OpenShift Container Platform 4.6

在裸机上安装

安装 OpenShift Container Platform 裸机集群

Last Updated: 2023-02-27

OpenShift Container Platform 4.6 在裸机上安装

安装 OpenShift Container Platform 裸机集群

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

法律通告

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

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

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

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

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

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

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

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

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

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

All other trademarks are the property of their respective owners.

摘要

本文档提供在裸机环境中安装和卸载 OpenShift Container Platform 集群的说明。

目录

第 1 章 在裸机上安装

1.1. 在裸机上安装集群

在 OpenShift Container Platform 版本 4.6 中，您可以在您置备的裸机基础架构上安装集群。

重要

虽然您可能能够按照此流程在虚拟化或云环境中部署集群，但您必须清楚非裸机平台的其他注意事项。在尝试在此类环境中安装 OpenShift Container Platform 集群前，请参阅有关在未经测试的平台上部署 OpenShift Container Platform 的指南中的信息。

1.1.1. 先决条件

- 查看有关 OpenShift Container Platform 安装和更新流程的详细信息。
- 如果使用防火墙，则必须将其配置为允许集群需要访问的站点。

注意

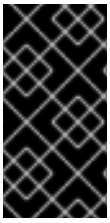
如果您要配置代理，请务必也要查看此站点列表。

1.1.2. OpenShift Container Platform 的互联网访问

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

您必须具有以下互联网访问权限：

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



重要

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

1.1.3. 具有用户置备基础架构的集群的机器要求

对于含有用户置备的基础架构的集群，您必须部署所有所需的机器。

1.1.3.1. 所需的机器

最小的 OpenShift Container Platform 集群需要下列主机：

- 一个临时 bootstrap 机器
- 三台 control plane 或 master 机器

- 至少两台计算机器，也称为 worker 机器。如果您正在运行三节点集群，则支持运行零个计算机器。不支持运行一台计算机器。

注意

集群要求 bootstrap 机器在三台 control plane 机器上部署 OpenShift Container Platform 集群。您可在安装集群后删除 bootstrap 机器。

重要

要保持集群的高可用性，请将独立的物理主机用于这些集群机器。

bootstrap 和 control plane 机器必须使用 Red Hat Enterprise Linux CoreOS (RHCOS) 作为操作系统。但是，计算机器可以在 Red Hat Enterprise Linux CoreOS(RHCOS)或 Red Hat Enterprise Linux(RHEL)7.9 间进行选择。

请注意，RHCOS 基于 Red Hat Enterprise Linux（RHEL） 8，并继承其所有硬件认证和要求。请查看Red Hat Enterprise Linux 技术功能及限制 。

1.1.3.2. 网络连接要求

所有 Red Hat Enterprise Linux CoreOS (RHCOS) 机器在启动过程中需要 **initramfs** 中的网络从 Machine Config Server 获取 Ignition 配置文件。在初次启动过程中，需要一个 DHCP 服务器或设置了静态 IP 地址来建立网络连接，以下载它们的 Ignition 配置文件。另外，集群中的每个 OpenShift Container Platform 节点都必须有权访问网络时间协议（NTP）服务器。如果 DHCP 服务器提供 NTP 服务器信息，Red Hat Enterprise Linux CoreOS（RHCOS）机器上的 chrony 时间服务会读取信息，并可与 NTP 服务器同步时钟。

1.1.3.3. 最低资源要求

每台集群机器都必须满足以下最低要求：

机器	操作系统	CPU [1]	RAM	存储	IOPS [2]
bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS 或 RHEL 7.9	2	8 GB	100 GB	300

1. 当未启用并发多线程(SMT)或超线程时，一个 CPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比率：（每个内核数的线程）LIMIT 插槽 = CPU。

2. OpenShift Container Platform 和 Kubernetes 对磁盘性能非常敏感，建议使用更快的存储速度，特别是 control plane 节点上需要 10 ms p99 fsync 持续时间的 etcd。请注意，在许多云平台上，存储大小和 IOPS 可一起扩展，因此您可能需要过度分配存储卷来获取足够的性能。

1.1.3.4. 证书签名请求管理

在使用您置备的基础架构时，集群只能有限地访问自动机器管理，因此您必须提供一种在安装后批准集群

证书签名请求 (CSR) 的机制。**kube-controller-manager** 只能批准 kubelet 客户端 CSR。 **machine-approver** 无法保证使用 kubelet 凭证请求的提供证书的有效性，因为它不能确认是正确的机器发出了该请求。您必须决定并实施一种方法，以验证 kubelet 提供证书请求的有效性并进行批准。

1.1.4. 创建用户置备的基础架构

在部署采用用户置备的基础架构的 OpenShift Container Platform 集群前，您必须创建底层基础架构。

先决条件

- 在为集群创建支持基础架构之前，请参阅OpenShift Container Platform 4.x Tested Integrations页。

流程

1. 在每个节点上配置 DHCP 或设置静态 IP 地址。
2. 提供所需的负载均衡器。
3. 配置机器的端口。
4. 配置 DNS。
5. 确保网络可以正常工作。

1.1.4.1. 用户置备的基础架构对网络的要求

所有 Red Hat Enterprise Linux CoreOS（RHCOS）机器在启动过程中需要 **initramfs** 中的网络从机器配置服务器获取 Ignition 配置。

在初次启动过程中，需要一个 DHCP 服务器或集群中的每个机器都设置了静态 IP 地址来建立网络连接，以下载它们的 Ignition 配置文件。

建议您使用 DHCP 服务器为集群进行长期机器管理。确保 DHCP 服务器已配置为向集群机器提供持久 IP 地址和主机名。

Kubernetes API 服务器必须能够解析集群机器的节点名称。如果 API 服务器和 worker 节点位于不同的区域中，您可以配置默认 DNS 搜索区域，以便 API 服务器能够解析节点名称。另一种支持的方法是始终在节点对象和所有 DNS 请求中使用完全限定域名来指代主机。

您必须配置机器间的网络连接，以便集群组件进行通信。每台机器都必须能够解析集群中所有其他机器的主机名。

表 1.1. 所有机器到所有机器

协议	端口	描述
ICMP	N/A	网络可访问性测试
TCP	**1936**	指标
	9000-**9999**	主机级别的服务，包括端口 **9100**-**9101** 上的节点导出器和端口 **9099** 上的 Cluster Version Operator。

协议	端口	描述
	10250-10259	Kubernetes 保留的默认端口
	10256	openshift-sdn
UDP	**4789**	VXLAN 和 Geneve
	6081	VXLAN 和 Geneve
	9000-9999	主机级别的服务，包括端口 **9100-9101** 上的节点导出器。
TCP/UDP	**30000-32767**	Kubernetes 节点端口

表 1.2. 要通过控制平面的所有机器

协议	端口	描述
TCP	**6443**	Kubernetes API

表 1.3. control plane 机器到 control plane 机器

协议	端口	描述
TCP	**2379-2380**	etcd 服务器和对等端口

网络拓扑要求

您为集群置备的基础架构必须满足下列网络拓扑要求。

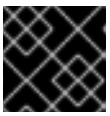
重要

OpenShift Container Platform 要求所有节点都能访问互联网，以便为平台容器提取镜像并向红帽提供遥测数据。

负载均衡器

在安装 OpenShift Container Platform 前，您必须置备两个满足以下要求的负载均衡器：

1. **API 负载均衡器**：提供一个通用端点，供用户（包括人和机器）与平台交互和配置。配置以下条件：
   - 只适用于第 4 层负载均衡。这可被称为 Raw TCP、SSL Passthrough 或者 SSL 桥接模式。如果使用 SSL Bridge 模式，必须为 API 路由启用 Server Name Indication（SNI）。
   - 无状态负载平衡算法。这些选项根据负载均衡器的实现而有所不同。



重要

不要为 API 负载均衡器配置会话持久性。

在负载均衡器的前端和后台配置以下端口：

表 1.4. API 负载均衡器

端口	后端机器（池成员）	内部	外部	描述
6443	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。您必须为 API 服务器健康检查探测配置 **/readyz** 端点。	X	X	Kubernetes API 服务器
22623	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。	X		机器配置服务器

注意

负载均衡器必须配置为，从 API 服务器关闭 **/readyz** 端点到从池中删除 API 服务器实例时最多需要 30 秒。在 **/readyz** 返回错误或处于健康状态后的时间范围内，端点必须被删除或添加。每 5 秒或 10 秒探测一次，有两个成功请求处于健康状态，三个成为不健康的请求经过测试。

2. **应用程序入口负载均衡器**:提供来自集群外部的应用程序流量流量的 Ingress 点。配置以下条件：

   - 只适用于第 4 层负载均衡。这可被称为 Raw TCP、SSL Passthrough 或者 SSL 桥接模式。如果使用 SSL Bridge 模式，您必须为 Ingress 路由启用Server Name Indication（SNI）。
   - 建议根据可用选项以及平台上托管的应用程序类型，使用基于连接的或者基于会话的持久性。

   在负载均衡器的前端和后台配置以下端口：

   表 1.5. 应用程序入口负载均衡器

端口	后端机器（池成员）	内部	外部	描述
443	默认运行入口路由器 Pod、计算或 worker 的机器。	X	X	HTTPS 流量
80	默认运行入口路由器 Pod、计算或 worker 的机器。	X	X	HTTP 流量

提示

如果负载均衡器可以看到客户端的真实 IP 地址，启用基于 IP 的会话持久性可提高使用端到端 TLS 加密的应用程序的性能。

注意

OpenShift Container Platform 集群需要正确配置入口路由器。control plane 初始化后，您必须配置入口路由器。

NTP 配置

OpenShift Container Platform 集群默认配置为使用公共网络时间协议（NTP）服务器。如果要使用本地企业 NTP 服务器，或者集群部署在断开连接的网络中，您可以将集群配置为使用特定的时间服务器。如需更多信息，请参阅*配置 chrony 时间服务*的文档。

如果 DHCP 服务器提供 NTP 服务器信息，Red Hat Enterprise Linux CoreOS（RHCOS）机器上的 chrony 时间服务会读取信息，并可与 NTP 服务器同步时钟。

其他资源

- 配置 chrony 时间服务

1.1.4.2. 用户置备 DNS 要求

DNS 用于名称解析和反向名称解析。DNS A/AAAA 或 CNAME 记录用于名称解析，PTR 记录用于反向解析名称。反向记录很重要，因为 Red Hat Enterprise Linux CoreOS（RHCOS）使用反向记录为所有节点设置主机名。另外，反向记录用于生成 OpenShift Container Platform 需要操作的证书签名请求（CSR）。

采用用户置备的基础架构的 OpenShift Container Platform 集群需要以下 DNS 记录。在每一记录中，**<cluster_name>** 是集群名称，**<base_domain>** 则是您在 **install-config.yaml** 文件中指定的集群基域。完整的 DNS 记录采用如下格式: **<component>.<cluster_name>.<base_domain>.**。

表 1.6. 所需的 DNS 记录

组件	记录	描述
Kubernetes API	**api.<cluster_name>.<base_domain>.**	添加 DNS A/AAAA 或 CNAME 记录，以及 DNS PTR 记录，以识别 control plane 机器的负载均衡器。这些记录必须由集群外的客户端以及集群中的所有节点解析。
	api-int.<cluster_name>.<base_domain>.	添加 DNS A/AAAA 或 CNAME 记录，以及 DNS PTR 记录，以识别 control plane 机器的负载均衡器。这些记录必须可以从集群中的所有节点解析。  **重要** API 服务器必须能够根据在 Kubernetes 中记录的主机名解析 worker 节点。如果 API 服务器无法解析节点名称，则代理的 API 调用会失败，且您无法从 pod 检索日志。
Routes	***.apps.<cluster_name>.<base_domain>.**	添加通配符 DNS A/AAAA 或 CNAME 记录，指向以运行入口路由器 Pod 的机器（默认为 worker 节点）为目标的负载均衡器。这些记录必须由集群外的客户端以及集群中的所有节点解析。

组件	记录	描述
bootstrap	**bootstrap.<cluster_name>.<base_domain>.**	添加 DNS A/AAAA 或 CNAME 记录，以及 DNS PTR 记录来识别 bootstrap 机器。这些记录必须由集群中的节点解析。
Master 主机	**<master><n>.<cluster_name>.<base_domain>.**	DNS A/AAAA 或 CNAME 记录，以识别 control plane 节点（也称为 master 节点）的每台机器。这些记录必须由集群中的节点解析。
Worker 主机	**<worker><n>.<cluster_name>.<base_domain>.**	添加 DNS A/AAAA 或 CNAME 记录，以识别 worker 节点的每台机器。这些记录必须由集群中的节点解析。

提示

您可以使用 **nslookup <hostname>** 命令来验证名称解析。您可以使用 **dig -x <ip_address>** 命令来验证 PTR 记录的反向名称解析。

下面的 BIND 区文件的例子展示了关于名字解析的 A 记录的例子。这个示例的目的是显示所需的记录。这个示例不是为选择一个名称解析服务提供建议。

例 1.1. DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
   2019070700 ; serial
   3H  ; refresh (3 hours)
   30M  ; retry (30 minutes)
   2W  ; expiry (2 weeks)
   1W )  ; minimum (1 week)
 IN NS ns1.example.com.
 IN MX 10 smtp.example.com.
;
;
ns1 IN A 192.168.1.5
smtp IN A 192.168.1.5
;
helper IN A 192.168.1.5
helper.ocp4 IN A 192.168.1.5
;
; The api identifies the IP of your load balancer.
api.ocp4  IN A 192.168.1.5
api-int.ocp4  IN A 192.168.1.5
;
; The wildcard also identifies the load balancer.
*.apps.ocp4  IN A 192.168.1.5
;
; Create an entry for the bootstrap host.
bootstrap.ocp4 IN A 192.168.1.96
;
; Create entries for the master hosts.
master0.ocp4  IN A 192.168.1.97
```

```
master1.ocp4  IN A 192.168.1.98
master2.ocp4  IN A 192.168.1.99
;
; Create entries for the worker hosts.
worker0.ocp4  IN A 192.168.1.11
worker1.ocp4  IN A 192.168.1.7
;
;EOF
```

下面的 BIND 区文件示例显示了反向名字解析的 PTR 记录示例。

例 1.2. 反向记录的 DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
   2019070700 ; serial
   3H  ; refresh (3 hours)
   30M  ; retry (30 minutes)
   2W  ; expiry (2 weeks)
   1W )  ; minimum (1 week)
 IN NS ns1.example.com.
;
; The syntax is "last octet" and the host must have an FQDN
; with a trailing dot.
97 IN PTR master0.ocp4.example.com.
98 IN PTR master1.ocp4.example.com.
99 IN PTR master2.ocp4.example.com.
;
96 IN PTR bootstrap.ocp4.example.com.
;
5 IN PTR api.ocp4.example.com.
5 IN PTR api-int.ocp4.example.com.
;
11 IN PTR worker0.ocp4.example.com.
7 IN PTR worker1.ocp4.example.com.
;
;EOF
```

1.1.5. 生成 SSH 私钥并将其添加到代理中

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

注意

在生产环境中，您需要进行灾难恢复和调试。

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

注意

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

流程

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

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

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

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。

注意

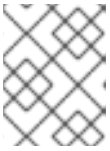
如果您计划在 **x86_64** 架构中安装使用 FIPS 验证的/Modules in Process 加密库的 OpenShift Container Platform 集群，不要创建使用 **ed25519** 算法的密钥。反之，创建一个使用 **rsa** 或 **ecdsa** 算法的密钥。

2. 作为后台任务启动 **ssh-agent** 进程：

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

输**出示例**

```
Agent pid 31874
```



注意

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

3. 将 SSH 私钥添加到 **ssh-agent**：

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

输**出示例**

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

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

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。如果在您置备的基础架构上安装集群，您必须将此密钥提供给集群的机器。

1.1.6. 获取安装程序

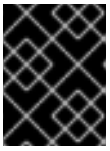
在安装 OpenShift Container Platform 之前，将安装文件下载到本地计算机上。

先决条件

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

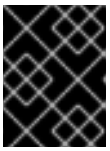
流程

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



重要

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



重要

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

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

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

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

1.1.7. 通过下载二进制文件安装 OpenShift CLI

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

重要

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

1.1.7.1. 在 Linux 上安装 OpenShift CLI

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

流程

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

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

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

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

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

1.1.7.2. 在 Windows 上安装 OpenShift CLI

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

流程

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

```
C:\> path
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

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

1.1.7.3. 在 macOS 上安装 OpenShift CLI

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

流程

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

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

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

1.1.8. 手动创建安装配置文件

对于使用用户置备的基础架构的 OpenShift Container Platform 安装，您必须手动生成安装配置文件。

先决条件

- 获取 OpenShift Container Platform 安装程序和集群的访问令牌。

流程

1. 创建用来存储您所需的安装资产的安装目录：

```
$ mkdir <installation_directory>
```

重要

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

2. 自定义以下 **install-config.yaml** 文件模板，并将它保存到 **<installation_directory>** 中。

注意

此配置文件必须命名为 **install-config.yaml**。

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

重要

install-config.yaml 文件会在安装过程的下一步骤中消耗掉。现在必须备份它。

1.1.8.1. 安装配置参数

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

注意

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

重要

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

1.1.8.1.1. 所需的配置参数

下表描述了所需的安装配置参数：

表 1.7. 所需的参数

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

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

1.1.8.1.2. 网络配置参数

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

只支持 IPv4 地址。

表 1.8. 网络参数

参数	描述	值
networking	集群网络的配置。	对象 **注意** 您不能在安装后修改 **networking** 对象指定的参数。
networking.networkType	要安装的集群网络供应商 Container Network Interface（CNI）插件。	**OpenShiftSDN** 或 **OVNKubernetes**。默认值为 **OpenShiftSDN**。
networking.clusterNetwork	pod 的 IP 地址块。 默认值为 **10.128.0.0/14**，主机前缀为 **/23**。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： <pre>networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23</pre>
networking.clusterNetwork.cidr	使用 **networking.clusterNetwork** 时需要此项。IP 地址块。 一个 IPv4 网络。	使用 CIDR 形式的 IP 地址块。IPv4 块的前缀长度介于 **0** 到 **32** 之间。

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

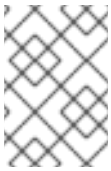
1.1.8.1.3. 可选配置参数

下表描述了可选安装配置参数：

表 1.9. 可选参数

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

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

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

参数	描述	值
imageContentSources	release-image 内容的源和仓库。	对象数组。包括一个 **source** 以及可选的 **mirrors**，如下表所示。
imageContentSources.source	使用 **imageContentSources** 时需要。指定用户在镜像拉取规格中引用的仓库。	字符串
imageContentSources.mirrors	指定可能还包含同一镜像的一个或多个仓库。	字符串数组
publish	如何发布或公开集群的面向用户的端点，如 Kubernetes API、OpenShift 路由。	**Internal** 或 **External**。默认值为 **External**。 在非云平台上不支持将此字段设置为 **Internal**。  **重要** 如果将字段的值设为 **Internal**，集群将无法运行。如需更多信息，请参阅 BZ#1953035。
sshKey	用于验证集群机器访问的 SSH 密钥或密钥。  **注意** 对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。	一个或多个密钥。例如： sshKey: <key1> <key2> <key3>

1.1.8.2. 裸机 install-config.yaml 文件示例

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

```
apiVersion: v1
baseDomain: example.com (1)
compute: (2)
- hyperthreading: Enabled (3)
  name: worker
  replicas: 0 (4)
controlPlane: (5)
  hyperthreading: Enabled (6)
```

```
  name: master
  replicas: 3 7
metadata:
  name: test 8
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14 9
    hostPrefix: 23 10
  networkType: OpenShiftSDN
  serviceNetwork: 11
  - 172.30.0.0/16
platform:
  none: {} 12
fips: false 13
pullSecret: '{"auths": ...}' 14
sshKey: 'ssh-ed25519 AAAA...' 15
```

1 集群的基域。所有 DNS 记录都必须是这个基域的子域，并包含集群名称。

2 5 **controlPlane** 部分是一个单映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，**controlPlane** 部分的第一行则不可以连字符开头。只使用一个 control plane 池。

3 6 是否要启用或禁用并发多线程（SMT）或**超线程**。默认情况下，启用 SMT 可提高机器内核的性能。您可以通过将参数值设为 **Disabled** 来禁用。如果禁用 SMT，则必须在所有集群机器中禁用它，其中包括 control plane 和计算机器。

注意

默认启用并发多线程（SMT）。如果在 BIOS 设置中没有启用 SMT，**hyperthreading** 参数不会起作用。

重要

如果您禁用 **hyperthreading**（无论是在 BIOS 中还是在 **install-config.yaml** 中），请确保您对可能会造成的机器性能显著降低的情况有所考虑。

4 **replicas** 参数的值必须设置为 **0**。此参数控制集群为您创建和管理的 worker 数量，使用用户置备的基础架构时集群不会执行这些功能。在完成 OpenShift Container Platform 安装前，您必须手动为集群部署 worker 机器。

7 您添加到集群的 control plane 机器数量。由于集群将这个值用作集群中 etcd 端点的数量，因此该值必须与您部署的 control plane 机器数量匹配。

8 您在 DNS 记录中指定的集群名称。

9 从中分配 pod IP 地址的 IP 地址块。此块不得与现有的物理网络重叠。这些 IP 地址用于 pod 网络。如果您需要从外部网络访问 pod，请配置负载均衡器和路由器来管理流量。

注意

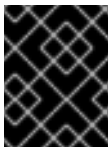
类 E CIDR 范围保留给以后使用。要使用 Class E CIDR 范围，您必须确保您的网络环境接受 Class E CIDR 范围内的 IP 地址。

10 分配给每个单独节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从所给的 **cidr** 中分配一个 **/23** 子网，这样就能有 510 (2^(32 - 23) - 2) 个 Pod IP 地址。如果您需要从外部网络访

11 用于服务 IP 地址的 IP 地址池。您只能输入一个 IP 地址池。此块不得与现有的物理网络重叠。如果您需要从外部网络访问服务，请配置负载均衡器和路由器来管理流量。

12 您必须将平台设置为 **none**。您不能为您的平台提供额外的平台配置变量。

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

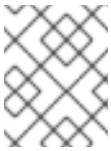


重要

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

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

15 Red Hat Enterprise Linux CoreOS (RHCOS) 中 **core** 用户的默认 SSH 密钥的公钥部分。



注意

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

1.1.8.3. 在安装过程中配置集群范围代理

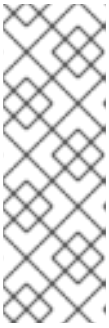
生产环境可能会拒绝直接访问互联网，而是提供 HTTP 或 HTTPS 代理。您可以通过在 **install-config.yaml** 文件中配置代理设置，将新的 OpenShift Container Platform 集群配置为使用代理。

注意

对于裸机安装，如果您没有从 **install-config.yaml** 文件中的 **networking.machineNetwork[].cidr** 字段指定的范围分配节点 IP 地址，您必须将其包括在 **proxy.noProxy** 字段中。

先决条件

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



注意

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

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

流程

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

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

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

注意

安装程序不支持代理的 **readinessEndpoints** 字段。

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

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

注意

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

1.1.9. 配置三节点集群

您可在没有 worker 的 OpenShift Container Platform 中安装和运行三节点集群。这为集群管理员和开发人员提供了较小的、效率更高的集群，用于开发、生产及测试。

流程

- 编辑 **install-config.yaml** 文件，将计算副本（也称为 worker 副本）数设为 **0**，如以下 **compute** 小节中所示：

