        False         False      31m
kube-apiserver                             4.16.0    True        False         False      26m
kube-controller-manager                    4.16.0    True        False         False      36m
kube-scheduler                             4.16.0    True        False         False      36m
kube-storage-version-migrator              4.16.0    True        False         False      37m
machine-api                                4.16.0    True        False         False      29m
machine-approver                           4.16.0    True        False         False      37m
machine-config                             4.16.0    True        False         False      36m
marketplace                                4.16.0    True        False         False      37m
monitoring                                 4.16.0    True        False         False      29m
network                                    4.16.0    True        False         False      38m
node-tuning                                4.16.0    True        False         False      37m
openshift-apiserver                        4.16.0    True        False         False      32m
openshift-controller-manager               4.16.0    True        False         False      30m
openshift-samples                          4.16.0    True        False         False      32m
operator-lifecycle-manager                 4.16.0    True        False         False      37m
operator-lifecycle-manager-catalog         4.16.0    True        False         False      37m
operator-lifecycle-manager-packageserver   4.16.0    True        False         False      32m
service-ca                                 4.16.0    True        False         False      38m
storage                                    4.16.0    True        False         False      37m
```

2. 配置不可用的 Operator。

23.3.5.15.1. 禁用默认的 OperatorHub 目录源

在 OpenShift Container Platform 安装过程中，默认为 OperatorHub 配置由红帽和社区项目提供的源内容的 operator 目录。在受限网络环境中，必须以集群管理员身份禁用默认目录。

流程

- 通过在 OperatorHub 对象中添加 disableAllDefaultSources: true 来 禁用默认目录的源：

```
$ oc patch OperatorHub cluster --type json \
    -p '[{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]'
```

提示

或者，您可以使用 Web 控制台管理目录源。在 Administration → Cluster Settings → Configuration → OperatorHub 页面中，点 Sources 选项卡，您可以在其中创建、更新、删除、禁用和启用单独的源。

23.3.5.15.2. 镜像 registry 存储配置

对于不提供默认存储的平台，Image Registry Operator 最初不可用。安装后，您必须将 registry 配置为使用存储，以便 Registry Operator 可用。

显示配置生产集群所需的持久性卷的说明。如果适用，显示有关将空目录配置为存储位置的说明，这仅适用于非生产集群。

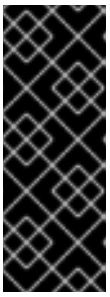
提供了在升级过程中使用 Recreate rollout 策略来允许镜像 registry 使用块存储类型的说明。

23.3.5.15.2.1. 为 VMware vSphere 配置 registry 存储

作为集群管理员，在安装后需要配置 registry 来使用存储。

先决条件

- 集群管理员权限。
- VMware vSphere 上有一个集群。
- 为集群置备的持久性存储，如 Red Hat OpenShift Data Foundation。



重要

当您只有一个副本时，OpenShift Container Platform 支持对镜像 registry 存储的 ReadWriteOnce 访问。ReadWriteOnce 访问还要求 registry 使用 Recreate rollout 策略。要部署支持高可用性的镜像 registry，需要两个或多个副本，ReadWriteMany 访问。

- 必须具有"100Gi"容量.

重要

测试显示在 RHEL 中使用 NFS 服务器作为核心服务的存储后端的问题。这包括 OpenShift Container Registry 和 Quay，Prometheus 用于监控存储，以及 Elasticsearch 用于日志存储。因此，不建议使用 RHEL NFS 作为 PV 后端用于核心服务。

市场上的其他 NFS 实现可能没有这些问题。如需了解更多与此问题相关的信息，请联络相关的 NFS 厂商。

流程

1. 要将 registry 配置为使用存储，修改 configs.imageregistry/cluster 资源中的 spec.storage.pvc。

注意

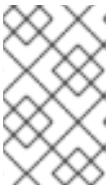
使用共享存储时，请查看您的安全设置以防止外部访问。

2. 验证您没有 registry pod:

```
$ oc get pod -n openshift-image-registry -l docker-registry=default
```

输出示例

```
No resourses found in openshift-image-registry namespace
```



注意

如果您的输出中有一个 registry pod，则不需要继续这个过程。

3. 检查 **registry** 配置：

```
$ oc edit configs.imageregistry.operator.openshift.io
```

输出示例

```
storage:
  pvc:
    claim: 1
```

1

将 claim 字段留空以允许自动创建 image-registry-storage 持久性卷声明(PVC)。PVC 基于默认存储类生成。但请注意，默认存储类可能会提供 ReadWriteOnce (RWO) 卷，如 RADOS 块设备(RBD)，这可能会在复制到多个副本时导致问题。

4. 检查 **clusteroperator** 状态：

```
$ oc get clusteroperator image-registry
```

输出示例

```
NAME             VERSION                             AVAILABLE   PROGRESSING   DEGRADED
SINCE   MESSAGE
image-registry   4.7                                 True        False         False      6h50m
```

23.3.5.15.2.2. 在非生产集群中为镜像 registry 配置存储

您必须为 Image Registry Operator 配置存储。对于非生产集群，您可以将镜像 registry 设置为空目录。如果您这样做，重启 registry 时会丢失所有镜像。

流程

- 将镜像 registry 存储设置为空目录：

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch
'{"spec":{"storage":{"emptyDir":{}}}}'
```

> **警告**
>
> 仅为非生产集群配置这个选项。

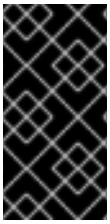
如果在 Image Registry Operator 初始化其组件前运行这个命令，oc patch 命令会失败并显示以下错误：

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster"
not found
```

等待几分钟，然后再次运行 命令。

23.3.5.15.2.3. 为 VMware vSphere 配置块 registry 存储

要允许镜像 registry 在作为集群管理员升级过程中使用块存储类型，如 vSphere Virtual Machine Disk(VMDK)，您可以使用 Recreate rollout 策略。



重要

支持块存储卷，但不建议在生产环境中用于镜像 registry。在块存储上配置 registry 的安装不具有高可用性，因为 registry 无法具有多个副本。

流程

1. 输入以下命令将镜像 registry 存储设置为块存储类型，对 registry 进行补丁，使其使用 Recreate rollout 策略，并只使用一个副本运行：

```
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec":{"rolloutStrategy":"Recreate","replicas":1}}'
```

2. 为块存储设备置备 PV，并为该卷创建 PVC。请求的块卷使用 ReadWriteOnce(RWO)访问模式。

   a. 创建包含以下内容的 pvc.yaml 文件以定义 VMware vSphere PersistentVolumeClaim 对象：

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: image-registry-storage 1
  namespace: openshift-image-registry 2
spec:
  accessModes:
  - ReadWriteOnce 3
  resources:
    requests:
      storage: 100Gi 4
```

1 代表 PersistentVolumeClaim 对象的唯一名称。

2 PersistentVolumeClaim 对象的命名空间，即 openshift-image-registry。

3 持久性卷声明的访问模式。使用 ReadWriteOnce 时，单个节点可以通过读写权限挂载该卷。

4 持久性卷声明的大小。

b. 输入以下命令从文件创建 PersistentVolumeClaim 对象：

```
$ oc create -f pvc.yaml -n openshift-image-registry
```

3. 输入以下命令编辑 registry 配置，使其引用正确的 PVC：

```
$ oc edit config.imageregistry.operator.openshift.io -o yaml
```

输出示例

```
storage:
  pvc:
    claim: 1
```

1 通过创建自定义 PVC，您可以将 claim 字段留空，以便默认自动创建 image-registry-storage PVC。

有关配置 registry 存储以便引用正确的 PVC 的说明，请参阅 为 vSphere 配置 registry。

23.3.5.16. 在用户置备的基础架构上完成安装

完成 Operator 配置后，可以在您提供的基础架构上完成集群安装。

先决条件

- 您的 control plane 已初始化。

- 已完成初始 Operator 配置。

流程

1. 使用以下命令确认所有集群组件都在线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION   AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.16.0    True        False         False      19m
baremetal                                  4.16.0    True        False         False      37m
cloud-credential                           4.16.0    True        False         False      40m
cluster-autoscaler                         4.16.0    True        False         False      37m
config-operator                            4.16.0    True        False         False      38m
console                                    4.16.0    True        False         False      26m
csi-snapshot-controller                    4.16.0    True        False         False      37m
dns                                        4.16.0    True        False         False      37m
etcd                                       4.16.0    True        False         False      36m
image-registry                             4.16.0    True        False         False      31m
ingress                                    4.16.0    True        False         False      30m
insights                                   4.16.0    True        False         False      31m
kube-apiserver                             4.16.0    True        False         False      26m
kube-controller-manager                    4.16.0    True        False         False      36m
kube-scheduler                             4.16.0    True        False         False      36m
kube-storage-version-migrator              4.16.0    True        False         False      37m
machine-api                                4.16.0    True        False         False      29m
machine-approver                           4.16.0    True        False         False      37m
machine-config                             4.16.0    True        False         False      36m
marketplace                                4.16.0    True        False         False      37m
monitoring                                 4.16.0    True        False         False      29m
network                                    4.16.0    True        False         False      38m
node-tuning                                4.16.0    True        False         False      37m
openshift-apiserver                        4.16.0    True        False         False      32m
openshift-controller-manager               4.16.0    True        False         False      30m
openshift-samples                          4.16.0    True        False         False      32m
operator-lifecycle-manager                 4.16.0    True        False         False      37m
operator-lifecycle-manager-catalog         4.16.0    True        False         False      37m
operator-lifecycle-manager-packageserver   4.16.0    True        False         False      32m
service-ca                                 4.16.0    True        False         False      38m
storage                                    4.16.0    True        False         False      37m
```

另外，当所有集群都可用时，以下命令会通知您。它还检索并显示凭证：

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

1

对于 <installation_directory>，请指定安装文件保存到的目录的路径。

输出示例

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

Cluster Version Operator 完成从 Kubernetes API 服务器部署 OpenShift Container Platform 集群时，该命令会成功。

重要

- 安装程序生成的 Ignition 配置文件包含 24 小时后过期的证书，然后在该时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 node-bootstrapper 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。

- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

2. 确认 Kubernetes API 服务器正在与 pod 通信。

   a. 要查看所有 pod 的列表，请使用以下命令：

   ```
   $ oc get pods --all-namespaces
   ```

输出示例

```
NAMESPACE                    NAME                                        READY   STATUS
RESTARTS   AGE
openshift-apiserver-operator    openshift-apiserver-operator-85cb746d55-zqhs8
1/1    Running    1        9m
openshift-apiserver          apiserver-67b9g                             1/1     Running    0
3m
openshift-apiserver          apiserver-ljcmx                             1/1     Running    0
1m
openshift-apiserver          apiserver-z25h4                             1/1     Running    0
2m
openshift-authentication-operator authentication-operator-69d5d8bf84-vh2n8
1/1    Running    0        5m
...
```

b. 使用以下命令，查看上一命令的输出中所列 pod 的日志：

```
$ oc logs <pod_name> -n <namespace> 1
```

1 指定 pod 名称和命名空间，如上一命令的输出中所示。

如果 pod 日志显示，Kubernetes API 服务器可以与集群机器通信。

3. 对于使用光纤通道协议(FCP)的安装，还需要额外的步骤才能启用多路径。不要在安装过程中启用多路径。

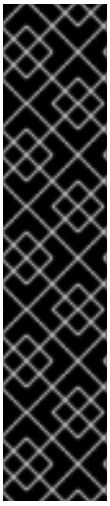
如需更多信息，请参阅 *安装后机器配置任务* 文档中的"使用 RHCOS 上使用内核参数启用多路径"。

4. 在 Cluster registration 页面注册 您的集群。

您可以按照将计算机器 添加到 vSphere 的内容在集群安装后添加额外的计算机器。

23.3.5.17. 为 control plane 节点配置 vSphere DRS 反关联性规则

可将 vSphere 分布式资源调度程序 DRS) 关联性规则配置为支持 OpenShift Container Platform Control Plane 节点的高可用性。反关联性规则确保 OpenShift Container Platform Control Plane 节点的 vSphere 虚拟机没有调度到同一 vSphere 主机。



重要

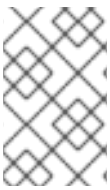
- 以下信息只适用于计算 DRS，不适用于存储 DRS。
- govc 命令是 VMware 提供的开源命令；它不是红帽提供的。红帽不支持 govc 命令。
- 有关下载和安装 govc 的说明，请参阅 VMware 文档网站。

运行以下命令来创建反关联性规则：

示例命令

```
$ govc cluster.rule.create \
  -name openshift4-control-plane-group \
  -dc MyDatacenter -cluster MyCluster \
  -enable \
  -anti-affinity master-0 master-1 master-2
```

创建规则后，您的 control plane 节点由 vSphere 自动迁移，以便它们不会在同一主机上运行。当 vSphere 协调新规则时，这可能需要一些时间。以下流程中会显示成功的命令完成。



注意

迁移会自动进行，并可能导致 OpenShift API 中断或延迟，直到迁移完成为止。

当 control plane 虚拟机名称发生变化或迁移到新的 vSphere 集群时，需要手动更新 vSphere DRS

反关联性规则。

流程

1. 运行以下命令来删除任何现有的 DRS 反关联性规则：

```
$ govc cluster.rule.remove \
  -name openshift4-control-plane-group \
  -dc MyDatacenter -cluster MyCluster
```

输出示例

```
[13-10-22 09:33:24] Reconfigure /MyDatacenter/host/MyCluster...OK
```

2. 运行以下命令，使用更新的名称再次创建规则：

```
$ govc cluster.rule.create \
  -name openshift4-control-plane-group \
  -dc MyDatacenter -cluster MyOtherCluster \
  -enable \
  -anti-affinity master-0 master-1 master-2
```

23.3.5.18. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于 远程健康监控

23.3.5.19. 后续步骤

- 自定义集群。
- 如果您用来安装集群的镜像 registry 具有可信任的 CA，请通过 配置额外的信任存储将其添加到集群中。
- 如果需要，您可以选择 不使用远程健康报告。
- 可选： 查看 vSphere 问题检测器 Operator 中的事件，以确定集群是否有权限或存储配置问题。
- 可选：如果您创建了加密的虚拟机，请创建一个加密的存储类。

23.4. 使用 ASSISTED INSTALLER 在 VSPHERE 上安装集群

您可以使用辅助安装程序在内部硬件或内部虚拟机中安装 OpenShift Container Platform。使用 Assisted Installer 安装 OpenShift Container Platform 支持 x86_64、AArch 64、ppc64le 和 s390x CPU 架构。

Assisted Installer 是一个在 Red Hat Hybrid Cloud Console 上提供的用户友好的安装解决方案。

23.4.1. 其他资源

- 使用 Assisted Installer 安装 OpenShift Container Platform

23.5. 使用基于代理的安装程序在 VSPHERE 上安装集群

基于代理的安装方法提供了以任何方式引导内部服务器的灵活性。它将辅助安装服务的使用与离线运行的功能相结合，包括在 air-gapped 环境中。

基于代理的安装是 OpenShift Container Platform 安装程序的子命令。它生成可引导 ISO 镜像，其中包含使用可用发行镜像部署 OpenShift Container Platform 集群所需的所有信息。

23.5.1. 其他资源

- 准备使用基于代理的安装程序安装

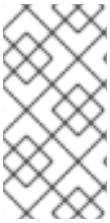
23.6. 在 VSPHERE 上安装三节点集群

在 OpenShift Container Platform 版本 4.16 中，您可以在 VMware vSphere 上安装三节点集群。三节点集群包含三个 control plane 机器，它们也可以充当计算机器。这种类型的集群提供了一个较小的、效率更高的集群，供集群管理员和开发人员用于测试、开发和生产。

您可以使用安装程序置备或用户置备的基础架构安装三节点集群。

23.6.1. 配置三节点集群

在部署集群前，您可以通过将 install-config.yaml 文件中的 worker 节点数量设置为 0 来配置三节点集群。将 worker 节点数量设置为 0 可确保 control plane 机器可以调度。这允许调度应用程序工作负载从 control plane 节点运行。



注意

因为应用程序工作负载从 control plane 节点运行，所以需要额外的订阅，因为 control plane 节点被视为计算节点。

先决条件

- 您有一个现有的 install-config.yaml 文件。

流程

1. 将 install-config.yaml 文件中的计算副本数量设置为 0，如以下 compute 小节中所示：

三节点集群的 install-config.yaml 文件示例

```
apiVersion: v1
baseDomain: example.com
compute:
- name: worker
```

```
  platform: {}
  replicas: 0
# ...
```

2. 如果您使用用户置备的基础架构部署集群：

   - 配置应用程序入口负载均衡器，将 HTTP 和 HTTPS 流量路由到 control plane 节点。在三节点集群中，Ingress Controller pod 在 control plane 节点上运行。如需更多信息，请参阅"用户置备的基础架构负载平衡要求"。

   - 创建 Kubernetes 清单文件后，请确保在 cluster-scheduler-02-config.yml 文件中将 spec.mastersSchedulable 参数设置为 true。您可以在 <installation_directory>/manifests 中找到此文件。如需更多信息，请参阅"使用用户置备的基础架构在 vSphere 上安装集群"中的"创建 Kubernetes 清单和 Ignition 配置文件"。

   - 不要创建额外的 worker 节点。

三节点集群的 cluster-scheduler-02-config.yml 文件示例

```
apiVersion: config.openshift.io/v1
kind: Scheduler
metadata:
  creationTimestamp: null
  name: cluster
spec:
  mastersSchedulable: true
  policy:
    name: ""
status: {}
```

23.6.2. 后续步骤

- 使用自定义在 vSphere 上安装集群

- 使用用户置备的基础架构在 vSphere 上安装集群

23.7. 在使用安装程序置备的基础架构的 VSPHERE 上卸载集群

您可以使用安装程序置备的基础架构删除您在 VMware vSphere 实例中部署的集群。

注意

运行 openshift-install destroy cluster 命令时，卸载 OpenShift Container Platform 时，vSphere 卷不会被自动删除。集群管理员必须手动找到 vSphere 卷并删除它们。

23.7.1. 删除使用安装程序置备的基础架构的集群

您可以从云中删除使用安装程序置备的基础架构的集群。

注意

卸载后，检查云供应商是否有未正确删除的资源，特别是在用户置备基础架构(UPI)集群中。可能存在安装程序未创建或安装程序无法访问的资源。

先决条件

- 有用于部署集群的安装程序副本。
- 有创建集群时安装程序生成的文件。

流程

1. 在用来安装集群的计算机中包含安装程序的目录中，运行以下命令：

```
$ ./openshift-install destroy cluster \
--dir <installation_directory> --log-level info 1 2
```

1. 对于 <installation_directory>，请指定安装文件保存到的目录的路径。

2

要查看不同的详情，请指定 warn、debug 或 error，而不是 info。

注意

您必须为集群指定包含集群定义文件的目录。安装程序需要此目录中的 metadata.json 文件来删除集群。

2. 可选：删除 <installation_directory> 目录和 OpenShift Container Platform 安装程序。

23.8. 使用 VSPHERE 问题检测器 OPERATOR

23.8.1. 关于 vSphere 问题检测器 Operator

vSphere 问题检测器 Operator 会检查在 vSphere 上部署的集群，以获取与存储相关的常见安装和错误配置问题。

Operator 在 openshift-cluster-storage-operator 命名空间中运行，并在 Cluster Storage Operator 检测到集群部署到 vSphere 时由 Cluster Storage Operator 启动。vSphere 问题检测器 Operator 与 vSphere vCenter 服务器通信，以确定集群中的虚拟机、默认数据存储以及 vSphere vCenter Server 配置的其他信息。Operator 使用 Cloud Credential Operator 中的凭证连接到 vSphere。

Operator 根据以下调度运行检查：

- 检查每 8 小时运行一次。
- 如果检查失败，Operator 会以 1 分钟、2 分钟、4 分钟、8 分钟等间隔再次运行检查。Operator 间隔最多为 8 小时。
- 当所有检查都通过后，调度会返回到 8 小时间隔。

Operator 在失败后会增加检查的频率，以便 Operator 可以在修复失败条件后快速报告成功。您可以手动运行 Operator 来获取即时故障排除信息。

23.8.2. 运行 vSphere 问题检测器 Operator 检查

您可以覆盖运行 vSphere 问题检测器 Operator 检查的调度，并立即运行检查。

vSphere 问题检测器 Operator 每 8 小时自动运行检查。但是，当 Operator 启动时，它会立即运行检查。当 Cluster Storage Operator 启动并确定集群是否在 vSphere 上运行时，Operator 由 Cluster Storage Operator 启动。要立即运行检查，您可以将 vSphere 问题检测器 Operator 扩展至 0，并返回到 1，以便它重启 vSphere 问题检测器 Operator。

先决条件

- 使用具有 cluster-admin 角色的用户访问集群。

流程

1. 将 Operator 扩展至 0:

```
$ oc scale deployment/vsphere-problem-detector-operator --replicas=0 \
   -n openshift-cluster-storage-operator
```

如果部署没有立即缩减为零，您可以运行以下命令来等待 pod 退出：

```
$ oc wait pods -l name=vsphere-problem-detector-operator \
   --for=delete --timeout=5m -n openshift-cluster-storage-operator
```

2. 将 Operator 扩展至 1 ：

```
$ oc scale deployment/vsphere-problem-detector-operator --replicas=1 \
   -n openshift-cluster-storage-operator
```

3. 删除旧的领导锁定，以加快 Cluster Storage Operator 的新领导选举机制：

```
$ oc delete -n openshift-cluster-storage-operator \
   cm vsphere-problem-detector-lock
```

验证

- 查看 vSphere 问题检测器 Operator 生成的事件或日志。确认事件或日志具有最新的时间戳。

23.8.3. 从 vSphere 问题检测器 Operator 中查看事件

在 vSphere 问题检测器 Operator 运行并执行配置检查后，它会创建可从命令行或 OpenShift Container Platform Web 控制台查看的事件。

流程

- 要使用命令行查看事件，请运行以下命令：

```
$ oc get event -n openshift-cluster-storage-operator \
  --sort-by={.metadata.creationTimestamp}
```

输出示例

```
16m     Normal   Started              pod/vsphere-problem-detector-operator-xxxxx
Started container vsphere-problem-detector
16m     Normal   Created              pod/vsphere-problem-detector-operator-xxxxx
Created container vsphere-problem-detector
16m     Normal   LeaderElection       configmap/vsphere-problem-detector-lock
vsphere-problem-detector-operator-xxxxx became leader
```

- 要使用 OpenShift Container Platform Web 控制台查看事件，请导航到 Home → Events，然后从 Project 菜单中选择 openshift-cluster-storage-operator。

23.8.4. 从 vSphere 问题检测器 Operator 查看日志

在 vSphere 问题检测器 Operator 运行并执行配置检查后，它会创建日志记录，这些记录可以从命令行或 OpenShift Container Platform Web 控制台查看。

流程

- 要使用命令行查看日志，请运行以下命令：

```
$ oc logs deployment/vsphere-problem-detector-operator \
  -n openshift-cluster-storage-operator
```

输出示例

```
I0108 08:32:28.445696       1 operator.go:209] ClusterInfo passed
I0108 08:32:28.451029       1 datastore.go:57] CheckStorageClasses checked 1 storage
classes, 0 problems found
I0108 08:32:28.451047       1 operator.go:209] CheckStorageClasses passed
I0108 08:32:28.452160       1 operator.go:209] CheckDefaultDatastore passed
I0108 08:32:28.480648       1 operator.go:271] CheckNodeDiskUUID:<host_name>
passed
I0108 08:32:28.480685       1 operator.go:271] CheckNodeProviderID:<host_name>
passed
```

- 要使用 OpenShift Container Platform Web 控制台查看 Operator 日志，请执行以下步骤：

  a. 导航到 Workloads → Pods。

  b. 从 Projects 菜单中选择 openshift-cluster-storage-operator。

  c. 点 vsphere-problem-detector-operator pod 的链接。

  d. 点击 Pod 详情 页面上的 Logs 选项卡查看日志。

23.8.5. 由 vSphere 问题检测器 Operator 运行的配置检查

下表标识了配置检查是否运行 vSphere 问题检测程序 Operator。有些检查会验证集群的配置。其他检查验证集群中每个节点的配置。

表 23.40. 集群配置检查

名称	描述

名称	描述
CheckDefaultDatastore	验证 vSphere 配置中的默认数据存储名称是否足够短，可用于动态置备。 如果这个检查失败，您可以预期如下情况： ● **systemd** 将错误记录到日志中，如 **Failed to set up mount unit: Invalid argument**。 ● 如果在虚拟机没有从节点排空所有 pod 的情况下关闭或重新引导虚拟机，**systemd** 不会卸载卷。 如果这个检查失败，请重新配置 vSphere，其默认数据存储的名称较短。
CheckFolderPermissions	验证列出默认数据存储中卷的权限。创建卷时需要此权限。Operator 通过列出 **/** 和 **/kubevols** 目录来验证权限。根目录必须存在。如果检查运行时不存在 **/kubevols** 目录，则可以接受。如果不存在，当数据存储用于动态置备时 **/kubevols** 目录会被创建。 如果这个检查失败，请查看 OpenShift Container Platform 安装过程中指定的 vCenter 帐户所需的权限。
CheckStorageClasses	验证以下内容： ● 这个存储类置备的每个持久性卷的完全限定路径小于 255 个字符。 ● 如果存储类使用存储策略，存储类必须只使用一个策略，且必须定义该策略。
CheckTaskPermissions	验证列出最新任务和数据存储的权限。
ClusterInfo	从 vSphere vCenter 收集集群版本和 UUID。

表 23.41. 节点配置检查

名称	描述
CheckNodeDiskUUID	验证所有 vSphere 虚拟机是否都配置了 **disk.enableUUID=TRUE**。 如果这个检查失败，请参阅 如何在 vSphere Red Hat Knowledgebase 解决方案中检查虚拟机的 'disk.EnableUUID' 参数。

CheckNodeProviderID	验证所有节点是否都配置了来自 vSphere vCenter 的 **ProviderID**。当以下命令的输出不包括每个节点的供应商 ID 时，此检查会失败。
	
	``
	$ oc get nodes -o custom-columns=NAME:.metadata.name,PROVIDER_ID:.spec.providerID,UUID:.status.nodeInfo.systemUUID
	``
	
	如果这个检查失败，请参阅 vSphere 产品文档来获取集群中每个节点的供应商 ID 的信息。

名称	描述
CollectNodeESXiVersion	报告运行节点的 ESXi 主机的版本。
CollectNodeHWVersion	报告节点的虚拟机硬件版本。

23.8.6. 关于存储类配置检查

使用 vSphere 存储的持久性卷的名称与数据存储名称和集群 ID 相关。

创建持久性卷时，systemd 为持久性卷创建一个挂载单元。systemd 进程有 255 个字符限制，用于持久性卷的 VDMK 文件的完全限定路径长度。

完全限定路径基于 systemd 和 vSphere 的命名约定。命名惯例使用以下模式：

```
/var/lib/kubelet/plugins/kubernetes.io/vsphere-volume/mounts/[<datastore>] 00000000-0000-0000-0000-000000000000/<cluster_id>-dynamic-pvc-00000000-0000-0000-0000-000000000000.vmdk
```

- 命名惯例需要 255 个字符限制的 205 个字符。
- 数据存储名称和集群 ID 由部署决定。
- 数据存储名称和集群 ID 替换为上述模式。然后，路径通过 systemd-escape 命令处理，以转义特殊字符。例如，连字符在转义后使用四个字符。转义的值为 \x2d。
- 在使用 systemd-escape 处理后，确保 systemd 可以访问 VDMK 文件的完全限定路径后，路径的长度必须小于 255 个字符。

23.8.7. vSphere 问题检测器 Operator 的指标

vSphere 问题检测器 Operator 会公开以下指标，供 OpenShift Container Platform 监控堆栈使用。

表 23.42. vSphere 问题检测器 Operator 公开的指标

名称	描述
vsphere_cluster_check_total	检查 vSphere 问题检测程序 Operator 是否执行的集群级别的累积数量。这一计数包括成功和失败。
vsphere_cluster_check_errors	检查 vSphere 问题检测程序 Operator 是否执行失败的集群级别数量。例如，1 表示一个集群级别的检查失败。
vsphere_esxi_version_total	具有特定版本的 ESXi 主机数量。请注意，如果主机运行多个节点，则该主机仅计算一次。
vsphere_node_check_total	检查 vSphere 问题检测程序 Operator 是否执行的节点级别的累积数量。这一计数包括成功和失败。
vsphere_node_check_errors	是否执行 vSphere 问题检测程序 Operator 的失败节点级别检查数量。例如，1 表示一个节点级别的检查失败。
vsphere_node_hw_version_total	具有特定硬件版本的 vSphere 节点数。
vsphere_vcenter_info	有关 vSphere vCenter 服务器的信息。

23.8.8. 其他资源

- 监控概述

23.9. VSPHERE 的安装配置参数

在 vSphere 上部署 OpenShift Container Platform 集群前，您可以提供参数来自定义集群和托管它的平台。在创建 install-config.yaml 文件时，您可以通过命令行为所需参数提供值。然后，您可以修改 install-config.yaml 文件以进一步自定义集群。

23.9.1. vSphere 可用的安装配置参数

下表指定可设置为安装过程的一部分所需的、可选和特定于 vSphere 的安装配置参数。

注意

安装后，您无法在 install-config.yaml 文件中修改这些参数。

23.9.1.1. 所需的配置参数

下表描述了所需的安装配置参数：

表 23.43. 所需的参数

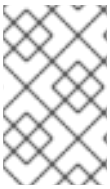
参数	描述	值
apiVersion:	**install-config.yaml** 内容的 API 版本。当前版本为 **v1**。安装程序可能还支持旧的 API 版本。	字符串
baseDomain:	云供应商的基域。基域用于创建到 OpenShift Container Platform 集群组件的路由。集群的完整 DNS 名称是 **baseDomain** 和 **metadata.name** 参数值的组合，其格式为 **<metadata.name>.<baseDomain>**。	完全限定域名或子域名，如 **example.com**。
metadata:	Kubernetes 资源 **ObjectMeta**，其中只消耗 **name** 参数。	对象
metadata: name:	集群的名称。集群的 DNS 记录是 **{{.metadata.name}}.{{.baseDomain}}** 的子域。	小写字母和连字符 (**-**) 的字符串，如 **dev**。
platform:	对于特定平台的配置取决于执行安装的环境： **aws**, **baremetal**, **azure**, **gcp**, **ibmcloud**, **nutanix**, **openstack**, **powervs**, **vsphere**, 或 **{}**.有关 **platform.<platform>** 参数的更多信息，请参考下表中您的特定平台。	对象

参数	描述	值
pullSecret:	从 Red Hat OpenShift Cluster Manager 获取 pull secret，验证从 Quay.io 等服务中下载 OpenShift Container Platform 组件的容器镜像。	{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }

23.9.1.2. 网络配置参数

您可以根据现有网络基础架构的要求自定义安装配置。例如，您可以扩展集群网络的 IP 地址块，或者提供不同于默认值的不同 IP 地址块。

- 如果使用 Red Hat OpenShift Networking OVN-Kubernetes 网络插件，则支持 IPv4 和 IPv6 地址系列。



注意

在 VMware vSphere 上，双栈网络可以指定 IPv4 或 IPv6 作为主要地址系列。

如果将集群配置为使用两个 IP 地址系列，请查看以下要求：

- 两个 IP 系列都必须将相同的网络接口用于默认网关。
- 两个 IP 系列都必须具有默认网关。
- 您必须为所有网络配置参数指定 IPv4 和 IPv6 地址。例如，以下配置 IPv4 地址列在 IPv6 地址的前面。