```
compute:
- name: worker
  platform: {}
  replicas: 0
```

1.1.10. 创建 Kubernetes 清单和 Ignition 配置文件

由于您必须修改一些集群定义文件并要手动启动集群机器，因此您必须生成 Kubernetes 清单和 Ignition 配置文件，集群需要这两项来创建其机器。

安装配置文件转换为 Kubernetes 清单。清单嵌套到 Ignition 配置文件中，稍后用于创建集群。

重要

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

先决条件

- 已获得 OpenShift Container Platform 安装程序。
- 已创建 **install-config.yaml** 安装配置文件。

流程

1. 切换到包含安装程序的目录，并为集群生成 Kubernetes 清单：

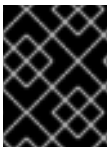
```
$ ./openshift-install create manifests --dir <installation_directory> 1
```

1 对于 **<installation_directory>**，请指定含有您创建的 **install-config.yaml** 文件的安装目录。

警告

如果要安装一个三节点集群，请跳过以下步骤，以便 control plane 节点可以调度。

+



重要

当您将 control plane 节点从默认不可调度配置为可以调度时，需要额外的订阅。这是因为 control plane 节点然后变为 worker 节点。

1. 检查 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** Kubernetes 清单文件中的 **mastersSchedulable** 参数是否已设置为 **false**。此设置可防止在 control plane 机器上调度 pod:
   a. 打开 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** 文件。
   b. 找到 **mastersSchedulable** 参数并确保它被设置为 **false**。
   c. 保存并退出文件。
2. 要创建 Ignition 配置文件，从包含安装程序的目录运行以下命令：

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

1 对于 **<installation_directory>**，请指定相同的安装目录。

该目录中将生成以下文件：

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

其他资源

- 如需有关 恢复 kubelet 证书的更多信息，请参阅恢复已过期的 control plane 证书。

1.1.11. 安装 RHCOS 并启动 OpenShift Container Platform bootstrap 过程

要在您置备的裸机基础架构上安装 OpenShift Container Platform，您必须在机器上安装 Red Hat Enterprise Linux CoreOS（RHCOS）。安装 RHCOS 时，您必须为 OpenShift Container Platform 安装程序生成的机器类型提供 Ignition 配置文件。如果您配置了合适的网络、DNS 和负载均衡基础架构，OpenShift Container Platform bootstrap 过程会在 RHCOS 机器重启后自动开始。

要在机器上安装 RHCOS，请按照以下步骤使用 ISO 镜像或网络 PXE 启动。

注意

本安装文档中包括的计算节点部署步骤特定于 RHCOS。如果您选择部署基于 RHEL 的计算节点，您将接管所有操作系统生命周期管理和维护，包括执行系统更新、应用补丁和完成所有其他必要的任务。RHEL 7 计算机器的使用已弃用，计划在以后的 OpenShift Container Platform 4 发行版本中删除。

您可以使用以下方法在 ISO 和 PXE 安装过程中配置 RHCOS:

- 内核参数： 您可以使用内核参数来提供特定于安装的信息。例如，您可以指定上传到 HTTP 服务器的 RHCOS 安装文件的位置，以及您要安装的节点类型的 Ignition 配置文件的位置。对于 PXE 安装，您可以使用 **APPEND** 参数将参数传递给实时安装程序的内核。对于 ISO 安装，您可以中断实时安装引导过程来添加内核参数。在这两种安装情况下，您可以使用特殊的 **coreos.inst.*** 参数来指示实时安装程序，以及标准安装引导参数来打开或关闭标准内核服务。

- Ignition 配置：OpenShift Container Platform Ignition 配置文件（***.ign**）特定于您要安装的节点类型。您可以在 RHCOS 安装过程中传递 bootstrap、control plane 或计算节点 Ignition 配置文件的位置，以便在第一次引导时生效。特殊情况下，您可以创建单独的、有限的 Ignition 配置来传递给 Live 系统。该 Ignition 配置可以执行特定任务，如在安装完成后向置备系统报告成功。这个特殊 Ignition 配置由 **coreos-installer** 使用，用于首次启动安装的系统。不要直接向 live ISO 提供标准 control plane 和计算节点 Ignition 配置。

- **coreos-installer**：您可以将 live ISO 安装程序引导到 shell 提示符，这可让您在首次引导前以多种方式准备持久性系统。特别是，您可以运行 **coreos-installer** 命令来识别包括的工件、使用磁盘分区以及设置联网。在有些情况下，您可以配置 live 系统上的功能并将其复制到安装的系统中。

使用 ISO 安装还是 PXE 安装要根据您的具体情况而定。PXE 安装需要可用的 DHCP 服务并进行更多准备，但可以使安装过程更自动化。ISO 安装是一个更手动的过程，如果您设置的机器较多，则可能不方便。

注意

自 OpenShift Container Platform 4.6 起，RHCOS ISO 和其他安装工件支持在带有 4K 扇区的磁盘上安装。

1.1.11.1. 使用 ISO 镜像创建 Red Hat Enterprise Linux CoreOS (RHCOS) 机器

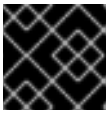
在您置备的基础架构上安装集群前，必须先创建 RHCOS 机器供其使用。您可以使用 ISO 镜像来创建这些机器。

先决条件

- 获取集群的 Ignition 配置文件。
- 具有可从计算机以及您创建的机器访问的 HTTP 服务器的访问权限。

流程

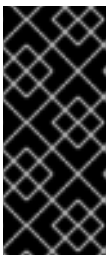
1. 将安装程序创建的 control plane、计算和 bootstrap Ignition 配置文件上传到 HTTP 服务器。记下这些文件的 URL。



重要

如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

2. 从 RHCOS 镜像镜像页面获取您选择的操作系统实例安装方法所需的 RHCOS 镜像。



重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每一发行版本都有改变。您必须下载最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。如果可用，请使用与 OpenShift Container Platform 版本匹配的镜像版本。此流程只使用 ISO 镜像。此安装类型不支持 RHCOS qcow2 镜像。

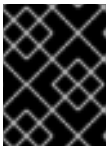
ISO 文件名类似以下示例：

rhcos-<version>-live.<architecture>.iso

3. 使用 ISO 启动 RHCOS 安装。使用如下安装选项之一：
   - 将 ISO 镜像刻录到磁盘并直接启动。
   - 通过 LOM 接口使用 ISO 重定向。
4. 引导 ISO 镜像。您可以中断安装引导过程来添加内核参数。然而，在这个 ISO 过程中，您应该使用 **coreos-installer** 命令而不是添加内核参数。如果您在没有选项或中断的情况下运行 live 安装程序，安装程序将引导至 live 系统上的 shell 提示符，准备好将 RHCOS 安装到磁盘中。
5. 在运行 **coreos-installer** 前，请参阅 *高级 RHCOS 安装参考* 部分，以了解配置功能的不同方法，如网络和磁盘分区。
6. 运行 **coreos-installer** 命令。您至少必须识别节点类型的 Ignition 配置文件位置，以及您要安装到的磁盘位置。下面是一个示例：

```
$ sudo coreos-installer install \
    --ignition-url=https://host/worker.ign /dev/sda
```

7. 安装 RHCOS 后，系统会重启。系统重启过程中，它会应用您指定的 Ignition 配置文件。
8. 继续为集群创建其他机器。



重要

此刻您必须创建 bootstrap 和 control plane 机器。如果 control plane 机器不可调度（这是默认调度），则在安装集群前至少会创建两台计算机器。

1.1.11.2. 通过 PXE 或 iPXE 启动来创建 Red Hat Enterprise Linux CoreOS (RHCOS) 机器

在安装使用手动置备 RHCOS 节点（如裸机）的集群前，您必须创建 RHCOS 机器供其使用。您可以使用 PXE 或 iPXE 启动来创建机器。

先决条件

- 获取集群的 Ignition 配置文件。

- 配置合适的 PXE 或 iPXE 基础架构。
- 具有 HTTP 服务器的访问权限，以便您可从计算机进行访问。

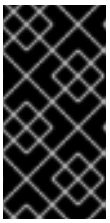
流程

1. 将安装程序创建的 master、worker 和 bootstrap Ignition 配置文件上传到 HTTP 服务器。记下这些文件的 URL。

重要

您可以在 Ignition 配置中添加或更改配置设置，然后将其保存到 HTTP 服务器。如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

2. 从 RHCOS **镜像镜像**页面获取 **RHCOS 内核**、**initramfs** 和 **rootfs** 文件。



重要

RHCOS 工件（artifact）可能不会随着 OpenShift Container Platform 的每个发行版本而改变。您必须下载最高版本的工件，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。这个过程只使用下面描述的正确 **kernel**、**initramfs** 和 **rootfs** 工件。此安装类型不支持 RHCOS qcow2 镜像。

文件名包含 OpenShift Container Platform 版本号。它们类似以下示例：

- **kernel**: **rhcos-<version>-live-kernel-<architecture>**
- **initramfs**: **rhcos-<version>-live-initramfs.<architecture>.img**
- **rootfs**: **rhcos-<version>-live-rootfs.<architecture>.img**

3. 上传引导方法所需的额外文件：
   - 对于传统的 PXE，将 **kernel** 和 **initramfs** 文件上传到 TFTP 服务器和 **rootfs** 文件到 HTTP 服务器。
   - 对于 iPXE，将 **内核**、**initramfs** 和 **rootfs** 文件上传到 HTTP 服务器。

重要

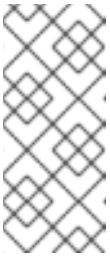
如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

4. 配置网络启动基础架构，以便在安装 RHCOS 后机器可从本地磁盘启动。
5. 为 RHCOS 镜像配置 PXE 或 iPXE 安装。
   针对您的环境修改以下示例菜单条目之一，并验证能否正确访问镜像和 Ignition 文件：
   - 对于 PXE：

```
DEFAULT pxeboot
TIMEOUT 20
PROMPT 0
LABEL pxeboot
    KERNEL http://<HTTP_server>/rhcos-<version>-live-kernel-<architecture> (1)
```

```
    APPEND initrd=http://<HTTP_server>/rhcos-<version>-live-initramfs.
<architecture>.img coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-
rootfs.<architecture>.img coreos.inst.install_dev=/dev/sda
coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign 2 3
```

1. 指定上传到 HTTP 服务器的 live **kernel** 文件位置。URL 必须是 HTTP、TFTP 或者 FTP ; 不支持 HTTPS 和 NFS。
2. 如果您使用多个 NIC，请在 **ip** 选项中指定一个接口。例如，要在名为 **eno1** 的 NIC 上使用 DHCP，请设置 **ip=eno1:dhcp**。
3. 指定上传到 HTTP 服务器的 RHCOS 文件的位置。**initrd** 参数值是 **initramfs** 文件的位置，**coreos.live.rootfs_url** 参数值是 **rootfs** 文件的位置，**coreos.inst.ignition_url** 参数值是 bootstrap Ignition 配置文件的位置。您还可以在 **APPEND** 行中添加更多内核参数来配置联网或其他引导选项。



注意

这个配置不会在使用图形控制台的机器上启用串口控制台访问。要配置不同的控制台，请在 **APPEND** 行中添加一个或多个 **console=** 参数。例如，添加 **console=tty0 console=ttyS0** 将第一个 PC 串口设置为主控制台，图形控制台作为二级控制台。如需更多信息，请参阅如何在 Red Hat Enterprise Linux 中设置串行终端和（或）控制台？

- 对于 iPXE：

```
kernel http://<HTTP_server>/rhcos-<version>-live-kernel-<architecture> initrd=main
coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-rootfs.
<architecture>.img coreos.inst.install_dev=/dev/sda
coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign 1 2
initrd --name main http://<HTTP_server>/rhcos-<version>-live-initramfs.
<architecture>.img 3
boot
```

1. 指定上传到 HTTP 服务器的 RHCOS 文件的位置。**kernel** 参数值是 **kernel** 文件的位置，在 UEFI 系统中引导时需要 **initrd=main** 参数。 **coreos.live.rootfs_url** 参数值是 **rootfs** 文件的位置， **coreos.inst.ignition_url** 参数值则是 bootstrap Ignition 配置文件的位置。
2. 如果您使用多个 NIC，请在 **ip** 选项中指定一个接口。例如，要在名为 **eno1** 的 NIC 上使用 DHCP，请设置 **ip=eno1:dhcp**。
3. 指定上传到 HTTP 服务器的 **initramfs** 文件的位置。

注意

这个配置不会在使用图形控制台的机器上启用串口控制台访问。要配置不同的控制台，请在 **kerne** 行中添加一个或多个 **console=** 参数。例如，添加 **console=tty0 console=ttyS0** 将第一个 PC 串口设置为主控制台，图形控制台作为二级控制台。如需更多信息，请参阅如何在 Red Hat Enterprise Linux 中设置串行终端和（或）控制台？

6. 如果使用 PXE UEFI，请执行以下操作：

   a. 提供启动系统所需的 **shim x64.efi** EFI 二进制文件和 **grub.cfg** 文件。

      - 通过将 RHCOS ISO 挂载到您的主机，然后将 **images/efiboot.img** 文件挂载到您的主机来提取所需的 EFI 二进制文件：

        ```
        $ mkdir -p /mnt/iso
        ```

        ```
        $ mkdir -p /mnt/efiboot
        ```

        ```
        $ mount -o loop rhcos-installer.x86_64.iso /mnt/iso
        ```

        ```
        $ mount -o loop,ro /mnt/iso/images/efiboot.img /mnt/efiboot
        ```

      - 从 **efiboot.img** 挂载点，将 **EFI/redhat/shimx64.efi** 和 **EFI/redhat/grubx64.efi** 文件复制到您的 TFTP 服务器中：

        ```
        $ cp /mnt/efiboot/EFI/redhat/shimx64.efi .
        ```

        ```
        $ cp /mnt/efiboot/EFI/redhat/grubx64.efi .
        ```

        ```
        $ umount /mnt/efiboot
        ```

        ```
        $ umount /mnt/iso
        ```

      - 将 RHCOS ISO 中包含的 **EFI/redhat/grub.cfg** 文件复制到您的 TFTP 服务器中。

   b. 编辑 **grub.cfg** 文件使其包含类似如下的参数：

      ```
      menuentry 'Install Red Hat Enterprise Linux CoreOS' --class fedora --class gnu-linux --
      class gnu --class os {
       linuxefi rhcos-<version>-live-kernel-<architecture> coreos.inst.install_dev=/dev/sda
      coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-rootfs.
      <architecture>.img coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign
       initrdefi rhcos-<version>-live-initramfs.<architecture>.img
      }
      ```

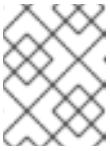
      其中：

      rhcos-<version>-live-kernel-<architecture>
      指定上传到 TFTP 服务器的 **kernel** 文件。

      http://<HTTP_server>/rhcos-<version>-live-rootfs.<architecture>.img
      指定上传到 HTTP 服务器的 live rootfs 镜像的位置。

      http://<HTTP_server>/bootstrap.ign
      指定上传到 HTTP 服务器的 bootstrap Ignition 配置文件的位置。

      rhcos-<version>-live-initramfs.<architecture>.img
      指定上传到 TFTP 服务器的 **initramfs** 文件的位置。



注意

有关如何为 UEFI 引导配置 PXE 服务器的详情，请查看红帽知识库文章：如何为 Red Hat Enterprise Linux 配置/设置 PXE 服务器？.

7. 继续为集群创建机器。

重要

此刻您必须创建 bootstrap 和 control plane 机器。如果 control plane 机器不可调度（这是默认调度），则在安装集群前至少会创建两台计算机器。

1.1.11.3. 高级 Red Hat Enterprise Linux CoreOS（RHCOS）安装配置

为 OpenShift Container Platform 手动置备 Red Hat Enterprise Linux CoreOS（RHCOS）节点的一个关键优点是能够进行通过默认的 OpenShift Container Platform 安装方法无法进行的配置。本节介绍了您可以使用的一些技术来进行配置，其中包括：

- 将内核参数传递给实时安装程序
- 从 live 系统手动运行 **coreos-installer**
- 将 Ignition 配置嵌入 ISO 中

本节详述了与 Red Hat Enterprise Linux CoreOS（RHCOS）手动安装的高级配置相关的内容，如磁盘分区、网络以及使用 Ignition 配置的不同方式相关。

1.1.11.3.1. 使用高级网络选项进行 PXE 和 ISO 安装

OpenShift Container Platform 节点的网络默认使用 DHCP 来收集所有必要配置设置。要设置静态 IP 地址或配置特殊的设置，如绑定,您可以执行以下操作之一：

- 引导 live 安装程序时会传递特殊的内核参数。
- 使用机器配置将网络文件复制到安装的系统中。
- 使用 live installer shell 提示配置网络，然后将那些设置复制到安装的系统上，以便在安装的系统第一次引导时生效。

要配置 PXE 或 iPXE 安装，请使用以下选项之一：

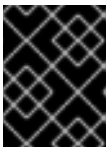
- 请参阅"高级 RHCOS 安装参考"表。
- 使用机器配置将网络文件复制到安装的系统中。

要配置 ISO 安装，请使用以下步骤。

流程

1. 引导 ISO 安装程序。
2. 在 live 系统 shell 提示下，使用可用的 RHEL 工具（如 **nmcli** 或 **nmtui**）为 Live 系统配置网络。
3. 运行 **coreos-installer** 命令来安装系统，添加 **--copy-network** 选项来复制网络配置。例如：

```
$ coreos-installer install --copy-network \
    --ignition-url=http://host/worker.ign /dev/sda
```



重要

copy-network 选项只复制 **/etc/NetworkManager/system-connections** 下的网络配置。特别是，它不会复制系统主机名。

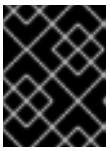
4. 重启安装的系统。

1.1.11.3.2. 磁盘分区

磁盘分区是在 Red Hat Enterprise Linux CoreOS（RHCOS）安装过程中在 OpenShift Container Platform 集群节点上创建的。特定架构的每个 RHCOS 节点都使用相同的分区布局，除非默认分区配置被覆盖。在 RHCOS 安装过程中，根文件系统的大小会增大，以使用目标设备中剩余的可用空间。

但是，在安装 OpenShift Container Platform 节点时，在两种情况下您可能需要覆盖默认分区：

- 创建单独的分区：对于在空磁盘中的 greenfield 安装，您可能想要在分区中添加单独的存储。这只在生成 **/var** 或者一个 **/var** 独立分区的子目录（如 **/var/lib/etcd** ）时被正式支持，但不支持两者。



重要

Kubernetes 只支持两个文件系统分区。如果您在原始配置中添加多个分区，Kubernetes 无法监控所有这些分区。

- 保留现有分区：对于 brownfield 安装，您要在现有节点上重新安装 OpenShift Container Platform，并希望保留从之前的操作系统中安装的数据分区，对于 **coreos-installer** 来说，引导选项和选项都允许您保留现有数据分区。

1.1.11.3.2.1. 创建一个独立的 /var 分区

通常情况下，OpenShift Container Platform 的磁盘分区应该留给安装程序。然而，在有些情况下您可能需要在文件系统的一部分中创建独立分区。

OpenShift Container Platform 支持添加单个分区来将存储附加到 **/var** 分区或 **/var** 的子目录。例如：

- **/var/lib/containers**：保存镜像相关的内容，随着更多镜像和容器添加到系统中，它所占用的存储会增加。
- **/var/lib/etcd**：保存您可能希望保持独立的数据，比如 etcd 存储的性能优化。
- **/var**：保存您希望独立保留的数据，用于特定目的（如审计）。

单独存储 **/var** 目录的内容可方便地根据需要对区域扩展存储，并可以在以后重新安装 OpenShift Container Platform 时保持该数据地完整。使用这个方法，您不必再次拉取所有容器，在更新系统时也无法复制大量日志文件。

因为 **/var** 在进行一个全新的 Red Hat Enterprise Linux CoreOS（RHCOS）安装前必需存在，所以这个流程会在 OpenShift Container Platform 安装过程的 **openshift-install** 准备阶段插入的机器配置来设置独立的 **/var** 分区。

流程

1. 创建存放 OpenShift Container Platform 安装文件的目录：

```
$ mkdir $HOME/clusterconfig
```

2. 运行 **openshift-install** 在 **manifest** 和 **openshift** 子目录中创建一组文件。在出现提示时回答系统问题：

```
$ openshift-install create manifests --dir $HOME/clusterconfig
? SSH Public Key ...
$ ls $HOME/clusterconfig/openshift/
99_kubeadmin-password-secret.yaml
99_openshift-cluster-api_master-machines-0.yaml
99_openshift-cluster-api_master-machines-1.yaml
99_openshift-cluster-api_master-machines-2.yaml
...
```

3. 创建 **MachineConfig** 对象并将其添加到 **openshift** 目录中的一个文件中。例如，把文件命名为 **98-var-partition.yaml**，将磁盘设备名称改为 **worker** 系统中存储设备的名称，并根据情况设置存储大小。这个示例将 **/var** 目录放在独立分区中：

```
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
spec:
  config:
    ignition:
      version: 3.1.0
    storage:
      disks:
      - device: /dev/<device_name> 1
        partitions:
        - label: var
          startMiB: <partition_start_offset> 2
          sizeMiB: <partition_size> 3
      filesystems:
        - device: /dev/disk/by-partlabel/var
          path: /var
          format: xfs
    systemd:
      units:
        - name: var.mount 4
          enabled: true
          contents: |
            [Unit]
            Before=local-fs.target
            [Mount]
            What=/dev/disk/by-partlabel/var
            Where=/var
            Options=defaults,prjquota 5
            [Install]
            WantedBy=local-fs.target
```

1. 要分区的磁盘的存储设备名称。
2. 当在引导磁盘中添加数据分区时，推荐最少使用 25000MB。root 文件系统会自动重新定义大小使其占据所有可用空间（最多到指定的偏移值）。如果没有指定值，或者指定的值小于推荐的最小值，则生成的 root 文件系统会太小，而在以后进行的 RHCOS 重新安装可能会覆盖数据分区的开始部分。
3. 数据分区的大小（以兆字节为单位）。
4. 挂载单元的名称必须与 where = 指令中指定的目录匹配。例如，对于挂载到 **/var/lib/containers** 的文件系统，这个单元必须命名为 **var-lib-containers.mount**。
5. 必须针对用于容器存储的文件系统启用 **prjquota** 挂载选项。

注意

在创建独立 **/var** 分区时，如果不同的实例类型没有相同的设备名称，则无法将不同的实例类型用于 worker 节点。

4. 再次运行 **openshift-install**，从 **manifest** 和 **openshift** 子目录中的一组文件创建 Ignition 配置：

```
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
auth  bootstrap.ign  master.ign  metadata.json  worker.ign
```

现在，可以使用 Ignition 配置文件作为 ISO 或 PXE 手动安装过程的输入来安装 Red Hat Enterprise Linux CoreOS（RHCOS）系统。

1.1.11.3.2.2. 保留现有分区

对于 ISO 安装，您可以在 **coreos-installer** 命令行中添加可让安装程序维护一个或多个现有分区的选项。对于 PXE 安装，您可以 **APPEND coreos.inst.*** 选项来保留分区。

保存的分区可能是来自现有 OpenShift Container Platform 系统中的分区，其中包括了您希望保留的数据分区。以下是几个提示：

- 如果您保存了现有分区，且这些分区没有为 RHCOS 留下足够空间，则安装将失败但不会损害已保存的分区。
- 通过分区标签或数字识别您要保留的磁盘分区。

对于 ISO 安装

这个示例保留分区标签以**数据**（**data***）开头的任何分区：

```
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign \
      --save-partlabel 'data*' /dev/sda
```

以下示例演示了在运行 **coreos-installer** 时要保留磁盘上的第 6 个分区：

```
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign \
      --save-partindex 6 /dev/sda
```

这个示例保留了分区 5 及更高分区：

```
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign
      --save-partindex 5- /dev/sda
```

在前面已保存分区的示例中，**coreos-installer** 会立即重新创建分区。

对于 PXE 安装

这个 **APPEND** 选项保留分区标签以 'data'('data*')开头的所有分区：

```
coreos.inst.save_partlabel=data*
```

这个 **APPEND** 选项保留分区 5 及其后的分区：

```
coreos.inst.save_partindex=5-
```

这个 **APPEND** 选项保留分区 6:

```
coreos.inst.save_partindex=6
```

1.1.11.3.3. 标识 Ignition 配置

在进行 RHCOS 手动安装时，您可以提供两种 Ignition 配置类型，它们有不同的原因：

- **永久安装 Ignition 配置**：每个手动 RHCOS 安装都需要传递 **openshift-installer** 生成的 Ignition 配置文件之一，如 **bootstrap.ign**、**master.ign** 和 **worker.ign**，才能进行安装。

  重要

  不建议修改这些文件。

  对于 PXE 安装，您可以使用 **coreos.inst.ignition_url=** 选项在 **APPEND** 行上传递 Ignition 配置。对于 ISO 安装，在 ISO 引导至 shell 提示符后，您可以使用 **--ignition-url=** 选项在 **coreos-installer** 命令行上识别 Ignition 配置。在这两种情况下，都只支持 HTTP 和 HTTPS 协议。

- **live 安装 Ignition 配置**：此类型必须手动创建，并应该尽可能避免，因为红帽不支持它。使用此方法，Ignition 配置会传递到 live 安装介质，在引导时立即运行，并在 RHCOS 系统安装到磁盘之前和/或之后执行设置任务。这个方法只用于必须执行一次且之后不能再次应用的任务，如不能使用机器配置进行的高级分区。
  对于 PXE 或 ISO 引导，您可以创建 Ignition 配置，**APPEND ignition.config.url=** 选项，以标识 Ignition 配置的位置。您还需要附加 **ignition.firstboot ignition.platform.id=metal** 或者 **ignition.config.url** 选项。

1.1.11.3.3.1. 在 RHCOS ISO 中嵌入 Ignition 配置

您可以直接嵌入 RHCOS ISO 镜像中的 live 安装 Ignition 配置。引导 ISO 镜像后，内嵌的配置将自动应用。

流程

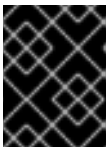
1. 从以下镜像镜像页面下载 **coreos-installer** 二进制文件：
   https://mirror.openshift.com/pub/openshift-v4/clients/coreos-installer/latest/。
2. 检索 RHCOS ISO 镜像和 Ignition 配置文件，并将其复制到可访问的目录中，如 **/mnt**:
   -

```
# cp rhcos-<version>-live.x86_64.iso bootstrap.ign /mnt/
# chmod 644 /mnt/rhcos-<version>-live.x86_64.iso
```

3. 运行以下命令将 Ignition 配置嵌入 ISO 中：

```
# ./coreos-installer iso ignition embed -i /mnt/bootstrap.ign \
    /mnt/rhcos-<version>-live.x86_64.iso
```

现在，您以使用该 ISO 使用指定的 live 安装 Ignition 配置来安装 RHCOS。



重要

不支持且不推荐使用 **coreos-installer iso ignition embed** 来嵌入由 **openshift-installer** 生成的文件，如 **bootstrap.ign**、**master.ign** 和 **worker.ign**。

4. 要显示嵌入的 Ignition 配置的内容并将其定向到文件中，请运行：

```
# ./coreos-installer iso ignition show /mnt/rhcos-<version>-live.x86_64.iso > mybootstrap.ign
```

```
# diff -s bootstrap.ign mybootstrap.ign
```

输**出示例**

```
Files bootstrap.ign and mybootstrap.ign are identical
```

5. 要删除 Ignition 配置并将 ISO 返回到其 pristine 状态（因此您可以重复使用它），请运行：

```
# ./coreos-installer iso ignition remove /mnt/rhcos-<version>-live.x86_64.iso
```

现在，您可以将另一个 Ignition 配置嵌入到 ISO 中，或者在其 pristine 状态下使用 ISO。

1.1.11.3.4. 高级 RHCOS 安装参考

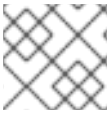
本节演示了网络配置和其他高级选项，允许您修改 Red Hat Enterprise Linux CoreOS（RHCOS）手动安装过程。下表描述了您可以与 RHCOS live installer 和 **coreos-installer** 命令一起使用的内核参数和命令行选项。

RHCOS 启动提示下的路由和绑定选项
如果从 ISO 镜像安装 RHCOS，您可以在引导该镜像时手动添加内核参数以配置节点的网络。如果没有使用网络参数，则安装默认为使用 DHCP。

重要

添加网络参数时，还必须添加 **rd.neednet=1** 内核参数。

下表描述了如何为实时 ISO 安装使用 **ip=** 、 **nameserver=** 和 **bond=** 内核参数。



注意

在添加内核参数时顺序非常重要： **ip=**，**nameserver=**，然后 **bond=**。

ISO 的路由和绑定选项

下表提供了配置 Red Hat Enterprise Linux CoreOS（RHCOS）节点网络的示例。这些是在系统引导过程中传递给 **dracut** 工具的网络选项。有关 **dracut** 支持的网络选项的详情，请参考 **dracut.cmdline** 手册页。

描述	例子
要配置一个 IP 地址，可以使用 DHCP(**ip=dhcp**)或者设置单独的静态 IP 地址(**ip=<host_ip>**)。然后在每个节点上指定 DNS 服务器 IP 地址(**nameserver=<dns_ip>**)。这个示例设置： • 节点的 IP 地址为 **10.10.10.2** • 网关地址为 **10.10.10.254** • 子网掩码为 **255.255.255.0** • 主机名为 **core0.example.com** • DNS 服务器地址为 **4.4.4.41**	ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none nameserver=4.4.4.41
通过指定多个 **ip=** 条目来指定多个网络接口。	ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none ip=10.10.10.3::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none
可选：您可以通过设置一个 **rd.route=** 值来配置到额外网络的路由。 如果额外网络网关与主要网络网关不同，则默认网关必须是主要网络网关。	配置默认网关： ip=::10.10.10.254:::: 为额外网络配置路由： rd.route=20.20.20.0/24:20.20.20.254:enp2s0
在单一接口中禁用 DHCP，比如当有两个或者多个网络接口时，且只有一个接口被使用。	ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none ip=::::core0.example.com:enp2s0:none
您可以将系统中 DHCP 和静态 IP 配置与多个网络接口结合在一起。	ip=enp1s0:dhcp ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none

描述	例子
可选： 您可以使用 **vlan=** 参数在单独的接口上配置 VLAN。	在网络接口中配置 VLAN 并使用静态 IP 地址： ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0.100:none vlan=enp2s0.100:enp2s0 在网络接口中配置 VLAN 并使用 DHCP： ip=enp2s0.100:dhcp vlan=enp2s0.100:enp2s0
您可以为每个服务器添加一个 **nameserver=** 条目来提供多个 DNS 服务器。	nameserver=1.1.1.1 nameserver=8.8.8.8
可选：使用 **bond=** 选项支持将多个网络接口绑定到一个接口。在这两个示例中： - 配置绑定接口的语法为：**bond=name[:network_interfaces][:options]** - *name* 是绑定设备名称（**bond0**），*network_interfaces* 代表用逗号分开的物理（以太网）接口(**em1,em2**)的列表，*options* 是用逗号分开的绑定选项列表。输入 **modinfo bonding** 查看可用选项。 - 当使用 **bond=** 创建绑定接口时，您必须指定如何分配 IP 地址以及绑定接口的其他信息。	要将绑定的接口配置为使用 DHCP，请将绑定的 IP 地址设置为 **dhcp**。例如： bond=bond0:em1,em2:mode=active-backup ip=bond0:dhcp 要将绑定接口配置为使用静态 IP 地址，请输入您需要的特定 IP 地址以及相关信息。例如： bond=bond0:em1,em2:mode=active-backup ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0:none
可选： 您可以使用 **vlan=** 参数在绑定接口上配置 VLAN。	使用 VLAN 配置绑定接口并使用 DHCP： ip=bond0.100:dhcp bond=bond0:em1,em2:mode=active-backup vlan=bond0.100:bond0 使用 VLAN 配置绑定接口，并使用静态 IP 地址： ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0.100:none bond=bond0:em1,em2:mode=active-backup vlan=bond0.100:bond0

描述	例子
可选：使用 **team=** 参数将网络团队用作绑定的替代选择。在本例中： • 配置组接口的语法为：**team=name[:network_interfaces]** *name* 是团队设备名称（**team0**），*network_interfaces* 代表用逗号分开的物理（以太网）接口（**em1、em2**）。  **注意** 当 RHCOS 切换到即将发布的 RHEL 版本时，团队计划被弃用。如需更多信息，请参阅 Red Hat 知识库文章。	配置网络团队： team=team0:em1,em2 ip=team0:dhcp

coreos.inst 引导选项用于 ISO 或 PXE 安装

虽然您可以将大多数标准安装引导参数传递给 live 安装程序，但也有一些特定于 RHCOS live 安装程序的参数。

- 对于 ISO，可以通过中断 RHCOS 安装程序来添加这些选项。
- 对于 PXE 或 iPXE，这些选项必须在启动 PXE 内核前添加到 **APPEND** 行中。您无法中断实时 PXE 安装。

下表显示了用于 ISO 和 PXE 安装的 RHCOS live installer 引导选项。

表 1.10. coreos.inst 引导选项

参数	描述
coreos.inst.install_dev	必需。要安装的系统中的块设备。虽然可以使用 **sda** 这样的相对路径，但建议使用完整路径，如 **/dev/sda**。
coreos.inst.ignition_url	可选：嵌入到已安装系统中的 Ignition 配置的 URL如果没有指定 URL，则不会嵌入 Ignition 配置。
coreos.inst.save_partlabel	可选：在安装过程中要保留的分区压缩标签。允许使用 glob 风格的通配符。指定分区不需要存在。
coreos.inst.save_partindex	可选：在安装过程中完成要保留的分区的分离索引。可以使用 **m-n** 指定范围，**m** 或 **n** 可以被省略。指定分区不需要存在。
coreos.inst.insecure	可选：将 **coreos.inst.image_url** 指定的 OS 镜像提交取消签名。

参数	描述
coreos.inst.image_url	可选：下载并安装指定的 RHCOS 镜像。 ● 这个参数不应该在生产环境中使用，而是只用于调试目的。 ● 虽然在 RHCOS 的安装版本与 live 介质的版本不匹配时可以使用这个参数，但建议使用与您要安装版本匹配的介质。 ● 如果您使用的是 **coreos.inst.image_url**。还必须使用 **coreos.inst.insecure**。这是因为，裸机介质没有为 OpenShift Container Platform 进行 GPG 签名。 ● 只支持 HTTP 和 HTTPS 协议。
coreos.inst.skip_reboot	可选：安装后该系统不会重启。安装完成后，您会收到提示，提示您检查在安装过程中发生的情况。这个参数不应该在生产环境中使用，而是只用于调试目的。
coreos.inst.platform_id	可选：安装 RHCOS 镜像的平台的 Ignition 平台 ID。默认为 **metal**。这个选项决定是否从云供应商（如 VMware）请求 Ignition 配置。例如：**coreos.inst.platform_id=vmware**。
ignition.config.url	可选：用于实时启动的 Ignition 配置的 URL。例如，它可以用来定制调用 **coreos-installer** 的方式，或者用来在安装前或安装后运行代码。这与 **coreos.inst.ignition_url**（这是已安装系统的 Ignition 配置）不同。

ISO 安装的 coreos-installer 选项

您还可以直接从命令行调用 **coreos-installer** 命令来安装 RHCOS。上表中的内核参数提供了在引导时自动调用 **coreos-installer** 的快捷方式，但您可以在 shell 提示符运行时将类似的参数直接传递给 **coreos-installer**。

下表显示了您可以在实时安装过程中从 shell 提示符传递给 **coreos-installer** 命令的选项和子命令。

表 1.11. CoreOS-installer 命令行选项、参数和子命令

命令行选项	
选项	**描述**
-u, **--image-url <url>**	手动指定镜像 URL。
-f, **--image-file <path>**	手动指定本地镜像文件。
-i, --ignition-file <path>	从文件中嵌入 Ignition 配置。

-I, --ignition-url <URL>	从 URL 嵌入 Ignition 配置。
--ignition-hash <digest>	Ignition config 的 **type-value** 的文摘值。
-p, --platform <name>	覆盖 Ignition 平台 ID。
--append-karg <arg>…	附加默认内核参数。
--delete-karg <arg>…	删除默认内核参数。
-n, --copy-network	从安装环境中复制网络配置。  **重要** **copy-network** 选项只复制 **/etc/NetworkManager/system-connections** 下的网络配置。特别是，它不会复制系统主机名。
--network-dir <path>	使用 **-n**。默认为 **/etc/NetworkManager/system-connections/**。
--save-partlabel <lx>..	使用这个标签 glob 保存分区。
--save-partindex <id>…	使用这个数值或者范围保存分区。
--offline	强制离线安装。
--insecure	跳过签名验证。
--insecure-ignition	允许没有 HTTPS 或 hash 的 Ignition URL。
--architecture <name>	目标 CPU 架构。默认为 **x86_64**。
--preserve-on-error	出现错误时不清除分区表。
-h, --help	打印帮助信息。
命令行参数	
参数	**描述**
<device>	目的设备。
CoreOS-installer 嵌入的 Ignition 命令	
命令	**描述**

$ coreos-installer iso ignition embed <options> --ignition-file <file_path> <ISO_image>	在 ISO 镜像中嵌入 Ignition 配置。
coreos-installer iso ignition show <options> <ISO_image>	显示来自 ISO 镜像的内嵌 Ignition 配置。
coreos-installer iso ignition remove <options> <ISO_image>	从 ISO 镜像中删除嵌入的 Ignition 配置。

coreos-installer ISO Ignition 选项

选项	描述
-f, --force	覆盖现有的 Ignition 配置。
-i, --ignition-file <path>	要使用的 Ignition 配置。默认为 **stdin**。
-o, --output <path>	将 ISO 写入到一个新输出文件。
-h，--help	打印帮助信息。

coreos-installer PXE Ignition 命令

命令	描述
请注意，不是所有子命令都接受这些选项。	
coreos-installer pxe ignition wrap <options>	在镜像中嵌套 Ignition 配置。
coreos-installer pxe ignition unwrap <options> <image_name>	显示在镜像中嵌套的 Ignition 配置。
coreos-installer pxe ignition unwrap <options> <initrd_name>	在 **initrd** 镜像中显示嵌套的 Ignition 配置。

coreos-installer PXE Ignition 选项

选项	描述
-i, --ignition-file <path>	要使用的 Ignition 配置。默认为 **stdin**。
-o, --output <path>	将 ISO 写入到一个新输出文件。
-h，--help	打印帮助信息。

1.1.12. 创建集群

要创建 OpenShift Container Platform 集群，请等待您通过安装程序生成的 Ignition 配置文件所置备的机器上完成 bootstrap 过程。

先决条件

- 为集群创建所需的基础架构。
- 已获得安装程序并为集群生成了 Ignition 配置文件。
- 已使用 Ignition 配置文件为集群创建 RHCOS 机器。
- 您的机器可直接访问互联网，或者可以使用 HTTP 或 HTTPS 代理。

流程

1. 监控 bootstrap 过程：

```
$ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \ 1
    --log-level=info 2
```

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

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

输出示例

```
INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...
INFO API v1.19.0 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
```

Kubernetes API 服务器提示已在 control plane 机器上完成 bootstrap 时，命令运行成功。

2. bootstrap 过程完成后，请从负载均衡器中删除 bootstrap 机器。

重要

此时您必须从负载均衡器中删除 bootstrap 机器。您还可以删除或重新格式化机器本身。

1.1.13. 使用 CLI 登录到集群

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

先决条件

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

流程

1. 导出 **kubeadmin** 凭证：

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

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

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输**出示例**

```
system:admin
```

1.1.14. 批准机器的证书签名请求

将机器添加到集群时，会为您添加的每台机器生成两个待处理证书签名请求（CSR）。您必须确认这些 CSR 已获得批准，或根据需要自行批准。客户端请求必须首先被批准，然后是服务器请求。

先决条件

- 您已将机器添加到集群中。

流程

1. 确认集群可以识别这些机器：

```
$ oc get nodes
```

输**出示例**

```
NAME      STATUS    ROLES   AGE  VERSION
master-0  Ready     master  63m  v1.19.0
master-1  Ready     master  63m  v1.19.0
master-2  Ready     master  64m  v1.19.0
```

输出将列出您创建的所有机器。

注意

在一些 CSR 被批准前，以上输出可能不包括计算节点（也称为 worker 节点）。

2. 检查待处理的 CSR，并确保可以看到添加到集群中的每台机器都有 **Pending** 或 **Approved** 状态的客户端请求：

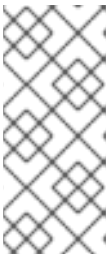
```
$ oc get csr
```

输**出示例**

```
NAME        AGE    REQUESTOR                                                   CONDITION
csr-8b2br   15m    system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
csr-8vnps   15m    system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
...
```

在本例中，两台机器加入了集群。您可能会在列表中看到更多已批准的 CSR。

3. 如果 CSR 没有获得批准，请在所添加机器的所有待处理 CSR 都处于 **Pending** 状态后，为您的集群机器批准这些 CSR：



注意

由于 CSR 会自动轮转，因此请在将机器添加到集群后一小时内批准您的 CSR。如果没有在一小时内批准，证书将会轮转，每个节点将会存在多个证书。您必须批准所有这些证书。批准客户端 CSR 后， Kubelet 为服务证书创建辅助 CSR，这需要手动批准。然后，如果 Kubelet 请求具有相同参数的新证书，则 **machine-approver** 会自动批准后续服务证书续订请求。

注意

对于在未启用机器 API 的平台中运行的集群，如裸机和其他用户置备的基础架构，必须采用一种方法自动批准 kubelet 提供证书请求（CSR）。如果没有批准请求，则 **oc exec**、**oc rsh** 和 **oc logs** 命令将无法成功，因为 API 服务器连接到 kubelet 时需要服务证书。与 Kubelet 端点联系的任何操作都需要此证书批准。这个方法必须监视新的 CSR，确认 CSR 由 **system:node** 或 **system:admin** 组中的 **node-bootstrapper** 服务帐户提交，并确认节点的身份。

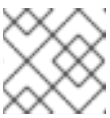
- 若要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```



注意

在有些 CSR 被批准前，一些 Operator 可能无法使用。

4. 现在，您的客户端请求已被批准，您必须查看添加到集群中的每台机器的服务器请求：

```
$ oc get csr
```

输**出示例**

```
NAME        AGE    REQUESTOR                                                   CONDITION
```

```
csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal
Pending
csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal
Pending
...
```

5. 如果剩余的 CSR 没有被批准，且处于 **Pending** 状态，请批准集群机器的 CSR：

   - 若要单独批准，请对每个有效的 CSR 运行以下命令：

     ```
     $ oc adm certificate approve <csr_name> 1
     ```

     1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

   - 要批准所有待处理的 CSR，请运行以下命令：

     ```
     $ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
     {{end}}{{end}}' | xargs oc adm certificate approve
     ```

6. 批准所有客户端和服务器 CSR 后，器将处于 **Ready** 状态。运行以下命令验证：

   ```
   $ oc get nodes
   ```

   输**出示例**

   ```
   NAME      STATUS    ROLES   AGE  VERSION
   master-0  Ready     master  73m  v1.20.0
   master-1  Ready     master  73m  v1.20.0
   master-2  Ready     master  74m  v1.20.0
   worker-0  Ready     worker  11m  v1.20.0
   worker-1  Ready     worker  11m  v1.20.0
   ```

注意

批准服务器 CSR 后可能需要几分钟时间让机器转换为 **Ready** 状态。

其他信息

- 如需有关 CSR 的更多信息，请参阅证书签名请求。

1.1.15. 初始 Operator 配置

在 control plane 初始化后，您必须立即配置一些 Operator 以便它们都可用。

先决条件

- 您的 control plane 已初始化。

流程

1. 观察集群组件上线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.6.0   True        False         False      3h56m
cloud-credential                           4.6.0   True        False         False      29h
cluster-autoscaler                         4.6.0   True        False         False      29h
config-operator                            4.6.0   True        False         False      6h39m
console                                    4.6.0   True        False         False      3h59m
csi-snapshot-controller                    4.6.0   True        False         False      4h12m
dns                                        4.6.0   True        False         False      4h15m
etcd                                       4.6.0   True        False         False      29h
image-registry                             4.6.0   True        False         False      3h59m
ingress                                    4.6.0   True        False         False      4h30m
insights                                   4.6.0   True        False         False      29h
kube-apiserver                             4.6.0   True        False         False      29h
kube-controller-manager                    4.6.0   True        False         False      29h
kube-scheduler                             4.6.0   True        False         False      29h
kube-storage-version-migrator              4.6.0   True        False         False      4h2m
machine-api                                4.6.0   True        False         False      29h
machine-approver                           4.6.0   True        False         False      6h34m
machine-config                             4.6.0   True        False         False      3h56m
marketplace                                4.6.0   True        False         False      4h2m
monitoring                                 4.6.0   True        False         False      6h31m
network                                    4.6.0   True        False         False      29h
node-tuning                                4.6.0   True        False         False      4h30m
openshift-apiserver                        4.6.0   True        False         False      3h56m
openshift-controller-manager               4.6.0   True        False         False      4h36m
openshift-samples                          4.6.0   True        False         False      4h30m
operator-lifecycle-manager                 4.6.0   True        False         False      29h
operator-lifecycle-manager-catalog         4.6.0   True        False         False      29h
operator-lifecycle-manager-packageserver   4.6.0   True        False         False      3h59m
service-ca                                 4.6.0   True        False         False      29h
storage                                    4.6.0   True        False         False      4h30m
```

2. 配置不可用的 Operator。

1.1.15.1. 安装过程中删除的镜像 registry

在不提供可共享对象存储的平台上，OpenShift Image Registry Operator bootstraps 本身的状态是 **Removed**。这允许 **openshift-installer** 在这些平台类型上完成安装。

将 **ManagementState** Image Registry Operator 配置从 **Removed** 改为 **Managed**。

注意

Prometheus 控制台提供了一个 **ImageRegistryRemoved** 警报，例如：

"Image Registry has been removed.**ImageStreamTags**, **BuildConfigs** and **DeploymentConfigs** which reference **ImageStreamTags** may not work as expected.Please configure storage and update the config to **Managed** state by editing configs.imageregistry.operator.openshift.io."

1.1.15.2. 镜像 registry 存储配置

对于不提供默认存储的平台，Image Registry Operator 最初将不可用。安装后，您必须配置 registry 使用的存储，这样 Registry Operator 才可用。

示配置生产集群所需的持久性卷的说明。如果适用，显示有关将空目录配置为存储位置的说明，该位置只可用于非生产集群。

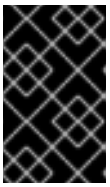
另外还提供了在升级过程中使用 **Recreate** rollout 策略来允许镜像 registry 使用块存储类型的说明。

1.1.15.2.1. 为裸机和其他手动安装配置 registry 存储

作为集群管理员，在安装后需要配置 registry 来使用存储。

先决条件

- 具有 Cluster Administrator 权限
- 使用手动置备的 Red Hat Enterprise Linux CoreOS（RHCOS）节点（如裸机）的集群。
- 为集群置备的持久性存储，如 Red Hat OpenShift Container Storage。



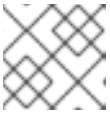
重要

如果您只有一个副本，OpenShift Container Platform 支持对镜像 registry 存储的 **ReadWriteOnce** 访问。要部署支持高可用性的、带有两个或多个副本的镜像 registry，需要 **ReadWriteMany** 访问设置。

- 必须具有 100Gi 容量。

流程

1. 为了配置 registry 使用存储，需要修改 **configs.imageregistry/cluster** 资源中的 **spec.storage.pvc** 。



注意

使用共享存储时，请查看您的安全设置以防止被外部访问。

2. 验证您没有 registry pod:

```
$ oc get pod -n openshift-image-registry
```

注意

如果存储类型为 **emptyDIR**，则副本数不能超过 **1**。

3. 检查 registry 配置：

```
$ oc edit configs.imageregistry.operator.openshift.io
```

输出示例

-

```
storage:
  pvc:
    claim:
```

将 **claim** 字段留空以允许自动创建一个 **image-registry-storage** PVC。

4. 检查 **clusteroperator** 的状态：

```
$ oc get clusteroperator image-registry
```

5. 确保您的 registry 设置为 manage，以启用镜像的构建和推送。

- 运行：

```
$ oc edit configs.imageregistry/cluster
```

然后将行改

```
managementState: Removed
```

为

```
managementState: Managed
```

1.1.15.2.2. 在非生产集群中配置镜像 registry 存储

您必须为 Image Registry Operator 配置存储。对于非生产集群，您可以将镜像 registry 设置为空目录。如果您这样做，重启 registry 后会丢失所有镜像。

流程

- 将镜像 registry 存储设置为空目录：

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec":{"storage":{"emptyDir":{}}}}'
```

警告

仅可为非生产集群配置这个选项。

如果在 Image Registry Operator 初始化其组件前运行此命令，**oc patch** 命令会失败并显示以下错误：

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster" not found
```

等待几分钟，然后再次运行该命令。

1.1.15.2.3. 配置块 registry 存储

要允许镜像 registry 在作为集群管理员升级过程中使用块存储类型，您可以使用 **Recreate** rollout 策略。

重要

支持块存储卷，但不建议将其与生产环境中的镜像 registry 一起使用。在块存储上配置 registry 的安装不具有高可用性，因为 registry 无法拥有多个副本。

流程

1. 要将镜像 registry 存储设置为块存储类型，对 registry 进行补丁，使其使用 **Recreate** rollout 策略，并只使用一个（**1**）副本运行：

```
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec":
{"rolloutStrategy":"Recreate","replicas":1}}'
```

2. 为块存储设备置备 PV，并为该卷创建 PVC。请求的块卷使用 ReadWriteOnce（RWO）访问模式。

3. 编辑 registry 配置，使其引用正确的 PVC。

1.1.16. 在用户置备的基础架构上完成安装

完成 Operator 配置后，可以在您提供的基础架构上完成集群安装。

先决条件

- 您的 control plane 已初始化。
- 已完成初始 Operator 配置。

流程

1. 使用以下命令确认所有集群组件都已在线：

```
$ watch -n5 oc get clusteroperators
```

输**出示例**

```
NAME                                       VERSION AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                              4.6.0   True        False         False      3h56m
cloud-credential                            4.6.0   True        False         False      29h
cluster-autoscaler                          4.6.0   True        False         False      29h
config-operator                             4.6.0   True        False         False      6h39m
console                                    4.6.0   True        False         False      3h59m
csi-snapshot-controller                          4.6.0   True        False         False      4h12m
dns                                        4.6.0   True        False         False      4h15m
etcd                                       4.6.0   True        False         False      29h
image-registry                              4.6.0   True        False         False      3h59m
ingress                                     4.6.0   True        False         False      4h30m
insights                                    4.6.0   True        False         False      29h
kube-apiserver                                   4.6.0   True        False         False      29h
```

```
kube-controller-manager                    4.6.0   True        False         False      29h
kube-scheduler                             4.6.0   True        False         False      29h
kube-storage-version-migrator              4.6.0   True        False         False      4h2m
machine-api                                4.6.0   True        False         False      29h
machine-approver                           4.6.0   True        False         False      6h34m
machine-config                             4.6.0   True        False         False      3h56m
marketplace                                4.6.0   True        False         False      4h2m
monitoring                                 4.6.0   True        False         False      6h31m
network                                    4.6.0   True        False         False      29h
node-tuning                                4.6.0   True        False         False      4h30m
openshift-apiserver                        4.6.0   True        False         False      3h56m
openshift-controller-manager               4.6.0   True        False         False      4h36m
openshift-samples                          4.6.0   True        False         False      4h30m
operator-lifecycle-manager                 4.6.0   True        False         False      29h
operator-lifecycle-manager-catalog         4.6.0   True        False         False      29h
operator-lifecycle-manager-packageserver   4.6.0   True        False         False      3h59m
service-ca                                 4.6.0   True        False         False      29h
storage                                    4.6.0   True        False         False      4h30m
```

或者，通过以下命令，如果所有集群都可用您会接到通知。它还检索并显示凭证：

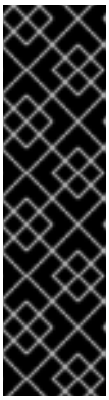
```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

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

输**出示例**

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

Cluster Version Operator 完成从 Kubernetes API 服务器部署 OpenShift Container Platform 集群时，命令运行成功。



重要

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

2. 确认 Kubernetes API 服务器正在与 pod 通信。
   a. 要查看所有 pod 的列表，请使用以下命令：

```
$ oc get pods --all-namespaces
```

输**出示例**

```
NAMESPACE                  NAME                                        READY   STATUS
```

```
RESTARTS   AGE
openshift-apiserver-operator      openshift-apiserver-operator-85cb746d55-zqhs8   1/1
Running   1          9m
openshift-apiserver               apiserver-67b9g                                 1/1     Running     0
3m
openshift-apiserver               apiserver-ljcmx                                 1/1     Running     0
1m
openshift-apiserver               apiserver-z25h4                                 1/1     Running     0
2m
openshift-authentication-operator authentication-operator-69d5d8bf84-vh2n8       1/1
Running   0          5m
...
```

b. 使用以下命令，查看上一命令的输出中所列 pod 的日志：