```
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  - cidr: fd00:10:128::/56
    hostPrefix: 64
  serviceNetwork:
  - 172.30.0.0/16
  - fd00:172:16::/112
```

注意

Red Hat OpenShift Data Foundation 灾难恢复解决方案不支持 Globalnet。对于区域灾难恢复场景，请确保为每个集群中的集群和服务网络使用非重叠的专用 IP 地址。

表 23.44. 网络参数

参数	描述	值
networking:	集群网络的配置。	对象 **注意** 您无法在安装后修改 **网络** 对象指定的参数。
networking: networkType:	要安装的 Red Hat OpenShift Networking 网络插件。	**OVNKubernetes**。**OVNKubernetes** 是 Linux 网络和包含 Linux 和 Windows 服务器的混合网络的 CNI 插件。默认值为 **OVNKubernetes**。
networking: clusterNetwork:	pod 的 IP 地址块。 默认值为 **10.128.0.0/14**，主机前缀为 **/23**。 如果您指定了多个 IP 地址块，块不得重叠。	对象数组。例如： networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23
networking: clusterNetwork: cidr:	使用 **networking.clusterNetwork** 时需要此项。IP 地址块。 IPv4 网络。	无类别域间路由(CIDR)表示法中的 IP 地址块。IPv4 块的前缀长度介于 **0 到 32** 之间。

参数	描述	值
networking: clusterNetwork: hostPrefix:	分配给每个节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从 given **cidr** 中分配 a/**23** 子网。**hostPrefix** 值 **23** 提供 510（2^(32 - 23)- 2）pod IP 地址。	子网前缀。 默认值为 **23**。
networking: serviceNetwork:	服务的 IP 地址块。默认值为 **172.30.0.0/16**。 OVN-Kubernetes 网络插件只支持服务网络的一个 IP 地址块。	CIDR 格式具有 IP 地址块的数组。例如： networking: serviceNetwork: - 172.30.0.0/16
networking: machineNetwork:	机器的 IP 地址块。 如果您指定了多个 IP 地址块，块不得重叠。	对象数组。例如： networking: machineNetwork: - cidr: 10.0.0.0/16
networking: machineNetwork: cidr:	使用 **networking.machineNetwork** 时需要此项。IP 地址块。对于 libvirt 和 IBM Power® Virtual Server 以外的所有平台，默认值为 **10.0.0.0/16**。对于 libvirt，默认值 为 **192.168.126.0/24**。对于 IBM Power® Virtual Server，默认值为 **192.168.0.0/24**。	CIDR 表示法中的 IP 网络块。 例如： **10.0.0.0/16**。 **注意** 将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

23.9.1.3. 可选的配置参数

下表描述了可选的安装配置参数：

表 23.45. 可选参数

参数	描述	值
additionalTrustBundle:	添加到节点可信证书存储中的 PEM 编码 X.509 证书捆绑包。配置了代理时，也可以使用此信任捆绑包。	字符串

参数	描述	值
capabilities:	控制可选核心组件的安装。您可以通过禁用可选组件来减少 OpenShift Container Platform 集群的空间。如需更多信息，请参阅*安装*中的"集群功能"页面。	字符串数组
capabilities: baselineCapabilitySet:	选择要启用的一组初始可选功能。有效值为 **None**、**v4.11**、**v4.12** 和 **vCurrent**。默认值为 **vCurrent**。	字符串
capabilities: additionalEnabledCapabilities:	将可选功能集合扩展到您在 **baselineCapabilitySet** 中指定的范围。您可以在此参数中指定多个功能。	字符串数组
cpuPartitioningMode:	启用工作负载分区，它会隔离 OpenShift Container Platform 服务、集群管理工作负载和基础架构 pod，以便在保留的一组 CPU 上运行。工作负载分区只能在安装过程中启用，且在安装后无法禁用。虽然此字段启用工作负载分区，但它不会将工作负载配置为使用特定的 CPU。如需更多信息，请参阅 *Scalability and Performance* 部分中的 *Workload partitioning* 页面。	**None** 或 **AllNodes**.**None** 是默认值。
compute:	组成计算节点的机器的配置。	**MachinePool** 对象的数组。
compute: architecture:	决定池中机器的指令集合架构。目前，不支持具有不同架构的集群。所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
compute: name:	使用 **compute** 时需要此项。机器池的名称。	**worker**
compute: platform:	使用 **compute** 时需要此项。使用此参数指定托管 worker 机器的云供应商。此参数值必须与 **controlPlane.platform** 参数值匹配。	**aws**, **azure**, **gcp**, **ibmcloud**, **nutanix**, **openstack**, **powervs**, **vsphere**, 或 **{}**

参数	描述	值
compute: replicas:	要置备的计算机器数量，也称为 worker 机器。	大于或等于 **2** 的正整数。默认值为 **3**。
featureSet:	为功能集启用集群。功能集是 OpenShift Container Platform 功能的集合，默认情况下不启用。有关在安装过程中启用功能集的更多信息，请参阅"使用功能门启用功能"。	字符串.要启用的功能集的名称，如 **TechPreviewNoUpgrade**。
controlPlane:	组成 control plane 的机器的配置。	**MachinePool** 对象的数组。
controlPlane: architecture:	决定池中机器的指令集合架构。目前，不支持具有不同架构的集群。所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
controlPlane: name:	使用 **controlPlane** 时需要此项。机器池的名称。	**master**
controlPlane: platform:	使用 **controlPlane** 时需要此项。使用此参数指定托管 control plane 机器的云供应商。此参数值必须与 **compute.platform** 参数值匹配。	**aws**, **azure**, **gcp**, **ibmcloud**, **nutanix**, **openstack**, **powervs**, **vsphere**, 或 **{}**
controlPlane: replicas:	要置备的 control plane 机器数量。	部署单节点 OpenShift 时支持的值为 **3** 或 **1**。
credentialsMode:	Cloud Credential Operator(CCO)模式。如果没有指定模式，CCO 会动态尝试决定提供的凭证的功能，在支持多个模式的平台上首选 mint 模式。	**Mint**、**Passthrough**、**Manual** 或空字符串(**""**)。[1]

参数	描述	值
fips:	启用或禁用 FIPS 模式。默认值为 **false**（禁用）。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。  **重要** 要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 Red Hat Enterprise Linux (RHEL) 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅在 FIPS 模式中安装该系统。 当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS) 时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。  **注意** 如果使用 Azure File 存储，则无法启用 FIPS 模式。	**false** 或 **true**

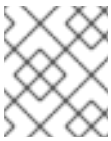
参数	描述	值
imageContentSources:	release-image 内容的源和存储库。	对象数组。包括一个 **source** 以及可选的 **mirrors**，如本表的以下行所述。
imageContentSources: source:	使用 **imageContentSources** 时需要此项。指定用户在镜像拉取规格中引用的存储库。	字符串
imageContentSources: mirrors:	指定可能还包含同一镜像的一个或多个仓库。	字符串数组
publish:	如何发布或公开集群的面向用户的端点，如 Kubernetes API、OpenShift 路由。	**内部或外部 .**默认值为 **External**。 在非云平台上不支持将此字段设置为 **Internal**。 **重要** 如果将字段的值设为 **Internal**，集群将无法运行。如需更多信息，请参阅 BZ#1953035。
sshKey:	用于验证对集群机器的访问的 SSH 密钥。 **注意** 对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。	例如，**sshKey: ssh-ed25519 AAAA..**。

1. **不是所有 CCO 模式都支持所有云供应商。有关 CCO 模式的更多信息，请参阅*身份验证和授权*内容中的"管理云供应商凭证"条目。**

23.9.1.4. 其他 VMware vSphere 配置参数

下表描述了其他 VMware vSphere 配置参数：

表 23.46. 其他 VMware vSphere 集群参数

参数	描述	值
platform: vsphere:	描述托管集群的云平台中的帐户。您可以使用参数来自定义平台。如果您为机器池中的 compute 和 control plane 机器提供额外的配置设置，则不需要该参数。您只能为 OpenShift Container Platform 集群指定一个 vCenter 服务器。	vSphere 配置对象的字典
platform: vsphere: apiVIPs:	为 control plane API 访问配置的虚拟 IP (VIP) 地址。  **注意** 这个参数只适用于安装程序置备的基础架构。	多个 IP 地址
platform: vsphere: diskType:	可选：磁盘置备方法。如果没有设置，则默认值为 vSphere 默认存储策略。	有效值为 **thin**、**thick** 或 **eagerZeroedThick**。
platform: vsphere: failureDomains:	建立地区和区域之间的关系。您可以使用 vCenter 对象（如 **datastore** 对象）定义故障域。故障域定义 OpenShift Container Platform 集群节点的 vCenter 位置。	故障域配置对象的数组。
platform: vsphere: failureDomains: name:	故障域的名称。	字符串
platform: vsphere: failureDomains: region:	如果为集群定义多个故障域，则必须将标签附加到每个 vCenter 数据中心。要定义区域，请使用 **openshift-region** 标签类别中的标签。对于单个 vSphere 数据中心环境，您不需要附加标签，但必须为参数输入一个字母数字值，如 **datacenter**。	字符串
platform: vsphere: failureDomains: server:	指定 VMware vCenter 服务器的完全限定主机名或 IP 地址，以便客户端可以访问故障域资源。您必须将 **server** 器角色应用到 vSphere vCenter 服务器位置。	字符串

参数	描述	值
platform: vsphere: failureDomains: zone:	如果为集群定义多个故障域，则必须为每个 vCenter 集群附加标签。要定义一个区，请使用 **openshift-zone** 标签类别中的标签。对于单个 vSphere 数据中心环境，您不需要附加标签，但必须为参数输入一个字母数字值，如 **cluster**。	字符串
platform: vsphere: failureDomains: topology: computeCluster:	vSphere 计算集群的路径。	字符串
platform: vsphere: failureDomains: topology: datacenter:	列出并定义 OpenShift Container Platform 虚拟机 (VM) 操作的数据中心。数据中心列表必须与 **vcenters** 字段中指定的数据中心列表匹配。	字符串
platform: vsphere: failureDomains: topology: datastore:	指定为故障域存储虚拟机文件的 vSphere 数据存储的路径。您必须将 **datastore** 角色应用到 vSphere vCenter 数据存储位置。	字符串
platform: vsphere: failureDomains: topology: folder:	可选：用户创建虚拟机的现有文件夹的绝对路径，例如 **/<datacenter_name>/vm/<folder_name>/<subfolder_name>**。如果没有提供这个值，安装程序会在数据中心虚拟机文件夹中创建一个顶层文件夹，其名称为基础架构 ID。如果您为集群提供基础架构，且您不想使用默认的 **StorageClass** 对象（名为 **thin**），您可以从 **install-config.yaml** 文件中省略 **folder** 参数。	字符串
platform: vsphere: failureDomains: topology: networks:	列出 vCenter 实例中包含您配置的虚拟 IP 地址和 DNS 记录的任何网络。	字符串

参数	描述	值
platform: vsphere: failureDomains: topology: resourcePool:	可选：安装程序创建虚拟机的现有资源池的绝对路径，例如 **/<datacenter_name>/host/<cluster_name>/Resources/<resource_pool_name>/<optional_nested_resource_pool_name>**。如果没有指定值，安装程序会在 **/<datacenter_name>/host/<cluster_name>/Resources** 下的集群的 root 中安装资源。	字符串
platform: vsphere: failureDomains: topology template:	指定预先存在的 Red Hat Enterprise Linux CoreOS (RHCOS) 镜像模板或虚拟机的绝对路径。安装程序可以使用镜像模板或虚拟机在 vSphere 主机上快速安装 RHCOS。考虑使用此参数作为在 vSphere 主机上上传 RHCOS 镜像的替代选择。此参数仅适用于安装程序置备的基础架构。	字符串
platform: vsphere: ingressVIPs:	为集群 Ingress 配置的虚拟 IP (VIP) 地址。 **注意** 这个参数只适用于安装程序置备的基础架构。	多个 IP 地址
platform: vsphere: vcenters:	配置连接详情，以便服务可以与 vCenter 服务器通信。目前，只支持单个 vCenter 服务器。	vCenter 配置对象的数组。
platform: vsphere: vcenters: datacenters:	列出并定义 OpenShift Container Platform 虚拟机 (VM) 操作的数据中心。数据中心列表必须与 **failureDomains** 字段中指定的数据中心列表匹配。	字符串
platform: vsphere: vcenters: password:	与 vSphere 用户关联的密码。	字符串

参数	描述	值
platform: vsphere: vcenters: port:	用于与 vCenter 服务器通信的端口号。	整数
platform: vsphere: vcenters: server:	vCenter 服务器的完全限定主机名(FQHN)或 IP 地址。	字符串
platform: vsphere: vcenters: user:	与 vSphere 用户关联的用户名。	字符串

23.9.1.5. 弃用的 VMware vSphere 配置参数

在 OpenShift Container Platform 4.13 中，以下 vSphere 配置参数已弃用。您可以继续使用这些参数，但安装程序不会在 install-config.yaml 文件中自动指定这些参数。

下表列出了每个已弃用的 vSphere 配置参数：

表 23.47. 弃用的 VMware vSphere 集群参数

参数	描述	值
platform: vsphere: apiVIP:	为 control plane API 访问配置的虚拟 IP(VIP)地址。 **注意** 在 OpenShift Container Platform 4.12 及更新的版本中，**apiVIP** 配置设置已弃用。反之，使用 **List** 格式在 **apiVIPs** 配置设置中输入值。	IP 地址，如 **128.0.0.1**。
platform: vsphere: cluster:	安装 OpenShift Container Platform 集群的 vCenter 集群。	字符串

参数	描述	值
platform: vsphere: datacenter:	定义 OpenShift Container Platform 虚拟机 (VM) 操作的数据中心。	字符串
platform: vsphere: defaultDatastore:	用于调配卷的默认数据存储名称。	字符串
platform: vsphere: folder:	可选：安装程序创建虚拟机的现有文件夹的绝对路径。如果没有提供这个值，安装程序会创建一个文件夹，它的名称为 datacenter 虚拟机文件夹中的基础架构 ID。	字符串，如 **/<datacenter_name>/vm/<folder_name>/<subfolder_name>**。
platform: vsphere: ingressVIP:	为集群 Ingress 配置的虚拟 IP (VIP) 地址。 **注意** 在 OpenShift Container Platform 4.12 及更新的版本中，**ingressVIP** 配置设置已弃用。反之，使用 **List** 格式在 **ingressVIPs** 配置设置中输入值。	IP 地址，如 **128.0.0.1**。
platform: vsphere: network:	包含您配置的虚拟 IP 地址和 DNS 记录的 vCenter 实例中的网络。	字符串
platform: vsphere: password:	vCenter 用户名的密码。	字符串
platform: vsphere: resourcePool:	可选：安装程序创建虚拟机的现有资源池的绝对路径。如果没有指定值，安装程序会在 **/<datacenter_name>/host/<cluster_name>/Resources** 下的集群的 root 中安装资源。	字符串，例如 **/<datacenter_name>/host/<cluster_name>/Resources/<resource_pool_name>/<optional_nested_resource_pool_name>**。

参数	描述	值
platform: vsphere: username:	用于连接 vCenter 实例的用户名。此用户必须至少具有 vSphere 中 静态或动态持久性卷置备 所需的角色和权限。	字符串
platform: vsphere: vCenter:	vCenter 服务器的完全限定主机名或 IP 地址。	字符串

23.9.1.6. 可选的 VMware vSphere 机器池配置参数

下表描述了可选的 VMware vSphere 机器池配置参数：

表 23.48. 可选的 VMware vSphere 机器池参数

参数	描述	值
platform: vsphere: clusterOSImage:	安装程序从中下载 Red Hat Enterprise Linux CoreOS (RHCOS) 镜像的位置。在为此参数设置路径值前，请确保 OpenShift Container Platform 发行版本中的默认 RHCOS 引导镜像与 RHCOS 镜像模板或虚拟机版本匹配；否则，集群安装可能会失败。	HTTP 或 HTTPS URL，可选使用 SHA-256 校验和。例如：**https://mirror.openshift.com/images/rhcos-<version>-vmware.<architecture>.ova**。
platform: vsphere: osDisk: diskSizeGB:	以 GB 为单位的磁盘大小。	整数
platform: vsphere: cpus:	用于分配虚拟机的虚拟处理器内核总数。**platform.vsphere.cpus** 的值必须是 **platform.vsphere.coresPerSocket** 值的倍数。	整数
platform: vsphere: coresPerSocket:	虚拟机中每个插槽的内核数。虚拟机上的虚拟套接字数量为 platform **.vsphere.cpus/platform.vsphere.coresPerSocket**。control plane 节点和 worker 节点的默认值为**4** 和 **2**。	整数

参数	描述	值

platform:
  vsphere:
  memoryMB:
	以 MB 为单位的虚拟机内存大小。	整数

第 24 章 在任意平台上安装

24.1. 在任意平台上安装集群

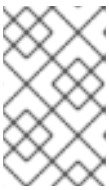
在 OpenShift Container Platform 版本 4.16 中，您可以在您置备的任何基础架构上安装集群，包括虚拟化和云环境。

重要

在尝试在虚拟化或云环境中安装 OpenShift Container Platform 集群前，请参阅有关在未经测试的平台上部署 OpenShift Container Platform 的指南 中的信息。

24.1.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读有关 选择集群安装方法的文档，并为用户准备它。
- 如果使用防火墙，则会 将其配置为允许集群需要访问的站点。



注意

如果要配置代理，请务必查看此站点列表。

24.1.2. OpenShift Container Platform 互联网访问

在 OpenShift Container Platform 4.16 中，您需要访问互联网来安装集群。

您必须具有以下互联网访问权限：

- 访问 OpenShift Cluster Manager 以下载安装程序并执行订阅管理。如果集群可以访问互联网，并且没有禁用 Telemetry，该服务会自动授权您的集群。

- 访问 Quay.io，以获取安装集群所需的软件包。
- 获取执行集群更新所需的软件包。

重要

如果您的集群无法直接访问互联网，则可以在置备的某些类型的基础架构上执行受限网络安装。在此过程中，您可以下载所需的内容，并使用它为镜像 registry 填充安装软件包。对于某些安装类型，集群要安装到的环境不需要访问互联网。在更新集群前，您要更新镜像 registry 的内容。

24.1.3. 具有用户置备基础架构的集群的要求

对于包含用户置备的基础架构的集群，您必须部署所有所需的机器。

本节论述了在用户置备的基础架构上部署 OpenShift Container Platform 的要求。

24.1.3.1. 集群安装所需的机器

最小的 OpenShift Container Platform 集群需要以下主机：

表 24.1. 最低所需的主机

主机	描述
一个临时 bootstrap 机器	集群需要 bootstrap 机器在三台 control plane 机器上部署 OpenShift Container Platform 集群。您可在安装集群后删除 bootstrap 机器。
三台 control plane 机器	control plane 机器运行组成 control plane 的 Kubernetes 和 OpenShift Container Platform 服务。
至少两台计算机器，也称为 worker 机器。	OpenShift Container Platform 用户请求的工作负载在计算机器上运行。

重要

要保持集群的高可用性，请将独立的物理主机用于这些集群机器。

bootstrap 和 control plane 机器必须使用 Red Hat Enterprise Linux CoreOS(RHCOS)作为操作系统。但是，计算机器可以在 Red Hat Enterprise Linux CoreOS(RHCOS)、Red Hat Enterprise Linux(RHEL) 8.6 和更高的版本。

请注意，RHCOS 基于 Red Hat Enterprise Linux(RHEL) 9.2，并继承其所有硬件认证和要求。查看红帽企业 Linux 技术功能和限制。

24.1.3.2. 集群安装的最低资源要求

每台集群机器都必须满足以下最低要求：

表 24.2. 最低资源要求

机器	操作系统	vCPU [1]	虚拟内存	Storage	每秒输入/输出(IOPS) [2]
bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane（控制平面）	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS、RHEL 8.6 及更新版本 [3]	2	8 GB	100 GB	300

1. 当未启用并发多线程(SMT)或超线程时，一个 vCPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比例：（每个内核数的线程）× sockets = vCPU。

2. OpenShift Container Platform 和 Kubernetes 对磁盘性能非常敏感，建议使用更快的存储速度，特别是 control plane 节点上需要 10 ms p99 fsync 持续时间的 etcd。请注意，在许多云平台上，存储大小和 IOPS 可一起扩展，因此您可能需要过度分配存储卷来获取足够的性能。

3. 与所有用户置备的安装一样，如果您选择在集群中使用 RHEL 计算机器，则负责所有操作系统生命周期管理和维护，包括执行系统更新、应用补丁和完成所有其他必要的任务。RHEL 7 计算机器的使用已弃用，并已在 OpenShift Container Platform 4.10 及更新的版本中删除。

注意

从 OpenShift Container Platform 版本 4.13 开始，RHCOS 基于 RHEL 版本 9.2，它更新了微架构要求。以下列表包含每个架构需要的最小指令集架构 (ISA)：

- x86-64 体系结构需要 x86-64-v2 ISA
- ARM64 架构需要 ARMv8.0-A ISA
- IBM Power 架构需要 Power 9 ISA
- s390x 架构需要 z14 ISA

如需更多信息，请参阅 RHEL 架构。

如果平台的实例类型满足集群机器的最低要求，则 OpenShift Container Platform 支持使用它。

24.1.3.3. 证书签名请求管理

在使用您置备的基础架构时，集群只能有限地访问自动机器管理，因此您必须提供一种在安装后批准集群证书签名请求 (CSR) 的机制。kube-controller-manager 只能批准 kubelet 客户端 CSR。machine-approver 无法保证使用 kubelet 凭证请求的提供证书的有效性，因为它不能确认是正确的机器发出了该请求。您必须决定并实施一种方法，以验证 kubelet 提供证书请求的有效性并进行批准。

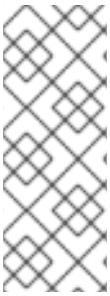
24.1.3.4. 用户置备的基础架构对网络的要求

所有 Red Hat Enterprise Linux CoreOS(RHCOS)机器都需要在启动时在 initramfs 中配置联网，以获取它们的 Ignition 配置文件。

在初次启动过程中，机器需要 IP 地址配置，该配置通过 DHCP 服务器或静态设置，提供所需的引导选项。建立网络连接后，机器会从 HTTP 或 HTTPS 服务器下载 Ignition 配置文件。然后，Ignition 配置文件用于设置每台机器的确切状态。Machine Config Operator 在安装后完成对机器的更多更改，如应用新证书或密钥。

建议使用 DHCP 服务器对集群机器进行长期管理。确保 DHCP 服务器已配置为向集群机器提供持久

的 IP 地址、DNS 服务器信息和主机名。



注意

如果用户置备的基础架构没有 DHCP 服务，您可以在 RHCOS 安装时向节点提供 IP 网络配置和 DNS 服务器地址。如果要从 ISO 镜像安装，这些参数可作为引导参数传递。如需有关静态 IP 置备和高级网络选项的更多信息，请参阅 *安装 RHCOS 并启动 OpenShift Container Platform bootstrap 过程* 部分。

Kubernetes API 服务器必须能够解析集群机器的节点名称。如果 API 服务器和 worker 节点位于不同的区域中，您可以配置默认 DNS 搜索区域，以允许 API 服务器解析节点名称。另一种支持的方法是始终通过节点对象和所有 DNS 请求中的完全限定域名引用主机。

24.1.3.4.1. 通过 DHCP 设置集群节点主机名

在 Red Hat Enterprise Linux CoreOS(RHCOS)机器上，主机名是通过 NetworkManager 设置的。默认情况下，机器通过 DHCP 获取其主机名。如果主机名不是由 DHCP 提供，请通过内核参数或者其它方法进行静态设置，请通过反向 DNS 查找获取。反向 DNS 查找在网络初始化后进行，可能需要一些时间来解决。其他系统服务可以在此之前启动，并将主机名检测为 localhost 或类似的内容。您可以使用 DHCP 为每个集群节点提供主机名来避免这种情况。

另外，通过 DHCP 设置主机名可以绕过实施 DNS split-horizon 的环境中的手动 DNS 记录名称配置错误。

24.1.3.4.2. 网络连接要求

您必须配置机器之间的网络连接，以允许 OpenShift Container Platform 集群组件进行通信。每台机器都必须能够解析集群中所有其他机器的主机名。

本节详细介绍了所需的端口。

重要

在连接的 OpenShift Container Platform 环境中，所有节点都需要访问互联网才能为平台容器拉取镜像，并向红帽提供遥测数据。

表 24.3. 用于全机器到所有机器通信的端口

协议	port	描述
ICMP	N/A	网络可访问性测试
TCP	**1936**	指标
	9000-9999	主机级别的服务，包括端口 9 **100-9101 上的节点导出器**，以及端口 **9099** 上的 Cluster Version Operator。
	10250-10259	Kubernetes 保留的默认端口
UDP	**4789**	VXLAN
	6081	Geneve
	9000-9999	主机级别的服务，包括端口 **9100**到**9101** 上的节点导出器。
	500	IPsec IKE 数据包
	4500	IPsec NAT-T 数据包
	123	UDP 端口 **123** 上的网络时间协议 (NTP) 如果配置了外部 NTP 时间服务器，需要打开 UDP 端口 **123**。
TCP/UDP	**30000-32767**	Kubernetes 节点端口
ESP	N/A	IPsec Encapsulating Security Payload(ESP)

表 24.4. 用于所有机器控制平面通信的端口

协议	port	描述
TCP	**6443**	Kubernetes API

表 24.5. control plane 机器用于 control plane 机器通信的端口

协议	port	描述
TCP	**2379-2380**	etcd 服务器和对等端口

用户置备的基础架构的 NTP 配置

OpenShift Container Platform 集群被配置为默认使用公共网络时间协议(NTP)服务器。如果要使用本地企业 NTP 服务器，或者集群部署在断开连接的网络中，您可以将集群配置为使用特定的时间服务

器。如需更多信息，*请参阅配置 chrony 时间服务* 的文档。

如果 DHCP 服务器提供 NTP 服务器信息，Red Hat Enterprise Linux CoreOS(RHCOS)机器上的 chrony 时间服务会读取信息，并可以把时钟与 NTP 服务器同步。

其他资源

- 配置 chrony 时间服务

24.1.3.5. 用户置备的 DNS 要求

在 OpenShift Container Platform 部署中，以下组件需要 DNS 名称解析：

- The Kubernetes API
- OpenShift Container Platform 应用程序通配符
- bootstrap、control plane 和计算机器

Kubernetes API、bootstrap 机器、control plane 机器和计算机器也需要反向 DNS 解析。

DNS A/AAAA 或 CNAME 记录用于名称解析，PTR 记录用于反向名称解析。反向记录很重要，因为 Red Hat Enterprise Linux CoreOS(RHCOS)使用反向记录为所有节点设置主机名，除非 DHCP 提供主机名。另外，反向记录用于生成 OpenShift Container Platform 需要操作的证书签名请求(CSR)。

注意

建议使用 DHCP 服务器为每个群集节点提供主机名。如需更多信息，*请参阅用户置备的基础架构部分的 DHCP 建议*。

用户置备的 OpenShift Container Platform 集群需要以下 DNS 记录，这些记录必须在安装前就位。在每个记录中，<cluster_name> 是集群名称，<base_domain> 是您在 install-config.yaml 文件中指定的基域。完整的 DNS 记录采用以下形式：<component>.<cluster_name>.<base_domain>.。

表 24.6. 所需的 DNS 记录

组件	记录	描述
Kubernetes API	**api.<cluster_name>.<base_domain>.**	DNS A/AAAA 或 CNAME 记录，以及用于标识 API 负载均衡器的 DNS PTR 记录。这些记录必须由集群外的客户端和集群中的所有节点解析。
	api-int.<cluster_name>.<base_domain>.	DNS A/AAAA 或 CNAME 记录，以及用于内部标识 API 负载均衡器的 DNS PTR 记录。这些记录必须可以从集群中的所有节点解析。 **重要** API 服务器必须能够根据 Kubernetes 中记录的主机名解析 worker 节点。如果 API 服务器无法解析节点名称，则代理的 API 调用会失败，且您无法从 pod 检索日志。
Routes	***.apps.<cluster_name>.<base_domain>.**	通配符 DNS A/AAAA 或 CNAME 记录，指向应用程序入口负载均衡器。应用程序入口负载均衡器以运行 Ingress Controller Pod 的机器为目标。默认情况下，Ingress Controller Pod 在计算机器上运行。这些记录必须由集群外的客户端和集群中的所有节点解析。 例如，console **-openshift-console.apps.<cluster_name>.<base_domain>** 用作到 OpenShift Container Platform 控制台的通配符路由。
bootstrap 机器	**bootstrap.<cluster_name>.<base_domain>.**	DNS A/AAAA 或 CNAME 记录，以及用于标识 bootstrap 机器的 DNS PTR 记录。这些记录必须由集群中的节点解析。
control plane 机器	**<control_plane><n>.<cluster_name>.<base_domain>.**	DNS A/AAAA 或 CNAME 记录，以识别 control plane 节点的每台机器。这些记录必须由集群中的节点解析。
计算机器	**<compute><n>.<cluster_name>.<base_domain>.**	DNS A/AAAA 或 CNAME 记录，用于识别 worker 节点的每台机器。这些记录必须由集群中的节点解析。

注意

在 OpenShift Container Platform 4.4 及更新的版本中，您不需要在 DNS 配置中指定 etcd 主机和 SRV 记录。

提示

您可以使用 dig 命令验证名称和反向名称解析。如需了解详细的 *验证步骤，请参阅为用户置备的基础架构验证 DNS 解析* 一节。

24.1.3.5.1. 用户置备的集群的 DNS 配置示例

本节提供 A 和 PTR 记录配置示例，它们满足了在用户置备的基础架构上部署 OpenShift Container Platform 的 DNS 要求。样本不是为选择一个 DNS 解决方案提供建议。

在这个示例中，集群名称为 ocp4，基域是 example.com。

用户置备的集群的 DNS A 记录配置示例

以下示例是 BIND 区域文件，其中显示了用户置备的集群中名称解析的 A 记录示例。

例 24.1. DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
  2019070700 ; serial
  3H  ; refresh (3 hours)
  30M  ; retry (30 minutes)
  2W  ; expiry (2 weeks)
  1W )  ; minimum (1 week)
 IN NS ns1.example.com.
 IN MX 10 smtp.example.com.
;
;
ns1.example.com.  IN A 192.168.1.5
smtp.example.com.  IN A 192.168.1.5
;
helper.example.com.  IN A 192.168.1.5
helper.ocp4.example.com. IN A 192.168.1.5
;
api.ocp4.example.com.  IN A 192.168.1.5 1
api-int.ocp4.example.com. IN A 192.168.1.5 2
;
*.apps.ocp4.example.com. IN A 192.168.1.5 3
;
bootstrap.ocp4.example.com. IN A 192.168.1.96 4
;
control-plane0.ocp4.example.com. IN A 192.168.1.97 5
control-plane1.ocp4.example.com. IN A 192.168.1.98 6
control-plane2.ocp4.example.com. IN A 192.168.1.99 7
;
compute0.ocp4.example.com. IN A 192.168.1.11 8
```

```
compute1.ocp4.example.com. IN A 192.168.1.7 9
;
;EOF
```

1

为 Kubernetes API 提供名称解析。记录引用 API 负载均衡器的 IP 地址。

2

为 Kubernetes API 提供名称解析。记录引用 API 负载均衡器的 IP 地址，用于内部集群通信。

3

为通配符路由提供名称解析。记录引用应用程序入口负载均衡器的 IP 地址。应用程序入口负载均衡器以运行 Ingress Controller Pod 的机器为目标。默认情况下，Ingress Controller Pod 在计算机器上运行。

注意

在这个示例中，将相同的负载均衡器用于 Kubernetes API 和应用入口流量。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。

4

为 bootstrap 机器提供名称解析。

5 6 7

为 control plane 机器提供名称解析。

8 9

为计算机器提供名称解析。

用户置备的集群的 DNS PTR 记录配置示例

以下示例 BIND 区域文件显示了用户置备的集群中反向名称解析的 PTR 记录示例。

例 24.2. 反向记录的 DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
  2019070700 ; serial
  3H  ; refresh (3 hours)
  30M  ; retry (30 minutes)
  2W  ; expiry (2 weeks)
  1W )  ; minimum (1 week)
 IN NS ns1.example.com.
;
5.1.168.192.in-addr.arpa. IN PTR api.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. IN PTR api-int.ocp4.example.com. 2
;
96.1.168.192.in-addr.arpa. IN PTR bootstrap.ocp4.example.com. 3
;
97.1.168.192.in-addr.arpa. IN PTR control-plane0.ocp4.example.com. 4
98.1.168.192.in-addr.arpa. IN PTR control-plane1.ocp4.example.com. 5
99.1.168.192.in-addr.arpa. IN PTR control-plane2.ocp4.example.com. 6
;
11.1.168.192.in-addr.arpa. IN PTR compute0.ocp4.example.com. 7
7.1.168.192.in-addr.arpa. IN PTR compute1.ocp4.example.com. 8
;
;EOF
```

1 为 Kubernetes API 提供反向 DNS 解析。PTR 记录引用 API 负载均衡器的记录名称。

2 为 Kubernetes API 提供反向 DNS 解析。PTR 记录引用 API 负载均衡器的记录名称，用于内部集群通信。

3 为 bootstrap 机器提供反向 DNS 解析。

4 5 6 为 control plane 机器提供反向 DNS 解析。

7 8 为计算机器提供反向 DNS 解析。

注意

OpenShift Container Platform 应用程序通配符不需要 PTR 记录。

24.1.3.6. 用户置备的基础架构的负载均衡要求

在安装 OpenShift Container Platform 前，您必须置备 API 和应用程序入口负载均衡基础架构。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。

注意

如果要使用 Red Hat Enterprise Linux (RHEL) 实例部署 API 和应用程序入口负载均衡器，您必须单独购买 RHEL 订阅。

负载平衡基础架构必须满足以下要求：

1. API 负载均衡器 ：提供一个通用端点，供用户（包括人工和机器）与平台交互和配置。配置以下条件：

   - 仅第 4 层负载均衡.这可被称为 Raw TCP 或 SSL Passthrough 模式。
   - 无状态负载平衡算法。这些选项根据负载均衡器的实施而有所不同。

重要

不要为 API 负载均衡器配置会话持久性。为 Kubernetes API 服务器配置会话持久性可能会导致出现过量 OpenShift Container Platform 集群应用程序流量，以及过量的在集群中运行的 Kubernetes API。

在负载均衡器的前端和后端配置以下端口：

表 24.7. API 负载均衡器

port	后端机器（池成员）	internal	外部	描述
6443	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。您必须为 API 服务器健康检查探测配置 **/readyz** 端点。	X	X	Kubernetes API 服务器
22623	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。	X		机器配置服务器

注意

负载均衡器必须配置为，从 API 服务器关闭 /readyz 端点到从池中移除 API 服务器实例时最多需要 30 秒。在 /readyz 返回错误或健康后的时间范围内，端点必须被删除或添加。每 5 秒或 10 秒探测一次，有两个成功请求处于健康状态，三个成为不健康的请求是经过良好测试的值。

2. 应用程序入口负载均衡器 ：为应用程序流量从集群外部流提供入口点。OpenShift Container Platform 集群需要正确配置入口路由器。

配置以下条件：

- 仅第 4 层负载均衡.这可被称为 Raw TCP 或 SSL Passthrough 模式。
- 建议根据可用选项以及平台上托管的应用程序类型，使用基于连接的或基于会话的持久性。

提示

如果应用程序入口负载均衡器可以看到客户端的真实 IP 地址，启用基于 IP 的会话持久性可以提高使用端到端 TLS 加密的应用程序的性能。

在负载均衡器的前端和后端配置以下端口：

表 24.8. 应用程序入口负载均衡器

port	后端机器（池成员）	internal	外部	描述
443	默认情况下，运行 Ingress Controller Pod、计算或 worker 的机器。	X	X	HTTPS 流量
80	默认情况下，运行 Ingress Controller Pod、计算或 worker 的机器。	X	X	HTTP 流量

注意

如果要部署一个带有零计算节点的三节点集群，Ingress Controller Pod 在 control plane 节点上运行。在三节点集群部署中，您必须配置应用程序入口负载均衡器，将 HTTP 和 HTTPS 流量路由到 control plane 节点。

24.1.3.6.1. 用户置备的集群的负载均衡器配置示例

本节提供了一个满足用户置备集群的负载均衡要求的 API 和应用程序入口负载均衡器配置示例。示例是 HAProxy 负载均衡器的 /etc/haproxy/haproxy.cfg 配置。这个示例不是为选择一个负载平衡解决方案提供建议。

在这个示例中，将相同的负载均衡器用于 Kubernetes API 和应用入口流量。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。

注意

如果您使用 HAProxy 作为负载均衡器，并且 SELinux 设置为 enforcing，您必须通过运行 setsebool -P haproxy_connect_any=1 来确保 HAProxy 服务可以绑定到配置的 TCP 端口。

例 24.3. API 和应用程序入口负载均衡器配置示例