```
$ oc logs <pod_name> -n <namespace> 1
```

1 指定 pod 名称和命名空间，如上一命令的输出中所示。

如果 pod 日志显示，Kubernetes API 服务器可以与集群机器通信。

1.1.17. OpenShift Container Platform 的 Telemetry 访问

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

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

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

1.1.18. 后续步骤

- 自定义集群。
- 如果需要，您可以选择不使用远程健康报告。
- 设置 registry 并配置 registry 存储。

1.2. 使用网络自定义在裸机上安装集群

在 OpenShift Container Platform 版本 4.6 中，您可以使用自定义的网络配置选项在裸机环境中安装集群。通过自定义网络配置，您的集群可以与环境中现有的 IP 地址分配共存，并与现有的 MTU 和 VXLAN 配置集成。

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

1.2.1. 先决条件

- 查看有关 OpenShift Container Platform 安装和更新流程的详细信息。
- 如果使用防火墙，您必须将其配置为访问 Red Hat Insights。

1.2.2. OpenShift Container Platform 的互联网访问

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

您必须具有以下互联网访问权限：

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

重要

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

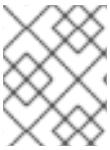
1.2.3. 具有用户置备基础架构的集群的机器要求

对于含有用户置备的基础架构的集群，您必须部署所有所需的机器。

1.2.3.1. 所需的机器

最小的 OpenShift Container Platform 集群需要下列主机：

- 一个临时 bootstrap 机器
- 三台 control plane 或 master 机器
- 至少两台计算机器，也称为 worker 机器。如果您正在运行三节点集群，则支持运行零个计算机器。不支持运行一台计算机器。



注意

集群要求 bootstrap 机器在三台 control plane 机器上部署 OpenShift Container Platform 集群。您可在安装集群后删除 bootstrap 机器。

重要

要保持集群的高可用性，请将独立的物理主机用于这些集群机器。

bootstrap 和 control plane 机器必须使用 Red Hat Enterprise Linux CoreOS (RHCOS) 作为操作系统。但是，计算机器可以在 Red Hat Enterprise Linux CoreOS(RHCOS)或 Red Hat Enterprise Linux(RHEL)7.9 间进行选择。

请注意，RHCOS 基于 Red Hat Enterprise Linux（RHEL） 8，并继承其所有硬件认证和要求。请查看Red Hat Enterprise Linux 技术功能及限制 。

1.2.3.2. 网络连接要求

所有 Red Hat Enterprise Linux CoreOS (RHCOS) 机器在启动过程中需要 **initramfs** 中的网络从 Machine Config Server 获取 Ignition 配置文件。在初次启动过程中，需要一个 DHCP 服务器或设置了静态 IP 地址来建立网络连接，以下载它们的 Ignition 配置文件。另外，集群中的每个 OpenShift Container Platform 节点都必须有权访问网络时间协议（NTP）服务器。如果 DHCP 服务器提供 NTP 服务器信息，Red Hat Enterprise Linux CoreOS（RHCOS）机器上的 chrony 时间服务会读取信息，并可与 NTP 服务器同步时钟。

1.2.3.3. 最低资源要求

每台集群机器都必须满足以下最低要求：

机器	操作系统	CPU [1]	RAM	存储	IOPS [2]
bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS 或 RHEL 7.9	2	8 GB	100 GB	300

1. 当未启用并发多线程(SMT)或超线程时，一个 CPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比率：（每个内核数的线程）LIMIT 插槽 = CPU。
2. OpenShift Container Platform 和 Kubernetes 对磁盘性能非常敏感，建议使用更快的存储速度，特别是 control plane 节点上需要 10 ms p99 fsync 持续时间的 etcd。请注意，在许多云平台上，存储大小和 IOPS 可一起扩展，因此您可能需要过度分配存储卷来获取足够的性能。

1.2.3.4. 证书签名请求管理

在使用您置备的基础架构时，集群只能有限地访问自动机器管理，因此您必须提供一种在安装后批准集群证书签名请求 (CSR) 的机制。**kube-controller-manager** 只能批准 kubelet 客户端 CSR。 **machine-approver** 无法保证使用 kubelet 凭证请求的提供证书的有效性，因为它不能确认是正确的机器发出了该请求。您必须决定并实施一种方法，以验证 kubelet 提供证书请求的有效性并进行批准。

1.2.4. 创建用户置备的基础架构

在部署采用用户置备的基础架构的 OpenShift Container Platform 集群前，您必须创建底层基础架构。

先决条件

- 在为集群创建支持基础架构之前，请参阅OpenShift Container Platform 4.x Tested Integrations页。

流程

1. 在每个节点上配置 DHCP 或设置静态 IP 地址。
2. 提供所需的负载均衡器。
3. 配置机器的端口。

4. 配置 DNS。

5. 确保网络可以正常工作。

1.2.4.1. 用户置备的基础架构对网络的要求

所有 Red Hat Enterprise Linux CoreOS（RHCOS）机器在启动过程中需要 **initramfs** 中的网络从机器配置服务器获取 Ignition 配置。

在初次启动过程中，需要一个 DHCP 服务器或集群中的每个机器都设置了静态 IP 地址来建立网络连接，以下载它们的 Ignition 配置文件。

建议您使用 DHCP 服务器为集群进行长期机器管理。确保 DHCP 服务器已配置为向集群机器提供持久 IP 地址和主机名。

Kubernetes API 服务器必须能够解析集群机器的节点名称。如果 API 服务器和 worker 节点位于不同的区域中，您可以配置默认 DNS 搜索区域，以便 API 服务器能够解析节点名称。另一种支持的方法是始终在节点对象和所有 DNS 请求中使用完全限定域名来指代主机。

您必须配置机器间的网络连接，以便集群组件进行通信。每台机器都必须能够解析集群中所有其他机器的主机名。

表 1.12. 所有机器到所有机器

协议	端口	描述
ICMP	N/A	网络可访问性测试
TCP	**1936**	指标
	9000-**9999**	主机级别的服务，包括端口 **9100**-**9101** 上的节点导出器和端口 **9099** 上的 Cluster Version Operator。
	10250-**10259**	Kubernetes 保留的默认端口
	10256	openshift-sdn
UDP	**4789**	VXLAN 和 Geneve
	6081	VXLAN 和 Geneve
	9000-**9999**	主机级别的服务，包括端口 **9100**-**9101** 上的节点导出器。
TCP/UDP	**30000**-**32767**	Kubernetes 节点端口

表 1.13. 要通过控制平面的所有机器

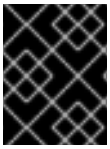
协议	端口	描述
TCP	**6443**	Kubernetes API

表 1.14. control plane 机器到 control plane 机器

协议	端口	描述
TCP	**2379-2380**	etcd 服务器和对等端口

网络拓扑要求

您为集群置备的基础架构必须满足下列网络拓扑要求。



重要

OpenShift Container Platform 要求所有节点都能访问互联网，以便为平台容器提取镜像并向红帽提供遥测数据。

负载均衡器

在安装 OpenShift Container Platform 前，您必须置备两个满足以下要求的负载均衡器：

1. **API 负载均衡器**：提供一个通用端点，供用户（包括人和机器）与平台交互和配置。配置以下条件：
   - 只适用于第 4 层负载均衡。这可被称为 Raw TCP、SSL Passthrough 或者 SSL 桥接模式。如果使用 SSL Bridge 模式，必须为 API 路由启用 Server Name Indication（SNI）。
   - 无状态负载平衡算法。这些选项根据负载均衡器的实现而有所不同。

重要

不要为 API 负载均衡器配置会话持久性。

在负载均衡器的前端和后台配置以下端口：

表 1.15. API 负载均衡器

端口	后端机器（池成员）	内部	外部	描述
6443	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。您必须为 API 服务器健康检查探测配置 **/readyz** 端点。	X	X	Kubernetes API 服务器
22623	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。	X		机器配置服务器

注意

负载均衡器必须配置为，从 API 服务器关闭 **/readyz** 端点到从池中删除 API 服务器实例时最多需要 30 秒。在 **/readyz** 返回错误或处于健康状态后的时间范围内，端点必须被删除或添加。每 5 秒或 10 秒探测一次，有两个成功请求处于健康状态，三个成为不健康的请求经过测试。

2. **应用程序入口负载均衡器**:提供来自集群外部的应用程序流量流量的 Ingress 点。配置以下条件：

   - 只适用于第 4 层负载均衡。这可被称为 Raw TCP、SSL Passthrough 或者 SSL 桥接模式。如果使用 SSL Bridge 模式，您必须为 Ingress 路由启用Server Name Indication（SNI）。
   - 建议根据可用选项以及平台上托管的应用程序类型，使用基于连接的或者基于会话的持久性。

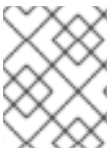
   在负载均衡器的前端和后台配置以下端口：

   表 1.16. 应用程序入口负载均衡器

端口	后端机器（池成员）	内部	外部	描述
443	默认运行入口路由器 Pod、计算或 worker 的机器。	X	X	HTTPS 流量
80	默认运行入口路由器 Pod、计算或 worker 的机器。	X	X	HTTP 流量

提示

如果负载均衡器可以看到客户端的真实 IP 地址，启用基于 IP 的会话持久性可提高使用端到端 TLS 加密的应用程序的性能。



注意

OpenShift Container Platform 集群需要正确配置入口路由器。control plane 初始化后，您必须配置入口路由器。

NTP 配置

OpenShift Container Platform 集群默认配置为使用公共网络时间协议（NTP）服务器。如果要使用本地企业 NTP 服务器，或者集群部署在断开连接的网络中，您可以将集群配置为使用特定的时间服务器。如需更多信息，请参阅*配置 chrony 时间服务*的文档。

如果 DHCP 服务器提供 NTP 服务器信息，Red Hat Enterprise Linux CoreOS（RHCOS）机器上的 chrony 时间服务会读取信息，并可与 NTP 服务器同步时钟。

其他资源

- 配置 chrony 时间服务

1.2.4.2. 用户置备 DNS 要求

DNS 用于名称解析和反向名称解析。DNS A/AAAA 或 CNAME 记录用于名称解析，PTR 记录用于反向解析名称。反向记录很重要，因为 Red Hat Enterprise Linux CoreOS（RHCOS）使用反向记录为所有节点

设置主机名。另外，反向记录用于生成 OpenShift Container Platform 需要操作的证书签名请求（CSR）。

采用用户置备的基础架构的 OpenShift Container Platform 集群需要以下 DNS 记录。在每一记录中，**<cluster_name>** 是集群名称，**<base_domain>** 则是您在 **install-config.yaml** 文件中指定的集群基域。完整的 DNS 记录采用如下格式: **<component>.<cluster_name>.<base_domain>.**。

表 1.17. 所需的 DNS 记录

组件	记录	描述
Kubernetes API	**api.<cluster_name>. <base_domain>.**	添加 DNS A/AAAA 或 CNAME 记录，以及 DNS PTR 记录，以识别 control plane 机器的负载均衡器。这些记录必须由集群外的客户端以及集群中的所有节点解析。
	api-int.<cluster_name>. <base_domain>.	添加 DNS A/AAAA 或 CNAME 记录，以及 DNS PTR 记录，以识别 control plane 机器的负载均衡器。这些记录必须可以从集群中的所有节点解析。  **重要** API 服务器必须能够根据在 Kubernetes 中记录的主机名解析 worker 节点。如果 API 服务器无法解析节点名称，则代理的 API 调用会失败，且您无法从 pod 检索日志。
Routes	***.apps.<cluster_name>. <base_domain>.**	添加通配符 DNS A/AAAA 或 CNAME 记录，指向以运行入口路由器 Pod 的机器（默认为 worker 节点）为目标的负载均衡器。这些记录必须由集群外的客户端以及集群中的所有节点解析。
bootstrap	**bootstrap.<cluster_name>. <base_domain>.**	添加 DNS A/AAAA 或 CNAME 记录，以及 DNS PTR 记录来识别 bootstrap 机器。这些记录必须由集群中的节点解析。
Master 主机	**<master><n>. <cluster_name>. <base_domain>.**	DNS A/AAAA 或 CNAME 记录，以识别 control plane 节点（也称为 master 节点）的每台机器。这些记录必须由集群中的节点解析。
Worker 主机	**<worker><n>. <cluster_name>. <base_domain>.**	添加 DNS A/AAAA 或 CNAME 记录，以识别 worker 节点的每台机器。这些记录必须由集群中的节点解析。

提示

您可以使用 **nslookup <hostname>** 命令来验证名称解析。您可以使用 **dig -x <ip_address>** 命令来验证 PTR 记录的反向名称解析。

下面的 BIND 区文件的例子展示了关于名字解析的 A 记录的例子。这个示例的目的是显示所需的记录。这个示例不是为选择一个名称解析服务提供建议。

例 1.3. DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
   2019070700 ; serial
   3H  ; refresh (3 hours)
   30M  ; retry (30 minutes)
   2W  ; expiry (2 weeks)
   1W )  ; minimum (1 week)
 IN NS ns1.example.com.
 IN MX 10 smtp.example.com.
;
;
ns1 IN A 192.168.1.5
smtp IN A 192.168.1.5
;
helper IN A 192.168.1.5
helper.ocp4 IN A 192.168.1.5
;
; The api identifies the IP of your load balancer.
api.ocp4  IN A 192.168.1.5
api-int.ocp4  IN A 192.168.1.5
;
; The wildcard also identifies the load balancer.
*.apps.ocp4  IN A 192.168.1.5
;
; Create an entry for the bootstrap host.
bootstrap.ocp4 IN A 192.168.1.96
;
; Create entries for the master hosts.
master0.ocp4  IN A 192.168.1.97
master1.ocp4  IN A 192.168.1.98
master2.ocp4  IN A 192.168.1.99
;
; Create entries for the worker hosts.
worker0.ocp4  IN A 192.168.1.11
worker1.ocp4  IN A 192.168.1.7
;
;EOF
```

下面的 BIND 区文件示例显示了反向名字解析的 PTR 记录示例。

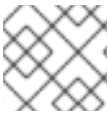
例 1.4. 反向记录的 DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
   2019070700 ; serial
   3H  ; refresh (3 hours)
   30M  ; retry (30 minutes)
   2W  ; expiry (2 weeks)
   1W )  ; minimum (1 week)
 IN NS ns1.example.com.
;
; The syntax is "last octet" and the host must have an FQDN
; with a trailing dot.
97 IN PTR master0.ocp4.example.com.
```

```
98 IN PTR master1.ocp4.example.com.
99 IN PTR master2.ocp4.example.com.
;
96 IN PTR bootstrap.ocp4.example.com.
;
5 IN PTR api.ocp4.example.com.
5 IN PTR api-int.ocp4.example.com.
;
11 IN PTR worker0.ocp4.example.com.
7 IN PTR worker1.ocp4.example.com.
;
;EOF
```

1.2.5. 生成 SSH 私钥并将其添加到代理中

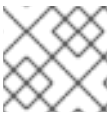
如果要在集群上执行安装调试或灾难恢复，则必须为 **ssh-agent** 和安装程序提供 SSH 密钥。您可以使用此密钥访问公共集群中的 bootstrap 机器来排除安装问题。



注意

在生产环境中，您需要进行灾难恢复和调试。

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



注意

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

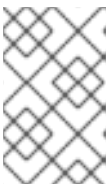
流程

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

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

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

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。



注意

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

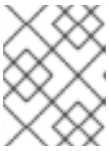
2. 作为后台任务启动 **ssh-agent** 进程：

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

-

输出示例

```
Agent pid 31874
```



注意

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

3. 将 SSH 私钥添加到 **ssh-agent**：

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

输出示例

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

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

后续步骤

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

1.2.6. 获取安装程序

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

先决条件

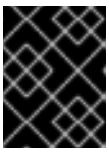
- 运行 Linux 或 macOS 的计算机，本地磁盘空间为 500 MB

流程

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

重要

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



重要

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

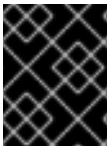
4. 提取安装程序。例如，在使用 Linux 操作系统的计算机上运行以下命令：

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

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

1.2.7. 通过下载二进制文件安装 OpenShift CLI

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



重要

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

1.2.7.1. 在 Linux 上安装 OpenShift CLI

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

流程

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

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

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

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

1.2.7.2. 在 Windows 上安装 OpenShift CLI

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

流程

1. 进入到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。

3. 单击 **OpenShift v4.6 Windows 客户端**条目旁边的 **Download Now**，再保存文件。
4. 使用 ZIP 程序解压存档。
5. 把 **oc** 二进制代码放到 **PATH** 中的目录中。
   要查看您的 **PATH**，请打开命令提示窗口并执行以下命令：

```
C:\> path
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

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

1.2.7.3. 在 macOS 上安装 OpenShift CLI

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

流程

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

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

1.2.8. 手动创建安装配置文件

对于使用用户置备的基础架构的 OpenShift Container Platform 安装，您必须手动生成安装配置文件。

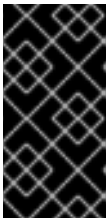
先决条件

- 获取 OpenShift Container Platform 安装程序和集群的访问令牌。

流程

1. 创建用来存储您所需的安装资产的安装目录：

```
$ mkdir <installation_directory>
```



重要

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

2. 自定义以下 **install-config.yaml** 文件模板，并将它保存到 **<installation_directory>** 中。

注意

此配置文件必须命名为 **install-config.yaml**。

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

重要

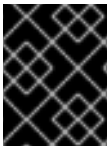
install-config.yaml 文件会在安装过程的下一步骤中消耗掉。现在必须备份它。

1.2.8.1. 安装配置参数

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

注意

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



重要

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

1.2.8.1.1. 所需的配置参数

下表描述了所需的安装配置参数：

表 1.18. 所需的参数

参数	描述	值
apiVersion	**install-config.yaml** 内容的 API 版本。当前版本是 **v1**。安装程序还可能支持旧的 API 版本。	字符串

参数	描述	值
baseDomain	云供应商的基域。此基础域用于创建到 OpenShift Container Platform 集群组件的路由。集群的完整 DNS 名称是 **baseDomain** 和 **metadata.name** 参数值的组合，其格式为 **<metadata.name>.<baseDomain>**。	完全限定域名或子域名，如 **example.com**。
metadata	Kubernetes 资源 **ObjectMeta**，其中只消耗 **name** 参数。	对象
metadata.name	集群的名称。集群的 DNS 记录是 **{{.metadata.name}}.{{.baseDomain}}** 的子域。	小写字母,连字符(-)和句点(.)的字符串，如 **dev**。
platform	执行安装的具体平台配置：**aws**、**baremetal**、**azure**、**openstack**、**ovirt**、**vsphere**。有关 **platform.<platform>** 参数的额外信息，请参考下表来了解您的具体平台。	对象
pullSecret	从 Red Hat OpenShift Cluster Manager 获取 pull secret，验证从 Quay.io 等服务中下载 OpenShift Container Platform 组件的容器镜像。	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

1.2.8.1.2. 网络配置参数

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

只支持 IPv4 地址。

表 1.19. 网络参数

参数	描述	值
networking	集群网络的配置。	对象 **注意** 您不能在安装后修改 **networking** 对象指定的参数。
networking.networkType	要安装的集群网络供应商 Container Network Interface（CNI）插件。	**OpenShiftSDN** 或 **OVNKubernetes**。默认值为 **OpenShiftSDN**。
networking.clusterNetwork	pod 的 IP 地址块。 默认值为 **10.128.0.0/14**，主机前缀为 **/23**。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23
networking.clusterNetwork.cidr	使用 **networking.clusterNetwork** 时需要此项。IP 地址块。 一个 IPv4 网络。	使用 CIDR 形式的 IP 地址块。IPv4 块的前缀长度介于 **0** 到 **32** 之间。
networking.clusterNetwork.hostPrefix	分配给每个单独节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从所给的 **cidr** 中分配一个 **/23** 子网。**hostPrefix** 值 **23** 提供 510（2^(32 - 23)- 2）个 pod IP 地址。	子网前缀。 默认值为 **23**。
networking.serviceNetwork	服务的 IP 地址块。默认值为 **172.30.0.0/16**。 OpenShift SDN 和 OVN-Kubernetes 网络供应商只支持服务网络的一个 IP 地址块。	CIDR 格式具有 IP 地址块的数组。例如： networking: serviceNetwork: - 172.30.0.0/16
networking.machineNetwork	机器的 IP 地址块。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: machineNetwork: - cidr: 10.0.0.0/16

参数	描述	值
networking.machineNetwork.cidr	使用 **networking.machineNetwork** 时需要。IP 地址块。libvirt 以外的所有平台的默认值为 **10.0.0.0/16**。对于 libvirt，默认值为 **192.168.126.0/24**。	CIDR 表示法中的 IP 网络块。 例如： **10.0.0.0/16**。  **注意** 将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

1.2.8.1.3. 可选配置参数

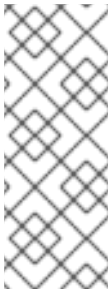
下表描述了可选安装配置参数：

表 1.20. 可选参数

参数	描述	值
additionalTrustBundle	添加到节点可信证书存储中的 PEM 编码 X.509 证书捆绑包。配置了代理时，也可以使用这个信任捆绑包。	字符串
compute	组成计算节点的机器的配置。	machine-pool 对象的数组。详情请查看以下"Machine-pool"表。
compute.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
compute.hyperthreading	是否在计算机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
compute.name	使用 **compute** 时需要此值。机器池的名称。	**worker**

参数	描述	值
compute.platform	使用 **compute** 时需要此值。使用此参数指定托管 worker 机器的云供应商。此参数值必须与 **controlPlane.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
compute.replicas	要置备的计算机器数量，也称为 worker 机器。	大于或等于 **2** 的正整数。默认值为 **3**。
controlPlane	组成 control plane 的机器的配置。	**MachinePool** 对象的数组。详情请查看以下"Machine-pool"表。
controlPlane.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
controlPlane.hyperthreading	是否在 control plane 机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
controlPlane.name	使用 **controlPlane** 时需要。机器池的名称。	**master**
controlPlane.platform	使用 **controlPlane** 时需要。使用此参数指定托管 control plane 机器的云供应商。此参数值必须与 **compute.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
controlPlane.replicas	要置备的 control plane 机器数量。	唯一支持的值是 **3**，它是默认值。

参数	描述	值
credentialsMode	Cloud Credential Operator（CCO）模式。如果没有指定任何模式，CCO 会动态地尝试决定提供的凭证的功能，在支持多个模式的平台上使用 mint 模式。  **注意** 不是所有 CCO 模式都支持所有云供应商。如需有关 CCO 模式的更多信息，请参阅*Red Hat Operator 参考指南*内容中的 *Cloud Credential Operator* 条目。	**Mint**、**Passthrough**、**Manual** 或空字符串(**""**)。
fips	启用或禁用 FIPS 模式。默认为 **false**（禁用）。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。  **重要** 只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。  **注意** 如果使用 Azure File 存储，则无法启用 FIPS 模式。	**false** 或 **true**
imageContentSources	release-image 内容的源和仓库。	对象数组。包括一个 **source** 以及可选的 **mirrors**，如下表所示。
imageContentSources.source	使用 **imageContentSources** 时需要。指定用户在镜像拉取规格中引用的仓库。	字符串
imageContentSources.mirrors	指定可能还包含同一镜像的一个或多个仓库。	字符串数组

参数	描述	值
publish	如何发布或公开集群的面向用户的端点，如 Kubernetes API、OpenShift 路由。	**Internal** 或 **External**。默认值为 **External**。 在非云平台上不支持将此字段设置为 **Internal**。  **重要** 如果将字段的值设为 **Internal**，集群将无法运行。如需更多信息，请参阅 BZ#1953035。
sshKey	用于验证集群机器访问的 SSH 密钥或密钥。  **注意** 对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。	一个或多个密钥。例如： `sshKey:` `  <key1>` `  <key2>` `  <key3>`

1.2.8.2. 裸机 install-config.yaml 文件示例

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

```
apiVersion: v1
baseDomain: example.com (1)
compute: (2)
- hyperthreading: Enabled (3)
  name: worker
  replicas: 0 (4)
controlPlane: (5)
  hyperthreading: Enabled (6)
  name: master
  replicas: 3 (7)
metadata:
  name: test (8)
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14 (9)
    hostPrefix: 23 (10)
  networkType: OpenShiftSDN
```

```
  serviceNetwork: 11
  - 172.30.0.0/16
platform:
  none: {} 12
fips: false 13
pullSecret: '{"auths": ...}' 14
sshKey: 'ssh-ed25519 AAAA...' 15
```

1 集群的基域。所有 DNS 记录都必须是这个基域的子域，并包含集群名称。

2 5 **controlPlane** 部分是一个单映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，**controlPlane** 部分的第一行则不可以连字符开头。只使用一个 control plane 池。

3 6 是否要启用或禁用并发多线程（SMT）或**超线程**。默认情况下，启用 SMT 可提高机器内核的性能。您可以通过将参数值设为 **Disabled** 来禁用。如果禁用 SMT，则必须在所有集群机器中禁用它，其中包括 control plane 和计算机器。

注意

默认启用并发多线程（SMT）。如果在 BIOS 设置中没有启用 SMT，**hyperthreading** 参数不会起作用。

重要

如果您禁用 **hyperthreading**（无论是在 BIOS 中还是在 **install-config.yaml** 中），请确保您对可能会造成的机器性能显著降低的情况有所考虑。

4 **replicas** 参数的值必须设置为 **0**。此参数控制集群为您创建和管理的 worker 数量，使用用户置备的基础架构时集群不会执行这些功能。在完成 OpenShift Container Platform 安装前，您必须手动为集群部署 worker 机器。

7 您添加到集群的 control plane 机器数量。由于集群将这个值用作集群中 etcd 端点的数量，因此该值必须与您部署的 control plane 机器数量匹配。

8 您在 DNS 记录中指定的集群名称。

9 从中分配 pod IP 地址的 IP 地址块。此块不得与现有的物理网络重叠。这些 IP 地址用于 pod 网络。如果您需要从外部网络访问 pod，请配置负载均衡器和路由器来管理流量。

注意

类 E CIDR 范围保留给以后使用。要使用 Class E CIDR 范围，您必须确保您的网络环境接受 Class E CIDR 范围内的 IP 地址。

10 分配给每个单独节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从所给的 **cidr** 中分配一个 **/23** 子网，这样就能有 510 (2^(32 - 23) - 2) 个 Pod IP 地址。如果您需要从外部网络访问节点，请配置负载均衡器和路由器来管理流量。

11 用于服务 IP 地址的 IP 地址池。您只能输入一个 IP 地址池。此块不得与现有的物理网络重叠。如果您需要从外部网络访问服务，请配置负载均衡器和路由器来管理流量。