```
global
  log         127.0.0.1 local2
  pidfile     /var/run/haproxy.pid
  maxconn     4000
  daemon
defaults
  mode                    http
  log                     global
  option                  dontlognull
  option http-server-close
  option                  redispatch
  retries                 3
```

```
  timeout http-request    10s
  timeout queue           1m
  timeout connect         10s
  timeout client          1m
  timeout server          1m
  timeout http-keep-alive 10s
  timeout check           10s
  maxconn                 3000
listen api-server-6443 (1)
  bind *:6443
  mode tcp
  option  httpchk GET /readyz HTTP/1.0
  option  log-health-checks
  balance roundrobin
  server bootstrap bootstrap.ocp4.example.com:6443 verify none check check-ssl inter 10s
fall 2 rise 3 backup (2)
  server master0 master0.ocp4.example.com:6443 weight 1 verify none check check-ssl
inter 10s fall 2 rise 3
  server master1 master1.ocp4.example.com:6443 weight 1 verify none check check-ssl
inter 10s fall 2 rise 3
  server master2 master2.ocp4.example.com:6443 weight 1 verify none check check-ssl
inter 10s fall 2 rise 3
listen machine-config-server-22623 (3)
  bind *:22623
  mode tcp
  server bootstrap bootstrap.ocp4.example.com:22623 check inter 1s backup (4)
  server master0 master0.ocp4.example.com:22623 check inter 1s
  server master1 master1.ocp4.example.com:22623 check inter 1s
  server master2 master2.ocp4.example.com:22623 check inter 1s
listen ingress-router-443 (5)
  bind *:443
  mode tcp
  balance source
  server compute0 compute0.ocp4.example.com:443 check inter 1s
  server compute1 compute1.ocp4.example.com:443 check inter 1s
listen ingress-router-80 (6)
  bind *:80
  mode tcp
  balance source
  server compute0 compute0.ocp4.example.com:80 check inter 1s
  server compute1 compute1.ocp4.example.com:80 check inter 1s
```

(1)

端口 6443 处理 Kubernetes API 流量并指向 control plane 机器。

(2) (4)

bootstrap 条目必须在 OpenShift Container Platform 集群安装前就位，且必须在 bootstrap 过程完成后删除它们。

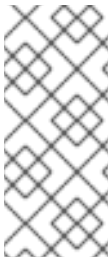
(3)

5

端口 443 处理 HTTPS 流量，并指向运行 Ingress Controller pod 的机器。默认情况下，Ingress Controller Pod 在计算机器上运行。

6

端口 80 处理 HTTP 流量，并指向运行 Ingress Controller pod 的机器。默认情况下，Ingress Controller Pod 在计算机器上运行。



注意

如果要部署一个带有零计算节点的三节点集群，Ingress Controller Pod 在 control plane 节点上运行。在三节点集群部署中，您必须配置应用程序入口负载均衡器，将 HTTP 和 HTTPS 流量路由到 control plane 节点。

提示

如果您使用 HAProxy 作为负载均衡器，您可以通过在 HAProxy 节点上运行 netstat -nltupe 来检查 haproxy 进程是否在侦听端口 6443、22623、443 和 80。

24.1.4. 准备用户置备的基础架构

在用户置备的基础架构上安装 OpenShift Container Platform 之前，您必须准备底层基础架构。

本节详细介绍了设置集群基础架构以准备 OpenShift Container Platform 安装所需的高级别步骤。这包括为您的集群节点配置 IP 网络和网络连接，通过防火墙启用所需的端口，以及设置所需的 DNS 和负载均衡基础架构。

准备后，集群基础架构必须满足 *带有用户置备的基础架构部分的集群要求*。

先决条件

- 您已参阅 OpenShift Container Platform 4.x Tested Integrations 页面。

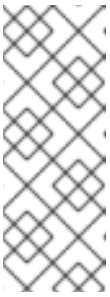
- 您已查看了 *具有用户置备基础架构的集群要求部分中详述的基础架构* 要求。

流程

1. 如果您使用 DHCP 向集群节点提供 IP 网络配置，请配置 DHCP 服务。

   a. 将节点的持久 IP 地址添加到您的 DHCP 服务器配置。在您的配置中，将相关网络接口的 MAC 地址与每个节点的预期 IP 地址匹配。

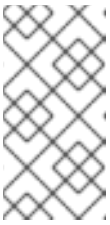
   b. 当您使用 DHCP 为集群机器配置 IP 寻址时，机器还通过 DHCP 获取 DNS 服务器信息。定义集群节点通过 DHCP 服务器配置使用的持久性 DNS 服务器地址。

   注意

   如果没有使用 DHCP 服务，则必须在 RHCOS 安装时为节点提供 IP 网络配置和 DNS 服务器地址。如果要从 ISO 镜像安装，这些参数可作为引导参数传递。如需有关静态 IP 置备和高级网络选项的更多信息，请参阅 *安装 RHCOS 并启动 OpenShift Container Platform bootstrap 过程* 部分。

   c. 在 DHCP 服务器配置中定义集群节点的主机名。有关 *主机名注意事项的详情，请参阅通过 DHCP 设置集群节点* 主机名部分。

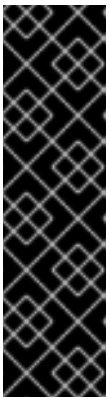
   

   注意

   如果没有使用 DHCP 服务，集群节点可以通过反向 DNS 查找来获取其主机名。

2. 确保您的网络基础架构提供集群组件之间所需的网络连接。有关 *要求的详情，请参阅用户置备的基础架构* 的网络要求部分。

3. 将防火墙配置为启用 OpenShift Container Platform 集群组件进行通信所需的端口。*如需有关所需端口的详细信息，请参阅用户置备的基础架构* 部分的网络要求。



重要

默认情况下，OpenShift Container Platform 集群可以访问端口 1936，因为每个 control plane 节点都需要访问此端口。

避免使用 Ingress 负载均衡器公开此端口，因为这样做可能会导致公开敏感信息，如统计信息和指标（与 Ingress Controller 相关的统计信息和指标）。

4. 为集群设置所需的 DNS 基础架构。

   a. 为 Kubernetes API、应用程序通配符、bootstrap 机器、control plane 机器和计算机器配置 DNS 名称解析。

   b. 为 Kubernetes API、bootstrap 机器、control plane 机器和计算机器配置反向 DNS 解析。

   如需有关 *OpenShift Container Platform DNS 要求的更多信息，请参阅用户置备* DNS 要求部分。

5. 验证您的 DNS 配置。

   a. 从安装节点，针对 Kubernetes API 的记录名称、通配符路由和集群节点运行 DNS 查找。验证响应中的 IP 地址是否与正确的组件对应。

   b. 从安装节点，针对负载均衡器和集群节点的 IP 地址运行反向 DNS 查找。验证响应中的记录名称是否与正确的组件对应。

   有关详细的 *DNS 验证步骤，请参阅用户置备的基础架构*验证 DNS 解析部分。

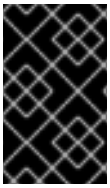
6. 置备所需的 API 和应用程序入口负载平衡基础架构。有关 *要求的更多信息，请参阅用户置备的基础架构的负载平衡* 要求部分。

注意

某些负载平衡解决方案要求在初始化负载平衡之前，对群集节点进行 DNS 名称解析。

24.1.5. 验证用户置备的基础架构的 DNS 解析

您可以在在用户置备的基础架构上安装 OpenShift Container Platform 前验证 DNS 配置。



重要

本节中详述的验证步骤必须在安装集群前成功。

先决条件

- 已为您的用户置备的基础架构配置了所需的 DNS 记录。

流程

1. 从安装节点，针对 Kubernetes API 的记录名称、通配符路由和集群节点运行 DNS 查找。验证响应中包含的 IP 地址是否与正确的组件对应。

   a. 对 Kubernetes API 记录名称执行查询。检查结果是否指向 API 负载均衡器的 IP 地址：

   ```
   $ dig +noall +answer @<nameserver_ip> api.<cluster_name>.<base_domain> 1
   ```

   1 将 <nameserver_ip> 替换为 nameserver 的 IP 地址，<cluster_name> 替换为您的集群名称，<base_domain> 替换为您的基本域名。

   输出示例

   ```
   api.ocp4.example.com.  604800 IN A 192.168.1.5
   ```

b.

对 Kubernetes 内部 API 记录名称执行查询。检查结果是否指向 API 负载均衡器的 IP 地址：

```
$ dig +noall +answer @<nameserver_ip> api-int.<cluster_name>.<base_domain>
```

输出示例

```
api-int.ocp4.example.com.  604800 IN A 192.168.1.5
```

c.

测试 *.apps.<cluster_name>.<base_domain> DNS 通配符查找示例。所有应用程序通配符查询都必须解析为应用程序入口负载均衡器的 IP 地址：

```
$ dig +noall +answer @<nameserver_ip> random.apps.<cluster_name>.
<base_domain>
```

输出示例

```
random.apps.ocp4.example.com.  604800 IN A 192.168.1.5
```

注意

在示例中，将相同的负载均衡器用于 Kubernetes API 和应用程序入口流量。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。

您可以使用另一个通配符值替换 random。例如，您可以查询到 OpenShift Container Platform 控制台的路由：

```
$ dig +noall +answer @<nameserver_ip> console-openshift-console.apps.
<cluster_name>.<base_domain>
```

输出示例

```
console-openshift-console.apps.ocp4.example.com. 604800 IN A 192.168.1.5
```

d. 针对 bootstrap DNS 记录名称运行查询。检查结果是否指向 bootstrap 节点的 IP 地址：

```
$ dig +noall +answer @<nameserver_ip> bootstrap.<cluster_name>.
<base_domain>
```

输出示例

```
bootstrap.ocp4.example.com.  604800 IN A 192.168.1.96
```

e. 使用此方法对 control plane 和计算节点的 DNS 记录名称执行查找。检查结果是否与每个节点的 IP 地址对应。

2. 从安装节点，针对负载均衡器和集群节点的 IP 地址运行反向 DNS 查找。验证响应中包含的记录名称是否与正确的组件对应。

a. 对 API 负载均衡器的 IP 地址执行反向查找。检查响应是否包含 Kubernetes API 和 Kubernetes 内部 API 的记录名称：

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.5
```

输出示例

```
5.1.168.192.in-addr.arpa. 604800 IN PTR api-int.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. 604800 IN PTR api.ocp4.example.com. 2
```

1

为 Kubernetes 内部 API 提供记录名称。

2

为 Kubernetes API 提供记录名称。

注意

OpenShift Container Platform 应用程序通配符不需要 PTR 记录。针对应用程序入口负载均衡器的 IP 地址解析反向 DNS 解析不需要验证步骤。

b. 对 bootstrap 节点的 IP 地址执行反向查找。检查结果是否指向 bootstrap 节点的 DNS 记录名称：

```
$ dig +noall +answer @<nameserver_ip> -x 192.168.1.96
```

输出示例

```
96.1.168.192.in-addr.arpa. 604800 IN PTR bootstrap.ocp4.example.com.
```

c. 使用此方法对 control plane 和计算节点的 IP 地址执行反向查找。检查结果是否与每个节点的 DNS 记录名称对应。

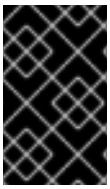
24.1.6. 为集群节点 SSH 访问生成密钥对

在 OpenShift Container Platform 安装过程中，您可以为安装程序提供 SSH 公钥。密钥通过它们的 Ignition 配置文件传递给 Red Hat Enterprise Linux CoreOS(RHCOS)节点，用于验证对节点的 SSH 访问。密钥添加到每个节点上 core 用户的 ~/.ssh/authorized_keys 列表中，这将启用免密码身份验证。

将密钥传递给节点后，您可以使用密钥对作为用户 核心 通过 SSH 连接到 RHCOS 节点。若要通过

SSH 访问节点，必须由 SSH 为您的本地用户管理私钥身份。

如果要通过 SSH 连接到集群节点来执行安装调试或灾难恢复，则必须在安装过程中提供 SSH 公钥。./openshift-install gather 命令还需要在集群节点上设置 SSH 公钥。



重要

不要在生产环境中跳过这个过程，在生产环境中需要灾难恢复和调试。

注意

您必须使用本地密钥，而不是使用特定平台方法配置 的密钥，如 AWS 密钥对。

流程

1. 如果您在本地计算机上没有可用于在集群节点上进行身份验证的现有 SSH 密钥对，请创建一个。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ ssh-keygen -t ed25519 -N '' -f <path>/<file_name> 1
```

1 指定新 SSH 密钥的路径和文件名，如 ~/.ssh/id_ed25519。如果您已有密钥对，请确保您的公钥位于 ~/.ssh 目录中。

注意

如果您计划在 x86_64、ppc64le 和 s390x 架构上安装使用 RHEL 加密库（这些加密库已提交给 NIST 用于 FIPS 140-2/140-3 验证）的 OpenShift Container Platform 集群，则不要创建使用 ed25519 算法的密钥。相反，创建一个使用 rsa 或 ecdsa 算法的密钥。

2. 查看公共 SSH 密钥：

```
$ cat <path>/<file_name>.pub
```

例如，运行以下命令来查看 ~/.ssh/id_ed25519.pub 公钥：

```
$ cat ~/.ssh/id_ed25519.pub
```

3. 将 SSH 私钥身份添加到本地用户的 SSH 代理（如果尚未添加）。在集群节点上，或者要使用 ./openshift-install gather 命令，需要对该密钥进行 SSH 代理管理，才能在集群节点上进行免密码 SSH 身份验证。

注意

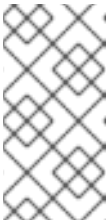
在某些发行版中，自动管理默认 SSH 私钥身份，如 ~/.ssh/id_rsa 和 ~/.ssh/id_dsa。

a. 如果 ssh-agent 进程尚未为您的本地用户运行，请将其作为后台任务启动：

```
$ eval "$(ssh-agent -s)"
```

输出示例

```
Agent pid 31874
```



注意

如果集群处于 FIPS 模式，则只使用 FIPS 兼容算法来生成 SSH 密钥。密钥必须是 RSA 或 ECDSA。

4. 将 SSH 私钥添加到 ssh-agent ：

```
$ ssh-add <path>/<file_name> 1
```

1 指定 SSH 私钥的路径和文件名，如 ~/.ssh/id_ed25519.pub

输出示例

```
Identity added: /home/<you>/<path>/<file_name> (<computer_name>)
```

后续步骤

- 安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。如果在您置备的基础架构上安装集群，则必须为安装程序提供密钥。

24.1.7. 获取安装程序

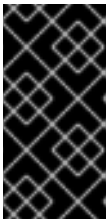
在安装 OpenShift Container Platform 前，将安装文件下载到您用于安装的主机上。

先决条件

- 您有一台运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB。

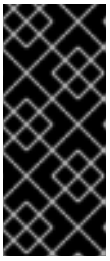
流程

1. 访问 OpenShift Cluster Manager 站点的 Infrastructure Provider 页面。如果您有红帽帐户，请使用您的凭证登录。如果没有，请创建一个帐户。
2. 选择您的基础架构供应商。
3. 进入到安装类型的页面，下载与您的主机操作系统和架构对应的安装程序，并将该文件放在您要存储安装配置文件的目录中。



重要

安装程序会在用来安装集群的计算机上创建几个文件。在完成集群安装后，您必须保留安装程序和安装程序所创建的文件。这两个文件都需要删除集群。



重要

删除安装程序创建的文件不会删除您的集群，即使集群在安装过程中失败也是如此。要删除集群，请为特定云供应商完成 OpenShift Container Platform 卸载流程。

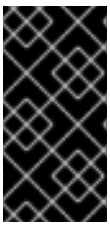
4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar -xvf openshift-install-linux.tar.gz
```

5. 从 Red Hat OpenShift Cluster Manager 下载安装 pull secret。此 pull secret 允许您与所含授权机构提供的服务进行身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

24.1.8. 安装 OpenShift CLI

您可以安装 OpenShift CLI(oc)来使用命令行界面与 OpenShift Container Platform 进行交互。您可以在 Linux、Windows 或 macOS 上安装 oc。



重要

如果安装了旧版本的 oc，则可能无法使用 OpenShift Container Platform 4.16 中的所有命令。下载并安装新版本的 oc。

在 Linux 上安装 OpenShift CLI

您可以按照以下流程在 Linux 上安装 OpenShift CLI(oc)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 产品变体 下拉列表中选择架构。
3. 从 版本 下拉列表中选择适当的版本。

4. 点 OpenShift v4.16 Linux Client 条目旁的 Download Now 来保存文件。

5. 解包存档：

```
$ tar xvf <file>
```

6. 将 oc 二进制文件放到 PATH 中的目录中。

   要查看您的 PATH，请执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 oc 命令：

```
$ oc <command>
```

在 Windows 上安装 OpenShift CLI

您可以按照以下流程在 Windows 上安装 OpenShift CLI(oc)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。

2. 从 版本 下拉列表中选择适当的版本。

3. 点 OpenShift v4.16 Windows Client 条目旁的 Download Now 来保存文件。

4. 使用 ZIP 程序解压存档。

5. 将 oc 二进制文件移到 PATH 中的目录中。

要查看您的 PATH，请打开命令提示并执行以下命令：

```
C:\> path
```

验证

- 安装 OpenShift CLI 后，可以使用 oc 命令：

```
C:\> oc <command>
```

在 macOS 上安装 OpenShift CLI

您可以按照以下流程在 macOS 上安装 OpenShift CLI(oc)二进制文件。

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 从 版本 下拉列表中选择适当的版本。
3. 点 OpenShift v4.16 macOS Client 条目旁的 Download Now 来保存文件。

注意

对于 macOS arm64，请选择 OpenShift v4.16 macOS arm64 Client 条目。

4. 解包和解压存档。
5. 将 oc 二进制文件移到 PATH 的目录中。

要查看您的 PATH，请打开终端并执行以下命令：

```
$ echo $PATH
```

验证

- 安装 OpenShift CLI 后，可以使用 oc 命令：

```
$ oc <command>
```

24.1.9. 手动创建安装配置文件

安装集群要求您手动创建安装配置文件。

先决条件

- 您在本地机器上有一个 SSH 公钥来提供给安装程序。该密钥将用于在集群节点上进行 SSH 身份验证，以进行调试和灾难恢复。
- 已获取 OpenShift Container Platform 安装程序和集群的 pull secret。

流程

1. 创建一个安装目录来存储所需的安装资产：

```
$ mkdir <installation_directory>
```

重要

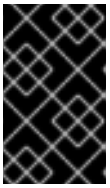
您必须创建一个目录。有些安装资产，如 bootstrap X.509 证书的过期间隔较短，因此不得重复使用安装目录。如果要重复使用另一个集群安装中的单个文件，您可以将它们复制到您的目录中。但是，安装资产的文件名可能会在发行版本间有所变化。从以前的 OpenShift Container Platform 版本中复制安装文件时请小心。

2. 自定义提供的 install-config.yaml 文件模板示例，并将其保存在 <installation_directory> 中。

注意

此配置文件必须命名为 install-config.yaml。

3. 备份 install-config.yaml 文件，以便您可以使用它安装多个集群。



重要

install-config.yaml 文件会在安装过程的下一步中使用。现在必须备份它。

24.1.9.1. 其他平台的 install-config.yaml 文件示例

您可以自定义 install-config.yaml 文件，以指定有关 OpenShift Container Platform 集群平台的更多详情，或修改所需参数的值。

```
apiVersion: v1
baseDomain: example.com (1)
compute: (2)
- hyperthreading: Enabled (3)
  name: worker
  replicas: 0 (4)
controlPlane: (5)
  hyperthreading: Enabled (6)
  name: master
  replicas: 3 (7)
metadata:
  name: test (8)
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14 (9)
    hostPrefix: 23 (10)
  networkType: OVNKubernetes (11)
  serviceNetwork: (12)
  - 172.30.0.0/16
platform:
  none: {} (13)
fips: false (14)
pullSecret: '{"auths": ...}' (15)
sshKey: 'ssh-ed25519 AAAA...' (16)
```

(1) 集群的基域。所有 DNS 记录都必须是这个基域的子域，并包含集群名称。

(2) (5) controlPlane 部分是一个单个映射，但 compute 部分是一系列映射。为满足不同数据结构的要求，compute 部分的第一行必须以连字符 - 开头，controlPlane 部分 的第一行则不以连字符开头。仅使用一个 control plane 池。

3 6

指定要启用或禁用并发多线程(SMT)还是超线程。默认情况下，启用 SMT 可提高机器中内核的性能。您可以通过将 参数值设置为 Disabled 来禁用它。如果禁用 SMT，则必须在所有集群机器中禁用它；这包括 control plane 和计算机器。

注意

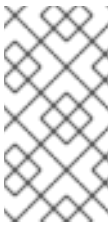
默认启用并发多线程(SMT)。如果您的 BIOS 设置中没有启用 SMT，超线程 参数无效。

重要

如果您禁用 超线程，无论是在 BIOS 中，还是在 install-config.yaml 文件中，请确保您的容量规划考虑机器性能显著降低的情况。

4

在用户置备的基础架构上安装 OpenShift Container Platform 时，必须将这个值设置为 0。在安装程序置备的安装中，参数控制集群为您创建和管理的计算机器数量。在用户置备的安装中，您必须在完成集群安装前手动部署计算机器。



注意

如果要安装一个三节点集群，在安装 Red Hat Enterprise Linux CoreOS(RHCOS)机器时不要部署任何计算机器。

7

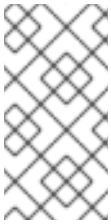
您添加到集群的 control plane 机器数量。由于集群使用这些值作为集群中的 etcd 端点数量，所以该值必须与您部署的 control plane 机器数量匹配。

8

您在 DNS 记录中指定的集群名称。

9

从中分配 Pod IP 地址的 IP 地址块。此块不得与现有物理网络重叠。这些 IP 地址用于 pod 网络。如果需要从外部网络访问 pod，您必须配置负载均衡器和路由器来管理流量。



注意

类 E CIDR 范围被保留以供以后使用。要使用 Class E CIDR 范围，您必须确保您的网络环境接受 Class E CIDR 范围内的 IP 地址。

10

分配给每个节点的子网前缀长度。例如，如果 hostPrefix 设为 23，则每个节点从 given cidr 中分配 a /23 子网，这样就能有 510（2^(32 - 23)- 2）个 pod IP 地址。如果需要从外部网络访问节点，请配置负载均衡器和路由器来管理流量。

11

要安装的集群网络插件。默认值 OVNKubernetes 是唯一支持的值。

12

用于服务 IP 地址的 IP 地址池。您只能输入一个 IP 地址池。此块不得与现有物理网络重叠。如果您需要从外部网络访问服务，请配置负载均衡器和路由器来管理流量。

13

您必须将平台设置为 none。您无法为您的平台提供额外的平台配置变量。

重要

使用平台类型 none 安装的集群无法使用一些功能，如使用 Machine API 管理计算机器。即使附加到集群的计算机器安装在通常支持该功能的平台上，也会应用这个限制。在安装后无法更改此参数。

14

是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS(RHCOS)机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。

重要

要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 Red Hat Enterprise Linux (RHEL) 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅在 FIPS 模式中安装该系统。

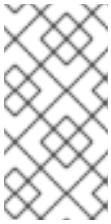
当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS)时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。

15

Red Hat OpenShift Cluster Manager 的 pull secret。此 pull secret 允许您与所含授权机构提供的服务进行身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

16

Red Hat Enterprise Linux CoreOS(RHCOS)中 core 用户的 SSH 公钥。



注意

对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 ssh-agent 进程使用的 SSH 密钥。

24.1.9.2. 在安装过程中配置集群范围的代理

生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 install-config.yaml 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

先决条件

- 您有一个现有的 install-config.yaml 文件。
- 您检查了集群需要访问的站点，并确定它们中的任何站点是否需要绕过代理。默认情况下，所有集群出口流量都经过代理，包括对托管云供应商 API 的调用。如果需要，您将在 Proxy 对象的 spec.noProxy 字段中添加站点来绕过代理。

注意

Proxy 对象 status.noProxy 字段使用安装配置中的 networking.machineNetwork[].cidr、networking.clusterNetwork[].cidr 和 networking.serviceNetwork[] 字段的值填充。

对于在 Amazon Web Services(AWS)、Google Cloud Platform(GCP)、Microsoft Azure 和 Red Hat OpenStack Platform(RHOSP)上安装，Proxy 对象 status.noProxy 字段也会使用实例元数据端点填充(169.254.169.254)。

流程

1. 编辑 install-config.yaml 文件并添加代理设置。例如：

```
apiVersion: v1
baseDomain: my.domain.com
proxy:
  httpProxy: http://<username>:<pswd>@<ip>:<port> 1
  httpsProxy: https://<username>:<pswd>@<ip>:<port> 2
  noProxy: example.com 3
additionalTrustBundle: | 4
    -----BEGIN CERTIFICATE-----
    <MY_TRUSTED_CA_CERT>
    -----END CERTIFICATE-----
additionalTrustBundlePolicy: <policy_to_add_additionalTrustBundle> 5
```

1. 用于创建集群外 HTTP 连接的代理 URL。URL 方案必须是 http。

2. 用于创建集群外 HTTPS 连接的代理 URL。

3. 要从代理中排除的目标域名、IP 地址或其他网络 CIDR 的逗号分隔列表。在域前面加上 . 以仅匹配子域。例如，.y.com 匹配 x.y.com，但不匹配 y.com。使用 * 绕过所有目的地的代理。

4. 如果提供，安装程序会在 openshift-config 命名空间中生成名为 user-ca-bundle 的配置映射，其包含代理 HTTPS 连接所需的一个或多个额外 CA 证书。然后，Cluster Network Operator 会创建 trusted-ca-bundle 配置映射，将这些内容与 Red Hat Enterprise Linux CoreOS（RHCOS）信任捆绑包合并， Proxy 对象的 trustedCA 字段中

也会引用此配置映射。additionalTrustBundle 字段是必需的，除非代理的身份证书由来自 RHCOS 信任捆绑包的颁发机构签名。

5 可选：决定 Proxy 对象的配置以引用 trustedCA 字段中 user-ca-bundle 配置映射的策略。允许的值是 Proxyonly 和 Always。仅在配置了 http/https 代理时，使用 Proxyonly 引用 user-ca-bundle 配置映射。使用 Always 始终引用 user-ca-bundle 配置映射。默认值为 Proxyonly。

注意

安装程序不支持代理的 readinessEndpoints 字段。

注意

如果安装程序超时，重启并使用安装程序的 wait-for 命令完成部署。例如：

```
$ ./openshift-install wait-for install-complete --log-level debug
```

2. 保存该文件并在安装 OpenShift Container Platform 时引用。

安装程序会创建一个名为 cluster 的集群范围代理，该代理 使用 提供的 install-config.yaml 文件中的代理设置。如果没有提供代理设置，仍然会创建一个 cluster Proxy 对象，但它会有一个空 spec。

注意

只支持名为 cluster 的 Proxy 对象，且无法创建额外的代理。

24.1.9.3. 配置三节点集群

另外，您可以在只由三台 control plane 机器组成的裸机集群中部署零台计算机器。这为集群管理员和开发人员提供了更小、效率更高的集群，用于测试、开发和生产。

在三节点 OpenShift Container Platform 环境中，三台 control plane 机器可以调度，这意味着应用程序工作负载被调度到它们上运行。

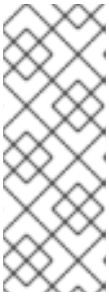
先决条件

- 您有一个现有的 install-config.yaml 文件。

流程

- 确保 install-config.yaml 文件中的计算副本数量设置为 0，如以下 计算 小节所示：

```
compute:
- name: worker
  platform: {}
  replicas: 0
```



注意

在用户置备的基础架构上安装 OpenShift Container Platform 时，无论您要部署的计算机器数量有多少，您必须将计算机器的 replicas 参数值设置为 0。在安装程序置备的安装中，参数控制集群为您创建和管理的计算机器数量。这不适用于手动部署计算机器的用户置备安装。

对于三节点集群安装，请按照以下步骤执行：

- 如果要部署一个带有零计算节点的三节点集群，Ingress Controller Pod 在 control plane 节点上运行。在三节点集群部署中，您必须配置应用程序入口负载均衡器，将 HTTP 和 HTTPS 流量路由到 control plane 节点。如需更多信息，*请参阅用户置备的基础架构的负载平衡要求* 部分。

- 在以下步骤中创建 Kubernetes 清单文件时，请确保 <installation_directory>/manifests/cluster-scheduler-02-config.yml 文件中的 mastersSchedulable 参数被设置为 true。这可让应用程序工作负载在 control plane 节点上运行。

- 在创建 Red Hat Enterprise Linux CoreOS(RHCOS)机器时，不要部署任何计算节点。

24.1.10. 创建 Kubernetes 清单和 Ignition 配置文件

由于您必须修改一些集群定义文件并手动启动集群机器，因此您必须生成 Kubernetes 清单和 Ignition 配置文件来配置机器。

安装配置文件转换为 Kubernetes 清单。清单嵌套到 Ignition 配置文件中，稍后用于配置集群机器。

重要

- OpenShift Container Platform 安装程序生成的 Ignition 配置文件包含 24 小时后过期的证书，然后在该时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 node-bootstrapper 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。

- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

先决条件

- 已获得 OpenShift Container Platform 安装程序。
- 已创建 install-config.yaml 安装配置文件。

流程

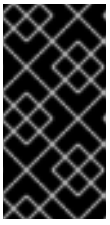
1. 进入包含 OpenShift Container Platform 安装程序的目录，并为集群生成 Kubernetes 清单：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 对于 <installation_directory>，请指定包含您创建的 install-config.yaml 文件的安装目录。

警告

如果您要安装一个三节点集群，请跳过以下步骤，以便可以调度 control plane 节点。



重要

当您将 control plane 节点从默认的不可调度配置为可以调度时，需要额外的订阅。这是因为 control plane 节点变为计算节点。

2. 检查 <installation_directory>/manifests/cluster-scheduler-02-config.yml Kubernetes 清单文件中的 mastersSchedulable 参数是否已设置为 false。此设置可防止在 control plane 机器上调度 pod：

   a. 打开 <installation_directory>/manifests/cluster-scheduler-02-config.yml 文件。

   b. 找到 mastersSchedulable 参数，并确保它被设置为 false。

   c. 保存并退出 文件。

3. 要创建 Ignition 配置文件，请从包含安装程序的目录运行以下命令：

```
$ ./openshift-install create ignition-configs --dir <installation_directory> 1
```

1 对于 <installation_directory>，请指定相同的安装目录。

为安装目录中的 bootstrap、control plane 和计算节点创建 Ignition 配置文件。kubeadmin-password 和 kubeconfig 文件在 ./<installation_directory>/auth 目录中创建：

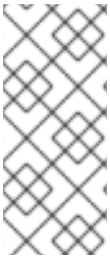
```
.
```

```
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

24.1.11. 安装 RHCOS 并启动 OpenShift Container Platform bootstrap 过程

要在您置备的裸机基础架构上安装 OpenShift Container Platform，您必须在机器上安装 Red Hat Enterprise Linux CoreOS(RHCOS)。安装 RHCOS 时，您必须为您要安装的机器类型提供 OpenShift Container Platform 安装程序生成的 Ignition 配置文件。如果您配置了适当的网络、DNS 和负载均衡基础架构，OpenShift Container Platform bootstrap 过程会在 RHCOS 机器重启后自动启动。

要在机器上安装 RHCOS，请按照以下步骤使用 ISO 镜像或网络 PXE 引导。



注意

本安装文档中包括的计算节点部署步骤特定于 RHCOS。如果您选择部署基于 RHEL 的计算节点，您需要负责所有操作系统生命周期管理和维护，包括执行系统更新、应用补丁和完成所有其他必要的任务。仅支持 RHEL 8 计算机器。

您可以使用以下方法在 ISO 和 PXE 安装过程中配置 RHCOS：

- 内核参数：您可以使用内核参数来提供特定于安装的信息。例如，您可以指定上传到 HTTP 服务器的 RHCOS 安装文件的位置以及您要安装的节点类型的 Ignition 配置文件的位置。对于 PXE 安装，您可以使用 APPEND 参数将参数传递给 live 安装程序的内核。对于 ISO 安装，您可以中断实时安装引导过程来添加内核参数。在这两个安装情形中，您可以使用特殊的 coreos.inst.* 参数来指示实时安装程序，以及标准安装引导参数来打开或关闭标准内核服务。

- Ignition 配置：OpenShift Container Platform Ignition 配置文件(*.ign)特定于您要安装的节点类型。您可以在 RHCOS 安装过程中传递 bootstrap、control plane 或计算节点 Ignition 配置文件的位置，以便在首次启动时生效。特殊情况下，您可以创建单独的、有限的 Ignition 配置以传递给 live 系统。该 Ignition 配置可以执行特定的任务，如在安装完成后向置备系统报告成功。此特殊的 Ignition 配置由 coreos-installer 使用，以便在首次引导安装的系统时应用。不要直接为实时 ISO 提供标准 control plane 和计算节点 Ignition 配置。

- coreos-installer：您可以将 live ISO 安装程序引导到 shell 提示符，这可让您在第一次引导前以多种方式准备持久性系统。特别是，您可以运行 coreos-installer 命令来识别要包含的各种

工件、使用磁盘分区和设置网络。在某些情况下，您可以配置 live 系统上的功能并将其复制到安装的系统中。

使用 ISO 安装还是 PXE 安装取决于您的情况。PXE 安装需要可用的 DHCP 服务并进行更多准备，但可以使安装过程更加自动化。ISO 安装是一个更手动的过程，如果您设置的机器数超过几台，则可能不方便。

注意

从 OpenShift Container Platform 4.6 开始，RHCOS ISO 和其他安装工件支持在带有 4K 扇区的磁盘上安装。

24.1.11.1. 使用 ISO 镜像安装 RHCOS

您可以使用 ISO 镜像在机器上安装 RHCOS。

先决条件

- 已为集群创建 Ignition 配置文件。
- 您已配置了适当的网络、DNS 和负载平衡基础架构。
- 您有一个可以从计算机以及您创建的机器访问的 HTTP 服务器。
- 您已参阅 *Advanced RHCOS 安装配置* 部分来了解配置功能的不同方法，如网络和磁盘分区。

流程

1. 获取每个 Ignition 配置文件的 SHA512 摘要。例如，您可以在运行 Linux 的系统上使用以下内容来获取 bootstrap.ign Ignition 配置文件的 SHA512 摘要：

```
$ sha512sum <installation_directory>/bootstrap.ign
```

后续步骤中会向 coreos-installer 提供摘要，以验证集群节点上 Ignition 配置文件的真实性。

2. 将安装程序创建的 bootstrap、control plane 和计算节点 Ignition 配置文件上传到 HTTP 服务器。注意这些文件的 URL。

重要

您可以在 Ignition 配置中添加或更改配置设置，然后将其保存到 HTTP 服务器。如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

3. 从安装主机上，验证 Ignition 配置文件是否在 URL 上可用。以下示例获取 bootstrap 节点的 Ignition 配置文件：

```
$ curl -k http://<HTTP_server>/bootstrap.ign 1
```

输出示例

```
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0{"ignition":
{"version":"3.2.0"},"passwd":{"users":[{"name":"core","sshAuthorizedKeys":["ssh-
rsa...
```

在命令中将 bootstrap.ign 替换为 master.ign 或 worker.ign，以验证 control plane 和计算节点的 Ignition 配置文件是否可用。

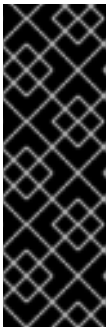
4. 虽然可以从 RHCOS 镜像镜像页面获取您选择的操作系统实例安装方法所需的 RHCOS 镜像，但推荐的方法是从 openshift-install 命令的输出获取 RHCOS 镜像的正确版本：

```
$ openshift-install coreos print-stream-json | grep '\.iso[^.]'
```

输出示例

```
"location": "<url>/art/storage/releases/rhcos-4.16-aarch64/<release>/aarch64/rhcos-
<release>-live.aarch64.iso",
"location": "<url>/art/storage/releases/rhcos-4.16-ppc64le/<release>/ppc64le/rhcos-
```

```
<release>-live.ppc64le.iso",
"location": "<url>/art/storage/releases/rhcos-4.16-s390x/<release>/s390x/rhcos-
<release>-live.s390x.iso",
"location": "<url>/art/storage/releases/rhcos-4.16/<release>/x86_64/rhcos-<release>-
live.x86_64.iso",
```



重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每个发行版本而改变。您必须下载最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。如果可用，请使用与 OpenShift Container Platform 版本匹配的镜像版本。这个过程只使用 ISO 镜像。此安装类型不支持 RHCOS qcow2 镜像。

ISO 文件名类似以下示例：

```
rhcos-<version>-live.<architecture>.iso
```

5. 使用 ISO 启动 RHCOS 安装。使用以下安装选项之一：

- 将 ISO 映像刻录到磁盘并直接启动。
- 使用 light-out 管理(LOM)接口使用 ISO 重定向。

6. 在不指定任何选项或中断实时引导序列的情况下引导 RHCOS ISO 镜像。等待安装程序在 RHCOS live 环境中引导进入 shell 提示符。

注意

可以中断 RHCOS 安装引导过程来添加内核参数。但是，在这个 ISO 过程中，您应该使用以下步骤中所述的 coreos-installer 命令，而不是添加内核参数。

7. 运行 coreos-installer 命令并指定满足您的安装要求的选项。您至少必须指定指向节点类型的 Ignition 配置文件的 URL，以及您要安装到的设备：

```
$ sudo coreos-installer install --ignition-url=http://<HTTP_server>/<node_type>.ign
<device> --ignition-hash=sha512-<digest> 1 2
```

1 1

您必须使用 sudo 运行 coreos-installer 命令，因为 core 用户没有执行安装所需的 root 权限。

2

当 Ignition 配置文件通过 HTTP URL 获取时，需要 --ignition-hash 选项来验证集群节点上 Ignition 配置文件的真实性。<digest> 是上一步中获取的 Ignition 配置文件 SHA512 摘要。

注意

如果要通过使用 TLS 的 HTTPS 服务器提供 Ignition 配置文件，您可以在运行 coreos-installer 前将内部证书颁发机构(CA)添加到系统信任存储中。

以下示例将引导节点安装初始化到 /dev/sda 设备。bootstrap 节点的 Ignition 配置文件从 IP 地址 192.168.1.2 的 HTTP Web 服务器获取：

```
$ sudo coreos-installer install --ignition-
url=http://192.168.1.2:80/installation_directory/bootstrap.ign /dev/sda --ignition-
hash=sha512-
a5a2d43879223273c9b60af66b44202a1d1248fc01cf156c46d4a79f552b6bad47bc8cc78dd
f0116e80c59d2ea9e32ba53bc807afbca581aa059311def2c3e3b
```

8. 在机器的控制台上监控 RHCOS 安装的进度。

重要

在开始安装 OpenShift Container Platform 之前，确保每个节点中安装成功。观察安装过程可以帮助确定可能会出现 RHCOS 安装问题的原因。

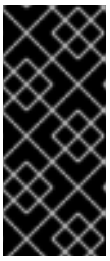
9. 安装 RHCOS 后，您必须重启系统。系统重启过程中，它会应用您指定的 Ignition 配置文件。

10. 检查控制台输出，以验证 Ignition 是否运行。

示例命令

```
Ignition: ran on 2022/03/14 14:48:33 UTC (this boot)
Ignition: user-provided config was applied
```

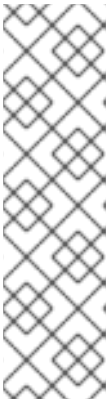
11. 继续为集群创建其他机器。



重要

此时您必须创建 bootstrap 和 control plane 机器。如果 control plane 机器不可调度，请在安装 OpenShift Container Platform 前至少创建两台计算机器。

如果存在所需的网络、DNS 和负载均衡器基础架构，OpenShift Container Platform bootstrap 过程会在 RHCOS 节点重启后自动启动。



注意

RHCOS 节点不包含 core 用户的默认密码。您可以使用可访问 SSH 私钥的用户的身份运行 ssh core@<node>.<cluster_name>.<base_domain > 来访问节点，该私钥与您在 install_config.yaml 文件中指定的公钥配对。运行 RHCOS 的 OpenShift Container Platform 4 集群节点不可变，它依赖于 Operator 来应用集群更改。不建议使用 SSH 访问集群节点。但是，当调查安装问题时，如果 OpenShift Container Platform API 不可用，或者 kubelet 在目标节点上无法正常工作，则调试或灾难恢复可能需要 SSH 访问。

24.1.11.2. 使用 PXE 或 iPXE 启动安装 RHCOS

您可以使用 PXE 或 iPXE 启动在机器上安装 RHCOS。

先决条件

- 已为集群创建 Ignition 配置文件。
- 您已配置了适当的网络、DNS 和负载平衡基础架构。
- 您已配置了合适的 PXE 或 iPXE 基础架构。
- 您有一个可以从计算机以及您创建的机器访问的 HTTP 服务器。
- 您已参阅 *Advanced RHCOS 安装配置* 部分来了解配置功能的不同方法，如网络和磁盘分区。

流程

1. 将安装程序创建的 bootstrap、control plane 和计算节点 Ignition 配置文件上传到 HTTP 服务器。注意这些文件的 URL。

重要

您可以在 Ignition 配置中添加或更改配置设置，然后将其保存到 HTTP 服务器。如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

2. 从安装主机上，验证 Ignition 配置文件是否在 URL 上可用。以下示例获取 bootstrap 节点的 Ignition 配置文件：

```
$ curl -k http://<HTTP_server>/bootstrap.ign 1
```

输出示例

```
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0{"ignition":
{"version":"3.2.0"},"passwd":{"users":[{"name":"core","sshAuthorizedKeys":["ssh-
rsa...
```

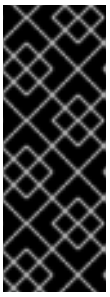
在命令中将 bootstrap.ign 替换为 master.ign 或 worker.ign，以验证 control plane 和计算节点的 Ignition 配置文件是否可用。

3. 虽然可以从 RHCOS image mirror 页面获取您选择的操作系统实例所需的 RHCOS kernel、initramfs 和 rootfs 文件，但推荐的方法是从 openshift-install 命令的输出中获取 RHCOS 文件的正确版本：

```
$ openshift-install coreos print-stream-json | grep -Eo '"https.*(kernel-
|initramfs.|rootfs.)\w+(\.img)?"'
```

输出示例

```
"<url>/art/storage/releases/rhcos-4.16-aarch64/<release>/aarch64/rhcos-<release>-live-
kernel-aarch64"
"<url>/art/storage/releases/rhcos-4.16-aarch64/<release>/aarch64/rhcos-<release>-live-
initramfs.aarch64.img"
"<url>/art/storage/releases/rhcos-4.16-aarch64/<release>/aarch64/rhcos-<release>-live-
rootfs.aarch64.img"
"<url>/art/storage/releases/rhcos-4.16-ppc64le/49.84.202110081256-0/ppc64le/rhcos-
<release>-live-kernel-ppc64le"
"<url>/art/storage/releases/rhcos-4.16-ppc64le/<release>/ppc64le/rhcos-<release>-live-
initramfs.ppc64le.img"
"<url>/art/storage/releases/rhcos-4.16-ppc64le/<release>/ppc64le/rhcos-<release>-live-
rootfs.ppc64le.img"
"<url>/art/storage/releases/rhcos-4.16-s390x/<release>/s390x/rhcos-<release>-live-
kernel-s390x"
"<url>/art/storage/releases/rhcos-4.16-s390x/<release>/s390x/rhcos-<release>-live-
initramfs.s390x.img"
"<url>/art/storage/releases/rhcos-4.16-s390x/<release>/s390x/rhcos-<release>-live-
rootfs.s390x.img"
"<url>/art/storage/releases/rhcos-4.16/<release>/x86_64/rhcos-<release>-live-kernel-
x86_64"
"<url>/art/storage/releases/rhcos-4.16/<release>/x86_64/rhcos-<release>-live-
initramfs.x86_64.img"
"<url>/art/storage/releases/rhcos-4.16/<release>/x86_64/rhcos-<release>-live-
rootfs.x86_64.img"
```



重要

RHCOS 工件可能不会随着 OpenShift Container Platform 的每个发行版本而改变。您必须下载最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。这个过程只使用下面描述的适当 kernel、initram fs 和 rootfs 工件。此安装类型不支持 RHCOS QCOW2 镜像。

文件名包含 OpenShift Container Platform 版本号。它们类似以下示例：

- **kernel:rhcos-<version>-live-kernel-<architecture>**
- **initramfs: rhcos-<version>-live-initramfs.<architecture>.img**
- **rootfs: rhcos-<version>-live-rootfs.<architecture>.img**

4. **将 rootfs、kernel 和 initramfs 文件上传到 HTTP 服务器。**

重要

如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

5. **配置网络引导基础架构，以便在安装 RHCOS 后机器从本地磁盘启动。**

6. **为 RHCOS 镜像配置 PXE 或 iPXE 安装并开始安装。**

为环境修改以下示例菜单条目之一，并验证能否正确访问镜像和 Ignition 文件：

- **对于 PXE(x86_64)：**

```
DEFAULT pxeboot
TIMEOUT 20
PROMPT 0
LABEL pxeboot
```

```
    KERNEL http://<HTTP_server>/rhcos-<version>-live-kernel-<architecture> 1
    APPEND initrd=http://<HTTP_server>/rhcos-<version>-live-initramfs.
<architecture>.img coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-
rootfs.<architecture>.img coreos.inst.install_dev=/dev/sda
coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign 2 3
```

1 1

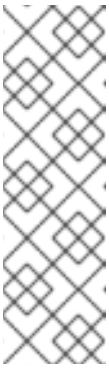
指定上传到 HTTP 服务器的 live kernel 文件位置。URL 必须是 HTTP、TFTP 或 FTP；不支持 HTTPS 和 NFS。

2

如果您使用多个 NIC，请在 ip 选项中指定一个接口。例如，要在名为 eno1 的 NIC 上使用 DHCP，请设置 ip=eno1:dhcp。

3

指定上传到 HTTP 服务器的 RHCOS 文件的位置。initrd 参数值是 initramfs 文件的位置，coreos.live.rootfs_url 参数值是 rootfs 文件的位置，coreos.inst.ignition_url 参数值则是 bootstrap Ignition 配置文件的位置。您还可以在 APPEND 行中添加更多内核参数来配置联网或其他引导选项。



注意

此配置不会在图形控制台的机器上启用串行控制台访问。要配置不同的控制台，请在 APPEND 行中添加一个或多个 console= 参数。例如，添加 console=tty0 console=ttyS0 以将第一个 PC 串口设置为主控制台，并将图形控制台设置为二级控制台。如需更多信息，请参阅如何在 Red Hat Enterprise Linux 中设置串行终端和/或控制台？ 和"启用 PXE 和 ISO 安装的串行控制台"部分。

- **对于 iPXE (x86_64 + aarch64) :**

```
kernel http://<HTTP_server>/rhcos-<version>-live-kernel-<architecture> initrd=main
coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-rootfs.
<architecture>.img coreos.inst.install_dev=/dev/sda
coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign 1 2
initrd --name main http://<HTTP_server>/rhcos-<version>-live-initramfs.
<architecture>.img 3
boot
```

1

指定上传到 HTTP 服务器的 RHCOS 文件的位置。kernel 参数值是 kernel 文件的位置，init rd=main 参数用于在 UEFI 系统中引导，coreos.live.rootfs_url 参数值是 rootfs 文件的位置，coreos.inst.ignition_url 参数值则是 bootstrap Ignition 配置文件

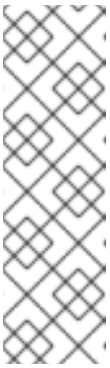
的位置。

2

如果您使用多个 NIC，请在 ip 选项中指定一个接口。例如，要在名为 eno1 的 NIC 上使用 DHCP，请设置 ip=eno1:dhcp。

3

指定上传到 HTTP 服务器的 initramfs 文件的位置。



注意

此配置不会在图形控制台的机器上启用串行控制台访问。要配置不同的控制台，请在 内核参数 中添加一个或多个 console= 参数。例如，添加 console=tty0 console=ttyS0 以将第一个 PC 串口设置为主控制台，并将图形控制台设置为二级控制台。如需更多信息，请参阅如何在 Red Hat Enterprise Linux 中设置串行终端和/或控制台？ 和"启用 PXE 和 ISO 安装的串行控制台"部分。

注意

要在 aarch64 架构中网络引导 CoreOS 内核，您需要使用启用了 IMAGE_GZIP 选项的 iPXE 构建版本。请参阅 iPXE 中的IMAGE_GZIP 选项。

- 对于 aarch64 中的 PXE（使用 UEFI 和 Grub 作为第二阶段）：

```
menuentry 'Install CoreOS' {
    linux rhcos-<version>-live-kernel-<architecture>
coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-rootfs.
<architecture>.img coreos.inst.install_dev=/dev/sda
coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign 1 2
    initrd rhcos-<version>-live-initramfs.<architecture>.img 3
}
```

1

指定上传到 HTTP/TFTP 服务器的 RHCOS 文件的位置。kernel 参数值是 TFTP 服务器中的 kernel 文件的位置。coreos.live.rootfs_url 参数值是 rootfs 文件的位置，coreos.inst.ignition_url 参数值是 HTTP 服务器上的 bootstrap Ignition 配置文件的位置。

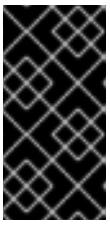
2

如果您使用多个 NIC，请在 ip 选项中指定一个接口。例如，要在名为 eno1 的 NIC 上使用 DHCP，请设置 ip=eno1:dhcp。

3

指定上传到 TFTP 服务器的 initramfs 文件的位置。

7. 在机器的控制台上监控 RHCOS 安装的进度。



重要

在开始安装 OpenShift Container Platform 之前，确保每个节点中安装成功。观察安装过程可以帮助确定可能会出现 RHCOS 安装问题的原因。

8. 安装 RHCOS 后，系统会重启。在重启过程中，系统会应用您指定的 Ignition 配置文件。

9. 检查控制台输出，以验证 Ignition 是否运行。

示例命令

```
Ignition: ran on 2022/03/14 14:48:33 UTC (this boot)
Ignition: user-provided config was applied
```

10. 继续为集群创建机器。

重要

此时您必须创建 bootstrap 和 control plane 机器。如果 control plane 机器不可调度，请在安装集群前至少创建两台计算机器。

如果存在所需的网络、DNS 和负载均衡器基础架构，OpenShift Container Platform bootstrap 过程会在 RHCOS 节点重启后自动启动。

注意

RHCOS 节点不包含 core 用户的默认密码。您可以使用可访问 SSH 私钥的用户的身份运行 ssh core@<node>.<cluster_name>.<base_domain > 来访问节点，该私钥与您在 install_config.yaml 文件中指定的公钥配对。运行 RHCOS 的 OpenShift Container Platform 4 集群节点不可变，它依赖于 Operator 来应用集群更改。不建议使用 SSH 访问集群节点。但是，当调查安装问题时，如果 OpenShift Container Platform API 不可用，或者 kubelet 在目标节点上无法正常工作，则调试或灾难恢复可能需要 SSH 访问。

24.1.11.3. 高级 RHCOS 安装配置

为 OpenShift Container Platform 手动置备 Red Hat Enterprise Linux CoreOS(RHCOS)节点的一个关键优点是能够进行通过默认的 OpenShift Container Platform 安装方法无法进行的配置。本节介绍了您可以使用的一些技术进行配置，其中包括：

- 将内核参数传递给实时安装程序
- 从 live 系统手动运行 coreos-installer
- 自定义实时 ISO 或 PXE 引导镜像

本节详述了与 Red Hat Enterprise Linux CoreOS(RHCOS)手动安装的高级配置相关的内容，如磁盘分区、网络以及使用 Ignition 配置的不同方式相关。

24.1.11.3.1. 使用高级网络选项进行 PXE 和 ISO 安装

OpenShift Container Platform 节点的网络默认使用 DHCP 来收集所有必要的配置设置。要设置静态 IP 地址或配置特殊设置，如绑定，您可以执行以下操作之一：

- 引导 live 安装程序时传递特殊内核参数。
- 使用机器配置将网络文件复制到安装的系统中。

- 从 live 安装程序 shell 提示符配置网络，然后将这些设置复制到安装的系统上，以便在安装的系统第一次引导时生效。

要配置 PXE 或 iPXE 安装，请使用以下选项之一：

- 请参阅"高级 RHCOS 安装参考"表。
- 使用机器配置将网络文件复制到安装的系统中。

要配置 ISO 安装，请使用以下步骤：

流程

1. 引导 ISO 安装程序.
2. 在 live 系统 shell 提示符下，使用可用的 RHEL 工具（如 nmcli 或 nmtui ）为 live 系统配置网络。
3. 运行 coreos-installer 命令以安装系统，添加 --copy-network 选项来复制网络配置。例如：

```
$ sudo coreos-installer install --copy-network \
    --ignition-url=http://host/worker.ign /dev/disk/by-id/scsi-<serial_number>
```

重要

copy-network 选项仅复制在 /etc/NetworkManager/system-connections 下找到的网络配置。特别是，它不会复制系统主机名。

4. 重启安装的系统。

其他资源

- 有关 nmcli 和 nmtui 工具的更多信息，请参阅 RHEL 8 文档中的 Getting started with

nmcli and Getting started with nmtui。

24.1.11.3.2. 磁盘分区

磁盘分区是在 Red Hat Enterprise Linux CoreOS(RHCOS)安装过程中在 OpenShift Container Platform 集群节点上创建的。特定架构的每个 RHCOS 节点使用相同的分区布局，除非覆盖默认的分区配置。在 RHCOS 安装过程中，根文件系统的大小会增加，以使用目标设备中剩余的可用空间。

重要

在节点上使用自定义分区方案可能会导致 OpenShift Container Platform 在某些节点分区上监控或警报。如果要覆盖默认分区，请参阅 了解 OpenShift 文件系统监控（驱除条件） 以了解有关 OpenShift Container Platform 如何监控主机文件系统的更多信息。

OpenShift Container Platform 监控以下两个文件系统标识符：

- nodefs，这是包含 /var/lib/kubelet 的文件系统
- imagefs，这是包含 /var/lib/containers 的文件系统

对于默认分区方案，nodefs 和 imagefs 监控相同的根文件系统 /。

要在 OpenShift Container Platform 集群节点上安装 RHCOS 时覆盖默认分区，您必须创建单独的分区。您可能想要为容器和容器镜像添加单独的存储分区。例如，通过在独立分区中挂载 /var/lib/containers，kubelet 会单独监控 /var/lib/containers 作为 imagefs 目录，以及 root 文件系统作为 nodefs 目录。

重要

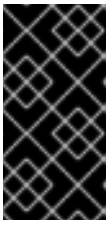
如果您已将磁盘大小调整为托管更大的文件系统，请考虑创建单独的 /var/lib/containers 分区。考虑重新定义具有 xfs 格式的磁盘大小，以减少大量分配组导致的 CPU 时间问题。

24.1.11.3.2.1. 创建独立 /var 分区

通常，您应该使用在 RHCOS 安装过程中创建的默认磁盘分区。然而，在有些情况下您可能需要为预期增长的目录创建独立分区。

OpenShift Container Platform 支持添加单个分区将存储附加到 /var 目录或 /var 的子目录中。例如：

- /var/lib/containers ：保存随着系统中添加更多镜像和容器而增长的容器相关内容。
- /var/lib/etcd ：保存您可能希望独立保留的数据，比如 etcd 存储的性能优化。
- /var ：保存您可能希望独立保留的数据，以满足审计等目的。



重要

对于大于 100GB 的磁盘大小，特别是磁盘大小大于 1TB，请创建一个独立的 /var 分区。

通过单独存储 /var 目录的内容，可以更轻松地根据需要为区域扩展存储，并在以后重新安装 OpenShift Container Platform，并保持该数据的完整性。使用这个方法，您不必再次拉取所有容器，在更新系统时也不必复制大量日志文件。

将独立分区用于 /var 目录或 /var 的子目录也会防止分区目录中的数据增加填充根文件系统。

以下流程通过添加机器配置清单来设置独立的 /var 分区，该清单会在安装准备阶段封装到节点类型的 Ignition 配置文件中。

流程

1. 在安装主机上，切换到包含 OpenShift Container Platform 安装程序的目录，并为集群生成 Kubernetes 清单：

```
$ openshift-install create manifests --dir <installation_directory>
```

2. 创建用于配置额外分区的 Butane 配置。例如，将文件命名为 $HOME/clusterconfig/98-var-partition.bu，将磁盘设备名称改为 worker 系统上存储设备的名称，并根据情况设置存储大小。这个示例将 /var 目录放在一个单独的分区中：

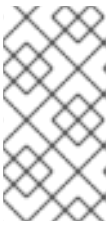
```
variant: openshift
version: 4.16.0
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
storage:
  disks:
  - device: /dev/disk/by-id/<device_name> 1
    partitions:
    - label: var
      start_mib: <partition_start_offset> 2
      size_mib: <partition_size> 3
      number: 5
  filesystems:
    - device: /dev/disk/by-partlabel/var
      path: /var
      format: xfs
      mount_options: [defaults, prjquota] 4
      with_mount_unit: true
```

1 要分区的磁盘的存储设备名称。

2 当在引导磁盘中添加数据分区时，推荐最少使用偏移值 25000 兆字节。root 文件系统会自动调整大小以填充所有可用空间（最多到指定的偏移值）。如果没有指定偏移值，或者指定的值小于推荐的最小值，则生成的 root 文件系统会太小，而在以后进行的 RHCOS 重新安装可能会覆盖数据分区的开始部分。

3 以兆字节为单位的数据分区大小。

4 对于用于容器存储的文件系统，必须启用 prjquota 挂载选项。



注意

在创建单独的 /var 分区时，如果不同的实例类型没有相同的设备名称，则无法将不同的实例类型用于计算节点。

3. 从 Butane 配置创建一个清单，并将它保存到 clusterconfig/openshift 目录中。例如，运行以下命令：

```
$ butane $HOME/clusterconfig/98-var-partition.bu -o
$HOME/clusterconfig/openshift/98-var-partition.yaml
```

4. 创建 Ignition 配置文件：

```
$ openshift-install create ignition-configs --dir <installation_directory> 1
```

1 对于 <installation_directory>，请指定相同的安装目录。

为安装目录中的 bootstrap、control plane 和计算节点创建 Ignition 配置文件：

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

<installation_directory>/manifest 和 <installation_directory>/openshift 目录中的文件被嵌套到 Ignition 配置文件中，包括包含 98-var-partition 自定义 MachineConfig 对象的文件。

后续步骤

- 您可以通过在 RHCOS 安装过程中引用 Ignition 配置文件来应用自定义磁盘分区。

24.1.11.3.2.2. 保留现有分区

对于 ISO 安装，您可以在 coreos-installer 命令中添加可让安装程序维护一个或多个现有分区的选项。对于 PXE 安装，您可以在 APPEND 参数中添加 coreos.inst.* 选项来保留分区。

保存的分区可能是来自现有 OpenShift Container Platform 系统的数据分区。您可以通过分区标签或编号识别您要保留的磁盘分区。

注意

如果您保存了现有分区，且这些分区没有为 RHCOS 留下足够空间，则安装将失败，而不影响保存的分区。

在 ISO 安装过程中保留现有分区

这个示例保留分区标签以 数据开头的任何分区(data *)：

```
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign \
      --save-partlabel 'data*' /dev/disk/by-id/scsi-<serial_number>
```

以下示例演示了在运行 coreos-installer 时要保留磁盘上的第 6 个分区：

```
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign \
      --save-partindex 6 /dev/disk/by-id/scsi-<serial_number>
```

这个示例保留分区 5 及更高分区：

```
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign
      --save-partindex 5- /dev/disk/by-id/scsi-<serial_number>
```

在前面已保存分区的示例中，coreos-installer 会立即重新创建分区。

在 PXE 安装过程中保留现有分区

这个 APPEND 选项保留分区标签以 'data'('data*')开头的任何分区：

```
coreos.inst.save_partlabel=data*
```

这个 APPEND 选项保留分区 5 及更高分区：

```
coreos.inst.save_partindex=5-
```

这个 APPEND 选项保留分区 6:

```
coreos.inst.save_partindex=6
```

24.1.11.3.3. 识别 Ignition 配置

在进行 RHCOS 手动安装时，您可以提供两种 Ignition 配置类型，它们有不同的原因：

- 永久安装 Ignition 配置 ：每个手动 RHCOS 安装都需要传递 openshift-installer 生成的 Ignition 配置文件之一，如 bootstrap.ign、master.ign 和 worker.ign，才能进行安装。

重要

不建议直接修改这些 Ignition 配置文件。您可以更新嵌套到 Ignition 配置文件中的清单文件，如上一节示例中所述。

对于 PXE 安装，您可以使用 coreos.inst.ignition_url= 选项在 APPEND 行上传递 Ignition 配置。对于 ISO 安装，在 ISO 引导至 shell 提示符后，您可以使用带有 --ignition-url= 选项的 coreos-installer 命令行。在这两种情况下，只支持 HTTP 和 HTTPS 协议。

- 实时安装 Ignition 配置 ：可使用 coreos-installer customize 子命令及其各种选项来创建此类型。使用此方法，Ignition 配置会传递到 live 安装介质，在引导时立即运行，并在 RHCOS 系统安装到磁盘之前或之后执行设置任务。这个方法只用于必须执行一次且之后不能再次应用的任务，比如不能使用机器配置进行的高级分区。

对于 PXE 或 ISO 引导，您可以创建 Ignition 配置，APP END ignition.config.url= 选项来标识 Ignition 配置的位置。您还需要附加 ignition.firstboot ignition.platform.id=metal 或 ignition.config.url 选项。

24.1.11.3.4. 高级 RHCOS 安装参考

本节演示了网络配置和其他高级选项，允许您修改 Red Hat Enterprise Linux CoreOS(RHCOS)手动安装过程。下表描述了您可以用于 RHCOS live 安装程序和 coreos-installer 命令的内核参数和命令行选项。

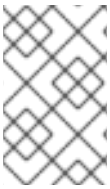
24.1.11.3.4.1. ISO 安装的网络和绑定选项

如果从 ISO 镜像安装 RHCOS，您可以在引导镜像时手动添加内核参数，以便为节点配置网络。如果没有指定网络参数，当 RHCOS 检测到需要网络来获取 Ignition 配置文件时，在 initramfs 中激活 DHCP。

重要

在手动添加网络参数时，还必须添加 rd.neednet=1 内核参数，以便在 initramfs 中启动网络。

以下信息提供了在 RHCOS 节点上为 ISO 安装配置网络和绑定的示例。示例描述了如何使用 ip=、name server = 和 bond= 内核参数。



注意

添加内核参数时顺序非常重要： ip=、 name server=，然后 bond=。

网络选项在系统引导过程中传递给 dracut 工具。有关 dracut 支持的网络选项的更多信息，请参阅 dracut.cmdline 手册页。

以下示例是 ISO 安装的网络选项。

配置 DHCP 或静态 IP 地址

要配置 IP 地址，可使用 DHCP(ip=dhcp)或设置单独的静态 IP 地址(ip=<host_ip>)。如果设置静态 IP，则必须在每个节点上识别 DNS 服务器 IP 地址（名称服务器=<dns_ip>）。以下示例集：

- 节点的 IP 地址为 10.10.10.2
- 网关地址为 10.10.10.254
- 子网掩码为 255.255.255.0
- 到 core0.example.com 的主机名
- DNS 服务器地址为 4.4.4.41

- 自动配置值为 none。当以静态方式配置 IP 网络时，不需要自动配置。

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
nameserver=4.4.4.41
```