12 您必须将平台设置为 **none**。您不能为您的平台提供额外的平台配置变量。

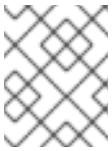
13 是否启用或禁用 FIPS 模式。默认情况下不启用 FIPS 模式。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes

重要

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

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

15 Red Hat Enterprise Linux CoreOS (RHCOS) 中 **core** 用户的默认 SSH 密钥的公钥部分。



注意

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

1.2.9. 网络配置阶段

当在安装前指定集群配置时，在安装过程中的几个阶段可以修改网络配置：

阶段 1

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

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

注意

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

阶段 2

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

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

1.2.10. 指定高级网络配置

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

重要

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

先决条件

- 创建 **install-config.yaml** 文件并完成对其所做的任何修改。
- 为集群生成 Ignition 配置文件。

流程

1. 进入包含安装程序的目录并创建清单：

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

其中：

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

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

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

其中：

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

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

为 **OpenShift SDN 网络供应商指定不同的 VXLAN 端口**

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

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

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

1.2.11. Cluster Network Operator 配置

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

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

clusterNetwork

从中分配 pod IP 地址的 IP 地址池。

serviceNetwork

服务的 IP 地址池。

defaultNetwork.type

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

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

1.2.11.1. Cluster Network Operator 配置对象

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

表 1.21. Cluster Network Operator 配置对象

字段	类型	Description
metadata.name	**字符串**	CNO 对象的名称。这个名称始终是 **cluster**。
spec.clusterNetwork	**数组**	用于指定从哪些 IP 地址块分配 Pod IP 地址以及分配给集群中每个节点的子网前缀长度的列表。例如： spec: clusterNetwork: - cidr: 10.128.0.0/19 hostPrefix: 23 - cidr: 10.128.32.0/19 hostPrefix: 23 此值是只读的，并在 **install-config.yaml** 文件中指定。

字段	类型	Description
spec.serviceNetwork	**数组**	服务的 IP 地址块。OpenShift SDN 和 OVN-Kubernetes Container Network Interface（CNI）网络供应商只支持服务网络具有单个 IP 地址块。例如： spec: serviceNetwork: - 172.30.0.0/14 此值是只读的，并在 **install-config.yaml** 文件中指定。
spec.defaultNetwork	**对象**	为集群网络配置 Container Network Interface（CNI）集群网络供应商。
spec.kubeProxyConfig	**对象**	此对象的字段指定 kube-proxy 配置。如果您使用 OVN-Kubernetes 集群网络供应商，则 kube-proxy 的配置不会起作用。

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

表 1.22. defaultNetwork 对象

字段	类型	Description
type	**字符串**	**OpenShiftSDN** 或 **OVNKubernetes**。在安装过程中选择了集群网络供应商。集群安装后无法更改这个值。 **注意** OpenShift Container Platform 默认使用 OpenShift SDN Container Network Interface（CNI）集群网络供应商。
openshiftSDNConfig	**对象**	此对象仅对 OpenShift SDN 集群网络供应商有效。
ovnKubernetesConfig	**对象**	此对象仅对 OVN-Kubernetes 集群网络供应商有效。

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

表 1.23. openshiftSDNConfig 对象

字段	类型	Description

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

OpenShift SDN 配置示例

```
defaultNetwork:
  type: OpenShiftSDN
  openshiftSDNConfig:
    mode: NetworkPolicy
    mtu: 1450
    vxlanPort: 4789
```

配置 OVN-Kubernetes CNI 集群网络供应商

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

表 1.24. ovnKubernetesConfig 对象

字段	类型	Description

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

OVN-Kubernetes 配置示例

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

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

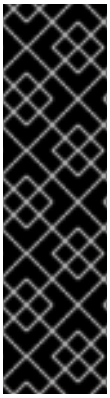
表 1.25. kubeProxyConfig 对象

字段	类型	Description
iptablesSyncPeriod	**字符串**	**iptables** 规则的刷新周期。默认值为 **30s**。有效的后缀包括 **s**、**m** 和 **h**，具体参见 Go **time** 软件包文档。 **注意** 由于 OpenShift Container Platform 4.3 及更高版本中引进了性能上的改进，现在不再需要调整 **iptablesSyncPeriod** 参数。

字段	类型	Description
proxyArguments.iptables-min-sync-period	**数组**	刷新 **iptables** 规则前的最短时长。此字段确保刷新的频率不会过于频繁。有效的后缀包括 **s**、**m** 和 **h**，具体参见 Go **time** 软件包。默认值为： kubeProxyConfig: proxyArguments: iptables-min-sync-period: - 0s

1.2.12. 创建 Ignition 配置文件

由于需要手工启动集群机器，因此您必须生成 Ignition 配置文件，集群需要它来创建其机器。



重要

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

先决条件

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

流程

- 获取 Ignition 配置文件：

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

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

重要

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

该目录中将生成以下文件：

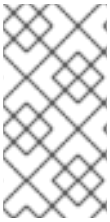
```
.
```

```
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

1.2.13. 安装 RHCOS 并启动 OpenShift Container Platform bootstrap 过程

要在您置备的裸机基础架构上安装 OpenShift Container Platform，您必须在机器上安装 Red Hat Enterprise Linux CoreOS（RHCOS）。安装 RHCOS 时，您必须为 OpenShift Container Platform 安装程序生成的机器类型提供 Ignition 配置文件。如果您配置了合适的网络、DNS 和负载均衡基础架构，OpenShift Container Platform bootstrap 过程会在 RHCOS 机器重启后自动开始。

要在机器上安装 RHCOS，请按照以下步骤使用 ISO 镜像或网络 PXE 启动。



注意

本安装文档中包括的计算节点部署步骤特定于 RHCOS。如果您选择部署基于 RHEL 的计算节点，您将接管所有操作系统生命周期管理和维护，包括执行系统更新、应用补丁和完成所有其他必要的任务。RHEL 7 计算机器的使用已弃用，计划在以后的 OpenShift Container Platform 4 发行版本中删除。

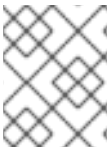
您可以使用以下方法在 ISO 和 PXE 安装过程中配置 RHCOS:

- 内核参数： 您可以使用内核参数来提供特定于安装的信息。例如，您可以指定上传到 HTTP 服务器的 RHCOS 安装文件的位置，以及您要安装的节点类型的 Ignition 配置文件的位置。对于 PXE 安装，您可以使用 **APPEND** 参数将参数传递给实时安装程序的内核。对于 ISO 安装，您可以中断实时安装引导过程来添加内核参数。在这两种安装情况下，您可以使用特殊的 **coreos.inst.*** 参数来指示实时安装程序，以及标准安装引导参数来打开或关闭标准内核服务。

- Ignition 配置：OpenShift Container Platform Ignition 配置文件（***.ign**）特定于您要安装的节点类型。您可以在 RHCOS 安装过程中传递 bootstrap、control plane 或计算节点 Ignition 配置文件的位置，以便在第一次引导时生效。特殊情况下，您可以创建单独的、有限的 Ignition 配置来传递给 Live 系统。该 Ignition 配置可以执行特定任务，如在安装完成后向置备系统报告成功。这个特殊 Ignition 配置由 **coreos-installer** 使用，用于首次启动安装的系统。不要直接向 live ISO 提供标准 control plane 和计算节点 Ignition 配置。

- **coreos-installer**：您可以将 live ISO 安装程序引导到 shell 提示符，这可让您在首次引导前以多种方式准备持久性系统。特别是，您可以运行 **coreos-installer** 命令来识别包括的工件、使用磁盘分区以及设置联网。在有些情况下，您可以配置 live 系统上的功能并将其复制到安装的系统中。

使用 ISO 安装还是 PXE 安装要根据您的具体情况而定。PXE 安装需要可用的 DHCP 服务并进行更多准备，但可以使安装过程更自动化。ISO 安装是一个更手动的过程，如果您设置的机器较多，则可能不方便。



注意

自 OpenShift Container Platform 4.6 起，RHCOS ISO 和其他安装工件支持在带有 4K 扇区的磁盘上安装。

1.2.13.1. 使用 ISO 镜像创建 Red Hat Enterprise Linux CoreOS (RHCOS) 机器

在您置备的基础架构上安装集群前，必须先创建 RHCOS 机器供其使用。您可以使用 ISO 镜像来创建这些机器。

先决条件

- 获取集群的 Ignition 配置文件。
- 具有可从计算机以及您创建的机器访问的 HTTP 服务器的访问权限。

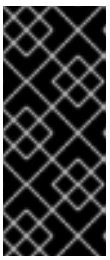
流程

1. 将安装程序创建的 control plane、计算和 bootstrap Ignition 配置文件上传到 HTTP 服务器。记下这些文件的 URL。

重要

如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

2. 从 RHCOS 镜像镜像页面获取您选择的操作系统实例安装方法所需的 RHCOS 镜像。



重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每一发行版本都有改变。您必须下载最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。如果可用，请使用与 OpenShift Container Platform 版本匹配的镜像版本。此流程只使用 ISO 镜像。此安装类型不支持 RHCOS qcow2 镜像。

ISO 文件名类似以下示例：

rhcos-<version>-live.<architecture>.iso

3. 使用 ISO 启动 RHCOS 安装。使用如下安装选项之一：
   - 将 ISO 镜像刻录到磁盘并直接启动。
   - 通过 LOM 接口使用 ISO 重定向。
4. 引导 ISO 镜像。您可以中断安装引导过程来添加内核参数。然而，在这个 ISO 过程中，您应该使用 **coreos-installer** 命令而不是添加内核参数。如果您在没有选项或中断的情况下运行 live 安装程序，安装程序将引导至 live 系统上的 shell 提示符，准备好将 RHCOS 安装到磁盘中。
5. 在运行 **coreos-installer** 前，请参阅 *高级 RHCOS 安装参考* 部分，以了解配置功能的不同方法，如网络和磁盘分区。
6. 运行 **coreos-installer** 命令。您至少必须识别节点类型的 Ignition 配置文件位置，以及您要安装到的磁盘位置。下面是一个示例：

```
$ sudo coreos-installer install \
    --ignition-url=https://host/worker.ign /dev/sda
```

7. 安装 RHCOS 后，系统会重启。系统重启过程中，它会应用您指定的 Ignition 配置文件。
8. 继续为集群创建其他机器。

重要

此刻您必须创建 bootstrap 和 control plane 机器。如果 control plane 机器不可调度（这是默认调度），则在安装集群前至少会创建两台计算机器。

1.2.13.2. 通过 PXE 或 iPXE 启动来创建 Red Hat Enterprise Linux CoreOS (RHCOS) 机器

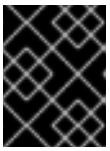
在安装使用手动置备 RHCOS 节点（如裸机）的集群前，您必须创建 RHCOS 机器供其使用。您可以使用 PXE 或 iPXE 启动来创建机器。

先决条件

- 获取集群的 Ignition 配置文件。
- 配置合适的 PXE 或 iPXE 基础架构。
- 具有 HTTP 服务器的访问权限，以便您可从计算机进行访问。

流程

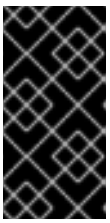
1. 将安装程序创建的 master、worker 和 bootstrap Ignition 配置文件上传到 HTTP 服务器。记下这些文件的 URL。



重要

您可以在 Ignition 配置中添加或更改配置设置，然后将其保存到 HTTP 服务器。如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

2. 从 RHCOS **镜像镜像**页面获取 **RHCOS 内核**、**initramfs** 和 **rootfs** 文件。



重要

RHCOS 工件（artifact）可能不会随着 OpenShift Container Platform 的每个发行版本而改变。您必须下载最高版本的工件，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。这个过程只使用下面描述的正确 **kernel**、**initramfs** 和 **rootfs** 工件。此安装类型不支持 RHCOS qcow2 镜像。

文件名包含 OpenShift Container Platform 版本号。它们类似以下示例：

- **kernel**: **rhcos-<version>-live-kernel-<architecture>**
- **initramfs**: **rhcos-<version>-live-initramfs.<architecture>.img**
- **rootfs**: **rhcos-<version>-live-rootfs.<architecture>.img**

3. 上传引导方法所需的额外文件：
   - 对于传统的 PXE，将 **kernel** 和 **initramfs** 文件上传到 TFTP 服务器和 **rootfs** 文件到 HTTP 服务器。
   - 对于 iPXE，将 **内核**、**initramfs** 和 **rootfs** 文件上传到 HTTP 服务器。

重要

如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

4. 配置网络启动基础架构，以便在安装 RHCOS 后机器可从本地磁盘启动。

5. 为 RHCOS 镜像配置 PXE 或 iPXE 安装。
针对您的环境修改以下示例菜单条目之一，并验证能否正确访问镜像和 Ignition 文件：

- 对于 PXE：

```
DEFAULT pxeboot
TIMEOUT 20
PROMPT 0
LABEL pxeboot
    KERNEL http://<HTTP_server>/rhcos-<version>-live-kernel-<architecture> (1)
    APPEND initrd=http://<HTTP_server>/rhcos-<version>-live-initramfs.
<architecture>.img coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-
rootfs.<architecture>.img coreos.inst.install_dev=/dev/sda
coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign (2) (3)
```

(1) 指定上传到 HTTP 服务器的 live **kernel** 文件位置。URL 必须是 HTTP、TFTP 或者 FTP ; 不支持 HTTPS 和 NFS。

(2) 如果您使用多个 NIC，请在 **ip** 选项中指定一个接口。例如，要在名为 **eno1** 的 NIC 上使用 DHCP，请设置 **ip=eno1:dhcp**。

(3) 指定上传到 HTTP 服务器的 RHCOS 文件的位置。**initrd** 参数值是 **initramfs** 文件的位置，**coreos.live.rootfs_url** 参数值是 **rootfs** 文件的位置，**coreos.inst.ignition_url** 参数值是 bootstrap Ignition 配置文件的位置。您还可以在 **APPEND** 行中添加更多内核参数来配置联网或其他引导选项。

注意

这个配置不会在使用图形控制台的机器上启用串口控制台访问。要配置不同的控制台，请在 **APPEND** 行中添加一个或多个 **console=** 参数。例如，添加 **console=tty0 console=ttyS0** 将第一个 PC 串口设置为主控制台，图形控制台作为二级控制台。如需更多信息，请参阅如何在 Red Hat Enterprise Linux 中设置串行终端和（或）控制台？

- 对于 iPXE：

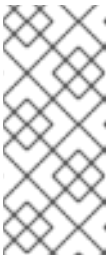
```
kernel http://<HTTP_server>/rhcos-<version>-live-kernel-<architecture> initrd=main
coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-rootfs.
<architecture>.img coreos.inst.install_dev=/dev/sda
coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign (1) (2)
initrd --name main http://<HTTP_server>/rhcos-<version>-live-initramfs.
<architecture>.img (3)
boot
```

(1) 指定上传到 HTTP 服务器的 RHCOS 文件的位置。**kernel** 参数值是 **kernel** 文件的位置，在 UEFI 系统中引导时需要 **initrd=main** 参数。 **coreos.live.rootfs_url** 参数值是 **rootfs** 文件的位置， **coreos.inst.ignition_url** 参数值则是 bootstrap Ignition 配置文件

的位置。

2 如果您使用多个 NIC，请在 **ip** 选项中指定一个接口。例如，要在名为 **eno1** 的 NIC 上使用 DHCP，请设置 **ip=eno1:dhcp**。

3 指定上传到 HTTP 服务器的 **initramfs** 文件的位置。



注意

这个配置不会在使用图形控制台的机器上启用串口控制台访问。要配置不同的控制台，请在 **kerne** 行中添加一个或多个 **console=** 参数。例如，添加 **console=tty0 console=ttyS0** 将第一个 PC 串口设置为主控制台，图形控制台作为二级控制台。如需更多信息，请参阅如何在 Red Hat Enterprise Linux 中设置串行终端和（或）控制台？

6. 如果使用 PXE UEFI，请执行以下操作：

   a. 提供启动系统所需的 **shim x64.efi** EFI 二进制文件和 **grub.cfg** 文件。

   - 通过将 RHCOS ISO 挂载到您的主机，然后将 **images/efiboot.img** 文件挂载到您的主机来提取所需的 EFI 二进制文件：

     ```
     $ mkdir -p /mnt/iso
     ```

     ```
     $ mkdir -p /mnt/efiboot
     ```

     ```
     $ mount -o loop rhcos-installer.x86_64.iso /mnt/iso
     ```

     ```
     $ mount -o loop,ro /mnt/iso/images/efiboot.img /mnt/efiboot
     ```

   - 从 **efiboot.img** 挂载点，将 **EFI/redhat/shimx64.efi** 和 **EFI/redhat/grubx64.efi** 文件复制到您的 TFTP 服务器中：

     ```
     $ cp /mnt/efiboot/EFI/redhat/shimx64.efi .
     ```

     ```
     $ cp /mnt/efiboot/EFI/redhat/grubx64.efi .
     ```

     ```
     $ umount /mnt/efiboot
     ```

     ```
     $ umount /mnt/iso
     ```

   - 将 RHCOS ISO 中包含的 **EFI/redhat/grub.cfg** 文件复制到您的 TFTP 服务器中。

   b. 编辑 **grub.cfg** 文件使其包含类似如下的参数：

   ```
   menuentry 'Install Red Hat Enterprise Linux CoreOS' --class fedora --class gnu-linux --class gnu --class os {
    linuxefi rhcos-<version>-live-kernel-<architecture> coreos.inst.install_dev=/dev/sda coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-rootfs.
   ```

```
<architecture>.img coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign
 initrdefi rhcos-<version>-live-initramfs.<architecture>.img
}
```

其中：

rhcos-<version>-live-kernel-<architecture>

指定上传到 TFTP 服务器的 **kernel** 文件。

http://<HTTP_server>/rhcos-<version>-live-rootfs.<architecture>.img

指定上传到 HTTP 服务器的 live rootfs 镜像的位置。

http://<HTTP_server>/bootstrap.ign

指定上传到 HTTP 服务器的 bootstrap Ignition 配置文件的位置。

rhcos-<version>-live-initramfs.<architecture>.img

指定上传到 TFTP 服务器的 **initramfs** 文件的位置。

注意

有关如何为 UEFI 引导配置 PXE 服务器的详情，请查看红帽知识库文章：如何为 Red Hat Enterprise Linux 配置/设置 PXE 服务器？.

7. 继续为集群创建机器。

重要

此刻您必须创建 bootstrap 和 control plane 机器。如果 control plane 机器不可调度（这是默认调度），则在安装集群前至少会创建两台计算机器。

1.2.13.3. 高级 Red Hat Enterprise Linux CoreOS（RHCOS）安装配置

为 OpenShift Container Platform 手动置备 Red Hat Enterprise Linux CoreOS（RHCOS）节点的一个关键优点是能够进行通过默认的 OpenShift Container Platform 安装方法无法进行的配置。本节介绍了您可以使用的一些技术来进行配置，其中包括：

- 将内核参数传递给实时安装程序
- 从 live 系统手动运行 **coreos-installer**
- 将 Ignition 配置嵌入 ISO 中

本节详述了与 Red Hat Enterprise Linux CoreOS（RHCOS）手动安装的高级配置相关的内容，如磁盘分区、网络以及使用 Ignition 配置的不同方式相关。

1.2.13.3.1. 使用高级网络选项进行 PXE 和 ISO 安装

OpenShift Container Platform 节点的网络默认使用 DHCP 来收集所有必要配置设置。要设置静态 IP 地址或配置特殊的设置，如绑定,您可以执行以下操作之一：

- 引导 live 安装程序时会传递特殊的内核参数。
- 使用机器配置将网络文件复制到安装的系统中。

- 使用 live installer shell 提示配置网络，然后将那些设置复制到安装的系统上，以便在安装的系统第一次引导时生效。

要配置 PXE 或 iPXE 安装，请使用以下选项之一：

- 请参阅"高级 RHCOS 安装参考"表。
- 使用机器配置将网络文件复制到安装的系统中。

要配置 ISO 安装，请使用以下步骤。

流程

1. 引导 ISO 安装程序。
2. 在 live 系统 shell 提示下，使用可用的 RHEL 工具（如 **nmcli** 或 **nmtui**）为 Live 系统配置网络。
3. 运行 **coreos-installer** 命令来安装系统，添加 **--copy-network** 选项来复制网络配置。例如：

```
$ coreos-installer install --copy-network \
    --ignition-url=http://host/worker.ign /dev/sda
```

重要

copy-network 选项只复制 **/etc/NetworkManager/system-connections** 下的网络配置。特别是，它不会复制系统主机名。

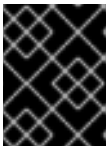
4. 重启安装的系统。

1.2.13.3.2. 磁盘分区

磁盘分区是在 Red Hat Enterprise Linux CoreOS（RHCOS）安装过程中在 OpenShift Container Platform 集群节点上创建的。特定架构的每个 RHCOS 节点都使用相同的分区布局，除非默认分区配置被覆盖。在 RHCOS 安装过程中，根文件系统的大小会增大，以使用目标设备中剩余的可用空间。

但是，在安装 OpenShift Container Platform 节点时，在两种情况下您可能需要覆盖默认分区：

- 创建单独的分区：对于在空磁盘中的 greenfield 安装，您可能想要在分区中添加单独的存储。这只在生成 **/var** 或者一个 **/var** 独立分区的子目录（如 **/var/lib/etcd** ）时被正式支持，但不支持两者。



重要

Kubernetes 只支持两个文件系统分区。如果您在原始配置中添加多个分区，Kubernetes 无法监控所有这些分区。

- 保留现有分区：对于 brownfield 安装，您要在现有节点上重新安装 OpenShift Container Platform，并希望保留从之前的操作系统中安装的数据分区，对于 **coreos-installer** 来说，引导选项和选项都允许您保留现有数据分区。

1.2.13.3.2.1. 创建一个独立的 /var 分区

通常情况下，OpenShift Container Platform 的磁盘分区应该留给安装程序。然而，在有些情况下您可能需要在文件系统的一部分中创建独立分区。

OpenShift Container Platform 支持添加单个分区来将存储附加到 **/var** 分区或 **/var** 的子目录。例如：

- **/var/lib/containers**：保存镜像相关的内容，随着更多镜像和容器添加到系统中，它所占用的存储会增加。
- **/var/lib/etcd**：保存您可能希望保持独立的数据，比如 etcd 存储的性能优化。
- **/var**：保存您希望独立保留的数据，用于特定目的（如审计）。

单独存储 **/var** 目录的内容可方便地根据需要对区域扩展存储，并可以在以后重新安装 OpenShift Container Platform 时保持该数据地完整。使用这个方法，您不必再次拉取所有容器，在更新系统时也无法复制大量日志文件。

因为 **/var** 在进行一个全新的 Red Hat Enterprise Linux CoreOS（RHCOS）安装前必需存在，所以这个流程会在 OpenShift Container Platform 安装过程的 **openshift-install** 准备阶段插入的机器配置来设置独立的 **/var** 分区。

流程

1. 创建存放 OpenShift Container Platform 安装文件的目录：

```
$ mkdir $HOME/clusterconfig
```

2. 运行 **openshift-install** 在 **manifest** 和 **openshift** 子目录中创建一组文件。在出现提示时回答系统问题：

```
$ openshift-install create manifests --dir $HOME/clusterconfig
? SSH Public Key ...
$ ls $HOME/clusterconfig/openshift/
99_kubeadmin-password-secret.yaml
99_openshift-cluster-api_master-machines-0.yaml
99_openshift-cluster-api_master-machines-1.yaml
99_openshift-cluster-api_master-machines-2.yaml
...
```

3. 创建 **MachineConfig** 对象并将其添加到 **openshift** 目录中的一个文件中。例如，把文件命名为 **98-var-partition.yaml**，将磁盘设备名称改为 **worker** 系统中存储设备的名称，并根据情况设置存储大小。这个示例将 **/var** 目录放在独立分区中：

```
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
spec:
  config:
    ignition:
      version: 3.1.0
    storage:
      disks:
      - device: /dev/<device_name> 1
        partitions:
        - label: var
          startMiB: <partition_start_offset> 2
```

```
      sizeMiB: <partition_size> 3
  filesystems:
    - device: /dev/disk/by-partlabel/var
      path: /var
      format: xfs
systemd:
  units:
    - name: var.mount 4
      enabled: true
      contents: |
        [Unit]
        Before=local-fs.target
        [Mount]
        What=/dev/disk/by-partlabel/var
        Where=/var
        Options=defaults,prjquota 5
        [Install]
        WantedBy=local-fs.target
```

1. 要分区的磁盘的存储设备名称。
2. 当在引导磁盘中添加数据分区时，推荐最少使用 25000MB。root 文件系统会自动重新定义大小使其占据所有可用空间（最多到指定的偏移值）。如果没有指定值，或者指定的值小于推荐的最小值，则生成的 root 文件系统会太小，而在以后进行的 RHCOS 重新安装可能会覆盖数据分区的开始部分。
3. 数据分区的大小（以兆字节为单位）。
4. 挂载单元的名称必须与 where = 指令中指定的目录匹配。例如，对于挂载到 **/var/lib/containers** 的文件系统，这个单元必须命名为 **var-lib-containers.mount**。
5. 必须针对用于容器存储的文件系统启用 **prjquota** 挂载选项。

注意

在创建独立 **/var** 分区时，如果不同的实例类型没有相同的设备名称，则无法将不同的实例类型用于 worker 节点。

4. 再次运行 **openshift-install**，从 **manifest** 和 **openshift** 子目录中的一组文件创建 Ignition 配置：

```
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
auth  bootstrap.ign  master.ign  metadata.json  worker.ign
```

现在，可以使用 Ignition 配置文件作为 ISO 或 PXE 手动安装过程的输入来安装 Red Hat Enterprise Linux CoreOS（RHCOS）系统。

1.2.13.3.2.2. 保留现有分区

对于 ISO 安装，您可以在 **coreos-installer** 命令行中添加可让安装程序维护一个或多个现有分区的选项。对于 PXE 安装，您可以 **APPEND coreos.inst.*** 选项来保留分区。

保存的分区可能是来自现有 OpenShift Container Platform 系统中的分区，其中包括了您希望保留的数据分区。以下是几个提示：

- 如果您保存了现有分区，且这些分区没有为 RHCOS 留下足够空间，则安装将失败但不会损害已保存的分区。
- 通过分区标签或数字识别您要保留的磁盘分区。

对于 ISO 安装

这个示例保留分区标签以**数据（data***）开头的任何分区：