注意

当您使用 DHCP 为 RHCOS 机器配置 IP 寻址时，机器还通过 DHCP 获取 DNS 服务器信息。对于基于 DHCP 的部署，您可以通过 DHCP 服务器配置定义 RHCOS 节点使用的 DNS 服务器地址。

配置没有静态主机名的 IP 地址

您可以在不分配静态主机名的情况下配置 IP 地址。如果用户没有设置静态主机名，则会提取并通过反向 DNS 查找自动设置。要在没有静态主机名的情况下配置 IP 地址，请参考以下示例：

- 节点的 IP 地址为 10.10.10.2
- 网关地址为 10.10.10.254
- 子网掩码为 255.255.255.0
- DNS 服务器地址为 4.4.4.41
- 自动配置值为 none。当以静态方式配置 IP 网络时，不需要自动配置。

```
ip=10.10.10.2::10.10.10.254:255.255.255.0::enp1s0:none
nameserver=4.4.4.41
```

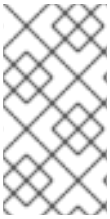
指定多个网络接口

您可以通过设置多个 ip= 条目来指定多个网络接口。

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
ip=10.10.10.3::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none
```

配置默认网关和路由

可选：您可以通过设置 a rd.route= 值来配置到额外网络的路由。



注意

当您配置一个或多个网络时，需要一个默认网关。如果额外网络网关与主要网络网关不同，则默认网关必须是主要网络网关。

- 运行以下命令来配置默认网关：

```
ip=::10.10.10.254::::
```

- 输入以下命令为额外网络配置路由：

```
rd.route=20.20.20.0/24:20.20.20.254:enp2s0
```

在单个接口中禁用 DHCP

您可以在单一接口中禁用 DHCP，例如当有两个或者多个网络接口时，且只有一个接口被使用。在示例中，enp1s0 接口具有一个静态网络配置，而 enp2s0 禁用了 DHCP，不使用它：

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none
ip=::::core0.example.com:enp2s0:none
```

合并 DHCP 和静态 IP 配置

您可以将系统上的 DHCP 和静态 IP 配置与多个网络接口合并，例如：

```
ip=enp1s0:dhcp
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none
```

在独立接口上配置 VLAN

可选： 您可以使用 vlan= 参数在单个接口上配置 VLAN。

- 要在网络接口中配置 VLAN 并使用静态 IP 地址，请运行以下命令：

```
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0.100:none
vlan=enp2s0.100:enp2s0
```

- 要在网络接口中配置 VLAN 并使用 DHCP，请运行以下命令：

```
ip=enp2s0.100:dhcp
vlan=enp2s0.100:enp2s0
```

提供多个 DNS 服务器

您可以通过为每个服务器添加一个 nameserver= 条目来提供多个 DNS 服务器，例如

```
nameserver=1.1.1.1
nameserver=8.8.8.8
```

将多个网络接口绑定到一个接口

可选：您可以使用 bond= 选项将多个网络接口绑定到一个接口。请参见以下示例：

- 配置绑定接口的语法为：bond=<name>[:<network_interfaces>][:options]

  <name> 是绑定设备名称 (bond0)、<network_interfaces> 代表以逗号分隔的物理（以太网）接口列表(em1,em2)，*options* 是用逗号分开的绑定选项列表。输入 modinfo bonding 查看可用选项。

- 当使用 bond= 创建绑定接口时，您必须指定如何分配 IP 地址以及绑定接口的其他信息。

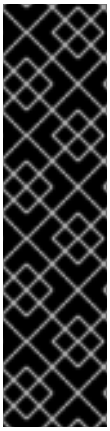
  - 要将绑定接口配置为使用 DHCP，请将绑定的 IP 地址设置为 dhcp。例如：

    ```
    bond=bond0:em1,em2:mode=active-backup
    ip=bond0:dhcp
    ```

  - 要将绑定接口配置为使用静态 IP 地址，请输入您需要的特定 IP 地址和相关信息。例如：

    ```
    bond=bond0:em1,em2:mode=active-backup
    ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0:none
    ```

将多个 SR-IOV 网络接口绑定到双端口 NIC 接口



重要

支持与为 SR-IOV 设备启用 NIC 分区关联的第 1 天操作只是一个技术预览功能。技术预览功能不受红帽产品服务等级协议（SLA）支持，且功能可能并不完整。红帽不推荐在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。

有关红帽技术预览功能支持范围的更多信息，请参阅技术预览功能支持范围。

可选： 您可以使用 bond= 选项将多个 SR-IOV 网络接口绑定到双端口 NIC 接口。

在每个节点上，您必须执行以下任务：

1. 按照管理 SR-IOV 设备中的指导创建 SR-IOV 虚拟功能(VF)。按照"将 SR-IOV 网络设备附加到虚拟机"部分中的步骤操作。

2. 创建绑定，将所需的 VF 附加到绑定，并根据配置网络绑定的指导设置绑定链接状态。按照任何描述的步骤创建绑定。

以下示例演示了您必须使用的语法：

- 配置绑定接口的语法为：bond=<name>[:<network_interfaces>][:options]

  <name> 是绑定设备名称 (bond0)、<network_interfaces> 由内核中已知的名称来代表虚拟功能(VF)，并显示在 ip link 命令的输出中 (eno1f0,eno2f0)， *options* 是以逗号分隔的绑定选项列表。输入 modinfo bonding 查看可用选项。

- 当使用 bond= 创建绑定接口时，您必须指定如何分配 IP 地址以及绑定接口的其他信息。

  - 要将绑定接口配置为使用 DHCP，请将绑定的 IP 地址设置为 dhcp。例如：

    ```
    bond=bond0:eno1f0,eno2f0:mode=active-backup
    ip=bond0:dhcp
    ```

  -

要将绑定接口配置为使用静态 IP 地址，请输入您需要的特定 IP 地址和相关信息。例如：

```
bond=bond0:eno1f0,eno2f0:mode=active-backup
ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0:none
```

使用网络团队

可选：您可以使用 team= 参数来将网络团队用作绑定的替代选择：

- 配置组接口的语法为：team=name[:network_interfaces]

name 是组设备名称(team0)，*network_interfaces* 代表以逗号分隔的物理（以太网）接口（em1、em2）列表。

注意

当 RHCOS 切换到即将推出的 RHEL 版本时，团队(team)功能被计划弃用。如需更多信息，请参阅红帽知识库文章。

使用以下示例配置网络团队：

```
team=team0:em1,em2
ip=team0:dhcp
```

24.1.11.3.4.2. ISO 和 PXE 安装的 coreos-installer 选项

从 ISO 镜像引导 RHCOS live 环境后，您可以通过 在命令提示符下运行 coreos-installer install <options> <device > 来安装 RHCOS。

下表显示了您可以传递给 coreos-installer 命令的子命令、选项和参数。

表 24.9. coreos-installer 子命令、命令行选项和参数

coreos-installer install 子命令	
子命令	*描述*
$ coreos-installer install <options> <device>	在 ISO 镜像中嵌入 Ignition 配置。

coreos-installer install 子命令选项	
选项	***描述***
-u, --image-url <url>	手动指定镜像 URL。
-f,--image-file <path>	手动指定本地镜像文件。用于调试。
-i, --ignition-file <path>	从文件中嵌入 Ignition 配置。
-i,--ignition-url <URL>	从 URL 嵌入 Ignition 配置。
--ignition-hash <digest>	Ignition 配置 **的 type-value** 的摘要值。
-p, --platform <name>	覆盖已安装系统的 Ignition 平台 ID。
--console <spec>	为安装的系统设置内核和引导装载程序控制台。有关 **<spec>** 格式的更多信息，请参阅 Linux 内核串口控制台文档。
--append-karg <arg>...	将默认内核参数附加到安装的系统中。
--delete-karg <arg>...	从安装的系统中删除默认内核参数。
-n, --copy-network	从安装环境中复制网络配置。 **重要** **copy-network** 选项仅复制在 **/etc/NetworkManager/system-connections** 下找到的网络配置。特别是，它不会复制系统主机名。
--network-dir <path>	用于 **-n**。默认为 **/etc/NetworkManager/system-connections/**。
--save-partlabel <lx>..	使用这个标签 glob 保存分区。
--save-partindex <id>...	使用这个数值或范围保存分区。
--insecure	跳过 RHCOS 镜像签名验证。
--insecure-ignition	允许没有 HTTPS 或 hash 的 Ignition URL。
--architecture <name>	目标 CPU 架构.有效值为 **x86_64** 和 **aarch64**。
--preserve-on-error	出错时不要清除分区表。

-h,--help	打印帮助信息.
coreos-installer install 子命令参数	
参数	***描述***
<device>	目标设备.
coreos-installer ISO 子命令	
子命令	***描述***
$ coreos-installer iso customize <options> <ISO_image>	自定义 RHCOS live ISO 镜像。
coreos-installer iso reset <options> <ISO_image>	将 RHCOS live ISO 镜像恢复到默认设置。
coreos-installer iso ignition remove <options> <ISO_image>	从 ISO 镜像中删除嵌入的 Ignition 配置。
coreos-installer ISO customize 子命令选项	
选项	***描述***
--dest-ignition <path>	将指定的 Ignition 配置文件合并到目标系统的新配置片段中。
--dest-console <spec>	为目标系统指定内核和引导装载程序控制台。
--dest-device <path>	安装并覆盖指定的目标设备。
--dest-karg-append <arg>	为每个目标系统引导添加一个内核参数。
--dest-karg-delete <arg>	从目标系统的每个引导中删除内核参数。
--network-keyfile <path>	使用指定的 NetworkManager 密钥文件进行实时和目标系统配置网络。
--ignition-ca <path>	指定要被 Ignition 信任的额外 TLS 证书颁发机构。
--pre-install <path>	在安装之前运行指定的脚本。
--post-install <path>	安装后运行指定的脚本。
--installer-config <path>	应用指定的安装程序配置文件。

--live-ignition <path>	将指定的 Ignition 配置文件合并到实时环境的新配置片段中。
--live-karg-append <arg>	为每个实时环境引导添加一个内核参数。
--live-karg-delete <arg>	从实时环境每次引导时删除内核参数。
--live-karg-replace <k=o=n>	在每次启动 live 环境时替换内核参数，格式为 **key=old=new**。
-f,--force	覆盖现有的 Ignition 配置。
-o,--output <path>	将 ISO 写入到新的输出文件。
-h,--help	打印帮助信息.
coreos-installer PXE 子命令	
子命令	***描述***
请注意，并非所有子命令都接受所有这些选项。	
coreos-installer pxe customize <options> <path>	自定义 RHCOS live PXE 引导配置。
coreos-installer pxe ignition wrap <options>	在镜像中嵌套 Ignition 配置。
coreos-installer pxe ignition unwrap <options> <image_name>	在镜像中显示嵌套的 Ignition 配置。
coreos-installer PXE customize 子命令选项	
选项	***描述***
请注意，并非所有子命令都接受所有这些选项。	
--dest-ignition <path>	将指定的 Ignition 配置文件合并到目标系统的新配置片段中。
--dest-console <spec>	为目标系统指定内核和引导装载程序控制台。
--dest-device <path>	安装并覆盖指定的目标设备。
--network-keyfile <path>	使用指定的 NetworkManager 密钥文件进行实时和目标系统配置网络。
--ignition-ca <path>	指定要被 Ignition 信任的额外 TLS 证书颁发机构。

--pre-install <path>	在安装之前运行指定的脚本。
post-install <path>	安装后运行指定的脚本。
--installer-config <path>	应用指定的安装程序配置文件。
--live-ignition <path>	将指定的 Ignition 配置文件合并到实时环境的新配置片段中。
-o, --output <path>	将 initramfs 写入一个新输出文件。  **注意** PXE 环境需要这个选项。
-h,--help	打印帮助信息.

24.1.11.3.4.3. coreos.inst 引导选项用于 ISO 或 PXE 安装

您可以通过将 coreos.inst boot 参数传递给 RHCOS live 安装程序，在引导时自动调用 coreos-installer 选项。这些是在标准引导参数之外提供的。

- 对于 ISO 安装，可以通过在启动加载器菜单中中断自动引导来添加 coreos.inst 选项。您可以在突出显示 RHEL CoreOS(Live) 菜单选项时按 TAB 来中断自动引导。
- 对于 PXE 或 iPXE 安装，在引导 RHCOS live 安装程序前，coreos.inst 选项必须添加到 APPEND 行。

下表显示了用于 ISO 和 PXE 安装的 RHCOS live 安装程序 coreos.inst 引导选项。

表 24.10. coreos.inst 引导选项

参数	描述
coreos.inst.install_dev	必需。要安装到的系统中的块设备。建议您使用完整路径，如 **/dev/sda**，但 **允许使用**。
coreos.inst.ignition_url	可选：嵌入到安装的系统中的 Ignition 配置的 URL。如果没有指定 URL，则不会嵌入 Ignition 配置。仅支持 HTTP 和 HTTPS 协议。

参数	描述
coreos.inst.save_partlabel	可选：在安装过程中要保留的分区分离标签。允许使用 glob 风格的通配符。指定分区不需要存在。
coreos.inst.save_partindex	可选：在安装过程中压缩要保留的分区索引。允许 **m-n** 范围，**m** 或 **n** 可以被省略。指定分区不需要存在。
coreos.inst.insecure	可选：将 **coreos.inst.image_url** 指定的 OS 镜像提交取消签名。
coreos.inst.image_url	可选：下载并安装指定的 RHCOS 镜像。 ● 这个参数不应该在生产环境中使用，而是只用于调试目的。 ● 虽然此参数可用于安装与 live 介质不匹配的 RHCOS 版本，但建议您使用与您要安装版本匹配的介质。 ● 如果您使用 **coreos.inst.image_url**，还必须使用 **coreos.inst.insecure**。这是因为，裸机介质没有为 OpenShift Container Platform 进行 GPG 签名。 ● 仅支持 HTTP 和 HTTPS 协议。
coreos.inst.skip_reboot	可选：安装后系统不会重启。安装完成后，您将收到提示，提示您检查在安装过程中发生的情况。这个参数不应该在生产环境中使用，而是只用于调试目的。
coreos.inst.platform_id	可选：安装 RHCOS 镜像的平台的 Ignition 平台 ID。默认为 **metal**。这个选项决定是否从云供应商（如 VMware）请求 Ignition 配置。例如：**coreos.inst.platform_id=vmware**。
ignition.config.url	可选：用于实时引导的 Ignition 配置的 URL。例如，这可用于自定义调用 **coreos-installer** 的方式，或者用于在安装前或安装后运行代码。这与 **coreos.inst.ignition_url**（这是已安装系统的 Ignition 配置）不同。

24.1.12. 等待 bootstrap 过程完成

OpenShift Container Platform bootstrap 过程在集群节点首次引导到安装到磁盘的持久 RHCOS 环境后开始。通过 Ignition 配置文件提供的配置信息用于初始化 bootstrap 过程并在机器上安装 OpenShift Container Platform。您必须等待 bootstrap 过程完成。

先决条件

- 已为集群创建 Ignition 配置文件。
- 您已配置了适当的网络、DNS 和负载平衡基础架构。
- 已获得安装程序，并为集群生成 Ignition 配置文件。
- 已在集群机器上安装 RHCOS，并提供 OpenShift Container Platform 安装程序生成的 Ignition 配置文件。
- 您的机器可以直接访问互联网，或者有 HTTP 或 HTTPS 代理可用。

流程

1. 监控 bootstrap 过程：

```
$ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \ 1
    --log-level=info 2
```

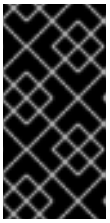
1. 对于 <installation_directory>，请指定安装文件保存到的目录的路径。
2. 要查看不同的安装详情，请指定 warn、debug 或 error，而不是 info。

输出示例

```
INFO Waiting up to 30m0s for the Kubernetes API at
https://api.test.example.com:6443...
INFO API v1.29.4 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
```

当 Kubernetes API 服务器提示已在 control plane 机器上引导它时，该命令会成功。

2. bootstrap 过程完成后，从负载均衡器中删除 bootstrap 机器。



重要

此时您必须从负载均衡器中删除 bootstrap 机器。您还可以删除或重新格式化 bootstrap 机器本身。

24.1.13. 使用 CLI 登录集群

您可以通过导出集群 kubeconfig 文件，以默认系统用户身份登录集群。kubeconfig 文件包含有关集群的信息，供 CLI 用于将客户端连接到正确的集群和 API 服务器。该文件特定于集群，在 OpenShift Container Platform 安装过程中创建。

先决条件

- 已部署 OpenShift Container Platform 集群。
- 已安装 oc CLI。

流程

1. 导出 kubeadmin 凭证：

```
$ export KUBECONFIG=<installation_directory>/auth/kubeconfig 1
```

1 对于 <installation_directory>，请指定安装文件保存到的目录的路径。

2. 验证您可以使用导出的配置成功运行 oc 命令：

```
$ oc whoami
```

输出示例

```
system:admin
```

24.1.14. 批准机器的证书签名请求

当您将机器添加到集群时，会为您添加的每台机器生成两个待处理证书签名请求(CSR)。您必须确认这些 CSR 已获得批准，或根据需要自行批准。必须首先批准客户端请求，然后批准服务器请求。

先决条件

- 您已将机器添加到集群中。

流程

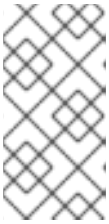
1. 确认集群可以识别这些机器：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0  Ready    master  63m  v1.29.4
master-1  Ready    master  63m  v1.29.4
master-2  Ready    master  64m  v1.29.4
```

输出中列出了您创建的所有机器。



注意

在有些 CSR 被批准前，前面的输出可能不包括计算节点（也称为 worker 节点）。

2. 检查待处理的 CSR，并确保添加到集群中的每台机器都有 Pending 或 Approved 状态的客户端请求：

```
$ oc get csr
```

输出示例

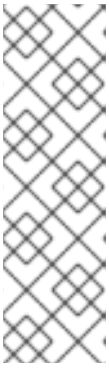
```
NAME        AGE    REQUESTOR                                                   CONDITION
csr-8b2br   15m    system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
csr-8vnps   15m    system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
...
```

在本例中，两台机器加入集群。您可能会在列表中看到更多已批准的 CSR。

3. 如果 CSR 没有获得批准，在您添加的机器的所有待处理 CSR 都处于 Pending 状态 后，请批准集群机器的 CSR：

注意

由于 CSR 会自动轮转，因此请在将机器添加到集群后一小时内批准您的 CSR。如果没有在一小时内批准它们，证书将会轮转，每个节点会存在多个证书。您必须批准所有这些证书。批准客户端 CSR 后，Kubelet 为服务证书创建一个二级 CSR，这需要手动批准。然后，如果 Kubelet 请求具有相同参数的新证书，则后续提供证书续订请求由 machine-approver 自动批准。



注意

对于在未启用机器 API 的平台上运行的集群，如裸机和其他用户置备的基础架构，您必须实施一种方法来自动批准 kubelet 提供证书请求(CSR)。如果没有批准请求，则 oc exec、ocrsh 和 oc logs 命令将无法成功，因为 API 服务器连接到 kubelet 时需要服务证书。与 Kubelet 端点联系的任何操作都需要此证书批准。该方法必须监视新的 CSR，确认 CSR 由 system: node 或 system:admin 组中的 node-bootstrapper 服务帐户提交，并确认节点的身份。

- 要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 <csr_name> 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}
{{"\n"}}{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```

注意

在有些 CSR 被批准前，一些 Operator 可能无法使用。

4. 现在，您的客户端请求已被批准，您必须查看添加到集群中的每台机器的服务器请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE     REQUESTOR                                                   CONDITION
csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

5. 如果剩余的 CSR 没有被批准，且处于 Pending 状态，请批准集群机器的 CSR：

- 要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1

<csr_name> 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

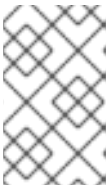
```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}
{{"\n"}}{{end}}{{end}}' | xargs oc adm certificate approve
```

6. 批准所有客户端和服务器 CSR 后，机器将 处于 Ready 状态。运行以下命令验证：

```
$ oc get nodes
```

输出示例

```
NAME     STATUS   ROLES   AGE  VERSION
master-0  Ready    master  73m  v1.29.4
master-1  Ready    master  73m  v1.29.4
master-2  Ready    master  74m  v1.29.4
worker-0  Ready    worker  11m  v1.29.4
worker-1  Ready    worker  11m  v1.29.4
```



注意

批准服务器 CSR 后可能需要几分钟时间让机器过渡到 Ready 状态。

其他信息

- 如需有关 CSR 的更多信息，请参阅 证书签名请求。

24.1.15. 初始 Operator 配置

在 control plane 初始化后，您必须立即配置一些 Operator，以便它们都可用。

先决条件

- 您的 control plane 已初始化。

流程

1. 观察集群组件上线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION   AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.16.0    True        False         False      19m
baremetal                                  4.16.0    True        False         False      37m
cloud-credential                           4.16.0    True        False         False      40m
cluster-autoscaler                         4.16.0    True        False         False      37m
config-operator                            4.16.0    True        False         False      38m
console                                    4.16.0    True        False         False      26m
csi-snapshot-controller                    4.16.0    True        False         False      37m
dns                                        4.16.0    True        False         False      37m
etcd                                       4.16.0    True        False         False      36m
image-registry                             4.16.0    True        False         False      31m
ingress                                    4.16.0    True        False         False      30m
insights                                   4.16.0    True        False         False      31m
kube-apiserver                             4.16.0    True        False         False      26m
kube-controller-manager                    4.16.0    True        False         False      36m
kube-scheduler                             4.16.0    True        False         False      36m
kube-storage-version-migrator              4.16.0    True        False         False      37m
machine-api                                4.16.0    True        False         False      29m
machine-approver                           4.16.0    True        False         False      37m
machine-config                             4.16.0    True        False         False      36m
marketplace                                4.16.0    True        False         False      37m
monitoring                                 4.16.0    True        False         False      29m
network                                    4.16.0    True        False         False      38m
node-tuning                                4.16.0    True        False         False      37m
openshift-apiserver                        4.16.0    True        False         False      32m
openshift-controller-manager               4.16.0    True        False         False      30m
openshift-samples                          4.16.0    True        False         False      32m
operator-lifecycle-manager                 4.16.0    True        False         False      37m
operator-lifecycle-manager-catalog         4.16.0    True        False         False      37m
operator-lifecycle-manager-packageserver   4.16.0    True        False         False      32m
service-ca                                 4.16.0    True        False         False      38m
storage                                    4.16.0    True        False         False      37m
```

2. 配置不可用的 Operator。

24.1.15.1. 禁用默认的 OperatorHub 目录源

在 OpenShift Container Platform 安装过程中，默认为 OperatorHub 配置由红帽和社区项目提供的源内容的 operator 目录。在受限网络环境中，必须以集群管理员身份禁用默认目录。

流程

- 通过在 OperatorHub 对象中添加 disableAllDefaultSources: true 来 禁用默认目录的源：

```
$ oc patch OperatorHub cluster --type json \
    -p '[{"op": "add", "path": "/spec/disableAllDefaultSources", "value": true}]'
```

提示

或者，您可以使用 Web 控制台管理目录源。在 Administration → Cluster Settings → Configuration → OperatorHub 页面中，点 Sources 选项卡，您可以在其中创建、更新、删除、禁用和启用单独的源。

24.1.15.2. 安装过程中删除的镜像 registry

在不提供可共享对象存储的平台上，OpenShift Image Registry Operator bootstraps 本身为 Removed。这允许 openshift-installer 在这些平台类型上完成安装。

安装后，您必须编辑 Image Registry Operator 配置，将 managementState 从 Removed 切换到 Managed。完成此操作后，您必须配置存储。

24.1.15.3. 镜像 registry 存储配置

对于不提供默认存储的平台，Image Registry Operator 最初不可用。安装后，您必须将 registry 配置为使用存储，以便 Registry Operator 可用。

显示配置生产集群所需的持久性卷的说明。如果适用，显示有关将空目录配置为存储位置的说明，这仅适用于非生产集群。

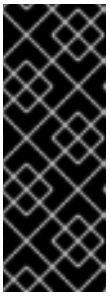
提供了在升级过程中使用 Recreate rollout 策略来允许镜像 registry 使用块存储类型的说明。

24.1.15.3.1. 为裸机和其他手动安装配置 registry 存储

作为集群管理员，在安装后需要配置 registry 来使用存储。

先决条件

- 您可以使用具有 cluster-admin 角色的用户访问集群。
- 您有一个使用手动置备的 Red Hat Enterprise Linux CoreOS(RHCOS)节点（如裸机）的集群。
- 您已为集群置备持久性存储，如 Red Hat OpenShift Data Foundation。



重要

当您只有一个副本时，OpenShift Container Platform 支持对镜像 registry 存储的 ReadWriteOnce 访问。ReadWriteOnce 访问还要求 registry 使用 Recreate rollout 策略。要部署支持高可用性的镜像 registry，需要两个或多个副本，ReadWriteMany 访问。

- 必须具有 100Gi 容量。

流程

1. 要将 registry 配置为使用存储，修改 configs.imageregistry/cluster 资源中的 spec.storage.pvc。

注意

使用共享存储时，请查看您的安全设置以防止外部访问。

2. 验证您没有 registry pod:

```
$ oc get pod -n openshift-image-registry -l docker-registry=default
```

输出示例

```
No resources found in openshift-image-registry namespace
```

注意

如果您的输出中有一个 registry pod，则不需要继续这个过程。

3. 检查 registry 配置：

```
$ oc edit configs.imageregistry.operator.openshift.io
```

输出示例

```
storage:
  pvc:
    claim:
```

将 claim 字段留空以允许自动创建 image-registry-storage PVC。

4. 检查 clusteroperator 状态：

```
$ oc get clusteroperator image-registry
```

输出示例

```
NAME              VERSION        AVAILABLE   PROGRESSING   DEGRADED   SINCE
MESSAGE
image-registry    4.16           True        False         False      6h50m
```

5. 确保 registry 设置为 managed，以启用镜像的构建和推送。

- 运行：

```
$ oc edit configs.imageregistry/cluster
```

然后，更改行

```
managementState: Removed
```

至

```
managementState: Managed
```

24.1.15.3.2. 在非生产集群中为镜像 registry 配置存储

您必须为 Image Registry Operator 配置存储。对于非生产集群，您可以将镜像 registry 设置为空目录。如果您这样做，重启 registry 时会丢失所有镜像。

流程

- 将镜像 registry 存储设置为空目录：

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch
'{"spec":{"storage":{"emptyDir":{}}}}'
```

警告

仅为非生产集群配置这个选项。

如果在 Image Registry Operator 初始化其组件前运行这个命令，oc patch 命令会失败并显示以下错误：

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster" not found
```

等待几分钟，然后再次运行 命令。

24.1.15.3.3. 为裸机配置块 registry 存储

要允许镜像 registry 在作为集群管理员升级过程中使用块存储类型，您可以使用 Recreate rollout 策略。

重要

支持块存储卷或块持久性卷，但不建议在生产环境中使用镜像 registry。在块存储上配置 registry 的安装不具有高可用性，因为 registry 无法具有多个副本。

如果您选择将块存储卷与镜像 registry 搭配使用，则必须使用文件系统持久性卷声明(PVC)。

流程

1. 输入以下命令将镜像 registry 存储设置为块存储类型，对 registry 进行补丁，使其使用 Recreate rollout 策略，并只使用一个副本运行：

```
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec":{"rolloutStrategy":"Recreate","replicas":1}}'
```

2. 为块存储设备置备 PV，并为该卷创建 PVC。请求的块卷使用 ReadWriteOnce(RWO)访问模式。

a. 创建包含以下内容的 pvc.yaml 文件以定义 VMware vSphere PersistentVolumeClaim 对象：

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: image-registry-storage 1
  namespace: openshift-image-registry 2
spec:
  accessModes:
  - ReadWriteOnce 3
  resources:
    requests:
      storage: 100Gi 4
```

1 代表 PersistentVolumeClaim 对象的唯一名称。

2 PersistentVolumeClaim 对象的命名空间，即 openshift-image-registry。

3 持久性卷声明的访问模式。使用 ReadWriteOnce 时，单个节点可以通过读写权限挂载该卷。

4 持久性卷声明的大小。

b. 输入以下命令从文件创建 PersistentVolumeClaim 对象：

```
$ oc create -f pvc.yaml -n openshift-image-registry
```

3. 输入以下命令编辑 registry 配置，使其引用正确的 PVC：

```
$ oc edit config.imageregistry.operator.openshift.io -o yaml
```

输出示例

```
storage:
  pvc:
    claim: 1
```

1

通过创建自定义 PVC，您可以将 claim 字段留空，以便默认自动创建 image-registry-storage PVC。

24.1.16. 在用户置备的基础架构上完成安装

完成 Operator 配置后，可以在您提供的基础架构上完成集群安装。

先决条件

- 您的 control plane 已初始化。
- 已完成初始 Operator 配置。

流程

1. 使用以下命令确认所有集群组件都在线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION   AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.16.0    True        False         False      19m
baremetal                                  4.16.0    True        False         False      37m
cloud-credential                           4.16.0    True        False         False      40m
cluster-autoscaler                         4.16.0    True        False         False      37m
config-operator                            4.16.0    True        False         False      38m
```

```
console                                    4.16.0    True        False         False      26m
csi-snapshot-controller                    4.16.0    True        False         False      37m
dns                                        4.16.0    True        False         False      37m
etcd                                       4.16.0    True        False         False      36m
image-registry                             4.16.0    True        False         False      31m
ingress                                    4.16.0    True        False         False      30m
insights                                   4.16.0    True        False         False      31m
kube-apiserver                             4.16.0    True        False         False      26m
kube-controller-manager                    4.16.0    True        False         False      36m
kube-scheduler                             4.16.0    True        False         False      36m
kube-storage-version-migrator              4.16.0    True        False         False      37m
machine-api                                4.16.0    True        False         False      29m
machine-approver                           4.16.0    True        False         False      37m
machine-config                             4.16.0    True        False         False      36m
marketplace                                4.16.0    True        False         False      37m
monitoring                                 4.16.0    True        False         False      29m
network                                    4.16.0    True        False         False      38m
node-tuning                                4.16.0    True        False         False      37m
openshift-apiserver                        4.16.0    True        False         False      32m
openshift-controller-manager               4.16.0    True        False         False      30m
openshift-samples                          4.16.0    True        False         False      32m
operator-lifecycle-manager                 4.16.0    True        False         False      37m
operator-lifecycle-manager-catalog         4.16.0    True        False         False      37m
operator-lifecycle-manager-packageserver   4.16.0    True        False         False      32m
service-ca                                 4.16.0    True        False         False      38m
storage                                    4.16.0    True        False         False      37m
```

另外，当所有集群都可用时，以下命令会通知您。它还检索并显示凭证：

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

1

对于 <installation_directory>，请指定安装文件保存到的目录的路径。

输出示例

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

Cluster Version Operator 完成从 Kubernetes API 服务器部署 OpenShift Container

Platform 集群时，该命令会成功。

重要

- 安装程序生成的 Ignition 配置文件包含 24 小时后过期的证书，然后在该时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外是，您必须手动批准待处理的 node-bootstrapper 证书签名请求(CSR)来恢复 kubelet 证书。如需更多信息，*请参阅从过期的 control plane 证书* 中恢复的文档。

- 建议您在 Ignition 配置文件生成后的 12 小时内使用它们，因为 24 小时的证书会在集群安装后的 16 小时到 22 小时间进行轮转。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中因为执行了证书更新而导致安装失败的问题。

2. 确认 Kubernetes API 服务器正在与 pod 通信。

   a. 要查看所有 pod 的列表，请使用以下命令：

```
$ oc get pods --all-namespaces
```

输出示例

```
NAMESPACE                  NAME                                        READY   STATUS
RESTARTS   AGE
openshift-apiserver-operator    openshift-apiserver-operator-85cb746d55-zqhs8
1/1    Running    1          9m
openshift-apiserver        apiserver-67b9g                             1/1     Running    0
3m
openshift-apiserver        apiserver-ljcmx                             1/1     Running    0
1m
openshift-apiserver        apiserver-z25h4                             1/1     Running    0
2m
openshift-authentication-operator authentication-operator-69d5d8bf84-vh2n8
1/1    Running    0          5m
...
```

b. 使用以下命令，查看上一命令的输出中所列 pod 的日志：

```
$ oc logs <pod_name> -n <namespace> 1
```

1 指定 pod 名称和命名空间，如上一命令的输出中所示。

如果 pod 日志显示，Kubernetes API 服务器可以与集群机器通信。

3. 对于使用光纤通道协议(FCP)的安装，还需要额外的步骤才能启用多路径。不要在安装过程中启用多路径。

如需更多信息，请参阅 *安装后机器配置任务* 文档中的"使用 RHCOS 上使用内核参数启用多路径"。

24.1.17. OpenShift Container Platform 的 Telemetry 访问

在 OpenShift Container Platform 4.16 中，默认运行的 Telemetry 服务提供有关集群健康状况和成功更新的指标，需要访问互联网。如果您的集群连接到互联网，Telemetry 会自动运行，而且集群会注册到 OpenShift Cluster Manager。

确认 OpenShift Cluster Manager 清单正确后，可以由 Telemetry 自动维护，也可以使用 OpenShift Cluster Manager 手动维护，使用订阅监控来跟踪帐户或多集群级别的 OpenShift Container Platform 订阅。

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于 远程健康监控

24.1.18. 后续步骤

- 自定义集群。
- 如果需要，您可以选择 不使用远程健康报告。

- 设置 registry 并配置 registry 存储。

第 25 章 安装配置

25.1. 自定义节点

OpenShift Container Platform 通过 Ignition 支持集群范围和每机器配置，允许对操作系统进行任意分区和文件内容更改。通常，如果在 Red Hat Enterprise Linux (RHEL) 中记录了配置文件，则支持通过 Ignition 修改它。

部署机器配置更改的方法有两种：

- 创建包含在清单文件中的机器配置，以便在 openshift-install 期间启动集群。
- 创建通过 Machine Config Operator 传递给运行的 OpenShift Container Platform 节点的机器配置。

另外，修改引用配置，比如在安装裸机节点时传递给 coreos-installer 的 Ignition 配置允许每个机器配置。这些更改目前对 Machine Config Operator 不可见。

以下小节描述了您可能需要以这种方式在节点上配置的功能。

25.1.1. 使用 Butane 创建机器配置

机器配置用于配置 control plane 和 worker 机器，方法是指示机器如何创建用户和文件系统、设置网络、安装 systemd 单元等。

因为修改机器配置可能比较困难，所以您可以使用 Butane 配置为您创建机器配置，从而使节点配置更容易。

25.1.1.1. 关于 Butane

但ane 是一个命令行实用程序，OpenShift Container Platform 使用它为编写机器配置提供便捷的简写语法，并对机器配置进行额外的验证。Butane 接受的 Butane 配置文件的格式在 OpenShift Butane 配置规格中定义。

25.1.1.2. 安装 Butane

您可以安装 Butane 工具（但ane），以便使用命令行界面创建 OpenShift Container Platform 机器配置。您可以通过下载对应的二进制文件，在 Linux、Windows 或 macOS 上安装但可 正常工作。

提示

但ane 版本与旧版本以及 Fedora CoreOS Config Transpiler(FCCT)向后兼容。

流程

1. 导航到位于 https://mirror.openshift.com/pub/openshift-v4/clients/butane/ 的 Butane 映像下载页面。

2. 获取 withane 二进制文件：

   a. 对于 Butane 的最新版本，请将最新的 但ane 镜像保存到当前目录中：

   ```
   $ curl https://mirror.openshift.com/pub/openshift-v4/clients/butane/latest/butane --output butane
   ```

   b. 可选：对于您要在其中安装的特定类型的架构，如 aarch64 或 ppc64le，代表适当的 URL。例如：

   ```
   $ curl https://mirror.openshift.com/pub/openshift-v4/clients/butane/latest/butane-aarch64 --output butane
   ```

3. 使下载的二进制文件可执行：

   ```
   $ chmod +x butane
   ```

4. 将 -ane 二进制文件移到 PATH 上的目录中。

   要查看您的 PATH，请打开终端并执行以下命令：

   ```
   $ echo $PATH
   ```

验证步骤

- 现在，您可以通过运行 butane 命令使用 But ane 工具：

```
$ butane <butane_file>
```

25.1.1.3. 使用 Butane 创建 MachineConfig 对象

您可以使用 Butane 生成 MachineConfig 对象，以便在安装时或通过 Machine Config Operator 配置 worker 或 control plane 节点。

先决条件

- 您已安装了 with ane 实用程序。

流程

1. 创建一个 Butane 配置文件。以下示例创建一个名为 99-worker-custom.bu 的文件，该文件将系统控制台配置为显示内核调试信息并为 chrony 时间服务指定自定义设置：

```
variant: openshift
version: 4.16.0
metadata:
  name: 99-worker-custom
  labels:
    machineconfiguration.openshift.io/role: worker
openshift:
  kernel_arguments:
    - loglevel=7
storage:
  files:
    - path: /etc/chrony.conf
      mode: 0644
      overwrite: true
      contents:
        inline: |
          pool 0.rhel.pool.ntp.org iburst
          driftfile /var/lib/chrony/drift
          makestep 1.0 3
          rtcsync
          logdir /var/log/chrony
```

注意

99-worker-custom.bu 文件设置为为 worker 节点创建机器配置。要在 control plane 节点上部署，请将角色从 worker 改为 master。要进行这两个操作，您可以在两种部署中使用不同的文件名来重复整个过程。

2. **通过提供您在上一步中创建的文件，创建 MachineConfig 对象：**

```
$ butane 99-worker-custom.bu -o ./99-worker-custom.yaml
```

已为您创建一个 MachineConfig 对象 YAML 文件，以完成机器的配置。

3. **如果将来需要更新 MachineConfig 对象，请保存 Butane 配置。**

4. **如果集群还没有运行，生成清单文件并将 MachineConfig 对象 YAML 文件添加到 openshift 目录中。如果集群已在运行，按如下所示应用该文件：**

```
$ oc create -f 99-worker-custom.yaml
```

其他资源

- **在节点中添加内核模块**
- **在安装过程中加密和镜像磁盘**

25.1.2. 添加 day-1 内核参数

虽然修改内核参数通常应做为第 2 天的任务，但您可能希望在初始集群安装过程中将内核参数添加到所有 master 节点或 worker 节点中。以下是您可能希望在集群安装过程中添加内核参数以便在系统第一次引导前生效的一些原因：

- **您需要在系统启动前进行一些低级网络配置。**
- **您希望禁用某个功能，如 SELinux，因此在系统首次启动时不会受到影响。**

警告

不支持在生产环境中禁用 RHCOS 上的 SELinux。在节点上禁用 SELinux 后，必须在生产集群中重新设置前重新置备它。

要在 master 节点或 worker 节点中添加内核参数，您可以创建一个 MachineConfig 对象，并将该对象注入 Ignition 在集群设置过程中使用的清单文件集合中。

有关您可以在引导时传递给 RHEL 8 内核的参数列表，请参阅 Kernel.org 内核参数。如果需要参数来完成初始 OpenShift Container Platform 安装，最好使用此流程添加内核参数。

流程

1. 进入包含安装程序的目录，并为集群生成 Kubernetes 清单：

```
$ ./openshift-install create manifests --dir <installation_directory>
```

2. 决定您要将内核参数添加到 worker 节点还是 control plane 节点。

3. 在 openshift 目录中，创建一个文件（例如： 99-openshift-machineconfig-master-kargs.yaml）来定义 MachineConfig 对象以添加内核设置。这个示例在 control plane 节点中添加了一个 loglevel=7 内核参数：

```
$ cat << EOF > 99-openshift-machineconfig-master-kargs.yaml
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: master
  name: 99-openshift-machineconfig-master-kargs
spec:
  kernelArguments:
    - loglevel=7
EOF
```

您可以将 master 改为 worker，将内核参数添加到 worker 节点。创建单独的 YAML 文件以添加到 master 和 worker 节点。

现在，您可以继续创建集群。

25.1.3. 在节点中添加内核模块

对于大多数常见硬件，Linux 内核包含了计算机启动时使用该硬件所需的设备驱动程序模块。然而，对于某些硬件，Linux 中无法使用模块。因此，您必须找到一种方法来为每个主机计算机提供这些模块。此流程描述了如何为 OpenShift Container Platform 集群中的节点执行此操作。

当首先按照这些说明部署内核模块时，该模块将提供给当前内核。如果安装了新内核，kmods-via-containers 软件将重建并部署该模块，以便新内核可以使用该模块的兼容版本。

使这个功能能够在每个节点中保持模块最新的方法是：

- 在引导时启动的每个节点中添加 systemd 服务，以检测是否安装了新内核。
- 如果检测到新内核，该服务会重建该模块并将其安装到内核中

有关此流程所需软件的详情，请查看 kmods-via-containers github 站点。

需要记住的几个重要问题：

- 这个过程是技术预览。
- 软件工具和示例还没有官方的 RPM，现只能从非官方的 github.com 站点获得。
- 红帽不支持您通过这些步骤添加的第三方内核模块。
- 在此过程中，构建内核模块所需的软件部署在 RHEL 8 容器中。请记住，当节点有新内核时，每个节点上会自动重新构建模块。因此，每个节点都需要访问 yum 存储库，该存储库包含重建该模块所需的内核和相关软件包。该内容最好由有效的 RHEL 订阅提供。

25.1.3.1. 构建和测试内核模块容器

在将内核模块部署到 OpenShift Container Platform 集群之前，您可以在单独的 RHEL 系统上测试该过程。收集内核模块的源代码、KVC 框架和 kmod-via-containers 软件。然后构建并测试模块。要在 RHEL 8 系统中做到这一点，请执行以下操作：

流程

1. 注册 RHEL 8 系统：

```
# subscription-manager register
```

2. 为 RHEL 8 系统附加订阅：

```
# subscription-manager attach --auto
```

3. 安装构建软件和容器所需的软件：

```
# yum install podman make git -y
```

4. 克隆 kmod-via-containers 存储库：

   a. 为存储库创建一个文件夹：

   ```
   $ mkdir kmods; cd kmods
   ```

   b. 克隆存储库：

   ```
   $ git clone https://github.com/kmods-via-containers/kmods-via-containers
   ```

5. 在 RHEL 8 构建主机上安装 KVC 框架实例来测试模块。这会添加 kmods-via-container systemd 服务并加载它：

   a. 进入 kmod-via-containers 目录：

   ```
   $ cd kmods-via-containers/
   ```

b. 安装 KVC 框架实例：

```
$ sudo make install
```

c. 重新载入 systemd Manager 配置：

```
$ sudo systemctl daemon-reload
```

6. 获取内核模块源代码。源代码可用于构建您无法控制但由其他人提供的第三方模块。您需要类似 kvc-simple-kmod 示例中显示的内容，该示例可克隆到您的系统中，如下所示：

```
$ cd .. ; git clone https://github.com/kmods-via-containers/kvc-simple-kmod
```

7. 编辑本例中的配置文件 simple-kmod.conf，并将 Dockerfile 的名称改为 Dockerfile.rhel：

a. 进入 kvc-simple-kmod 目录：

```
$ cd kvc-simple-kmod
```

b. 重命名 Dockerfile:

```
$ cat simple-kmod.conf
```

Dockerfile 示例

```
KMOD_CONTAINER_BUILD_CONTEXT="https://github.com/kmods-via-containers/kvc-simple-kmod.git"
KMOD_CONTAINER_BUILD_FILE=Dockerfile.rhel
KMOD_SOFTWARE_VERSION=dd1a7d4
KMOD_NAMES="simple-kmod simple-procfs-kmod"
```

8. 为您的内核模块创建一个 kmods-via-containers@.service 实例，本例中为 simple-kmod：

```
$ sudo make install
```

9. 启用 kmods-via-containers@.service 实例：

```
$ sudo kmods-via-containers build simple-kmod $(uname -r)
```

10. 启用并启动 systemd 服务：

```
$ sudo systemctl enable kmods-via-containers@simple-kmod.service --now
```

a. 查看服务状态：

```
$ sudo systemctl status kmods-via-containers@simple-kmod.service
```

输出示例

```
● kmods-via-containers@simple-kmod.service - Kmods Via Containers - simple-
kmod
  Loaded: loaded (/etc/systemd/system/kmods-via-containers@.service;
      enabled; vendor preset: disabled)
  Active: active (exited) since Sun 2020-01-12 23:49:49 EST; 5s ago...
```

11. 要确认载入了内核模块，使用 lsmod 命令列出模块：

```
$ lsmod | grep simple_
```

输出示例

```
simple_procfs_kmod     16384  0
simple_kmod            16384  0
```

12. 可选。使用其他方法检查 simple-kmod 是否正常工作：

- 使用 dmesg 在内核环缓冲中查找 "Hello world" 信息：

```
$ dmesg | grep 'Hello world'
```

输出示例

```
[ 6420.761332] Hello world from simple_kmod.
```

- 检查 /proc 中的 simple-procfs-kmod 值：

```
$ sudo cat /proc/simple-procfs-kmod
```

输出示例

```
simple-procfs-kmod number = 0
```

- 运行 spkut 命令从模块中获取更多信息：

```
$ sudo spkut 44
```

输出示例

```
KVC: wrapper simple-kmod for 4.18.0-147.3.1.el8_1.x86_64
Running userspace wrapper using the kernel module container...
+ podman run -i --rm --privileged
  simple-kmod-dd1a7d4:4.18.0-147.3.1.el8_1.x86_64 spkut 44
simple-procfs-kmod number = 0
simple-procfs-kmod number = 44
```

下一步，当系统引导此服务时，将检查新内核是否在运行。如果有新的内核，该服务会构建内核模块的新版本，然后载入它。如果已经构建了该模块，它将只加载它。

25.1.3.2. 为 OpenShift Container Platform 置备内核模块

根据 OpenShift Container Platform 集群首次引导时是否必须存在内核模块，您可以使用以下两种方式之一设置内核模块部署：

- 在 集群安装时(day-1)置备内核模块 ：您可以通过一个 MachineConfig 对象创建内容，并通过包括一组清单文件来将其提供给 openshift-install。
- 通过 Machine Config Operator(day-2)置备内核模块： 如果您可以等到集群启动并运行后再添加内核模块，您可以通过 Machine Config Operator(MCO)部署内核模块软件。

在这两种情况下，每个节点都需要能够在检测到新内核时获取内核软件包和相关软件包。您可以通过几种方法设置每个节点来获取该内容。

- 为每个节点提供 RHEL 权利。
- 从现有 RHEL 主机获取 RHEL 权利，从 /etc/pki/entitlement 目录中获取，并将它们复制到与您构建 Ignition 配置时提供的其他文件相同的位置。
- 在 Dockerfile 中，添加指针到包含内核和其他软件包的 yum 存储库。这必须包括新内核包，因为它们需要与新安装的内核相匹配。

25.1.3.2.1. 通过 MachineConfig 对象置备内核模块

通过将内核模块软件与 MachineConfig 对象一起打包，您可以在安装时或通过 Machine Config Operator 向 worker 或 control plane 节点提供该软件。

流程

1. 注册 RHEL 8 系统：

```
# subscription-manager register
```

2. 为 RHEL 8 系统附加订阅：

```
# subscription-manager attach --auto
```

3. 安装构建软件所需的软件：

```
# yum install podman make git -y
```

4. 创建托管内核模块和工具的目录：

```
$ mkdir kmods; cd kmods
```

5. 获取 kmods-via-containers 软件：

   a. 克隆 kmods-via-containers 存储库：

   ```
   $ git clone https://github.com/kmods-via-containers/kmods-via-containers
   ```

   b. 克隆 kvc-simple-kmod 存储库：

   ```
   $ git clone https://github.com/kmods-via-containers/kvc-simple-kmod
   ```

6. 获取您的模块软件。本例中使用 kvc-simple-kmod。

7. 使用之前克隆的存储库，创建一个 fakeroot 目录，并在其中填充您要通过 Ignition 提供的文件：

   a. 创建目录：

   ```
   $ FAKEROOT=$(mktemp -d)
   ```

b. 进入 kmod-via-containers 目录：

```
$ cd kmods-via-containers
```

c. 安装 KVC 框架实例：

```
$ make install DESTDIR=${FAKEROOT}/usr/local CONFDIR=${FAKEROOT}/etc/
```

d. 进入 kvc-simple-kmod 目录：

```
$ cd ../kvc-simple-kmod
```

e. 创建实例：

```
$ make install DESTDIR=${FAKEROOT}/usr/local CONFDIR=${FAKEROOT}/etc/
```

8. 运行以下命令，克隆 fakeroot 目录，将任何符号链接替换为目标副本：

```
$ cd .. && rm -rf kmod-tree && cp -Lpr ${FAKEROOT} kmod-tree
```

9. 创建一个 Butane 配置文件 99-simple-kmod.bu，它嵌入内核模块树并启用 systemd 服务。

注意

如需有关 Butane 的信息，请参阅"使用 Butane 创建机器配置"。

```
variant: openshift
version: 4.16.0
metadata:
  name: 99-simple-kmod
  labels:
    machineconfiguration.openshift.io/role: worker (1)
storage:
  trees:
    - local: kmod-tree
systemd:
```

```
units:
 - name: kmods-via-containers@simple-kmod.service
   enabled: true
```

1

要在 control plane 节点上部署，请将 worker 改为 master。要在 control plane 和 worker 节点上部署，请对每个节点类型执行一次这些指令的其余部分。

10. 使用 Butane 生成机器配置 YAML 文件 99-simple-kmod.yaml，其中包含要交付的文件和配置：

```
$ butane 99-simple-kmod.bu --files-dir . -o 99-simple-kmod.yaml
```

11. 如果集群还没有启动，生成清单文件并将该文件添加到 openshift 目录中。如果集群已在运行，按如下所示应用该文件：

```
$ oc create -f 99-simple-kmod.yaml
```

您的节点将启动 kmods-via-containers@simple-kmod.service 服务，并将载入内核模块。

12. 要确认内核模块已加载，您可以登录到节点（使用 oc debug node/<openshift-node>，然后 chroot /host）。要列出模块，请使用 lsmod 命令：

```
$ lsmod | grep simple_
```

输出示例

```
simple_procfs_kmod     16384  0
simple_kmod            16384  0
```

25.1.4. 在安装过程中加密和镜像磁盘

在 OpenShift Container Platform 安装过程中，您可以在集群节点上启用引导磁盘加密和镜像功能。

25.1.4.1. 关于磁盘加密

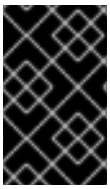
您可以在安装时在 control plane 和计算节点上为引导磁盘启用加密。OpenShift Container Platform 支持 Trusted Platform 模块(TPM)v2 和 Tang 加密模式。

TPM v2

这是首选模式。TPM v2 将密码短语存储在服务器的安全加密处理器中。如果从服务器中删除磁盘，您可以使用此模式来防止在集群节点上解密引导磁盘数据。

tang

Tang 和 Clevis 是启用网络绑定磁盘加密(NBDE)的服务器和客户端组件。您可以将集群节点中的引导磁盘数据绑定到一个或多个 Tang 服务器。这会防止解密数据，除非节点位于可访问 Tang 服务器的安全网络中。Clevis 是一种自动化解密框架，用于在客户端中实施解密。



重要

使用 Tang 加密模式加密磁盘只支持在用户置备的基础架构上安装裸机和 vSphere。

在以前的 Red Hat Enterprise Linux CoreOS(RHCOS)版本中，磁盘加密是通过在 Ignition 配置中指定 /etc/clevis.json 来配置的。使用 OpenShift Container Platform 4.7 或更高版本创建的集群不支持该文件。使用以下步骤配置磁盘加密。

启用 TPM v2 或 Tang 加密模式时，RHCOS 引导磁盘将使用 LUKS2 格式进行加密。

这个功能：

- 可用于安装程序置备的基础架构、用户置备的基础架构和辅助安装程序部署
- 对于辅助安装程序部署：
  - 每个集群只能有一个加密方法 Tang 或 TPM
  - 加密可以在某些或所有节点上启用

- 没有 Tang 阈值；所有服务器都必须有效且可操作
- 加密只适用于安装磁盘，不适用于工作负载磁盘

- 只在 Red Hat Enterprise Linux CoreOS(RHCOS)系统上支持
- 在清单安装过程中设置磁盘加密，加密写入磁盘的所有数据
- 不需要用户干预来提供密码短语
- 如果启用了 FIPS 模式，则使用 AES-256-XTS 加密或 AES-256-CBC

25.1.4.1.1. 配置加密阈值

在 OpenShift Container Platform 中，您可以指定多个 Tang 服务器的要求。您还可以同时配置 TPM v2 和 Tang 加密模式。这只有在存在 TPM 安全加密处理器且 Tang 服务器可以通过安全网络访问时启用引导磁盘数据解密。

您可以使用 Butane 配置中的 threshold 属性来定义解密所需的 TPM v2 和 Tang 加密条件的最小数量。

通过声明的条件的任意组合达到声明的值时，会满足阈值。如果是离线调配，使用包含的公告访问离线服务器，并且仅在在线服务器数量没有满足集合阈值时使用该广告。

例如，在以下的配置中，threshold 的值为 2，这在访问带有可用的离线服务器作为备份的两个 Tang 服务器时就达到了；或访问 TPM 安全加密处理器和一个 Tang 服务器达到：

用于磁盘加密的 Butane 配置示例

```
variant: openshift
version: 4.16.0
metadata:
  name: worker-storage
  labels:
```

```
    machineconfiguration.openshift.io/role: worker
boot_device:
  layout: x86_64 (1)
  luks:
    tpm2: true (2)
    tang: (3)
      - url: http://tang1.example.com:7500
        thumbprint: jwGN5tRFK-kF6pIX89ssF3khxxX
      - url: http://tang2.example.com:7500
        thumbprint: VCJsvZFjBSIHSldw78rOrq7h2ZF
      - url: http://tang3.example.com:7500
        thumbprint: PLjNyRdGw03zlRoGjQYMahSZGu9
        advertisement: "{\"payload\": \"...\", \"protected\": \"...\", \"signature\": \"...\"}" (4)
    threshold: 2 (5)
openshift:
  fips: true
```

(1)

将此字段设置为集群节点的指令集合架构。一些示例包括、x86_64、aarch64 或 ppc64le。

(2)

如果要使用受信任的平台模块(TPM)加密根文件系统，请包含此字段。

(3)

如果要使用一个或多个 Tang 服务器，请包含此部分。

(4)

可选：包含用于离线置备的此字段。Ignition 将置备 Tang 服务器绑定，而不是在运行时从服务器获取公告。这样，服务器在置备时不可用。

(5)

指定进行解密所需的最小 TPM v2 和 Tang 加密条件。

重要

默认 阈值 为 1。如果您在配置中包含多个加密条件，但没有指定阈值，则会在满足任何条件时进行解密。

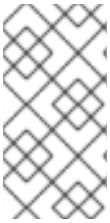
注意

如果需要 TPM v2 *和* Tang 进行，则 threshold 属性的值必须等于声明的 Tang 服务器总数再加一。如果阈值较低，可以使用单一加密模式达到阈值。例如，如果您将 tpm2 设置为 true 并指定两个 Tang 服务器，则可以通过访问两个 Tang 服务器来满足阈值 2，即使 TPM 安全加密处理器不可用。

25.1.4.2. 关于磁盘镜像

在 control plane 和 worker 节点上安装 OpenShift Container Platform 时，您可以将引导和其他磁盘镜像到两个或者多个冗余存储设备。存储设备失败后节点将继续正常工作，提供一个设备仍然可用。

镜像不支持替换失败的磁盘。重新置备节点，将镜像恢复到正常的非降级状态。



注意

对于用户置备的基础架构部署，镜像只在 RHCOS 系统上可用。使用 BIOS 或 UEFI 和 ppc64le 节点上引导的 x86_64 节点上支持镜像。

25.1.4.3. 配置磁盘加密和镜像

您可以在 OpenShift Container Platform 安装过程中启用并配置加密和镜像功能。

先决条件

- 您已在安装节点上下载了 OpenShift Container Platform 安装程序。
- 在安装节点上安装了 Butane。

注意

Butane 是一个命令行实用程序，OpenShift Container Platform 用来为编写和验证机器配置提供方便的简短语法。如需更多信息，请参阅"使用 Butane 创建机器配置"。

- 您可以使用 Red Hat Enterprise Linux(RHEL)8 机器来生成 Tang Exchange 密钥的指纹。

流程

1. 如果要使用 TPM v2 加密集群，请检查每个节点的主机固件中是否需要启用 TPM v2 加密。这在大多数 Dell 系统中是必需的。检查具体系统的手册。

2. 如果要使用 Tang 加密集群，请按照以下步骤操作：

   a. 设置 Tang 服务器或访问现有服务器。具体步骤请查看 网络绑定磁盘加密。

   b. 如果尚未安装，在 RHEL 8 机器上安装 clevis 软件包：

```
$ sudo yum install clevis
```

   c. 在 RHEL 8 计算机上，运行以下命令来生成交换密钥的指纹。使用 Tang 服务器的 URL 替换 http://tang1.example.com:7500

```
$ clevis-encrypt-tang '{"url":"http://tang1.example.com:7500"}' < /dev/null > /dev/null 1
```

1. 在本例中，tang d.socket 正在侦听 Tang 服务器上的端口 7500。

注意

clevis-encrypt-tang 命令生成交换密钥的指纹。此步骤中不会将数据传递给加密命令；/dev/null 作为输入而不是纯文本存在。加密的输出也会发送到 /dev/null，因为此过程不需要它。

输出示例

```
The advertisement contains the following signing keys:

PLjNyRdGw03zlRoGjQYMahSZGu9 1
```

1 Exchange 键的指纹。

当 Do 您希望信任这些密钥时，显示 [ynYN] 提示时，键入 Y。

d. 可选：对于离线 Tang 置备：

i. 使用 curl 命令从服务器获取公告。使用 Tang 服务器的 URL 替换 http://tang2.example.com:7500 ：

```
$ curl -f http://tang2.example.com:7500/adv > adv.jws && cat adv.jws
```

预期输出

```
{"payload": "eyJrZXlzIjogW3siYWxnIjogIkV",
"eyJhbGciOiJFUzUxMiIsImN0eSI",
"ADLgk7fZdE3Yt4FyYsm0pHiau7Q"}
```

ii. 为 Clevis 提供广告文件进行加密：

```
$ clevis-encrypt-tang '{"url":"http://tang2.example.com:7500","adv":"adv.jws"}'
< /dev/null > /dev/null
```

e. 如果节点配置了静态 IP 寻址，请运行 coreos-installer iso custom --dest-karg-append 或者在安装 RHCOS 节点时使用 coreos-installer --append-karg 选项来设置已安装系统的 IP 地址。为您的 网络附加 ip= 和其他参数。

重要

有些配置静态 IP 的方法在第一次引导后不会影响 initramfs，且不适用于 Tang 加密。这包括 coreos-installer --copy-network 选项、coreos-installer iso customize --network-keyfile 选项和 coreos-installer pxe customize --network-keyfile 选项，以及在安装过程中在 live ISO 或 PXE 镜像的内核命令行中添加 ip= 参数。静态 IP 配置不正确会导致节点第二次引导失败。

3. 在安装节点上，切换到包含安装程序的目录，并为集群生成 Kubernetes 清单：

```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 将 <installation_directory> 替换为您要存储安装文件的目录的路径。

4. 创建一个 Butane 配置来配置磁盘加密、镜像或两者。例如，若要为计算节点配置存储，请创建一个 $HOME/clusterconfig/worker-storage.bu 文件。

引导设备的ane 配置示例

```
variant: openshift
version: 4.16.0
metadata:
  name: worker-storage 1
  labels:
    machineconfiguration.openshift.io/role: worker 2
boot_device:
  layout: x86_64 3
  luks: 4
    tpm2: true 5
    tang: 6
      - url: http://tang1.example.com:7500 7
        thumbprint: PLjNyRdGw03zlRoGjQYMahSZGu9 8
      - url: http://tang2.example.com:7500
        thumbprint: VCJsvZFjBSIHSldw78rOrq7h2ZF
        advertisement: "{"payload": "eyJrZXlzIjogW3siYWxnIjogIkV", "protected":
"eyJhbGciOiJFUzUxMiIsImN0eSI", "signature": "ADLgk7fZdE3Yt4FyYsm0pHiau7Q"}"
9
    threshold: 1 10
  mirror: 11
    devices: 12
```

```
    - /dev/sda
    - /dev/sdb
openshift:
 fips: true 13
```

1 2

对于 control plane 配置，在这两个位置中将 worker 替换为 master。

3

将此字段设置为集群节点的指令集合架构。一些示例包括、x86_64、aarch64 或 ppc64le。

4

如果要加密 root 文件系统，请包含此部分。如需了解更多详细信息，请参阅"关于磁盘加密"。

5

如果要使用受信任的平台模块(TPM)加密根文件系统，请包含此字段。

6

如果要使用一个或多个 Tang 服务器，请包含此部分。

7

指定 Tang 服务器的 URL。在本例中，tang d.socket 正在侦听 Tang 服务器上的端口 7500。

8

指定上一步中生成的 Exchange key thumbprint。

9

可选：以有效 JSON 格式指定离线 Tang 服务器的公告。

10

指定进行解密时必须满足的最小 TPM v2 和 Tang 加密条件。默认值为 1。有关此主题的更多信息，请参阅"配置加密阈值"。

11 如果要镜像引导磁盘，请包含此部分。如需了解更多详细信息，请参阅"关于磁盘镜像"。

12 列出引导磁盘镜像中包含的所有磁盘设备，包括 RHCOS 将安装到的磁盘。

13 包含此指令以在集群中启用 FIPS 模式。

重要

要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 Red Hat Enterprise Linux (RHEL) 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅在 FIPS 模式中安装该系统。如果要将节点配置为同时使用磁盘加密和镜像，则必须在同一 Butane 配置文件中配置这两个功能。如果要在启用了 FIPS 模式的节点上配置磁盘加密，则必须在同一 Butane 配置文件中包含 fips 指令，即使在单独的清单中也启用了 FIPS 模式。

5. 从对应的 Butane 配置文件创建一个 control plane 或计算节点清单，并将它保存到 <installation_directory>/openshift 目录。例如，要为计算节点创建清单，请运行以下命令：

```
$ butane $HOME/clusterconfig/worker-storage.bu -o
<installation_directory>/openshift/99-worker-storage.yaml
```

对需要磁盘加密或镜像的每种节点类型重复此步骤。

6. 保存 Butane 配置文件，以防将来需要更新清单。

7. 继续进行 OpenShift Container Platform 安装的其余部分。

提示

您可以在安装过程中监控 RHCOS 节点上的控制台日志，以了解与磁盘加密或镜像相关的错误消息。

重要

如果您配置附加数据分区，除非明确请求加密，否则不会加密它们。

验证

安装 OpenShift Container Platform 后，您可以验证是否在集群节点上启用了引导磁盘加密或镜像功能。

1. **在安装主机上，使用 debug pod 访问集群节点：**

   a. 为节点启动 **debug pod，例如：**

   ```
   $ oc debug node/compute-1
   ```

   b. **将 /host 设置为 debug shell 中的根目录。debug pod 在 pod 中的 /host 中挂载节点的根文件系统。通过将根目录改为 /host，您可以运行节点上可执行路径中包含的二进制文件：**

   ```
   # chroot /host
   ```

   注意

   运行 **Red Hat Enterprise Linux CoreOS(RHCOS)的 OpenShift Container Platform 集群节点不可变，它依赖于 Operator 来应用集群更改。不建议使用 SSH 访问集群节点。但是，如果 OpenShift Container Platform API 不可用，或者 kubelet 在目标节点上无法正常工作，oc 操作将会受到影响。在这种情况下，可以使用 ssh core@<node>.<cluster_name>.<base_domain> 来访问节点。**

2. **如果配置了引导磁盘加密，请验证是否启用它：**

   a. **在 debug shell 中查看节点上 root 映射的状态：**

   ```
   # cryptsetup status root
   ```

   输出示例

```
/dev/mapper/root is active and is in use.
  type:    LUKS2 1
  cipher:  aes-xts-plain64 2
  keysize: 512 bits
  key location: keyring
  device:  /dev/sda4 3
  sector size:  512
  offset:  32768 sectors
  size:    15683456 sectors
  mode:    read/write
```

1

加密格式。启用 TPM v2 或 Tang 加密模式时，RHCOS 引导磁盘将使用 LUKS2 格式进行加密。

2

用于加密 LUKS2 卷的加密算法。如果启用了 FIPS 模式，则使用 aes-cbc-essiv:sha256 密码。

3

包含加密 LUKS2 卷的设备。如果启用了镜像，该值将表示软件镜像设备，如 /dev/md126。

b. 列出绑定到加密设备的 Clevis 插件：

```
# clevis luks list -d /dev/sda4 1
```

1

指定上一步中输出中 device 字段中列出的设备。

输出示例

```
1: sss '{"t":1,"pins":{"tang":[{"url":"http://tang.example.com:7500"}]}}' 1
```

1

在示例输出中，Tang 插件由 /dev/sda4 设备的 Shamir 的 Secret 共享 SSS) Clevis 插件使用。

3. 如果配置了镜像(mirror)，请验证是否启用它：

a. 在 debug shell 中列出节点上的软件 RAID 设备：