```
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign \
      --save-partlabel 'data*' /dev/sda
```

以下示例演示了在运行 **coreos-installer** 时要保留磁盘上的第 6 个分区：

```
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign \
      --save-partindex 6 /dev/sda
```

这个示例保留了分区 5 及更高分区：

```
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign
      --save-partindex 5- /dev/sda
```

在前面已保存分区的示例中，**coreos-installer** 会立即重新创建分区。

对于 PXE 安装

这个 **APPEND** 选项保留分区标签以 'data'('data*')开头的所有分区：

```
coreos.inst.save_partlabel=data*
```

这个 **APPEND** 选项保留分区 5 及其后的分区：

```
coreos.inst.save_partindex=5-
```

这个 **APPEND** 选项保留分区 6:

```
coreos.inst.save_partindex=6
```

1.2.13.3.3. 标识 Ignition 配置

在进行 RHCOS 手动安装时，您可以提供两种 Ignition 配置类型，它们有不同的原因：

- **永久安装 Ignition 配置**：每个手动 RHCOS 安装都需要传递 **openshift-installer** 生成的 Ignition 配置文件之一，如 **bootstrap.ign**、**master.ign** 和 **worker.ign**，才能进行安装。

  重要

  不建议修改这些文件。

  对于 PXE 安装，您可以使用 **coreos.inst.ignition_url=** 选项在 **APPEND** 行上传递 Ignition 配置。对于 ISO 安装，在 ISO 引导至 shell 提示符后，您可以使用 **--ignition-url=** 选项在 **coreos-installer** 命令行上识别 Ignition 配置。在这两种情况下，都只支持 HTTP 和 HTTPS 协议。

- **live 安装 Ignition 配置**：此类型必须手动创建，并应该尽可能避免，因为红帽不支持它。使用此方法，Ignition 配置会传递到 live 安装介质，在引导时立即运行，并在 RHCOS 系统安装到磁盘之前和/或之后执行设置任务。这个方法只用于必须执行一次且之后不能再次应用的任务，如不能使用机器配置进行的高级分区。
  对于 PXE 或 ISO 引导，您可以创建 Ignition 配置，**APPEND ignition.config.url=** 选项，以标识 Ignition 配置的位置。您还需要附加 **ignition.firstboot ignition.platform.id=metal** 或者 **ignition.config.url** 选项。

1.2.13.3.3.1. 在 RHCOS ISO 中嵌入 Ignition 配置

您可以直接嵌入 RHCOS ISO 镜像中的 live 安装 Ignition 配置。引导 ISO 镜像后，内嵌的配置将自动应用。

流程

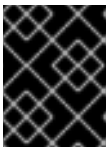
1. 从以下镜像镜像页面下载 **coreos-installer** 二进制文件：
   https://mirror.openshift.com/pub/openshift-v4/clients/coreos-installer/latest/。
2. 检索 RHCOS ISO 镜像和 Ignition 配置文件，并将其复制到可访问的目录中，如 **/mnt**:

```
# cp rhcos-<version>-live.x86_64.iso bootstrap.ign /mnt/
# chmod 644 /mnt/rhcos-<version>-live.x86_64.iso
```

3. 运行以下命令将 Ignition 配置嵌入 ISO 中：

```
# ./coreos-installer iso ignition embed -i /mnt/bootstrap.ign \
    /mnt/rhcos-<version>-live.x86_64.iso
```

   现在，您以使用该 ISO 使用指定的 live 安装 Ignition 配置来安装 RHCOS。



重要

不支持且不推荐使用 **coreos-installer iso ignition embed** 来嵌入由 **openshift-installer** 生成的文件，如 **bootstrap.ign**、**master.ign** 和 **worker.ign**。

4. 要显示嵌入的 Ignition 配置的内容并将其定向到文件中，请运行：

```
# ./coreos-installer iso ignition show /mnt/rhcos-<version>-live.x86_64.iso > mybootstrap.ign
```

```
# diff -s bootstrap.ign mybootstrap.ign
```

   输**出示例**

```
Files bootstrap.ign and mybootstrap.ign are identical
```

5. 要删除 Ignition 配置并将 ISO 返回到其 pristine 状态（因此您可以重复使用它），请运行：

```
# ./coreos-installer iso ignition remove /mnt/rhcos-<version>-live.x86_64.iso
```

   现在，您可以将另一个 Ignition 配置嵌入到 ISO 中，或者在其 pristine 状态下使用 ISO。

1.2.13.3.4. 高级 RHCOS 安装参考

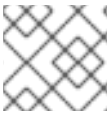
本节演示了网络配置和其他高级选项，允许您修改 Red Hat Enterprise Linux CoreOS（RHCOS）手动安装过程。下表描述了您可以与 RHCOS live installer 和 **coreos-installer** 命令一起使用的内核参数和命令行选项。

RHCOS 启动提示下的路由和绑定选项
如果从 ISO 镜像安装 RHCOS，您可以在引导该镜像时手动添加内核参数以配置节点的网络。如果没有使用网络参数，则安装默认为使用 DHCP。

重要

添加网络参数时，还必须添加 **rd.neednet=1** 内核参数。

下表描述了如何为实时 ISO 安装使用 **ip=** 、 **nameserver=** 和 **bond=** 内核参数。



注意

在添加内核参数时顺序非常重要： **ip=**，**nameserver=**，然后 **bond=**。

ISO 的路由和绑定选项

下表提供了配置 Red Hat Enterprise Linux CoreOS（RHCOS）节点网络的示例。这些是在系统引导过程中传递给 **dracut** 工具的网络选项。有关 **dracut** 支持的网络选项的详情，请参考 **dracut.cmdline** 手册页。

描述	例子
要配置一个 IP 地址，可以使用 DHCP(**ip=dhcp**)或者设置单独的静态 IP 地址(**ip=<host_ip>**)。然后在每个节点上指定 DNS 服务器 IP 地址(**nameserver=<dns_ip>**)。这个示例设置： • 节点的 IP 地址为 **10.10.10.2** • 网关地址为 **10.10.10.254** • 子网掩码为 **255.255.255.0** • 主机名为 **core0.example.com** • DNS 服务器地址为 **4.4.4.41**	ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none nameserver=4.4.4.41
通过指定多个 **ip=** 条目来指定多个网络接口。	ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none ip=10.10.10.3::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none

描述	例子
可选：您可以通过设置一个 **rd.route=** 值来配置到额外网络的路由。 如果额外网络网关与主要网络网关不同，则默认网关必须是主要网络网关。	配置默认网关： `ip=::10.10.10.254::::` 为额外网络配置路由： `rd.route=20.20.20.0/24:20.20.20.254:enp2s0`
在单一接口中禁用 DHCP，比如当有两个或者多个网络接口时，且只有一个接口被使用。	`ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none` `ip=::::core0.example.com:enp2s0:none`
您可以将系统中 DHCP 和静态 IP 配置与多个网络接口结合在一起。	`ip=enp1s0:dhcp` `ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none`
可选： 您可以使用 **vlan=** 参数在单独的接口上配置 VLAN。	在网络接口中配置 VLAN 并使用静态 IP 地址： `ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0.100:none` `vlan=enp2s0.100:enp2s0` 在网络接口中配置 VLAN 并使用 DHCP： `ip=enp2s0.100:dhcp` `vlan=enp2s0.100:enp2s0`
您可以为每个服务器添加一个 **nameserver=** 条目来提供多个 DNS 服务器。	`nameserver=1.1.1.1` `nameserver=8.8.8.8`
可选：使用 **bond=** 选项支持将多个网络接口绑定到一个接口。在这两个示例中： ● 配置绑定接口的语法为：**bond=name[:network_interfaces][:options]** ● *name* 是绑定设备名称（**bond0**），*network_interfaces* 代表用逗号分开的物理（以太网）接口(**em1,em2**)的列表，*options* 是用逗号分开的绑定选项列表。输入 **modinfo bonding** 查看可用选项。 ● 当使用 **bond=** 创建绑定接口时，您必须指定如何分配 IP 地址以及绑定接口的其他信息。	要将绑定的接口配置为使用 DHCP，请将绑定的 IP 地址设置为 **dhcp**。例如： `bond=bond0:em1,em2:mode=active-backup` `ip=bond0:dhcp` 要将绑定接口配置为使用静态 IP 地址，请输入您需要的特定 IP 地址以及相关信息。例如： `bond=bond0:em1,em2:mode=active-backup` `ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0:none`

描述	例子
可选： 您可以使用 **vlan=** 参数在绑定接口上配置 VLAN。	使用 VLAN 配置绑定接口并使用 DHCP： ip=bond0.100:dhcp bond=bond0:em1,em2:mode=active-backup vlan=bond0.100:bond0 使用 VLAN 配置绑定接口，并使用静态 IP 地址： ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0.100:none bond=bond0:em1,em2:mode=active-backup vlan=bond0.100:bond0
可选：使用 **team=** 参数将网络团队用作绑定的替代选择。在本例中： • 配置组接口的语法为：**team=name[:network_interfaces]** *name* 是团队设备名称（**team0**），*network_interfaces* 代表用逗号分开的物理（以太网）接口（**em1、em2**）。 **注意** 当 RHCOS 切换到即将发布的 RHEL 版本时，团队计划被弃用。如需更多信息，请参阅 Red Hat 知识库文章。	配置网络团队： team=team0:em1,em2 ip=team0:dhcp

coreos.inst 引导选项用于 ISO 或 PXE 安装

虽然您可以将大多数标准安装引导参数传递给 live 安装程序，但也有一些特定于 RHCOS live 安装程序的参数。

- 对于 ISO，可以通过中断 RHCOS 安装程序来添加这些选项。
- 对于 PXE 或 iPXE，这些选项必须在启动 PXE 内核前添加到 **APPEND** 行中。您无法中断实时 PXE 安装。

下表显示了用于 ISO 和 PXE 安装的 RHCOS live installer 引导选项。

表 1.26. coreos.inst 引导选项

参数	描述
coreos.inst.install_dev	必需。要安装的系统中的块设备。虽然可以使用 **sda** 这样的相对路径，但建议使用完整路径，如 **/dev/sda**。
coreos.inst.ignition_url	可选：嵌入到已安装系统中的 Ignition 配置的 UR如果没有指定 URL，则不会嵌入 Ignition 配置。

参数	描述
coreos.inst.save_partlabel	可选：在安装过程中要保留的分区压缩标签。允许使用 glob 风格的通配符。指定分区不需要存在。
coreos.inst.save_partindex	可选：在安装过程中完成要保留的分区的分离索引。可以使用 **m**-**n** 指定范围，**m** 或 **n** 可以被省略。指定分区不需要存在。
coreos.inst.insecure	可选：将 **coreos.inst.image_url** 指定的 OS 镜像提交取消签名。
coreos.inst.image_url	可选：下载并安装指定的 RHCOS 镜像。 ● 这个参数不应该在生产环境中使用，而是只用于调试目的。 ● 虽然在 RHCOS 的安装版本与 live 介质的版本不匹配时可以使用这个参数，但建议使用与您要安装版本匹配的介质。 ● 如果您使用的是 **coreos.inst.image_url**。还必须使用 **coreos.inst.insecure**。这是因为，裸机介质没有为 OpenShift Container Platform 进行 GPG 签名。 ● 只支持 HTTP 和 HTTPS 协议。
coreos.inst.skip_reboot	可选：安装后该系统不会重启。安装完成后，您会收到提示，提示您检查在安装过程中发生的情况。这个参数不应该在生产环境中使用，而是只用于调试目的。
coreos.inst.platform_id	可选：安装 RHCOS 镜像的平台的 Ignition 平台 ID。默认为 **metal**。这个选项决定是否从云供应商（如 VMware）请求 Ignition 配置。例如：**coreos.inst.platform_id=vmware**。
ignition.config.url	可选：用于实时启动的 Ignition 配置的 URL。例如，它可以用来定制调用 **coreos-installer** 的方式，或者用来在安装前或安装后运行代码。这与 **coreos.inst.ignition_url**（这是已安装系统的 Ignition 配置）不同。

ISO 安装的 coreos-installer 选项

您还可以直接从命令行调用 **coreos-installer** 命令来安装 RHCOS。上表中的内核参数提供了在引导时自动调用 **coreos-installer** 的快捷方式，但您可以在 shell 提示符运行时将类似的参数直接传递给 **coreos-installer**。

下表显示了您可以在实时安装过程中从 shell 提示符传递给 **coreos-installer** 命令的选项和子命令。

表 1.27. CoreOS-installer 命令行选项、参数和子命令

命令行选项	
选项	**描述**
-u, **--image-url <url>**	手动指定镜像 URL。
-f, **--image-file <path>**	手动指定本地镜像文件。
-i, --ignition-file <path>	从文件中嵌入 Ignition 配置。
-I, **--ignition-url <URL>**	从 URL 嵌入 Ignition 配置。
--ignition-hash <digest>	Ignition config 的 **type-value** 的文摘值。
-p, **--platform <name>**	覆盖 Ignition 平台 ID。
--append-karg <arg>…	附加默认内核参数。
--delete-karg <arg>…	删除默认内核参数。
-n, **--copy-network**	从安装环境中复制网络配置。  **重要** **copy-network** 选项只复制 **/etc/NetworkManager/system-connections** 下的网络配置。特别是，它不会复制系统主机名。
--network-dir <path>	使用 **-n**。默认为 **/etc/NetworkManager/system-connections/**。
--save-partlabel <lx>..	使用这个标签 glob 保存分区。
--save-partindex <id>…	使用这个数值或者范围保存分区。
--offline	强制离线安装。
--insecure	跳过签名验证。
--insecure-ignition	允许没有 HTTPS 或 hash 的 Ignition URL。
--architecture <name>	目标 CPU 架构。默认为 **x86_64**。
--preserve-on-error	出现错误时不清除分区表。
-h，**--help**	打印帮助信息。

命令行参数	
参数	**描述**
<device>	目的设备。
CoreOS-installer 嵌入的 Ignition 命令	
命令	**描述**
$ coreos-installer iso ignition embed <options> --ignition-file <file_path> <ISO_image>	在 ISO 镜像中嵌入 Ignition 配置。
coreos-installer iso ignition show <options> <ISO_image>	显示来自 ISO 镜像的内嵌 Ignition 配置。
coreos-installer iso ignition remove <options> <ISO_image>	从 ISO 镜像中删除嵌入的 Ignition 配置。
coreos-installer ISO Ignition 选项	
选项	**描述**
-f, **--force**	覆盖现有的 Ignition 配置。
-i, **--ignition-file <path>**	要使用的 Ignition 配置。默认为 **stdin**。
-o, **--output <path>**	将 ISO 写入到一个新输出文件。
-h, **--help**	打印帮助信息。
coreos-installer PXE Ignition 命令	
命令	**描述**
请注意，不是所有子命令都接受这些选项。	
coreos-installer pxe ignition wrap <options>	在镜像中嵌套 Ignition 配置。
coreos-installer pxe ignition unwrap <options> <image_name>	显示在镜像中嵌套的 Ignition 配置。
coreos-installer pxe ignition unwrap <options> <initrd_name>	在 **initrd** 镜像中显示嵌套的 Ignition 配置。
coreos-installer PXE Ignition 选项	

选项	描述
-i, --ignition-file <path>	要使用的 Ignition 配置。默认为 **stdin**。
-o, --output <path>	将 ISO 写入到一个新输出文件。
-h, --help	打印帮助信息。

1.2.14. 创建集群

要创建 OpenShift Container Platform 集群，请等待您通过安装程序生成的 Ignition 配置文件所置备的机器上完成 bootstrap 过程。

先决条件

- 为集群创建所需的基础架构。
- 已获得安装程序并为集群生成了 Ignition 配置文件。
- 已使用 Ignition 配置文件为集群创建 RHCOS 机器。
- 您的机器可直接访问互联网，或者可以使用 HTTP 或 HTTPS 代理。

流程

1. 监控 bootstrap 过程：

```
$ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \ 1
    --log-level=info 2
```

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

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

输出示例

```
INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...
INFO API v1.19.0 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
```

Kubernetes API 服务器提示已在 control plane 机器上完成 bootstrap 时，命令运行成功。

2. bootstrap 过程完成后，请从负载均衡器中删除 bootstrap 机器。

重要

此时您必须从负载均衡器中删除 bootstrap 机器。您还可以删除或重新格式化机器本身。

1.2.15. 使用 CLI 登录到集群

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

先决条件

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

流程

1. 导出 **kubeadmin** 凭证：

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

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

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输**出示例**

```
system:admin
```

1.2.16. 批准机器的证书签名请求

将机器添加到集群时，会为您添加的每台机器生成两个待处理证书签名请求（CSR）。您必须确认这些 CSR 已获得批准，或根据需要自行批准。客户端请求必须首先被批准，然后是服务器请求。

先决条件

- 您已将机器添加到集群中。

流程

1. 确认集群可以识别这些机器：

```
$ oc get nodes
```

输**出示例**

```
NAME     STATUS   ROLES   AGE  VERSION
master-0 Ready    master  63m  v1.19.0
master-1 Ready    master  63m  v1.19.0
master-2 Ready    master  64m  v1.19.0
```

输出将列出您创建的所有机器。

注意

在一些 CSR 被批准前，以上输出可能不包括计算节点（也称为 worker 节点）。

2. 检查待处理的 CSR，并确保可以看到添加到集群中的每台机器都有 **Pending** 或 **Approved** 状态的客户端请求：

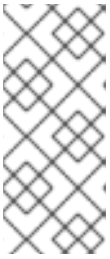
```
$ oc get csr
```

输**出示例**

```
NAME        AGE    REQUESTOR                                                   CONDITION
csr-8b2br   15m    system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
csr-8vnps   15m    system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
...
```

在本例中，两台机器加入了集群。您可能会在列表中看到更多已批准的 CSR。

3. 如果 CSR 没有获得批准，请在所添加机器的所有待处理 CSR 都处于 **Pending** 状态后，为您的集群机器批准这些 CSR：



注意

由于 CSR 会自动轮转，因此请在将机器添加到集群后一小时内批准您的 CSR。如果没有在一小时内批准，证书将会轮转，每个节点将会存在多个证书。您必须批准所有这些证书。批准客户端 CSR 后， Kubelet 为服务证书创建辅助 CSR，这需要手动批准。然后，如果 Kubelet 请求具有相同参数的新证书，则 **machine-approver** 会自动批准后续服务证书续订请求。

注意

对于在未启用机器 API 的平台中运行的集群，如裸机和其他用户置备的基础架构，必须采用一种方法自动批准 kubelet 提供证书请求（CSR）。如果没有批准请求，则 **oc exec**、**oc rsh** 和 **oc logs** 命令将无法成功，因为 API 服务器连接到 kubelet 时需要服务证书。与 Kubelet 端点联系的任何操作都需要此证书批准。这个方法必须监视新的 CSR，确认 CSR 由 **system:node** 或 **system:admin** 组中的 **node-bootstrapper** 服务帐户提交，并确认节点的身份。

- 若要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```

注意

在有些 CSR 被批准前，一些 Operator 可能无法使用。

4. 现在，您的客户端请求已被批准，您必须查看添加到集群中的每台机器的服务器请求：

```
$ oc get csr
```

输出示例

```
NAME        AGE     REQUESTOR                                                   CONDITION
csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal       Pending
csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal       Pending
...
```

5. 如果剩余的 CSR 没有被批准，且处于 **Pending** 状态，请批准集群机器的 CSR：

- 若要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs oc adm certificate approve
```

6. 批准所有客户端和服务器 CSR 后，器将处于 **Ready** 状态。运行以下命令验证：

```
$ oc get nodes
```

输出示例

```
NAME      STATUS    ROLES   AGE  VERSION
master-0  Ready     master  73m  v1.20.0
master-1  Ready     master  73m  v1.20.0
master-2  Ready     master  74m  v1.20.0
worker-0  Ready     worker  11m  v1.20.0
worker-1  Ready     worker  11m  v1.20.0
```

注意

批准服务器 CSR 后可能需要几分钟时间让机器转换为 **Ready** 状态。

其他信息

- 如需有关 CSR 的更多信息，请参阅证书签名请求。

1.2.17. 初始 Operator 配置

在 control plane 初始化后，您必须立即配置一些 Operator 以便它们都可用。

先决条件

- 您的 control plane 已初始化。

流程

1. 观察集群组件上线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.6.0   True        False         False      3h56m
cloud-credential                           4.6.0   True        False         False      29h
cluster-autoscaler                         4.6.0   True        False         False      29h
config-operator                            4.6.0   True        False         False      6h39m
console                                    4.6.0   True        False         False      3h59m
csi-snapshot-controller                    4.6.0   True        False         False      4h12m
dns                                        4.6.0   True        False         False      4h15m
etcd                                       4.6.0   True        False         False      29h
image-registry                             4.6.0   True        False         False      3h59m
ingress                                    4.6.0   True        False         False      4h30m
insights                                   4.6.0   True        False         False      29h
kube-apiserver                             4.6.0   True        False         False      29h
kube-controller-manager                    4.6.0   True        False         False      29h
kube-scheduler                             4.6.0   True        False         False      29h
kube-storage-version-migrator              4.6.0   True        False         False      4h2m
machine-api                                4.6.0   True        False         False      29h
machine-approver                           4.6.0   True        False         False      6h34m
machine-config                             4.6.0   True        False         False      3h56m
marketplace                                4.6.0   True        False         False      4h2m
monitoring                                 4.6.0   True        False         False      6h31m
network                                    4.6.0   True        False         False      29h
node-tuning                                4.6.0   True        False         False      4h30m
openshift-apiserver                        4.6.0   True        False         False      3h56m
openshift-controller-manager               4.6.0   True        False         False      4h36m
openshift-samples                          4.6.0   True        False         False      4h30m
operator-lifecycle-manager                 4.6.0   True        False         False      29h
operator-lifecycle-manager-catalog         4.6.0   True        False         False      29h
operator-lifecycle-manager-packageserver   4.6.0   True        False         False      3h59m
service-ca                                 4.6.0   True        False         False      29h
storage                                    4.6.0   True        False         False      4h30m
```

2. 配置不可用的 Operator。

1.2.17.1. 安装过程中删除的镜像 registry

在不提供可共享对象存储的平台上，OpenShift Image Registry Operator bootstraps 本身的状态是 **Removed**。这允许 **openshift-installer** 在这些平台类型上完成安装。

将 **ManagementState** Image Registry Operator 配置从 **Removed** 改为 **Managed**。

注意

Prometheus 控制台提供了一个 **ImageRegistryRemoved** 警报，例如：

"Image Registry has been removed.**ImageStreamTags**, **BuildConfigs** and **DeploymentConfigs** which reference **ImageStreamTags** may not work as expected.Please configure storage and update the config to **Managed** state by editing configs.imageregistry.operator.openshift.io."

1.2.17.2. 镜像 registry 存储配置

对于不提供默认存储的平台，Image Registry Operator 最初将不可用。安装后，您必须配置 registry 使用的存储，这样 Registry Operator 才可用。

示配置生产集群所需的持久性卷的说明。如果适用，显示有关将空目录配置为存储位置的说明，该位置只可用于非生产集群。

另外还提供了在升级过程中使用 **Recreate** rollout 策略来允许镜像 registry 使用块存储类型的说明。

1.2.17.3. 配置块 registry 存储

要允许镜像 registry 在作为集群管理员升级过程中使用块存储类型，您可以使用 **Recreate** rollout 策略。

重要

支持块存储卷，但不建议将其与生产环境中的镜像 registry 一起使用。在块存储上配置 registry 的安装不具有高可用性，因为 registry 无法拥有多个副本。

流程

1. 要将镜像 registry 存储设置为块存储类型，对 registry 进行补丁，使其使用 **Recreate** rollout 策略，并只使用一个（**1**）副本运行：

```
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec":
{"rolloutStrategy":"Recreate","replicas":1}}'
```

2. 为块存储设备置备 PV，并为该卷创建 PVC。请求的块卷使用 ReadWriteOnce（RWO）访问模式。

3. 编辑 registry 配置，使其引用正确的 PVC。

1.2.18. 在用户置备的基础架构上完成安装

完成 Operator 配置后，可以在您提供的基础架构上完成集群安装。

先决条件

- 您的 control plane 已初始化。

- 已完成初始 Operator 配置。

流程

1. 使用以下命令确认所有集群组件都已在线：

```
$ watch -n5 oc get clusteroperators
```

输**出示例**

```
NAME                                       VERSION AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.6.0   True        False         False     3h56m
cloud-credential                           4.6.0   True        False         False     29h
cluster-autoscaler                         4.6.0   True        False         False     29h
config-operator                            4.6.0   True        False         False     6h39m
console                                    4.6.0   True        False         False     3h59m
csi-snapshot-controller                    4.6.0   True        False         False     4h12m
dns                                        4.6.0   True        False         False     4h15m
etcd                                       4.6.0   True        False         False     29h
image-registry                             4.6.0   True        False         False     3h59m
ingress                                    4.6.0   True        False         False     4h30m
insights                                   4.6.0   True        False         False     29h
kube-apiserver                             4.6.0   True        False         False     29h
kube-controller-manager                    4.6.0   True        False         False     29h
kube-scheduler                             4.6.0   True        False         False     29h
kube-storage-version-migrator              4.6.0   True        False         False     4h2m
machine-api                                4.6.0   True        False         False     29h
machine-approver                           4.6.0   True        False         False     6h34m
machine-config                             4.6.0   True        False         False     3h56m
marketplace                                4.6.0   True        False         False     4h2m
monitoring                                 4.6.0   True        False         False     6h31m
network                                    4.6.0   True        False         False     29h
node-tuning                                4.6.0   True        False         False     4h30m
openshift-apiserver                        4.6.0   True        False         False     3h56m
openshift-controller-manager               4.6.0   True        False         False     4h36m
openshift-samples                          4.6.0   True        False         False     4h30m
operator-lifecycle-manager                 4.6.0   True        False         False     29h
operator-lifecycle-manager-catalog         4.6.0   True        False         False     29h
operator-lifecycle-manager-packageserver   4.6.0   True        False         False     3h59m
service-ca                                 4.6.0   True        False         False     29h
storage                                    4.6.0   True        False         False     4h30m
```

或者，通过以下命令，如果所有集群都可用您会接到通知。它还检索并显示凭证：

```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

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

输**出示例**

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

Cluster Version Operator 完成从 Kubernetes API 服务器部署 OpenShift Container Platform 集群时，命令运行成功。

重要

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

2. 确认 Kubernetes API 服务器正在与 pod 通信。

   a. 要查看所有 pod 的列表，请使用以下命令：

   ```
   $ oc get pods --all-namespaces
   ```

   输**出示例**

   ```
   NAMESPACE                       NAME                                        READY   STATUS
   RESTARTS   AGE
   openshift-apiserver-operator    openshift-apiserver-operator-85cb746d55-zqhs8   1/1
   Running    1          9m
   openshift-apiserver             apiserver-67b9g                               1/1     Running     0
   3m
   openshift-apiserver             apiserver-ljcmx                               1/1     Running     0
   1m
   openshift-apiserver             apiserver-z25h4                               1/1     Running     0
   2m
   openshift-authentication-operator authentication-operator-69d5d8bf84-vh2n8        1/1
   Running     0          5m
   ...
   ```

   b. 使用以下命令，查看上一命令的输出中所列 pod 的日志：

   ```
   $ oc logs <pod_name> -n <namespace> 1
   ```

   1 指定 pod 名称和命名空间，如上一命令的输出中所示。

   如果 pod 日志显示，Kubernetes API 服务器可以与集群机器通信。

1.2.19. OpenShift Container Platform 的 Telemetry 访问

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

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

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

1.2.20. 后续步骤

- 自定义集群。
- 如果需要，您可以选择不使用远程健康报告。
- 设置 registry 并配置 registry 存储。

1.3. 在受限网络中的裸机上安装集群

在 OpenShift Container Platform 版本 4.6 中，您可以在受限网络中置备的裸机基础架构上安装集群。

重要

虽然您可能能够按照此流程在虚拟化或云环境中部署集群，但您必须清楚非裸机平台的其他注意事项。在尝试在此类环境中安装 OpenShift Container Platform 集群前，请参阅有关在未经测试的平台上部署 OpenShift Container Platform 的指南中的信息。

1.3.1. 先决条件

- 在镜像主机上创建镜像 registry，并获取您的 OpenShift Container Platform 版本的 **imageContentSources** 数据。

  重要

  由于安装介质位于堡垒主机上，因此请使用该计算机完成所有安装步骤。

- 为集群置备持久性存储。若要部署私有镜像 registry，您的存储必须提供 ReadWriteMany 访问模式。
- 查看有关 OpenShift Container Platform 安装和更新流程的详细信息。
- 如果使用防火墙并计划使用遥测（telemetry），您必须将防火墙配置为允许集群需要访问的站点。

  注意

  如果您要配置代理，请务必也要查看此站点列表。

1.3.2. 关于在受限网络中安装

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

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

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

重要

由于用户置备安装配置的复杂性，在尝试使用用户置备的基础架构受限网络安装前，请考虑完成标准用户置备的基础架构安装。通过完成此测试安装，您可以更轻松地隔离和排查您在受限网络中安装时可能出现的问题。

1.3.2.1. 其他限制

受限网络中的集群还有以下额外限制：

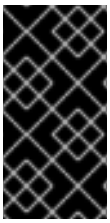
- **ClusterVersion** 状态包含一个 **Unable to retrieve available updates** 错误。
- 默认情况下，您无法使用 Developer Catalog 的内容，因为您无法访问所需的镜像流标签。

1.3.3. OpenShift Container Platform 的互联网访问

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

您必须具有以下互联网访问权限：

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



重要

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

1.3.4. 具有用户置备基础架构的集群的机器要求

对于含有用户置备的基础架构的集群，您必须部署所有所需的机器。

1.3.4.1. 所需的机器

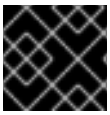
最小的 OpenShift Container Platform 集群需要下列主机：

- 一个临时 bootstrap 机器
- 三台 control plane 或 master 机器

- 至少两台计算机器，也称为 worker 机器。如果您正在运行三节点集群，则支持运行零个计算机器。不支持运行一台计算机器。

注意

集群要求 bootstrap 机器在三台 control plane 机器上部署 OpenShift Container Platform 集群。您可在安装集群后删除 bootstrap 机器。



重要

要保持集群的高可用性，请将独立的物理主机用于这些集群机器。

bootstrap 和 control plane 机器必须使用 Red Hat Enterprise Linux CoreOS (RHCOS) 作为操作系统。但是，计算机器可以在 Red Hat Enterprise Linux CoreOS(RHCOS)或 Red Hat Enterprise Linux(RHEL)7.9 间进行选择。

请注意，RHCOS 基于 Red Hat Enterprise Linux（RHEL） 8，并继承其所有硬件认证和要求。请查看Red Hat Enterprise Linux 技术功能及限制 。

1.3.4.2. 网络连接要求

所有 Red Hat Enterprise Linux CoreOS (RHCOS) 机器在启动过程中需要 **initramfs** 中的网络从 Machine Config Server 获取 Ignition 配置文件。在初次启动过程中，需要一个 DHCP 服务器或设置了静态 IP 地址来建立网络连接，以下载它们的 Ignition 配置文件。另外，集群中的每个 OpenShift Container Platform 节点都必须有权访问网络时间协议（NTP）服务器。如果 DHCP 服务器提供 NTP 服务器信息，Red Hat Enterprise Linux CoreOS（RHCOS）机器上的 chrony 时间服务会读取信息，并可与 NTP 服务器同步时钟。

1.3.4.3. 最低资源要求

每台集群机器都必须满足以下最低要求：

机器	操作系统	CPU [1]	RAM	存储	IOPS [2]
bootstrap	RHCOS	4	16 GB	100 GB	300
Control plane	RHCOS	4	16 GB	100 GB	300
Compute	RHCOS 或 RHEL 7.9	2	8 GB	100 GB	300

1. 当未启用并发多线程(SMT)或超线程时，一个 CPU 相当于一个物理内核。启用后，使用以下公式来计算对应的比率：（每个内核数的线程）LIMIT 插槽 = CPU。

2. OpenShift Container Platform 和 Kubernetes 对磁盘性能非常敏感，建议使用更快的存储速度，特别是 control plane 节点上需要 10 ms p99 fsync 持续时间的 etcd。请注意，在许多云平台上，存储大小和 IOPS 可一起扩展，因此您可能需要过度分配存储卷来获取足够的性能。

1.3.4.4. 证书签名请求管理

在使用您置备的基础架构时，集群只能有限地访问自动机器管理，因此您必须提供一种在安装后批准集群证书签名请求 (CSR) 的机制。**kube-controller-manager** 只能批准 kubelet 客户端 CSR。 **machine-**

approver 无法保证使用 kubelet 凭证请求的提供证书的有效性，因为它不能确认是正确的机器发出了该请求。您必须决定并实施一种方法，以验证 kubelet 提供证书请求的有效性并进行批准。

1.3.5. 创建用户置备的基础架构

在部署采用用户置备的基础架构的 OpenShift Container Platform 集群前，您必须创建底层基础架构。

先决条件

- 在为集群创建支持基础架构之前，请参阅OpenShift Container Platform 4.x Tested Integrations页。

流程

1. 在每个节点上配置 DHCP 或设置静态 IP 地址。
2. 提供所需的负载均衡器。
3. 配置机器的端口。
4. 配置 DNS。
5. 确保网络可以正常工作。

1.3.5.1. 用户置备的基础架构对网络的要求

所有 Red Hat Enterprise Linux CoreOS（RHCOS）机器在启动过程中需要 **initramfs** 中的网络从机器配置服务器获取 Ignition 配置。

在初次启动过程中，需要一个 DHCP 服务器或集群中的每个机器都设置了静态 IP 地址来建立网络连接，以下载它们的 Ignition 配置文件。

建议您使用 DHCP 服务器为集群进行长期机器管理。确保 DHCP 服务器已配置为向集群机器提供持久 IP 地址和主机名。

Kubernetes API 服务器必须能够解析集群机器的节点名称。如果 API 服务器和 worker 节点位于不同的区域中，您可以配置默认 DNS 搜索区域，以便 API 服务器能够解析节点名称。另一种支持的方法是始终在节点对象和所有 DNS 请求中使用完全限定域名来指代主机。

您必须配置机器间的网络连接，以便集群组件进行通信。每台机器都必须能够解析集群中所有其他机器的主机名。

表 1.28. 所有机器到所有机器

协议	端口	描述
ICMP	N/A	网络可访问性测试
TCP	**1936**	指标
	9000-**9999**	主机级别的服务，包括端口 **9100**-**9101** 上的节点导出器和端口 **9099** 上的 Cluster Version Operator。

协议	端口	描述
	10250-10259	Kubernetes 保留的默认端口
	10256	openshift-sdn
UDP	**4789**	VXLAN 和 Geneve
	6081	VXLAN 和 Geneve
	9000-9999	主机级别的服务，包括端口 **9100-9101** 上的节点导出器。
TCP/UDP	**30000-32767**	Kubernetes 节点端口

表 1.29. 要通过控制平面的所有机器

协议	端口	描述
TCP	**6443**	Kubernetes API

表 1.30. control plane 机器到 control plane 机器

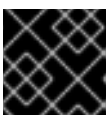
协议	端口	描述
TCP	**2379-2380**	etcd 服务器和对等端口

网络拓扑要求
您为集群置备的基础架构必须满足下列网络拓扑要求。

负载均衡器
在安装 OpenShift Container Platform 前，您必须置备两个满足以下要求的负载均衡器：

1. **API 负载均衡器**：提供一个通用端点，供用户（包括人和机器）与平台交互和配置。配置以下条件：

   - 只适用于第 4 层负载均衡。这可被称为 Raw TCP、SSL Passthrough 或者 SSL 桥接模式。如果使用 SSL Bridge 模式，必须为 API 路由启用 Server Name Indication（SNI）。
   - 无状态负载平衡算法。这些选项根据负载均衡器的实现而有所不同。

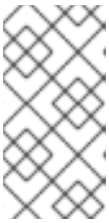
   重要

   不要为 API 负载均衡器配置会话持久性。

   在负载均衡器的前端和后台配置以下端口：

   表 1.31. API 负载均衡器

端口	后端机器（池成员）	内部	外部	描述
6443	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。您必须为 API 服务器健康检查探测配置 **/readyz** 端点。	X	X	Kubernetes API 服务器
22623	Bootstrap 和 control plane.bootstrap 机器初始化集群 control plane 后，您要从负载均衡器中删除 bootstrap 机器。	X		机器配置服务器



注意

负载均衡器必须配置为，从 API 服务器关闭 **/readyz** 端点到从池中删除 API 服务器实例时最多需要 30 秒。在 **/readyz** 返回错误或处于健康状态后的时间范围内，端点必须被删除或添加。每 5 秒或 10 秒探测一次，有两个成功请求处于健康状态，三个成为不健康的请求经过测试。

2. **应用程序入口负载均衡器**:提供来自集群外部的应用程序流量流量的 Ingress 点。配置以下条件：

   - 只适用于第 4 层负载均衡。这可被称为 Raw TCP、SSL Passthrough 或者 SSL 桥接模式。如果使用 SSL Bridge 模式，您必须为 Ingress 路由启用Server Name Indication（SNI）。
   - 建议根据可用选项以及平台上托管的应用程序类型，使用基于连接的或者基于会话的持久性。

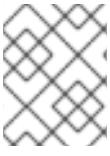
   在负载均衡器的前端和后台配置以下端口：

表 1.32. 应用程序入口负载均衡器

端口	后端机器（池成员）	内部	外部	描述
443	默认运行入口路由器 Pod、计算或 worker 的机器。	X	X	HTTPS 流量
80	默认运行入口路由器 Pod、计算或 worker 的机器。	X	X	HTTP 流量

提示

如果负载均衡器可以看到客户端的真实 IP 地址，启用基于 IP 的会话持久性可提高使用端到端 TLS 加密的应用程序的性能。



注意

OpenShift Container Platform 集群需要正确配置入口路由器。control plane 初始化后，您必须配置入口路由器。

NTP 配置

OpenShift Container Platform 集群默认配置为使用公共网络时间协议（NTP）服务器。如果要使用本地企业 NTP 服务器，或者集群部署在断开连接的网络中，您可以将集群配置为使用特定的时间服务器。如需更多信息，请参阅*配置 chrony 时间服务*的文档。

如果 DHCP 服务器提供 NTP 服务器信息，Red Hat Enterprise Linux CoreOS（RHCOS）机器上的 chrony 时间服务会读取信息，并可与 NTP 服务器同步时钟。

其他资源

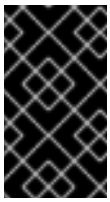
- 配置 chrony 时间服务

1.3.5.2. 用户置备 DNS 要求

DNS 用于名称解析和反向名称解析。DNS A/AAAA 或 CNAME 记录用于名称解析，PTR 记录用于反向解析名称。反向记录很重要，因为 Red Hat Enterprise Linux CoreOS（RHCOS）使用反向记录为所有节点设置主机名。另外，反向记录用于生成 OpenShift Container Platform 需要操作的证书签名请求（CSR）。

采用用户置备的基础架构的 OpenShift Container Platform 集群需要以下 DNS 记录。在每一记录中，**<cluster_name>** 是集群名称，**<base_domain>** 则是您在 **install-config.yaml** 文件中指定的集群基域。完整的 DNS 记录采用如下格式: **<component>.<cluster_name>.<base_domain>.**。

表 1.33. 所需的 DNS 记录

组件	记录	描述
Kubernetes API	**api.<cluster_name>.<base_domain>.**	添加 DNS A/AAAA 或 CNAME 记录，以及 DNS PTR 记录，以识别 control plane 机器的负载均衡器。这些记录必须由集群外的客户端以及集群中的所有节点解析。
	api-int.<cluster_name>.<base_domain>.	添加 DNS A/AAAA 或 CNAME 记录，以及 DNS PTR 记录，以识别 control plane 机器的负载均衡器。这些记录必须可以从集群中的所有节点解析。  **重要** API 服务器必须能够根据在 Kubernetes 中记录的主机名解析 worker 节点。如果 API 服务器无法解析节点名称，则代理的 API 调用会失败，且您无法从 pod 检索日志。
Routes	***.apps.<cluster_name>.<base_domain>.**	添加通配符 DNS A/AAAA 或 CNAME 记录，指向以运行入口路由器 Pod 的机器（默认为 worker 节点）为目标的负载均衡器。这些记录必须由集群外的客户端以及集群中的所有节点解析。
bootstrap	**bootstrap.<cluster_name>.<base_domain>.**	添加 DNS A/AAAA 或 CNAME 记录，以及 DNS PTR 记录来识别 bootstrap 机器。这些记录必须由集群中的节点解析。
Master 主机	**<master><n>.<cluster_name>.<base_domain>.**	DNS A/AAAA 或 CNAME 记录，以识别 control plane 节点（也称为 master 节点）的每台机器。这些记录必须由集群中的节点解析。

组件	记录	描述
Worker 主机	**<worker><n>. <cluster_name>. <base_domain>.**	添加 DNS A/AAAA 或 CNAME 记录，以识别 worker 节点的每台机器。这些记录必须由集群中的节点解析。

提示

您可以使用 **nslookup <hostname>** 命令来验证名称解析。您可以使用 **dig -x <ip_address>** 命令来验证 PTR 记录的反向名称解析。

下面的 BIND 区文件的例子展示了关于名字解析的 A 记录的例子。这个示例的目的是显示所需的记录。这个示例不是为选择一个名称解析服务提供建议。

例 1.5. DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
   2019070700 ; serial
   3H  ; refresh (3 hours)
   30M  ; retry (30 minutes)
   2W  ; expiry (2 weeks)
   1W )  ; minimum (1 week)
 IN NS ns1.example.com.
 IN MX 10 smtp.example.com.
;
;
ns1 IN A 192.168.1.5
smtp IN A 192.168.1.5
;
helper IN A 192.168.1.5
helper.ocp4 IN A 192.168.1.5
;
; The api identifies the IP of your load balancer.
api.ocp4  IN A 192.168.1.5
api-int.ocp4  IN A 192.168.1.5
;
; The wildcard also identifies the load balancer.
*.apps.ocp4  IN A 192.168.1.5
;
; Create an entry for the bootstrap host.
bootstrap.ocp4 IN A 192.168.1.96
;
; Create entries for the master hosts.
master0.ocp4  IN A 192.168.1.97
master1.ocp4  IN A 192.168.1.98
master2.ocp4  IN A 192.168.1.99
;
; Create entries for the worker hosts.
worker0.ocp4  IN A 192.168.1.11
worker1.ocp4  IN A 192.168.1.7
;
;EOF
```

下面的 BIND 区文件示例显示了反向名字解析的 PTR 记录示例。

例 1.6. 反向记录的 DNS 区数据库示例

```
$TTL 1W
@ IN SOA ns1.example.com. root (
  2019070700 ; serial
  3H  ; refresh (3 hours)
  30M  ; retry (30 minutes)
  2W  ; expiry (2 weeks)
  1W )  ; minimum (1 week)
 IN NS ns1.example.com.
;
; The syntax is "last octet" and the host must have an FQDN
; with a trailing dot.
97 IN PTR master0.ocp4.example.com.
98 IN PTR master1.ocp4.example.com.
99 IN PTR master2.ocp4.example.com.
;
96 IN PTR bootstrap.ocp4.example.com.
;
5 IN PTR api.ocp4.example.com.
5 IN PTR api-int.ocp4.example.com.
;
11 IN PTR worker0.ocp4.example.com.
7 IN PTR worker1.ocp4.example.com.
;
;EOF
```

1.3.6. 生成 SSH 私钥并将其添加到代理中

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

注意

在生产环境中，您需要进行灾难恢复和调试。

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

注意

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

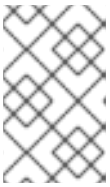
流程

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

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

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

运行此命令会在指定的位置生成不需要密码的 SSH 密钥。



注意

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

2. 作为后台任务启动 **ssh-agent** 进程：

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

输**出示例**

```
Agent pid 31874
```

注意

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

3. 将 SSH 私钥添加到 **ssh-agent**：

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

输**出示例**

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

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

后续步骤

- 在安装 OpenShift Container Platform 时，为安装程序提供 SSH 公钥。如果在您置备的基础架构上安装集群，您必须将此密钥提供给集群的机器。

1.3.7. 手动创建安装配置文件

对于使用用户置备的基础架构的 OpenShift Container Platform 安装，您必须手动生成安装配置文件。

先决条件

- 获取 OpenShift Container Platform 安装程序和集群的访问令牌。

- 获取命令输出中的 **imageContentSources** 部分来镜像存储库。
- 获取您的镜像 registry 的证书内容。

流程

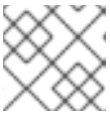
1. 创建用来存储您所需的安装资产的安装目录：

```
$ mkdir <installation_directory>
```

重要

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

2. 自定义以下 **install-config.yaml** 文件模板，并将它保存到 **<installation_directory>** 中。



注意

此配置文件必须命名为 **install-config.yaml**。

- 除非使用 RHCOS 默认信任的 registry，如 **docker.io**，否则必须在 **additionalTrustBundle** 部分中提供镜像存储库的证书内容。在大多数情况下，必须为您的镜像提供证书。
- 您必须包含命令输出中的 **imageContentSources** 部分，才能镜像存储库。

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

重要

install-config.yaml 文件会在安装过程的下一步骤中消耗掉。现在必须备份它。

1.3.7.1. 安装配置参数

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

注意

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

重要

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

1.3.7.1.1. 所需的配置参数

下表描述了所需的安装配置参数：

表 1.34. 所需的参数

参数	描述	值
apiVersion	**install-config.yaml** 内容的 API 版本。当前版本是 **v1**。安装程序还可能支持旧的 API 版本。	字符串
baseDomain	云供应商的基域。此基础域用于创建到 OpenShift Container Platform 集群组件的路由。集群的完整 DNS 名称是 **baseDomain** 和 **metadata.name** 参数值的组合，其格式为 **<metadata.name>.<baseDomain>**。	完全限定域名或子域名，如 **example.com**。
metadata	Kubernetes 资源 **ObjectMeta**，其中只消耗 **name** 参数。	对象
metadata.name	集群的名称。集群的 DNS 记录是 **{{.metadata.name}}.{{.baseDomain}}** 的子域。	小写字母,连字符(-)和句点(.)的字符串，如 **dev**。
platform	执行安装的具体平台配置：**aws**、**baremetal**、**azure**、**openstack**、**ovirt**、**vsphere**。有关 **platform.<platform>** 参数的额外信息，请参考下表来了解您的具体平台。	对象
pullSecret	从 Red Hat OpenShift Cluster Manager 获取 pull secret，验证从 Quay.io 等服务中下载 OpenShift Container Platform 组件的容器镜像。	<pre>{ "auths":{ "cloud.openshift.com":{ "auth":"b3Blb=", "email":"you@example.com" }, "quay.io":{ "auth":"b3Blb=", "email":"you@example.com" } } }</pre>

1.3.7.1.2. 网络配置参数

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

只支持 IPv4 地址。

表 1.35. 网络参数

参数	描述	值
networking	集群网络的配置。	对象 **注意** 您不能在安装后修改 **networking** 对象指定的参数。
networking.networkType	要安装的集群网络供应商 Container Network Interface（CNI）插件。	**OpenShiftSDN** 或 **OVNKubernetes**。默认值为 **OpenShiftSDN**。
networking.clusterNetwork	pod 的 IP 地址块。 默认值为 **10.128.0.0/14**，主机前缀为 **/23**。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23
networking.clusterNetwork.cidr	使用 **networking.clusterNetwork** 时需要此项。IP 地址块。 一个 IPv4 网络。	使用 CIDR 形式的 IP 地址块。IPv4 块的前缀长度介于 **0** 到 **32** 之间。
networking.clusterNetwork.hostPrefix	分配给每个单独节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从所给的 **cidr** 中分配一个 **/23** 子网。**hostPrefix** 值 **23** 提供 510（2^(32 - 23)- 2）个 pod IP 地址。	子网前缀。 默认值为 **23**。
networking.serviceNetwork	服务的 IP 地址块。默认值为 **172.30.0.0/16**。 OpenShift SDN 和 OVN-Kubernetes 网络供应商只支持服务网络的一个 IP 地址块。	CIDR 格式具有 IP 地址块的数组。例如： networking: serviceNetwork: - 172.30.0.0/16
networking.machineNetwork	机器的 IP 地址块。 如果您指定多个 IP 地址块，则块不得互相重叠。	一个对象数组。例如： networking: machineNetwork: - cidr: 10.0.0.0/16

参数	描述	值
networking.machineNetwork.cidr	使用 **networking.machineNetwork** 时需要。IP 地址块。libvirt 以外的所有平台的默认值为 **10.0.0.0/16**。对于 libvirt，默认值为 **192.168.126.0/24**。	CIDR 表示法中的 IP 网络块。 例如： **10.0.0.0/16**。 **注意** 将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。

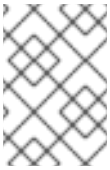
1.3.7.1.3. 可选配置参数

下表描述了可选安装配置参数：

表 1.36. 可选参数

参数	描述	值
additionalTrustBundle	添加到节点可信证书存储中的 PEM 编码 X.509 证书捆绑包。配置了代理时，也可以使用这个信任捆绑包。	字符串
compute	组成计算节点的机器的配置。	machine-pool 对象的数组。详情请查看以下"Machine-pool"表。
compute.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
compute.hyperthreading	是否在计算机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。 **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
compute.name	使用 **compute** 时需要此值。机器池的名称。	**worker**

参数	描述	值
compute.platform	使用 **compute** 时需要此值。使用此参数指定托管 worker 机器的云供应商。此参数值必须与 **controlPlane.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
compute.replicas	要置备的计算机器数量，也称为 worker 机器。	大于或等于 **2** 的正整数。默认值为 **3**。
controlPlane	组成 control plane 的机器的配置。	**MachinePool** 对象的数组。详情请查看以下"Machine-pool"表。
controlPlane.architecture	决定池中机器的指令集合架构。目前不支持异构集群，因此所有池都必须指定相同的架构。有效值为 **amd64** （默认值）。	字符串
controlPlane.hyperthreading	是否在 control plane 机器上启用或禁用并发多线程或**超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果禁用并发多线程，请确保在容量规划时考虑到机器性能可能会显著降低的问题。	**Enabled** 或 **Disabled**
controlPlane.name	使用 **controlPlane** 时需要。机器池的名称。	**master**
controlPlane.platform	使用 **controlPlane** 时需要。使用此参数指定托管 control plane 机器的云供应商。此参数值必须与 **compute.platform** 参数值匹配。	**aws**、**azure**、**gcp**、**openstack**、**ovirt**、**vsphere** 或 **{}**
controlPlane.replicas	要置备的 control plane 机器数量。	唯一支持的值是 **3**，它是默认值。

参数	描述	值
credentialsMode	Cloud Credential Operator（CCO）模式。如果没有指定任何模式，CCO 会动态地尝试决定提供的凭证的功能，在支持多个模式的平台上使用 mint 模式。  **注意** 不是所有 CCO 模式都支持所有云供应商。如需有关 CCO 模式的更多信息，请参阅*Red Hat Operator 参考指南*内容中的 *Cloud Credential Operator* 条目。	**Mint**、**Passthrough**、**Manual** 或空字符串(**""**)。
fips	启用或禁用 FIPS 模式。默认为 **false**（禁用）。如果启用了 FIPS 模式，运行 OpenShift Container Platform 的 Red Hat Enterprise Linux CoreOS (RHCOS) 机器会绕过默认的 Kubernetes 加密套件，并使用由 RHCOS 提供的加密模块。  **重要** 只有在 **x86_64** 架构中的 OpenShift Container Platform 部署支持 FIPS 验证的/Modules in Process 加密库。  **注意** 如果使用 Azure File 存储，则无法启用 FIPS 模式。	**false** 或 **true**
imageContentSources	release-image 内容的源和仓库。	对象数组。包括一个 **source** 以及可选的 **mirrors**，如下表所示。
imageContentSources.source	使用 **imageContentSources** 时需要。指定用户在镜像拉取规格中引用的仓库。	字符串

参数	描述	值
imageContentSources.mirrors	指定可能还包含同一镜像的一个或多个仓库。	字符串数组
publish	如何发布或公开集群的面向用户的端点，如 Kubernetes API、OpenShift 路由。	**Internal** 或 **External**。默认值为 **External**。 在非云平台上不支持将此字段设置为 **Internal**。 **重要** 如果将字段的值设为 **Internal**，集群将无法运行。如需更多信息，请参阅 BZ#1953035。
sshKey	用于验证集群机器访问的 SSH 密钥或密钥。 **注意** 对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。	一个或多个密钥。例如： sshKey: <key1> <key2> <key3>

1.3.7.2. 裸机 install-config.yaml 文件示例

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

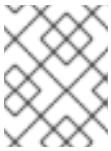
```
apiVersion: v1
baseDomain: example.com (1)
compute: (2)
- hyperthreading: Enabled (3)
  name: worker
  replicas: 0 (4)
controlPlane: (5)
  hyperthreading: Enabled (6)
  name: master
  replicas: 3 (7)
metadata:
  name: test (8)
networking:
```