```
# cat /proc/mdstat
```

输出示例

```
Personalities : [raid1]
md126 : active raid1 sdb3[1] sda3[0] 1
  393152 blocks super 1.0 [2/2] [UU]

md127 : active raid1 sda4[0] sdb4[1] 2
  51869632 blocks super 1.2 [2/2] [UU]

unused devices: <none>
```

1

/dev/md126 软件 RAID 镜像设备使用集群节点中的 /dev/sda3 和 /dev/sdb3 磁盘设备。

2

/dev/md127 软件 RAID 镜像设备使用集群节点中的 /dev/sda4 和 /dev/sdb4 磁盘设备。

b. 查看上一命令输出中列出的每个软件 RAID 设备的详细信息。以下示例列出了 /dev/md126 设备详情：

```
# mdadm --detail /dev/md126
```

输出示例

```
/dev/md126:
           Version : 1.0
     Creation Time : Wed Jul  7 11:07:36 2021
        Raid Level : raid1 (1)
        Array Size : 393152 (383.94 MiB 402.59 MB)
     Used Dev Size : 393152 (383.94 MiB 402.59 MB)
      Raid Devices : 2
     Total Devices : 2
       Persistence : Superblock is persistent

       Update Time : Wed Jul  7 11:18:24 2021
             State : clean (2)
    Active Devices : 2 (3)
   Working Devices : 2 (4)
    Failed Devices : 0 (5)
     Spare Devices : 0

Consistency Policy : resync

              Name : any:md-boot (6)
              UUID : ccfa3801:c520e0b5:2bee2755:69043055
            Events : 19

    Number   Major   Minor   RaidDevice State
       0     252       3        0      active sync   /dev/sda3 (7)
       1     252      19        1      active sync   /dev/sdb3 (8)
```

(1)

指定设备的 RAID 级别。raid1 表示 RAID 1 磁盘镜像。

(2)

指定 RAID 设备的状态。

(3) (4)

指出活跃且正常工作的底层磁盘设备数量。

(5)

说明处于故障状态的底层磁盘设备数量。

6

软件 RAID 设备的名称。

7 8

提供有关软件 RAID 设备使用的底层磁盘设备的信息。

c.

列出软件 RAID 设备中挂载的文件系统：

```
# mount | grep /dev/md
```

输出示例

```
/dev/md127 on / type xfs
(rw,relatime,seclabel,attr2,inode64,logbufs=8,logbsize=32k,prjquota)
/dev/md127 on /etc type xfs
(rw,relatime,seclabel,attr2,inode64,logbufs=8,logbsize=32k,prjquota)
/dev/md127 on /usr type xfs
(ro,relatime,seclabel,attr2,inode64,logbufs=8,logbsize=32k,prjquota)
/dev/md127 on /sysroot type xfs
(ro,relatime,seclabel,attr2,inode64,logbufs=8,logbsize=32k,prjquota)
/dev/md127 on /var type xfs
(rw,relatime,seclabel,attr2,inode64,logbufs=8,logbsize=32k,prjquota)
/dev/md127 on /var/lib/containers/storage/overlay type xfs
(rw,relatime,seclabel,attr2,inode64,logbufs=8,logbsize=32k,prjquota)
/dev/md127 on /var/lib/kubelet/pods/e5054ed5-f882-4d14-b599-
99c050d4e0c0/volume-subpaths/etc/tuned/1 type xfs
(rw,relatime,seclabel,attr2,inode64,logbufs=8,logbsize=32k,prjquota)
/dev/md127 on /var/lib/kubelet/pods/e5054ed5-f882-4d14-b599-
99c050d4e0c0/volume-subpaths/etc/tuned/2 type xfs
(rw,relatime,seclabel,attr2,inode64,logbufs=8,logbsize=32k,prjquota)
/dev/md127 on /var/lib/kubelet/pods/e5054ed5-f882-4d14-b599-
99c050d4e0c0/volume-subpaths/etc/tuned/3 type xfs
(rw,relatime,seclabel,attr2,inode64,logbufs=8,logbsize=32k,prjquota)
/dev/md127 on /var/lib/kubelet/pods/e5054ed5-f882-4d14-b599-
99c050d4e0c0/volume-subpaths/etc/tuned/4 type xfs
(rw,relatime,seclabel,attr2,inode64,logbufs=8,logbsize=32k,prjquota)
/dev/md127 on /var/lib/kubelet/pods/e5054ed5-f882-4d14-b599-
99c050d4e0c0/volume-subpaths/etc/tuned/5 type xfs
(rw,relatime,seclabel,attr2,inode64,logbufs=8,logbsize=32k,prjquota)
/dev/md126 on /boot type ext4 (rw,relatime,seclabel)
```

在示例输出中，/boot 文件系统挂载到 /dev/md126 软件 RAID 设备上，root 文件系统挂载到 /dev/md127。

4. 对每种 OpenShift Container Platform 节点类型重复验证步骤。

其他资源

- 有关 TPM v2 和 Tang 加密模式的更多信息，请参阅使用 基于策略的解密配置加密卷的自动解锁。

25.1.4.4. 配置启用了 RAID 的数据卷

您可以启用软件 RAID 分区以提供外部数据卷。OpenShift Container Platform 支持 RAID 0、RAID 1、RAID 4、RAID 5、RAID 6 和 RAID 10 用于数据保护和容错。如需了解更多详细信息，请参阅"关于磁盘镜像"。

先决条件

- 您已在安装节点上下载了 OpenShift Container Platform 安装程序。
- 您已在安装节点上安装了 Butane。

注意

但ane 是一个命令行实用程序，OpenShift Container Platform 使用它为编写机器配置提供便捷的简写语法，并对机器配置进行额外的验证。如需更多信息，*请参阅使用 Butane 创建机器配置* 部分。

流程

1. 创建一个 Butane 配置，以使用软件 RAID 配置数据卷。

   - 要在用于镜像引导磁盘的相同磁盘上配置带有 RAID 1 的数据卷，请创建一个 $HOME/clusterconfig/raid1-storage.bu 文件，例如：

镜像引导磁盘上的 RAID 1

```
variant: openshift
version: 4.16.0
metadata:
  name: raid1-storage
  labels:
    machineconfiguration.openshift.io/role: worker
boot_device:
  mirror:
    devices:
      - /dev/disk/by-id/scsi-3600508b400105e210000900000490000
      - /dev/disk/by-id/scsi-SSEAGATE_ST373453LW_3HW1RHM6
storage:
  disks:
    - device: /dev/disk/by-id/scsi-3600508b400105e210000900000490000
      partitions:
        - label: root-1
          size_mib: 25000 (1)
        - label: var-1
    - device: /dev/disk/by-id/scsi-SSEAGATE_ST373453LW_3HW1RHM6
      partitions:
        - label: root-2
          size_mib: 25000 (2)
        - label: var-2
  raid:
    - name: md-var
      level: raid1
      devices:
        - /dev/disk/by-partlabel/var-1
        - /dev/disk/by-partlabel/var-2
  filesystems:
    - device: /dev/md/md-var
      path: /var
      format: xfs
      wipe_filesystem: true
      with_mount_unit: true
```

(1) (2)

当在引导磁盘中添加数据分区时，推荐最少使用 25000 MB。如果没有指定值，或者指定的值小于推荐的最小值，则生成的 root 文件系统会太小，而在以后进行的 RHCOS 重新安装可能会覆盖数据分区的开始部分。

-

要在辅助磁盘上配置带有 RAID 1 的数据卷，请创建一个 $HOME/clusterconfig/raid1-alt-storage.bu 文件，例如：

辅助磁盘上的 RAID 1

```
variant: openshift
version: 4.16.0
metadata:
  name: raid1-alt-storage
  labels:
    machineconfiguration.openshift.io/role: worker
storage:
  disks:
    - device: /dev/sdc
      wipe_table: true
      partitions:
        - label: data-1
    - device: /dev/sdd
      wipe_table: true
      partitions:
        - label: data-2
  raid:
    - name: md-var-lib-containers
      level: raid1
      devices:
        - /dev/disk/by-partlabel/data-1
        - /dev/disk/by-partlabel/data-2
  filesystems:
    - device: /dev/md/md-var-lib-containers
      path: /var/lib/containers
      format: xfs
      wipe_filesystem: true
      with_mount_unit: true
```

2. 从您在上一步中创建的 Butane 配置创建 RAID 清单，并将它保存到 <installation_directory>/openshift 目录中。例如，要为计算节点创建清单，请运行以下命令：

```
$ butane $HOME/clusterconfig/<butane_config>.bu -o
<installation_directory>/openshift/<manifest_name>.yaml 1
```

1 将 <butane_config> 和 <manifest_name> 替换为上一步中的文件名。例如，对于辅助磁盘，raid1-alt-storage.bu 和 raid1-alt-storage.yaml。

3. 保存 Butane 配置，以防将来需要更新清单。

4. 继续进行 OpenShift Container Platform 安装的其余部分。

25.1.4.5. 在 CPU (VROC) 数据卷中配置 Intel® 虚拟 RAID

Intel® VROC 是混合 RAID 类型，其中有些维护被卸载到硬件，但显示为软件 RAID 到操作系统。

以下流程配置启用了 Intel® VROC 的 RAID1。

先决条件

- 您有一个启用了 Intel® Volume Management Device (VMD) 的系统。

流程

1. 运行以下命令来创建 Intel® Matrix Storage Manager (IMSM) RAID 容器：

```
$ mdadm -CR /dev/md/imsm0 -e \
 imsm -n2 /dev/nvme0n1 /dev/nvme1n1 (1)
```

(1) RAID 设备名称。本例中列出了两个设备。如果提供多个设备名称，您需要调整 -n 标志。例如，列出三个使用 -n3 的设备。

2. 在容器内创建 RAID1 存储：

   a. 运行以下命令，在实际 RAID1 卷前面创建一个 dummy RAID0 卷：

```
$ mdadm -CR /dev/md/dummy -l0 -n2 /dev/imsm0 -z10m --assume-clean
```

   b. 运行以下命令来创建实际的 RAID1 阵列：

```
$ mdadm -CR /dev/md/coreos -l1 -n2 /dev/imsm0
```

c. 停止 RAID0 和 RAID1 成员阵列，使用以下命令删除 dummy RAID0 阵列：

```
$ mdadm -S /dev/md/dummy \
  mdadm -S /dev/md/coreos \
  mdadm --kill-subarray=0 /dev/md/imsm0
```

d. 运行以下命令重启 RAID1 阵列：

```
$ mdadm -A /dev/md/coreos /dev/md/imsm0
```

3. 在 RAID1 设备上安装 RHCOS：

a. 运行以下命令，获取 IMSM 容器的 UUID：

```
$ mdadm --details --export /dev/md/imsm0
```

b. 运行以下命令安装 RHCOS 并包含 rd.md.uuid 内核参数：

```
$ coreos-installer install /dev/md/coreos \
  --append-karg rd.md.uuid=<md_UUID> 1
  ...
```

1 IMSM 容器的 UUID。

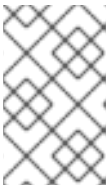
包括安装 RHCOS 所需的任何其他 coreos-installer 参数。

25.1.5. 配置 chrony 时间服务

您可以通过修改 chrony .conf 文件的内容，并将这些内容作为机器配置传递给节点，从而设置 chrony 时间服务(chronyd)使用的时间服务器和相关设置。

流程

1. 创建一个 Butane 配置，包括 chrony.conf 文件的内容。例如，要在 worker 节点上配置 chrony，请创建一个 99-worker-chrony.bu 文件。



注意

如需有关 Butane 的信息，请参阅"使用 Butane 创建机器配置"。

```
variant: openshift
version: 4.16.0
metadata:
  name: 99-worker-chrony (1)
  labels:
    machineconfiguration.openshift.io/role: worker (2)
storage:
  files:
  - path: /etc/chrony.conf
    mode: 0644 (3)
    overwrite: true
    contents:
      inline: |
        pool 0.rhel.pool.ntp.org iburst (4)
        driftfile /var/lib/chrony/drift
        makestep 1.0 3
        rtcsync
        logdir /var/log/chrony
```

(1) (2) 在 control plane 节点上，在这两个位置中将 master 替换为 worker。

(3) 为机器配置文件的 mode 字段指定数值模式。在创建文件并应用更改后，模式 将转换为十进制值。您可以使用 oc get mc <mc-name> -o yaml 命令来检查 YAML 文件。

(4) 指定任何有效的、可访问的时间源，如 DHCP 服务器提供的源。或者，您可以指定以下 NTP 服务器：1.rhel.pool.ntp.org, 2.rhel.pool.ntp.org, 或 3.rhel.pool.ntp.org。

2. 使用 Butane 生成 MachineConfig 对象文件 99-worker-chrony.yaml，其中包含要交付至节点的配置：

```
$ butane 99-worker-chrony.bu -o 99-worker-chrony.yaml
```

3. 使用以下两种方式之一应用配置：

- 如果集群还没有运行，在生成清单文件后，将 MachineConfig 对象文件添加到 <installation_directory>/openshift 目录中，然后继续创建集群。
- 如果集群已在运行，请应用该文件：

```
$ oc apply -f ./99-worker-chrony.yaml
```

25.1.6. 其他资源

- 有关 Butane 的详情，请参考使用 Butane 创建机器配置。
- 有关 FIPS 支持的详情，请参考对 FIPS 加密的支持。

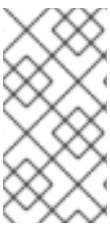
25.2. 配置防火墙

如果使用防火墙，您必须进行配置，以便 OpenShift Container Platform 可以访问正常工作所需的站点。您必须始终授予某些站点的访问权限，如果使用 Red Hat Insights、Telemetry 服务、托管集群的云以及某些构建策略，则还要授予更多站点的访问权限。

25.2.1. 为 OpenShift Container Platform 配置防火墙

在安装 OpenShift Container Platform 前，您必须配置防火墙，以授予 OpenShift Container Platform 所需站点的访问权限。在使用防火墙时，为防火墙提供额外的配置，以便 OpenShift Container Platform 可以访问正常工作所需的站点。

与 worker 节点相比，仅在控制器节点上运行的服务没有特殊的配置注意事项。



注意

如果您的环境在 OpenShift Container Platform 集群前面有一个专用的负载均衡器，请查看防火墙和负载均衡器之间的允许列表，以防止对集群造成不必要的网络限制。

流程

1. 为您的防火墙的允许列表设置以下 registry URL：

URL	port	功能
registry.redhat.io	443	提供核心容器镜像
access.redhat.com [1]	443	托管存储在 Red Hat Ecosytem Catalog 中的所有容器镜像，包括核心容器镜像。
quay.io	443	提供核心容器镜像
cdn.quay.io	443	提供核心容器镜像
cdn01.quay.io	443	提供核心容器镜像
cdn02.quay.io	443	提供核心容器镜像
cdn03.quay.io	443	提供核心容器镜像
cdn04.quay.io	443	提供核心容器镜像
cdn05.quay.io	443	提供核心容器镜像
cdn06.quay.io	443	提供核心容器镜像
sso.redhat.com	443	**https://console.redhat.com** 站点使用来自 **sso.redhat.com** 的身份验证

1. 在防火墙环境中，确保 access.redhat.com 资源位于允许列表中。此资源托管容器客户端在从 registry.access.redhat.com 中拉取镜像时验证镜像所需的签名存储。

您可以在 allowlist 中使用通配符 lfquay.io 和 lfopenshiftapps.com 而不是 cdn.quay.io 和 cdn0[1-3].quay.io。在 allowlist 中添加站点（如 quay.io ）时，不要向 denylist 添加通配符条目，如 *.quay.io。在大多数情况下，镜像 registry 使用内容交付网络（CDN）来提供镜像。如果防火墙阻止访问，则初始下载请求重定向到一个主机名（如 cdn01.quay.io ）时，镜像下载将被拒绝。

2. 将防火墙的允许列表设置为包含为构建所需的语言或框架提供资源的任何站点。

3. 如果不禁用 Telemetry，您必须授予对以下 URL 的访问权限，以访问 Red Hat Insights：

URL	port	功能
cert-api.access.redhat.com	443	Telemetry 所需
api.access.redhat.com	443	Telemetry 所需
infogw.api.openshift.com	443	Telemetry 所需
console.redhat.com	443	Telemetry 和 **insights-operator**需要

4. **如果使用 Alibaba Cloud、Amazon Web Services (AWS)、Microsoft Azure 或 Google Cloud Platform (GCP) 来托管您的集群，您必须授予对为该云提供云供应商 API 和 DNS 的 URL 的访问权限：**

云	URL	port	功能
Alibaba	***.aliyuncs.com**	443	需要此项以访问 Alibaba Cloud 服务和资源。查看 Alibaba endpoint_config.go 文件，以查找您使用的区域所允许的确切端点。
AWS	**aws.amazon.com**	443	用于在 AWS 环境中安装和管理集群。
	***.amazonaws.com** 另外，如果您选择不对 AWS API 使用通配符，则必须在允许列表中包含以下 URL：	443	需要此项以访问 AWS 服务和资源。请参阅 AWS 文档中的 AWS Service Endpoints，以查找您使用的区域所允许的确切端点。
	ec2.amazonaws.com	443	用于在 AWS 环境中安装和管理集群。
	events.amazonaws.com	443	用于在 AWS 环境中安装和管理集群。
	iam.amazonaws.com	443	用于在 AWS 环境中安装和管理集群。
	route53.amazonaws.com	443	用于在 AWS 环境中安装和管理集群。
	***.s3.amazonaws.com**	443	用于在 AWS 环境中安装和管理集群。
	***.s3.<aws_region>.amazonaws.com**	443	用于在 AWS 环境中安装和管理集群。
	***.s3.dualstack.<aws_region>.amazonaws.com**	443	用于在 AWS 环境中安装和管理集群。

云	URL	port	功能
	sts.amazonaws.com	443	用于在 AWS 环境中安装和管理集群。
	sts.<aws_region>.amazonaws.com	443	用于在 AWS 环境中安装和管理集群。
	tagging.us-east-1.amazonaws.com	443	用于在 AWS 环境中安装和管理集群。此端点始终为 **us-east-1**，无论集群要部署到的区域。
	ec2.<aws_region>.amazonaws.com	443	用于在 AWS 环境中安装和管理集群。
	elasticloadbalancing.<aws_region>.amazonaws.com	443	用于在 AWS 环境中安装和管理集群。
	servicequotas.<aws_region>.amazonaws.com	443	必需。用于确认用于部署该服务的配额。
	tagging.<aws_region>.amazonaws.com	443	允许以标签的形式分配 AWS 资源的元数据。
	***.cloudfront.net**	443	用于提供对 CloudFront 的访问。如果使用 AWS 安全令牌服务(STS)和私有 S3 存储桶，您必须提供对 CloudFront 的访问。
GCP	***.googleapis.com**	443	需要此项以访问 GCP 服务和资源。请参阅 GCP 文档中的 Cloud Endpoints，以查找您的 API 允许的端点。
	accounts.google.com	443	需要此项以访问您的 GCP 帐户。
Microsoft Azure	**management.azure.com**	443	需要此项以访问 Microsoft Azure 服务和资源。请参阅 Microsoft Azure 文档中的 Microsoft Azure REST API 参考，以查找允许您 API 的端点。
	***.blob.core.windows.net**	443	需要下载 Ignition 文件。
	login.microsoftonline.com	443	需要此项以访问 Microsoft Azure 服务和资源。请参阅 Microsoft Azure 文档中的 Azure REST API 参考，以查找您的 API 允许的端点。

5. **将以下 URL 列入允许列表：**

URL	port	功能
mirror.openshift.com	443	需要此项以访问镜像安装内容和镜像。此站点也是发行版本镜像签名的来源，但 Cluster Version Operator 只需要一个可正常工作的源。
storage.googleapis.com/openshift-release	443	发行版本镜像签名源，但 Cluster Version Operator 只需要一个可正常工作的源。
***.apps.<cluster_name>.<base_domain>**	443	需要此项以访问默认集群路由，除非您在安装过程中设置了入口通配符。
quayio-production-s3.s3.amazonaws.com	443	需要此项以访问 AWS 中的 Quay 镜像内容。
api.openshift.com	443	集群令牌需要，并检查集群是否有可用的更新。
rhcos.mirror.openshift.com	443	需要此项以下载 Red Hat Enterprise Linux CoreOS(RHCOS)镜像。
console.redhat.com	443	集群令牌所需。
sso.redhat.com	443	**https://console.redhat.com** 站点使用来自 **sso.redhat.com** 的身份验证

Operator 需要路由访问权限来执行健康检查。特别是，身份验证和 Web 控制台 Operator 会连接到两个路由，以验证路由是否正常工作。如果您是集群管理员，且不想允许 *.apps.<cluster_name>.<base_domain>，则允许这些路由：

- **oauth-openshift.apps.<cluster_name>.<base_domain>**
- **console-openshift-console.apps.<cluster_name>.<base_domain>，或在 consoles.operator/cluster 对象中的 spec.route.hostname 项指定的主机名（如果这个项不为空）。**

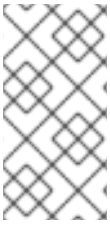
6. **将以下 URL 列入允许的可选第三方内容：**

URL	port	功能
registry.connect.redhat.com	443	所有第三方镜像和认证操作器必填.

URL	port	功能
rhc4tp-prod-z8cxf-image-registry-us-east-1-evenkyleffocxqvofrk.s3.dualstack.us-east-1.amazonaws.com	443	提供对托管在 **registry.connect.redhat.com**上的容器镜像的访问
oso-rhc4tp-docker-registry.s3-us-west-2.amazonaws.com	443	对于 Sonatype Nexus, F5 Big IP operator 是必需的。

7. **如果您使用默认的红帽网络时间协议(NTP)服务器允许以下 URL：**

- **1.rhel.pool.ntp.org**
- **2.rhel.pool.ntp.org**
- **3.rhel.pool.ntp.org**



注意

如果您不使用默认的 Red Hat NTP 服务器，请验证您的平台的 NTP 服务器并在防火墙中允许它。

其他资源

- **AWS STS 的 OpenID Connect 要求**

25.2.2. OpenShift Container Platform 网络流列表

网络流列表描述了 OpenShift Container Platform 服务的入站流。对于裸机和云环境，列表中的网络信息是准确的。使用网络流列表中的信息来帮助管理入口流量。您可以将入口流量限制为基本流以提高网络安全性。

要查看或下载原始 CSV 内容，请参阅此资源。

另外，在管理入口流量时请考虑以下动态端口范围：

- **9000-9999: 主机级别的服务**
- **3000-32767: Kubernetes 节点端口**
- **49152-65535: 动态或专用端口**

注意

网络流列表描述了基本 OpenShift Container Platform 安装的入口流量流。它没有描述其他组件的网络流，如 Red Hat Marketplace 提供的可选 Operator。列表不适用于 Hosted-Control-Plane、MicroShift 或独立集群。

表 25.1. 网络流列表

方向	协议	port	Namespace	service	Pod	Container	节点角色	选填
入口	TCP	22	主机系统服务	sshd			master	TRUE
入口	TCP	53	openshift-dns	dns-default	dnf-default	dns	master	FALSE
入口	TCP	111	主机系统服务	rpcbind			master	TRUE
入口	TCP	2379	openshift-etcd	etcd	etcd	etcdctl	master	FALSE
入口	TCP	2380	openshift-etcd	healthz	etcd	etcd	master	FALSE
入口	TCP	5050	openshift-machine-api		ironic-proxy	ironic-proxy	master	FALSE

方向	协议	port	Namespace	service	Pod	Container	节点角色	选填
入口	TCP	6080	openshift-kube-apiserver		kube-apiserver	kube-apiserver-insecure-readyz	master	FALSE
入口	TCP	6385	openshift-machine-api		ironic-proxy	ironic-proxy	master	FALSE
入口	TCP	6443	openshift-kube-apiserver	APIServer	kube-apiserver	kube-apiserver	master	FALSE
入口	TCP	8080	openshift-network-operator		network-operator	network-operator	master	FALSE
入口	TCP	8798	openshift-machine-config-operator	machine-config-daemon	machine-config-daemon	machine-config-daemon	master	FALSE
入口	TCP	9001	openshift-machine-config-operator	machine-config-daemon	machine-config-daemon	kube-rbac-proxy	master	FALSE
入口	TCP	9099	openshift-cluster-version	cluster-version-operator	cluster-version-operator	cluster-version-operator	master	FALSE
入口	TCP	9100	openshift-monitoring	node-exporter	node-exporter	kube-rbac-proxy	master	FALSE

方向	协议	port	Namespace	service	Pod	Container	节点角色	选填
入口	TCP	9103	openshift-ovn-kubernetes	ovn-kubernetes-node	ovnkube-node	kube-rbac-proxy-node	master	FALSE
入口	TCP	9104	openshift-network-operator	metrics	network-operator	network-operator	master	FALSE
入口	TCP	9105	openshift-ovn-kubernetes	ovn-kubernetes-node	ovnkube-node	kube-rbac-proxy-ovn-metrics	master	FALSE
入口	TCP	9107	openshift-ovn-kubernetes	egressip-node-healthcheck	ovnkube-node	ovnkube-controller	master	FALSE
入口	TCP	9108	openshift-ovn-kubernetes	ovn-kubernetes-control-plane	ovnkube-control-plane	kube-rbac-proxy	master	FALSE
入口	TCP	9192	openshift-cluster-machine-approver	machine-approver	machine-approver	kube-rbac-proxy	master	FALSE
入口	TCP	9258	openshift-cloud-controller-manager-operator	machine-approver	cluster-cloud-controller-manager	cluster-cloud-controller-manager	master	FALSE
入口	TCP	9444	openshift-kni-infra		hapoxy	hapoxy	master	FALSE

方向	协议	port	Namespace	service	Pod	Container	节点角色	选填
入口	TCP	9445	openshift-kni-infra		hapoxy	hapoxy	master	FALSE
入口	TCP	9447	openshift-machine-api		metal3-baremetal-operator		master	FALSE
入口	TCP	9537	主机系统服务	crio-metrics			master	FALSE
入口	TCP	9637	openshift-machine-config-operator	kube-rbac-proxy-crio	kube-rbac-proxy-crio	kube-rbac-proxy-crio	master	FALSE
入口	TCP	9978	openshift-etcd	etcd	etcd	etcd-metrics	master	FALSE
入口	TCP	9979	openshift-etcd	etcd	etcd	etcd-metrics	master	FALSE
入口	TCP	9980	openshift-etcd	etcd	etcd	etcd	master	FALSE
入口	TCP	10250	主机系统服务	kubelet			master	FALSE
入口	TCP	10256	openshift-ovn-kubernetes	ovnkube	ovnkube	ovnkube-controller	master	FALSE
入口	TCP	10257	openshift-kube-controller-manager	kube-controller-manager	kube-controller-manager	kube-controller-manager	master	FALSE

方向	协议	port	Namespace	service	Pod	Container	节点角色	选填
入口	TCP	10258	openshift-cloud-controller-manager-operator	cloud-controller	cloud-controller-manager	cloud-controller-manager	master	FALSE
入口	TCP	10259	openshift-kube-scheduler	scheduler	openshift-kube-scheduler	kube-scheduler	master	FALSE
入口	TCP	10260	openshift-cloud-controller-manager-operator	cloud-controller	cloud-controller-manager	cloud-controller-manager	master	FALSE
入口	TCP	10300	openshift-cluster-csi-drivers	csi-liveness probe	csi-driver-node	csi-driver	master	FALSE
入口	TCP	10309	openshift-cluster-csi-drivers	csi-node-driver	csi-driver-node	csi-node-driver-registrar	master	FALSE
入口	TCP	10357	openshift-kube-apiserver	openshift-kube-apiserver-healthz	kube-apiserver	kube-apiserver-check-endpoints	master	FALSE
入口	TCP	17697	openshift-kube-apiserver	openshift-kube-apiserver-healthz	kube-apiserver	kube-apiserver-check-endpoints	master	FALSE

方向	协议	port	Namespace	service	Pod	Container	节点角色	选填
入口	TCP	18080	openshift-kni-infra		coredns	coredns	master	FALSE
入口	TCP	22623	openshift-machine-config-operator	machine-config-server	machine-config-server	machine-config-server	master	FALSE
入口	TCP	22624	openshift-machine-config-operator	machine-config-server	machine-config-server	machine-config-server	master	FALSE
入口	UDP	53	openshift-dns	dns-default	dnf-default	dns	master	FALSE
入口	UDP	111	主机系统服务	rpcbind			master	TRUE
入口	UDP	6081	openshift-ovn-kubernetes	ovn-kubernetes geneve			master	FALSE
入口	TCP	22	主机系统服务	sshd			worker	TRUE
入口	TCP	53	openshift-dns	dns-default	dnf-default	dns	worker	FALSE
入口	TCP	80	openshift-ingress	router-default	router-default	路由器	worker	FALSE
入口	TCP	111	主机系统服务	rpcbind			worker	TRUE
入口	TCP	443	openshift-ingress	router-default	router-default	路由器	worker	FALSE
入口	TCP	8798	openshift-machine-config-operator	machine-config-daemon	machine-config-daemon	machine-config-daemon	worker	FALSE

方向	协议	port	Namespace	service	Pod	Container	节点角色	选填
入口	TCP	9001	openshift-machine-config-operator	machine-config-daemon	machine-config-daemon	kube-rbac-proxy	worker	FALSE
入口	TCP	9100	openshift-monitoring	node-exporter	node-exporter	kube-rbac-proxy	worker	FALSE
入口	TCP	9103	openshift-ovn-kubernetes	ovn-kubernetes-node	ovnkube-node	kube-rbac-proxy-node	worker	FALSE
入口	TCP	9105	openshift-ovn-kubernetes	ovn-kubernetes-node	ovnkube-node	kube-rbac-proxy-ovn-metrics	worker	FALSE
入口	TCP	9107	openshift-ovn-kubernetes	egressip-node-healthcheck	ovnkube-node	ovnkube-controller	worker	FALSE
入口	TCP	9537	主机系统服务	crio-metrics			worker	FALSE
入口	TCP	9637	openshift-machine-config-operator	kube-rbac-proxy-crio	kube-rbac-proxy-crio	kube-rbac-proxy-crio	worker	FALSE
入口	TCP	10250	主机系统服务	kubelet			worker	FALSE
入口	TCP	10256	openshift-ovn-kubernetes	ovnkube	ovnkube	ovnkube-controller	worker	TRUE

方向	协议	port	Namespace	service	Pod	Container	节点角色	选填
入口	TCP	10300	openshift-cluster-csi-drivers	csi-livenessprobe	csi-driver-node	csi-driver	worker	FALSE
入口	TCP	10309	openshift-cluster-csi-drivers	csi-node-driver-registrar	csi-driver-node	csi-node-driver-registrar	worker	FALSE
入口	TCP	18080	openshift-kni-infra		coredns	coredns	worker	FALSE
入口	UDP	53	openshift-dns	dns-default	dnf-default	dns	worker	FALSE
入口	UDP	111	主机系统服务	rpcbind			worker	TRUE
入口	UDP	6081	openshift-ovn-kubernetes	ovn-kubernetes geneve			worker	FALSE

25.3. 启用 LINUX 控制组版本 1 (CGROUP V1)

自 OpenShift Container Platform 4.14 起，OpenShift Container Platform 在集群中使用 Linux 控制组版本 2 (cgroup v2)。如果您在 OpenShift Container Platform 4.13 或更早版本中使用 cgroup v1，迁移到 OpenShift Container Platform 4.16 不会自动将 cgroup 配置更新至版本 2。全新安装 OpenShift Container Platform 4.14 或更高版本默认使用 cgroup v2。但是，您可以在安装时启用 Linux 控制组群版本 1 (cgroup v1)。在 OpenShift Container Platform 中启用 cgroup v1 禁用集群中的所有 cgroup v2 控制器和层次结构。

重要

cgroup v1 是一个已弃用的功能。弃用的功能仍然包含在 OpenShift Container Platform 中，并将继续被支持。但是，这个功能会在以后的发行版本中被删除，且不建议在新的部署中使用。

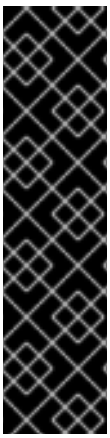
有关 OpenShift Container Platform 中已弃用或删除的主要功能的最新列表，请参阅 OpenShift Container Platform 发行注记中*已弃用和删除的功能*部分。

cgroup v2 是 Linux cgroup API 的当前版本。cgroup v2 比 cgroup v1 提供多种改进，包括统一层次结构、更安全的子树委派、新功能，如 Pressure Stall Information，以及增强的资源管理和隔离。但是，cgroup v2 与 cgroup v1 具有不同的 CPU、内存和 I/O 管理特征。因此，在运行 cgroup v2 的集群上，一些工作负载可能会遇到内存或 CPU 用量差异。

您可以通过编辑 node.config 对象在 cgroup v1 和 cgroup v2 间切换。如需更多信息，请参阅本节"添加资源"中的"在节点上配置 Linux cgroup"。

25.3.1. 在安装过程中启用 Linux cgroup v1

您可以通过创建安装清单来安装集群时启用 Linux 控制组群版本 1 (cgroup v1)。



重要

cgroup v1 是一个已弃用的功能。弃用的功能仍然包含在 OpenShift Container Platform 中，并将继续被支持。但是，这个功能会在以后的发行版本中被删除，且不建议在新的部署中使用。

有关 OpenShift Container Platform 中已弃用或删除的主要功能的最新列表，请参阅 OpenShift Container Platform 发行注记中*已弃用和删除的功能*部分。

流程

1. 创建或编辑 node.config 对象以指定 v1 cgroup：

```
apiVersion: config.openshift.io/v1
kind: Node
metadata:
```

```
  name: cluster
spec:
  cgroupMode: "v2"
```

2. 照常继续安装。

其他资源

- OpenShift Container Platform 安装概述
- 在节点上配置 Linux cgroup

第 26 章 验证安装

您可以按照本文档中的步骤在安装后检查 OpenShift Container Platform 集群的状态。

26.1. 查看安装日志

您可以在 OpenShift Container Platform 安装日志中查看安装概述。如果安装成功，日志中包括访问集群所需的信息。

先决条件

- 有访问安装主机的访问权限。

流程

- 查看安装主机上安装目录中的 .openshift_install.log 日志文件：

```
$ cat <install_dir>/.openshift_install.log
```

输出示例

如果安装成功，日志末尾会包括集群凭证，如下例所示：

```
...
time="2020-12-03T09:50:47Z" level=info msg="Install complete!"
time="2020-12-03T09:50:47Z" level=info msg="To access the cluster as the
system:admin user when using 'oc', run 'export
KUBECONFIG=/home/myuser/install_dir/auth/kubeconfig'"
time="2020-12-03T09:50:47Z" level=info msg="Access the OpenShift web-console
here: https://console-openshift-console.apps.mycluster.example.com"
time="2020-12-03T09:50:47Z" level=info msg="Login to the console with user:
\"kubeadmin\", and password: \"password\""
time="2020-12-03T09:50:47Z" level=debug msg="Time elapsed per stage:"
time="2020-12-03T09:50:47Z" level=debug msg="    Infrastructure: 6m45s"
time="2020-12-03T09:50:47Z" level=debug msg="Bootstrap Complete: 11m30s"
time="2020-12-03T09:50:47Z" level=debug msg=" Bootstrap Destroy: 1m5s"
time="2020-12-03T09:50:47Z" level=debug msg=" Cluster Operators: 17m31s"
time="2020-12-03T09:50:47Z" level=info msg="Time elapsed: 37m26s"
```

26.2. 查看镜像拉取源

对于没有网络连接的集群，您可以使用节点上的命令来查看拉取的镜像源，如 crictl 镜像。

但是，对于断开连接的安装，若要查看拉取镜像的来源，您必须查看 CRI-O 日志以查找 Trying 以访问日志条目，如下所示。查看镜像拉取源的其他方法，如 crictl images 命令，显示非镜像镜像的镜像名称，即使镜像是从镜像位置拉取的。

先决条件

- 您可以使用具有 cluster-admin 角色的用户访问集群。

流程

- 查看 master 或 worker 节点的 CRI-O 日志：

```
$  oc adm node-logs <node_name> -u crio
```

输出示例

用于访问日志条目的 Trying 指示镜像要从中拉取的位置。

```
...
Mar 17 02:52:50 ip-10-0-138-140.ec2.internal crio[1366]: time="2021-08-05
10:33:21.594930907Z" level=info msg="Pulling image: quay.io/openshift-release-
dev/ocp-release:4.10.0-ppc64le" id=abcd713b-d0e1-4844-ac1c-474c5b60c07c
name=/runtime.v1alpha2.ImageService/PullImage
Mar 17 02:52:50 ip-10-0-138-140.ec2.internal crio[1484]: time="2021-03-17
02:52:50.194341109Z" level=info msg="Trying to access \"li0317gcp1.mirror-
registry.qe.gcp.devcluster.openshift.com:5000/ocp/release@sha256:1926eae7cacb9c0
0f142ec98b00628970e974284b6ddaf9a6a086cb9af7a6c31\""
Mar 17 02:52:50 ip-10-0-138-140.ec2.internal crio[1484]: time="2021-03-17
02:52:50.226788351Z" level=info msg="Trying to access \"li0317gcp1.mirror-
registry.qe.gcp.devcluster.openshift.com:5000/ocp/release@sha256:1926eae7cacb9c0
0f142ec98b00628970e974284b6ddaf9a6a086cb9af7a6c31\""
...
```

日志可能会显示镜像拉取源两次，如上例中所示。

如果您的 ImageContentSourcePolicy 对象列出了多个镜像，OpenShift Container Platform 会尝试按照配置中列出的顺序拉取镜像，例如：

```
Trying to access \"li0317gcp1.mirror-
```

```
registry.qe.gcp.devcluster.openshift.com:5000/ocp/release@sha256:1926eae7cacb9c00f142ec
98b00628970e974284b6ddaf9a6a086cb9af7a6c31\"
Trying to access \"li0317gcp2.mirror-
registry.qe.gcp.devcluster.openshift.com:5000/ocp/release@sha256:1926eae7cacb9c00f142ec
98b00628970e974284b6ddaf9a6a086cb9af7a6c31\"
```

26.3. 获取集群版本、状态和更新详情

您可以通过运行 oc get clusterversion 命令来查看集群版本和状态。如果状态显示安装仍在进行，您可以查看 Operator 的状态以了解更多信息。

您还可以列出当前的更新频道并查看可用的集群更新。

先决条件

- 您可以使用具有 cluster-admin 角色的用户访问集群。
- 已安装 OpenShift CLI(oc)。

流程

1. 获取集群版本和总体状态：

```
$ oc get clusterversion
```

输出示例

```
NAME     VERSION   AVAILABLE   PROGRESSING   SINCE   STATUS
version   4.6.4     True        False         6m25s   Cluster version is 4.6.4
```

示例输出显示集群已被成功安装。

2. 如果集群状态表示安装仍在进行，您可以通过检查 Operator 状态来获取更详细的进度信息：

-

```
$ oc get clusteroperators.config.openshift.io
```

3. 查看集群规格、更新可用性和更新历史记录的详细概述：

```
$ oc describe clusterversion
```

4. 列出当前的更新频道：

```
$ oc get clusterversion -o jsonpath='{.items[0].spec}{"\n"}'
```

输出示例

```
{"channel":"stable-4.6","clusterID":"245539c1-72a3-41aa-9cec-72ed8cf25c5c"}
```

5. 查看可用的集群更新：

```
$ oc adm upgrade
```

输出示例

```
Cluster version is 4.6.4

Updates:

VERSION IMAGE
4.6.6   quay.io/openshift-release-dev/ocp-release@sha256:c7e8f18e8116356701bd23ae3a23fb9892dd5ea66c8300662ef30563d7104f39
```

其他资源

- 如需有关 查询 Operator 状态的更多信息，请参阅在安装后 查询 Operator 状态。

- 有关调查 Operator 问题的信息，请参阅对 Operator 进行故障排除。
- 有关更新集群的更多信息，请参阅使用 Web 控制台更新集群。
- 如需了解有关更新频道的概述，请参阅了解更新频道和 发行版本。

26.4. 验证集群是否使用短期凭证

您可以通过检查集群中的 Cloud Credential Operator (CCO) 配置和其他值来验证集群是否对各个组件使用简短安全凭证。

先决条件

- 已使用 Cloud Credential Operator 实用程序(ccoctl)部署了 OpenShift Container Platform 集群来实现短期凭证。
- 已安装 OpenShift CLI（oc）。
- 您以具有 cluster-admin 权限的用户身份登录。

流程

- 运行以下命令，验证 CCO 是否配置为以手动模式运行：

```
$ oc get cloudcredentials cluster \
  -o=jsonpath={.spec.credentialsMode}
```

以下输出确认 CCO 以手动模式运行：

输出示例

```
Manual
```

- 运行以下命令，验证集群没有 root 凭证：

```
$ oc get secrets \
  -n kube-system <secret_name>
```

其中 <secret_name> 是云供应商的 root secret 的名称。

平台	Secret 名称
Amazon Web Services (AWS)	**aws-creds**
Microsoft Azure	**azure-credentials**
Google Cloud Platform (GCP)	**gcp-credentials**

一个错误确认集群中不存在 root secret。

AWS 集群的输出示例

```
Error from server (NotFound): secrets "aws-creds" not found
```

- 运行以下命令，验证组件是否在单个组件中使用短期安全凭证：

```
$ oc get authentication cluster \
  -o jsonpath \
  --template='{ .spec.serviceAccountIssuer }'
```

此命令显示集群 Authentication 对象中 .spec.serviceAccountIssuer 参数的值。与云供应商关联的 URL 的输出表示集群使用从集群外部创建和管理的简短凭证的手动模式。

- Azure 集群：通过运行以下命令，验证组件假定 secret 清单中指定的 Azure 客户端 ID：

```
$ oc get secrets \
  -n openshift-image-registry installer-cloud-credentials \
  -o jsonpath='{.data}'
```

输出中包含了 azure_client_id 和 azure_federated_token_file 字段代表组件假定 Azure 客户端 ID。

- Azure 集群 ：运行以下命令来验证 pod 身份 Webhook 是否正在运行：

```
$ oc get pods \
  -n openshift-cloud-credential-operator
```

输出示例

```
NAME                                         READY   STATUS    RESTARTS   AGE
cloud-credential-operator-59cf744f78-r8pbq   2/2     Running   2          71m
pod-identity-webhook-548f977b4c-859lz        1/1     Running   1          70m
```

26.5. 使用 CLI 查询集群节点状态

您可以在安装后验证集群节点的状态。

先决条件

- 您可以使用具有 cluster-admin 角色的用户访问集群。
- 已安装 OpenShift CLI(oc)。

流程

1. 列出集群节点的状态。验证输出是否列出所有预期的 control plane 和计算节点，以及每个节点的状态是否为 Ready ：

```
$ oc get nodes
```

输出示例

```
NAME                         STATUS   ROLES    AGE   VERSION
compute-1.example.com        Ready    worker   33m   v1.29.4
control-plane-1.example.com  Ready    master   41m   v1.29.4
control-plane-2.example.com  Ready    master   45m   v1.29.4
compute-2.example.com        Ready    worker   38m   v1.29.4
compute-3.example.com        Ready    worker   33m   v1.29.4
control-plane-3.example.com  Ready    master   41m   v1.29.4
```

2. 查看每个集群节点的 CPU 和内存资源的可用性：

```
$ oc adm top nodes
```

输出示例

```
NAME                         CPU(cores)   CPU%   MEMORY(bytes)   MEMORY%
compute-1.example.com        128m         8%     1132Mi          16%
control-plane-1.example.com  801m         22%    3471Mi          23%
control-plane-2.example.com  1718m        49%    6085Mi          40%
compute-2.example.com        935m         62%    5178Mi          75%
compute-3.example.com        111m         7%     1131Mi          16%
control-plane-3.example.com  942m         26%    4100Mi          27%
```

其他资源

- 有关查看 节点健康状况和调查节点问题的更多详细信息，请参阅验证 节点健康状况。

26.6. 从 OPENSHIFT CONTAINER PLATFORM WEB 控制台查看集群状态

您可以在 OpenShift Container Platform Web 控制台中的 Overview 页面中查看以下信息：

- 集群的一般状态
- control plane、集群 Operator 和存储的状态
- CPU、内存、文件系统、网络传输和 pod 可用性
- 集群的 API 地址、集群 ID 和供应商名称
- 集群版本信息
- 集群更新状态，包括当前更新频道和可用更新的详情
- 详细节点、pod、存储类和持久性卷声明(PVC)信息的集群清单
- 持续集群活动和最近事件列表

先决条件

- 您可以使用具有 cluster-admin 角色的用户访问集群。

流程

- 在 Administrator 视角中，导航到 Home → Overview。

26.7. 查看 RED HAT OPENSHIFT CLUSTER MANAGER 中的集群状态

在 OpenShift Container Platform Web 控制台中，您可以查看 OpenShift Cluster Manager 上集群状态的详细信息。

先决条件

- 登录到 OpenShift Cluster Manager。
- 您可以使用具有 cluster-admin 角色的用户访问集群。

流程

1. 进入 OpenShift Cluster Manager 中的 Clusters 列表，并找到您的 OpenShift Container Platform 集群。
2. 点集群的 Overview 选项卡。
3. 查看有关集群的以下信息：
   - vCPU 和内存可用性和资源使用情况
   - 集群 ID、状态、类型、区域和供应商名称
   - 按节点类型划分的节点数
   - 集群版本详情、集群的创建日期和集群所有者的名称
   - 集群的生命周期支持状态
   - 订阅信息，包括服务级别协议(SLA)状态、订阅单元类型、集群的生产环境状态、订阅责任和服务级别

   提示

   要查看集群的历史记录，请点 Cluster history 选项卡。

4. 导航到 Monitoring 页面查看以下信息：

   - 已检测到的问题列表
   - 正在触发的警报列表
   - 集群 Operator 状态和版本
   - 集群的资源使用情况

5. 可选：您可以通过进入 Overview 菜单来查看 Red Hat Insights 收集的集群信息。在这个菜单中，您可以查看以下信息：

   - 集群可能会暴露的问题，按风险级别分类
   - 根据类别进行健康检查的状态

其他资源

- 如需了解更多 与查看集群中的潜在问题的信息，请参阅使用 Insights 发现集群中的问题。

26.8. 检查集群资源的可用性和使用

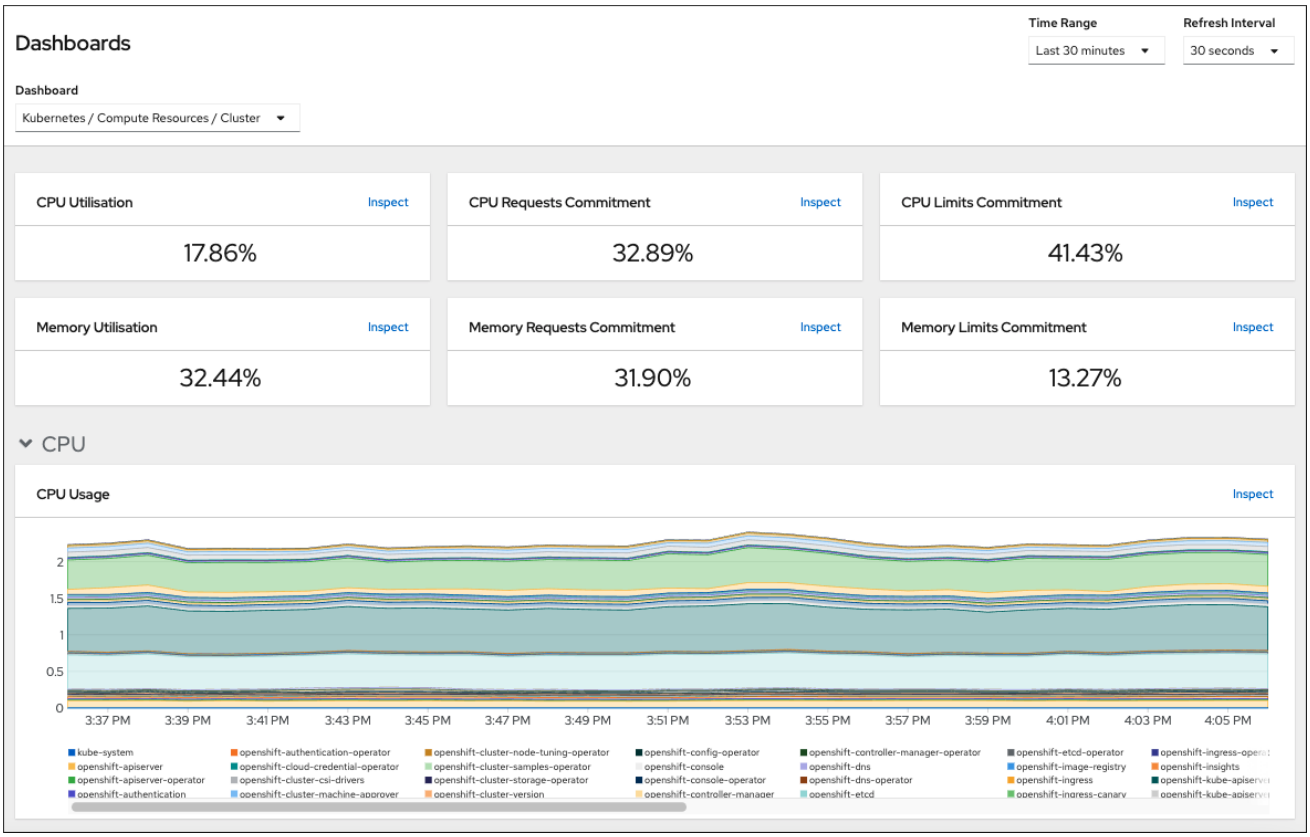
OpenShift Container Platform 提供了一组全面的监控仪表板，可帮助您了解集群组件的状态。

在 Administrator 视角中，您可以访问 OpenShift Container Platform 核心组件的仪表板，包括：

- etcd
- Kubernetes 计算资源

- Kubernetes 网络资源
- Prometheus
- 与集群和节点性能相关的仪表板

图 26.1. 计算资源仪表板示例



先决条件

- 您可以使用具有 cluster-admin 角色的用户访问集群。

流程

1. 在 OpenShift Container Platform web 控制台的 Administrator 视角中，进入到 Observe → Dashboards。
2. 在 Dashboard 列表中选择一个仪表板。有些仪表板（如 etcd 仪表板）在被选择时会生成额外的子菜单。

3. 可选：在 Time Range 列表中为图形选择一个时间范围。

   - 选择预定义的时间段。
   - 通过选择 Time Range 列表中的 Custom 时间范围 来设置自定义时间范围。

     a. 输入或选择 From 和 To date and time。

     b. 单击 Save 以保存自定义时间范围。

4. 可选：选择一个 Refresh Interval。

5. 将鼠标悬停在仪表板中的每个图形上，以显示特定项目的详细信息。

其他资源

- 如需有关 OpenShift Container Platform 监控堆栈的更多信息，请参阅监控概述。

26.9. 列出正在触发的警报

当 OpenShift Container Platform 集群中有一组定义的条件满足时，警报会提供通知。您可以使用 OpenShift Container Platform Web 控制台中的 Alerting UI 查看集群中触发的警报。

先决条件

- 您可以使用具有 cluster-admin 角色的用户访问集群。

流程

1. 在 Administrator 视角中，导航到 Observe → Alerting → Alerts 页面。

2. 查看正在触发的警报，包括 严重性 、 状态和 来源。

3. 在 Alert Details 页面中选择一个警报来查看更详细的信息。

其他资源

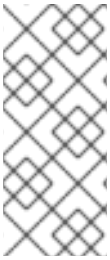
- 如需有关 OpenShift Container Platform 中警报的更多信息，请参阅管理 警报。

26.10. 后续步骤

- 如果您在安装集群时遇到问题，请参阅对 安装进行故障排除。
- 安装 OpenShift Container Platform 后，您可以 进一步扩展和自定义集群。

第 27 章 安装问题的故障排除

为了帮助对 OpenShift Container Platform 安装进行故障排除，您可以从 bootstrap 和 control plane 机器收集日志。您还可以从安装程序获取调试信息。如果您无法使用日志和调试信息解决问题，请参阅 确定组件特定故障排除的安装问题。



注意

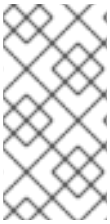
如果您的 OpenShift Container Platform 安装失败，且 debug 输出或日志包含网络超时或其他连接错误，请查看配置防火墙。从防火墙和负载均衡器收集日志可帮助您诊断与网络相关的错误。

27.1. 先决条件

- 已尝试安装 OpenShift Container Platform 集群，且安装失败。

27.2. 从失败安装中收集日志

如果您为安装程序提供了 SSH 密钥，则可以收集有关失败安装的数据。



注意

与从正在运行的集群收集日志相比，您可以使用其他命令收集失败安装的日志。如果必须从正在运行的集群中收集日志，请使用 oc adm must-gather 命令。

先决条件

- 在 bootstrap 过程完成前，OpenShift Container Platform 安装会失败。bootstrap 节点正在运行，并可通过 SSH 访问。
- ssh-agent 进程在您的计算机上处于活跃状态，并且为 ssh-agent 进程和安装程序提供了相同的 SSH 密钥。
- 如果尝试在您置备的基础架构上安装集群，则必须具有 bootstrap 和 control plane 节点的完全限定域名。

流程

1. 生成从 bootstrap 和 control plane 机器获取安装日志的命令：

- 如果您使用安装程序置备的基础架构，请切换到包含安装程序的目录，并运行以下命令：

```
$ ./openshift-install gather bootstrap --dir <installation_directory> 1
```

1 installation_directory 是您在运行时指定的目录 。/openshift-install create cluster。此目录包含安装程序创建的 OpenShift Container Platform 定义文件。

对于安装程序置备的基础架构，安装程序会保存有关集群的信息，因此您不用指定主机名或 IP 地址。

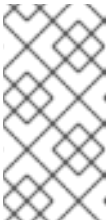
- 如果您使用您置备的基础架构，请切换到包含安装程序的目录，并运行以下命令：

```
$ ./openshift-install gather bootstrap --dir <installation_directory> \ 1
  --bootstrap <bootstrap_address> \ 2
  --master <master_1_address> \ 3
  --master <master_2_address> \ 4
  --master <master_3_address>" 5
```

1 对于 installation_directory，请指定您在运行时指定的同一目录 。/openshift-install create cluster。此目录包含安装程序创建的 OpenShift Container Platform 定义文件。

2 <bootstrap_address> 是集群 bootstrap 机器的完全限定域名或 IP 地址。

3 4 5 对于集群中的每个 control plane 或 master 机器，将 <master_*_address> 替换为其完全限定域名或 IP 地址。



注意

默认集群包含三台 control plane 机器。如所示，列出所有 control plane 机器，无论您的集群使用了多少个。

输出示例

```
INFO Pulling debug logs from the bootstrap machine
INFO Bootstrap gather logs captured here "<installation_directory>/log-bundle-<timestamp>.tar.gz"
```

如果您提交有关安装失败的红帽支持问题单，请在问题单中包含压缩日志。

27.3. 使用到主机的 SSH 访问手动收集日志

在 must-gather 或自动收集方法无法正常工作的情况下手动收集日志。

重要

默认情况下，基于 Red Hat OpenStack Platform(RHOSP)的安装禁用对 OpenShift Container Platform 节点的 SSH 访问。

先决条件

- 您必须有到主机的 SSH 访问权限。

流程

1. 运行以下命令，使用 journalctl 命令从 bootstrap 主机收集 bootkube.service 服务日志：

```
$ journalctl -b -f -u bootkube.service
```

2. 使用 podman logs 收集 bootstrap 主机的容器日志。这显示为从主机获取所有容器日志的循环：

```
$ for pod in $(sudo podman ps -a -q); do sudo podman logs $pod; done
```

3. 或者，运行以下命令来使用 tail 命令收集主机的容器日志：

```
# tail -f /var/lib/containers/storage/overlay-containers/*/userdata/ctr.log
```

4. 运行以下命令，使用 journalctl 命令从 master 和 worker 主机收集 kubelet.service 和 crio.service 服务日志：

```
$ journalctl -b -f -u kubelet.service -u crio.service
```

5. 运行以下命令，使用 tail 命令收集 master 和 worker 主机容器日志：

```
$ sudo tail -f /var/log/containers/*
```

27.4. 在不使用 SSH 访问主机的情况下手动收集日志

在 must-gather 或自动收集方法无法正常工作的情况下手动收集日志。

如果您无法通过 SSH 访问节点，您可以访问系统日志来调查主机上发生的情况。

先决条件

- OpenShift Container Platform 安装必须已完成。
- API 服务仍然可以正常工作。
- 您有系统管理员特权。

流程

1. 通过运行以下命令访问 /var/log 中的 Access journald 单元日志：

```
$ oc adm node-logs --role=master -u kubelet
```

2. 运行以下命令访问 /var/log 中的主机文件路径：

```
$ oc adm node-logs --role=master --path=openshift-apiserver
```

27.5. 从安装程序获取调试信息

您可以使用以下任一操作从安装程序获取调试信息。

- 在 hidden .openshift_install.log 文件中查看 来自过去安装的调试信息。例如，输入：

```
$ cat ~/<installation_directory>/.openshift_install.log 1
```

1 对于 installation_directory，请指定您在运行时指定的同一目录 。/openshift-install create cluster。

- 进入包含安装程序的目录，并使用 --log-level=debug 重新运行它：

```
$ ./openshift-install create cluster --dir <installation_directory> --log-level debug 1
```

1 对于 installation_directory，请指定您在运行时指定的同一目录 。/openshift-install create cluster。

27.6. 重新安装 OPENSHIFT CONTAINER PLATFORM 集群

如果您无法调试并解决失败的 OpenShift Container Platform 安装中的问题，请考虑安装新的 OpenShift Container Platform 集群。在再次开始安装过程前，您必须完成彻底的清理。对于用户置备的基础架构(UPI)安装，您必须手动销毁集群并删除所有关联的资源。以下流程用于安装程序置备的基础架构(IPI)安装。

流程

1. 销毁集群并删除与集群关联的所有资源，包括安装目录中的隐藏安装程序状态文件：

```
$ ./openshift-install destroy cluster --dir <installation_directory> 1
```

1 installation_directory 是您在运行时指定的目录 。/openshift-install create cluster。此目录包含安装程序创建的 OpenShift Container Platform 定义文件。

2. 在重新安装集群前，删除安装目录：

```
$ rm -rf <installation_directory>
```

3. 按照安装新 OpenShift Container Platform 集群的步骤进行操作。

其他资源

- 安装 OpenShift Container Platform 集群

第 28 章 支持 FIPS 加密

您可以使用 FIPS 模式安装 OpenShift Container Platform 集群。

OpenShift Container Platform 专为 FIPS 设计。当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS)时，OpenShift Container Platform 核心组件使用 RHEL 加密库，在 x86_64、ppc64le 和 s390x 架构上提交到 NIST FIPS 140-2/140-3 Validation。

有关 NIST 验证程序的更多信息，请参阅加密模块验证程序。有关为验证提交的 RHEL 加密库的单独版本的最新 NIST 状态，请参阅 Compliance Activities 和 Government Standards。

重要

要为集群启用 FIPS 模式，您必须从一个配置为以 FIPS 模式运行的 RHEL 9 计算机中运行安装程序，且必须使用安装程序支持的 FIPS 版本。请参阅 *使用 'oc adm extract' 的 FIPS 的安装程序*部分。

有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅在 FIPS 模式中安装该系统。

对于集群中的 Red Hat Enterprise Linux CoreOS(RHCOS)机器，当机器根据 install-config.yaml 文件中的选项的状态进行部署时，会应用这个更改，该文件管理用户在集群部署过程中可以更改的集群选项。在 Red Hat Enterprise Linux(RHEL)机器中，您必须在计划用作 worker 机器的机器上安装操作系统时启用 FIPS 模式。

因为 FIPS 必须在集群首次引导的操作系统前启用，所以您不能在部署集群后启用 FIPS。

28.1. 使用 OC ADM EXTRACT 获取支持 FIPS 的安装程序

OpenShift Container Platform 需要使用支持 FIPS 的安装二进制文件来在 FIPS 模式中安装集群。您可以使用 OpenShift CLI (oc) 从发行镜像中提取该二进制文件。获取二进制文件后，您可以继续集群安装，将 openshift-install 命令的所有实例替换为 openshift-install-fips。

先决条件

- 已使用版本 4.16 或更新版本安装了 OpenShift CLI (oc)。

流程

1. 运行以下命令，从安装程序中提取支持 FIPS 的二进制文件：

```
$ oc adm release extract --registry-config "${pullsecret_file}" --command=openshift-install-fips --to "${extract_dir}" ${RELEASE_IMAGE}
```

其中：

<pullsecret_file>

指定包含 pull secret 的文件的名称。

<extract_dir>

指定您要提取二进制文件的目录。

<RELEASE_IMAGE>

指定您使用的 OpenShift Container Platform 发行版本的 Quay.io URL。有关查找发行镜像的更多信息，请参阅*提取 OpenShift Container Platform 安装程序*。

2. 继续集群安装，将 openshift-install 命令的所有实例替换为 openshift-install-fips。

其他资源

- 提取 OpenShift Container Platform 安装程序

28.2. 使用公共 OPENSHIFT 镜像获取支持 FIPS 的安装程序

OpenShift Container Platform 需要使用支持 FIPS 的安装二进制文件来在 FIPS 模式中安装集群。您可以通过从公共 OpenShift 镜像下载来获取此二进制文件。获取二进制文件后，进行集群安装，将 openshift-install 二进制文件的所有实例替换为 openshift-install-fips。

先决条件

- 您可以访问互联网。

流程

1. 从 https://mirror.openshift.com/pub/openshift-v4/clients/ocp/latest-4.16/openshift-install-rhel9-amd64.tar.gz 下载安装程序。

2. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

```
$ tar -xvf openshift-install-rhel9-amd64.tar.gz
```

3. 继续集群安装，将 openshift-install 命令的所有实例替换为 openshift-install-fips。

28.3. OPENSHIFT CONTAINER PLATFORM 中的 FIPS 验证

OpenShift Container Platform 在 RHEL 和 RHCOS 中使用特定的 FIPS 验证或 Modules In Process 模块用于使用它们的操作系统组件。请参阅 RHEL 核心加密组件。例如，当用户使用 SSH 连接到 OpenShift Container Platform 集群和容器时，这些连接会被正确加密。

OpenShift Container Platform 组件以 Go 语言编写，并使用红帽的 golang 编译器构建。当您为集群启用 FIPS 模式时，需要加密签名的所有 OpenShift Container Platform 组件都会调用 RHEL 和 RHCOS 加密库。

表 28.1. OpenShift Container Platform 4.16 中的 FIPS 模式属性和限制

属性	限制
RHEL 9 和 RHCOS 操作系统支持 FIPS。	FIPS 实现没有使用在单一步骤中执行哈希计算和签名生成或验证的功能。在以后的 OpenShift Container Platform 版本中，将继续评估并改进此限制。
CRI-O 运行时支持 FIPS。	
OpenShift Container Platform 服务支持 FIPS。	
从 RHEL 9 和 RHCOS 二进制文件和镜像中获得的 FIPS 验证或模块加密模块和算法。	
使用 FIPS 兼容 golang 编译器。	TLS FIPS 并不完善，但计划在将来的 OpenShift Container Platform 版本中被支持。
支持多个架构中的 FIPS。	目前，只有使用 **x86_64**、**ppc64le** 和 **s390x** 架构的 OpenShift Container Platform 部署中才支持 FIPS。

28.4. 集群使用的组件支持 FIPS

虽然 OpenShift Container Platform 集群本身使用 FIPS 验证或Modules In Process 模块，但请确保支持 OpenShift Container Platform 集群的系统使用 FIPS 验证的或模块 In Process 模块进行加密。

28.4.1. etcd

要确保存储在 etcd 中的 secret 使用 FIPS 验证的/Modules in Process 加密，以 FIPS 模式引导节点。在以 FIPS 模式安装集群后，您可以使用 FIPS 批准的 aes cbc 加密算法加密 etcd 数据。

28.4.2. Storage

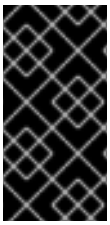
对于本地存储，使用 RHEL 提供的磁盘加密或使用 RHEL 提供的磁盘加密的容器原生存储。通过将所有数据存储到使用 RHEL 提供的磁盘加密的卷中，并为您的集群启用 FIPS 模式，静态数据和正在启动的数据或网络数据都受到 FIPS 验证的/Modules in Process 加密的保护。您可以将集群配置为加密每个节点的根文件系统，如 自定义节点 中所述。

28.4.3. runtimes

要确保容器知道它们在使用 FIPS 验证的/Modules in Process 加密模块的主机上运行，请使用 CRI-O 管理您的运行时。

28.5. 在 FIPS 模式下安装集群

要使用 FIPS 模式安装集群，请按照在首选基础架构上安装自定义集群的说明进行。在部署集群前，请确定在 install-config.yaml 文件中设置了 fips: true。



重要

要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 RHEL 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅在 FIPS 模式中安装该系统。

- Amazon Web Services
- Microsoft Azure
- 裸机

- Google Cloud Platform
- IBM Cloud®
- IBM Power®
- IBM Z® 和 IBM® LinuxONE
- IBM Z® 和 IBM® LinuxONE with RHEL KVM
- Red Hat OpenStack Platform(RHOSP)
- VMware vSphere

注意

如果使用 Azure File 存储，则无法启用 FIPS 模式。

要将 AES CBC 加密应用到 etcd 数据存储中，请在安装集群后按照 加密 etcd 数据 过程进行操作。

如果您在集群中添加 RHEL 节点，请确保在机器初始引导前启用 FIPS 模式。请参阅 将 RHEL 计算机器添加到 OpenShift Container Platform 集群 ， 以及以 FIPS 模式安装系统。