```
  clusterNetwork:
  - cidr: 10.128.0.0/14 9
    hostPrefix: 23 10
  networkType: OpenShiftSDN
  serviceNetwork: 11
  - 172.30.0.0/16
platform:
  none: {} 12
fips: false 13
pullSecret: '{"auths":{"<local_registry>": {"auth": "<credentials>","email": "you@example.com"}}}' 14
sshKey: 'ssh-ed25519 AAAA...' 15
additionalTrustBundle: | 16
  -----BEGIN CERTIFICATE-----
  ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
  -----END CERTIFICATE-----
imageContentSources: 17
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-release
- mirrors:
  - <local_registry>/<local_repository_name>/release
  source: quay.io/openshift-release-dev/ocp-v4.0-art-dev
```

1 集群的基域。所有 DNS 记录都必须是这个基域的子域，并包含集群名称。

2 5 **controlPlane** 部分是一个单映射，但 **compute** 部分是一系列映射。为满足不同数据结构的要求，**compute** 部分的第一行必须以连字符 **-** 开头，**controlPlane** 部分的第一行则不可以连字符开头。只使用一个 control plane 池。

3 6 是否要启用或禁用并发多线程（SMT）或**超线程**。默认情况下，启用 SMT 可提高机器内核的性能。您可以通过将参数值设为 **Disabled** 来禁用。如果禁用 SMT，则必须在所有集群机器中禁用它，其中包括 control plane 和计算机器。



注意

默认启用并发多线程（SMT）。如果在 BIOS 设置中没有启用 SMT，**hyperthreading** 参数不会起作用。

重要

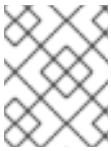
如果您禁用 **hyperthreading**（无论是在 BIOS 中还是在 **install-config.yaml** 中），请确保您对可能会造成的机器性能显著降低的情况有所考虑。

4 **replicas** 参数的值必须设置为 **0**。此参数控制集群为您创建和管理的 worker 数量，使用用户置备的基础架构时集群不会执行这些功能。在完成 OpenShift Container Platform 安装前，您必须手动为集群部署 worker 机器。

7 您添加到集群的 control plane 机器数量。由于集群将这个值用作集群中 etcd 端点的数量，因此该值必须与您部署的 control plane 机器数量匹配。

8 您在 DNS 记录中指定的集群名称。

9 从中分配 pod IP 地址的 IP 地址块。此块不得与现有的物理网络重叠。这些 IP 地址用于 pod 网络。如果您需要从外部网络访问 pod，请配置负载均衡器和路由器来管理流量。



注意

类 E CIDR 范围保留给以后使用。要使用 Class E CIDR 范围，您必须确保您的网络环境接受 Class E CIDR 范围内的 IP 地址。

10 分配给每个单独节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从所给的 **cidr** 中分配一个 **/23** 子网，这样就能有 510 (2^(32 - 23) - 2) 个 Pod IP 地址。如果您需要从外部网络访问节点，请配置负载均衡器和路由器来管理流量。

11 用于服务 IP 地址的 IP 地址池。您只能输入一个 IP 地址池。此块不得与现有的物理网络重叠。如果您需要从外部网络访问服务，请配置负载均衡器和路由器来管理流量。

12 您必须将平台设置为 **none**。您不能为您的平台提供额外的平台配置变量。

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

重要

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

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

15 Red Hat Enterprise Linux CoreOS (RHCOS) 中 **core** 用户的默认 SSH 密钥的公钥部分。

注意

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

16 提供用于镜像 registry 的证书文件内容。

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

1.3.7.3. 在安装过程中配置集群范围代理

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

注意

对于裸机安装，如果您没有从 **install-config.yaml** 文件中的 **networking.machineNetwork[].cidr** 字段指定的范围分配节点 IP 地址，您必须将其包括在 **proxy.noProxy** 字段中。

先决条件

- 您有一个现有的 **install-config.yaml** 文件。

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

注意

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

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

流程

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

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

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

注意

安装程序不支持代理的 **readinessEndpoints** 字段。

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

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

注意

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

1.3.8. 配置三节点集群

您可在没有 worker 的 OpenShift Container Platform 中安装和运行三节点集群。这为集群管理员和开发人员提供了较小的、效率更高的集群，用于开发、生产及测试。

流程

- 编辑 **install-config.yaml** 文件，将计算副本（也称为 worker 副本）数设为 **0**，如以下 **compute** 小节中所示：

```
compute:
- name: worker
  platform: {}
  replicas: 0
```

1.3.9. 创建 Kubernetes 清单和 Ignition 配置文件

由于您必须修改一些集群定义文件并要手动启动集群机器，因此您必须生成 Kubernetes 清单和 Ignition 配置文件，集群需要这两项来创建其机器。

安装配置文件转换为 Kubernetes 清单。清单嵌套到 Ignition 配置文件中，稍后用于创建集群。

重要

- 安装程序生成的 Ignition 配置文件包含在 24 小时后过期的证书，然后在过期时进行续订。如果在更新证书前关闭集群，且集群在 24 小时后重启，集群会自动恢复过期的证书。一个例外情况是，您需要手动批准待处理的 **node-bootstrapper** 证书签名请求（CSR）来恢复 kubelet 证书。如需更多信息，请参阅*从过期的 control plane 证书中恢复*的文档。

- 建议您在生成 12 小时后使用 Ignition 配置文件，因为集群安装后 24 小时证书从 16 小时轮转至 22 小时。通过在 12 小时内使用 Ignition 配置文件，您可以避免在安装过程中运行证书更新时避免安装失败。

先决条件

- 已获得 OpenShift Container Platform 安装程序。对于受限网络安装，这些文件位于您的堡垒主机上。

- 已创建 **install-config.yaml** 安装配置文件。

流程

1. 切换到包含安装程序的目录，并为集群生成 Kubernetes 清单：

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

1 对于 **<installation_directory>**，请指定含有您创建的 **install-config.yaml** 文件的安装目录。

警告

如果要安装一个三节点集群，请跳过以下步骤，以便 control plane 节点可以调度。

+

重要

当您将 control plane 节点从默认不可调度配置为可以调度时，需要额外的订阅。这是因为 control plane 节点然后变为 worker 节点。

1. 检查 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** Kubernetes 清单文件中的 **mastersSchedulable** 参数是否已设置为 **false**。此设置可防止在 control plane 机器上调度 pod:

   a. 打开 **<installation_directory>/manifests/cluster-scheduler-02-config.yml** 文件。

   b. 找到 **mastersSchedulable** 参数并确保它被设置为 **false**。

   c. 保存并退出文件。

2. 要创建 Ignition 配置文件，从包含安装程序的目录运行以下命令：

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

1 对于 **<installation_directory>**，请指定相同的安装目录。

该目录中将生成以下文件：

```
.
├── auth
│   ├── kubeadmin-password
│   └── kubeconfig
├── bootstrap.ign
├── master.ign
├── metadata.json
└── worker.ign
```

1.3.10. 配置 chrony 时间服务

您需要修改 **chrony.conf** 文件的内容来设置 chrony 时间服务（**chronyd**）使用的时间服务器和相关设置，并通过一个机器配置将这些内容传递给节点。

流程

1. 创建 **chrony.conf** 文件的内容并对其进行 base64 编码。例如：

```
$ cat << EOF | base64
```

```
    pool 0.rhel.pool.ntp.org iburst (1)
    driftfile /var/lib/chrony/drift
    makestep 1.0 3
    rtcsync
    logdir /var/log/chrony
EOF
```

(1) 指定任何有效的、可访问的时间源，如 DHCP 服务器提供的时间源。

输出示例

```
ICAgIHNlcnZlciBjbG9jay5yZWRoYXQuY29tIGlidXJzdAogICAgZHJpZnRmaWxlIC92YXIvbGli
L2Nocm9ueS9kcmlmdAogICAgbWFrZXN0ZXAgMS4wIDMKICAgIHJ0Y3N5bmMKICAgIGxvZ2
RpciAv
dmFyL2xvZy9jaHJvbnkK
```

2. 创建 **MachineConfig** 对象文件，将 base64 字符串替换为您刚刚创建的字符串。本例将文件添加到 **master** 节点。您可以将其更改为 **worker**，或为 **worker** 角色创建额外的 MachineConfig。为集群使用的每种机器创建 MachineConfig 文件：

```
$ cat << EOF > ./99-masters-chrony-configuration.yaml
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: master
  name: 99-masters-chrony-configuration
spec:
  config:
    ignition:
      config: {}
      security:
        tls: {}
      timeouts: {}
      version: 3.1.0
    networkd: {}
    passwd: {}
    storage:
      files:
      - contents:
          source: data:text/plain;charset=utf-
8;base64,ICAgIHNlcnZlciBjbG9jay5yZWRoYXQuY29tIGlidXJzdAogICAgZHJpZnRmaWxlIC92Y
XIvbGliL2Nocm9ueS9kcmlmdAogICAgbWFrZXN0ZXAgMS4wIDMKICAgIHJ0Y3N5bmMKICAg
IGxvZ2RpciAvdmFyL2xvZy9jaHJvbnkK
        mode: 420 (1)
        overwrite: true
        path: /etc/chrony.conf
  osImageURL: ""
EOF
```

(1) 为机器配置文件的 **mode** 字段指定数值模式。在创建文件并应用更改后，**模式** 将转换为十进制值。您可以使用 **oc get mc <mc-name> -o yaml** 命令来检查 YAML 文件。

3. 对配置文件做一个备份副本。

4. 使用两种方式之一应用配置：

   - 如果集群还没有启动，在生成清单文件后，将此文件添加到 **<installation_directory>/openshift** 目录中，然后继续创建集群。

   - 如果集群已在运行，请应用该文件：

     ```
     $ oc apply -f ./99-masters-chrony-configuration.yaml
     ```

1.3.11. 安装 RHCOS 并启动 OpenShift Container Platform bootstrap 过程

要在您置备的裸机基础架构上安装 OpenShift Container Platform，您必须在机器上安装 Red Hat Enterprise Linux CoreOS（RHCOS）。安装 RHCOS 时，您必须为 OpenShift Container Platform 安装程序生成的机器类型提供 Ignition 配置文件。如果您配置了合适的网络、DNS 和负载均衡基础架构，OpenShift Container Platform bootstrap 过程会在 RHCOS 机器重启后自动开始。

要在机器上安装 RHCOS，请按照以下步骤使用 ISO 镜像或网络 PXE 启动。

注意

本安装文档中包括的计算节点部署步骤特定于 RHCOS。如果您选择部署基于 RHEL 的计算节点，您将接管所有操作系统生命周期管理和维护，包括执行系统更新、应用补丁和完成所有其他必要的任务。RHEL 7 计算机器的使用已弃用，计划在以后的 OpenShift Container Platform 4 发行版本中删除。

您可以使用以下方法在 ISO 和 PXE 安装过程中配置 RHCOS:

- 内核参数： 您可以使用内核参数来提供特定于安装的信息。例如，您可以指定上传到 HTTP 服务器的 RHCOS 安装文件的位置，以及您要安装的节点类型的 Ignition 配置文件的位置。对于 PXE 安装，您可以使用 **APPEND** 参数将参数传递给实时安装程序的内核。对于 ISO 安装，您可以中断实时安装引导过程来添加内核参数。在这两种安装情况下，您可以使用特殊的 **coreos.inst.*** 参数来指示实时安装程序，以及标准安装引导参数来打开或关闭标准内核服务。

- Ignition 配置：OpenShift Container Platform Ignition 配置文件（***.ign**）特定于您要安装的节点类型。您可以在 RHCOS 安装过程中传递 bootstrap、control plane 或计算节点 Ignition 配置文件的位置，以便在第一次引导时生效。特殊情况下，您可以创建单独的、有限的 Ignition 配置来传递给 Live 系统。该 Ignition 配置可以执行特定任务，如在安装完成后向置备系统报告成功。这个特殊 Ignition 配置由 **coreos-installer** 使用，用于首次启动安装的系统。不要直接向 live ISO 提供标准 control plane 和计算节点 Ignition 配置。

- **coreos-installer**：您可以将 live ISO 安装程序引导到 shell 提示符，这可让您在首次引导前以多种方式准备持久性系统。特别是，您可以运行 **coreos-installer** 命令来识别包括的工件、使用磁盘分区以及设置联网。在有些情况下，您可以配置 live 系统上的功能并将其复制到安装的系统中。

使用 ISO 安装还是 PXE 安装要根据您的具体情况而定。PXE 安装需要可用的 DHCP 服务并进行更多准备，但可以使安装过程更自动化。ISO 安装是一个更手动的过程，如果您设置的机器较多，则可能不方便。

注意

自 OpenShift Container Platform 4.6 起，RHCOS ISO 和其他安装工件支持在带有 4K 扇区的磁盘上安装。

1.3.11.1. 使用 ISO 镜像创建 Red Hat Enterprise Linux CoreOS (RHCOS) 机器

在您置备的基础架构上安装集群前，必须先创建 RHCOS 机器供其使用。您可以使用 ISO 镜像来创建这些机器。

先决条件

- 获取集群的 Ignition 配置文件。
- 具有可从计算机以及您创建的机器访问的 HTTP 服务器的访问权限。

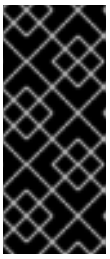
流程

1. 将安装程序创建的 control plane、计算和 bootstrap Ignition 配置文件上传到 HTTP 服务器。记下这些文件的 URL。

重要

如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

2. 从 RHCOS 镜像镜像页面获取您选择的操作系统实例安装方法所需的 RHCOS 镜像。



重要

RHCOS 镜像可能不会随着 OpenShift Container Platform 的每一发行版本都有改变。您必须下载最高版本的镜像，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。如果可用，请使用与 OpenShift Container Platform 版本匹配的镜像版本。此流程只使用 ISO 镜像。此安装类型不支持 RHCOS qcow2 镜像。

ISO 文件名类似以下示例：

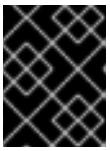
rhcos-<version>-live.<architecture>.iso

3. 使用 ISO 启动 RHCOS 安装。使用如下安装选项之一：
   - 将 ISO 镜像刻录到磁盘并直接启动。
   - 通过 LOM 接口使用 ISO 重定向。
4. 引导 ISO 镜像。您可以中断安装引导过程来添加内核参数。然而，在这个 ISO 过程中，您应该使用 **coreos-installer** 命令而不是添加内核参数。如果您在没有选项或中断的情况下运行 live 安装程序，安装程序将引导至 live 系统上的 shell 提示符，准备好将 RHCOS 安装到磁盘中。
5. 在运行 **coreos-installer** 前，请参阅 *高级 RHCOS 安装参考* 部分，以了解配置功能的不同方法，如网络和磁盘分区。
6. 运行 **coreos-installer** 命令。您至少必须识别节点类型的 Ignition 配置文件位置，以及您要安装到的磁盘位置。下面是一个示例：

```
$ sudo coreos-installer install \
    --ignition-url=https://host/worker.ign /dev/sda
```

7. 安装 RHCOS 后，系统会重启。系统重启过程中，它会应用您指定的 Ignition 配置文件。

8. 继续为集群创建其他机器。



重要

此刻您必须创建 bootstrap 和 control plane 机器。如果 control plane 机器不可调度（这是默认调度），则在安装集群前至少会创建两台计算机器。

1.3.11.2. 通过 PXE 或 iPXE 启动来创建 Red Hat Enterprise Linux CoreOS (RHCOS) 机器

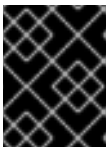
在安装使用手动置备 RHCOS 节点（如裸机）的集群前，您必须创建 RHCOS 机器供其使用。您可以使用 PXE 或 iPXE 启动来创建机器。

先决条件

- 获取集群的 Ignition 配置文件。
- 配置合适的 PXE 或 iPXE 基础架构。
- 具有 HTTP 服务器的访问权限，以便您可从计算机进行访问。

流程

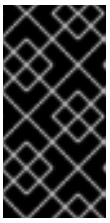
1. 将安装程序创建的 master、worker 和 bootstrap Ignition 配置文件上传到 HTTP 服务器。记下这些文件的 URL。



重要

您可以在 Ignition 配置中添加或更改配置设置，然后将其保存到 HTTP 服务器。如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

2. 从 RHCOS **镜像镜像**页面获取 **RHCOS 内核**、**initramfs** 和 **rootfs** 文件。



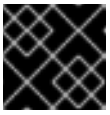
重要

RHCOS 工件（artifact）可能不会随着 OpenShift Container Platform 的每个发行版本而改变。您必须下载最高版本的工件，其版本号应小于或等于您安装的 OpenShift Container Platform 版本。这个过程只使用下面描述的正确 **kernel**、**initramfs** 和 **rootfs** 工件。此安装类型不支持 RHCOS qcow2 镜像。

文件名包含 OpenShift Container Platform 版本号。它们类似以下示例：

- **kernel**: **rhcos-<version>-live-kernel-<architecture>**
- **initramfs**: **rhcos-<version>-live-initramfs.<architecture>.img**
- **rootfs**: **rhcos-<version>-live-rootfs.<architecture>.img**

3. 上传引导方法所需的额外文件：
   - 对于传统的 PXE，将 **kernel** 和 **initramfs** 文件上传到 TFTP 服务器和 **rootfs** 文件到 HTTP 服务器。
   - 对于 iPXE，将 **内核**、**initramfs** 和 **rootfs** 文件上传到 HTTP 服务器。



重要

如果您计划在安装完成后在集群中添加更多计算机器，请不要删除这些文件。

4. 配置网络启动基础架构，以便在安装 RHCOS 后机器可从本地磁盘启动。

5. 为 RHCOS 镜像配置 PXE 或 iPXE 安装。
针对您的环境修改以下示例菜单条目之一，并验证能否正确访问镜像和 Ignition 文件：

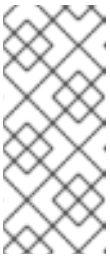
- 对于 PXE：

```
DEFAULT pxeboot
TIMEOUT 20
PROMPT 0
LABEL pxeboot
    KERNEL http://<HTTP_server>/rhcos-<version>-live-kernel-<architecture> 1
    APPEND initrd=http://<HTTP_server>/rhcos-<version>-live-initramfs.
<architecture>.img coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-
rootfs.<architecture>.img coreos.inst.install_dev=/dev/sda
coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign 2 3
```

1. 指定上传到 HTTP 服务器的 live **kernel** 文件位置。URL 必须是 HTTP、TFTP 或者 FTP；不支持 HTTPS 和 NFS。

2. 如果您使用多个 NIC，请在 **ip** 选项中指定一个接口。例如，要在名为 **eno1** 的 NIC 上使用 DHCP，请设置 **ip=eno1:dhcp**。

3. 指定上传到 HTTP 服务器的 RHCOS 文件的位置。**initrd** 参数值是 **initramfs** 文件的位置，**coreos.live.rootfs_url** 参数值是 **rootfs** 文件的位置，**coreos.inst.ignition_url** 参数值是 bootstrap Ignition 配置文件的位置。您还可以在 **APPEND** 行中添加更多内核参数来配置联网或其他引导选项。



注意

这个配置不会在使用图形控制台的机器上启用串口控制台访问。要配置不同的控制台，请在 **APPEND** 行中添加一个或多个 **console=** 参数。例如，添加 **console=tty0 console=ttyS0** 将第一个 PC 串口设置为主控制台，图形控制台作为二级控制台。如需更多信息，请参阅如何在 Red Hat Enterprise Linux 中设置串行终端和（或）控制台？

- 对于 iPXE：

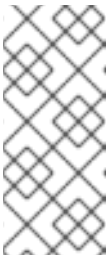
```
kernel http://<HTTP_server>/rhcos-<version>-live-kernel-<architecture> initrd=main
coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-rootfs.
<architecture>.img coreos.inst.install_dev=/dev/sda
coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign 1 2
initrd --name main http://<HTTP_server>/rhcos-<version>-live-initramfs.
<architecture>.img 3
boot
```

1. 指定上传到 HTTP 服务器的 RHCOS 文件的位置。**kernel** 参数值是 **kernel** 文件的位置，在 UEFI 系统中引导时需要 **initrd=main** 参数。 **coreos.live.rootfs_url** 参数值是 **rootfs** 文件的位置， **coreos.inst.ignition_url** 参数值则是 bootstrap Ignition 配置文件

的位置。

2 如果您使用多个 NIC，请在 **ip** 选项中指定一个接口。例如，要在名为 **eno1** 的 NIC 上使用 DHCP，请设置 **ip=eno1:dhcp**。

3 指定上传到 HTTP 服务器的 **initramfs** 文件的位置。



注意

这个配置不会在使用图形控制台的机器上启用串口控制台访问。要配置不同的控制台，请在 **kerne** 行中添加一个或多个 **console=** 参数。例如，添加 **console=tty0 console=ttyS0** 将第一个 PC 串口设置为主控制台，图形控制台作为二级控制台。如需更多信息，请参阅如何在 Red Hat Enterprise Linux 中设置串行终端和（或）控制台？

6. 如果使用 PXE UEFI，请执行以下操作：

   a. 提供启动系统所需的 **shim x64.efi** EFI 二进制文件和 **grub.cfg** 文件。

   - 通过将 RHCOS ISO 挂载到您的主机，然后将 **images/efiboot.img** 文件挂载到您的主机来提取所需的 EFI 二进制文件：

     ```
     $ mkdir -p /mnt/iso
     ```

     ```
     $ mkdir -p /mnt/efiboot
     ```

     ```
     $ mount -o loop rhcos-installer.x86_64.iso /mnt/iso
     ```

     ```
     $ mount -o loop,ro /mnt/iso/images/efiboot.img /mnt/efiboot
     ```

   - 从 **efiboot.img** 挂载点，将 **EFI/redhat/shimx64.efi** 和 **EFI/redhat/grubx64.efi** 文件复制到您的 TFTP 服务器中：

     ```
     $ cp /mnt/efiboot/EFI/redhat/shimx64.efi .
     ```

     ```
     $ cp /mnt/efiboot/EFI/redhat/grubx64.efi .
     ```

     ```
     $ umount /mnt/efiboot
     ```

     ```
     $ umount /mnt/iso
     ```

   - 将 RHCOS ISO 中包含的 **EFI/redhat/grub.cfg** 文件复制到您的 TFTP 服务器中。

   b. 编辑 **grub.cfg** 文件使其包含类似如下的参数：

   ```
   menuentry 'Install Red Hat Enterprise Linux CoreOS' --class fedora --class gnu-linux --class gnu --class os {
    linuxefi rhcos-<version>-live-kernel-<architecture> coreos.inst.install_dev=/dev/sda coreos.live.rootfs_url=http://<HTTP_server>/rhcos-<version>-live-rootfs.
   ```

```
<architecture>.img coreos.inst.ignition_url=http://<HTTP_server>/bootstrap.ign
 initrdefi rhcos-<version>-live-initramfs.<architecture>.img
}
```

其中：

rhcos-<version>-live-kernel-<architecture>

指定上传到 TFTP 服务器的 **kernel** 文件。

http://<HTTP_server>/rhcos-<version>-live-rootfs.<architecture>.img

指定上传到 HTTP 服务器的 live rootfs 镜像的位置。

http://<HTTP_server>/bootstrap.ign

指定上传到 HTTP 服务器的 bootstrap Ignition 配置文件的位置。

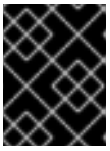
rhcos-<version>-live-initramfs.<architecture>.img

指定上传到 TFTP 服务器的 **initramfs** 文件的位置。

注意

有关如何为 UEFI 引导配置 PXE 服务器的详情，请查看红帽知识库文章：如何为 Red Hat Enterprise Linux 配置/设置 PXE 服务器？.

7. 继续为集群创建机器。



重要

此刻您必须创建 bootstrap 和 control plane 机器。如果 control plane 机器不可调度（这是默认调度），则在安装集群前至少会创建两台计算机器。

1.3.11.3. 高级 Red Hat Enterprise Linux CoreOS（RHCOS）安装配置

为 OpenShift Container Platform 手动置备 Red Hat Enterprise Linux CoreOS（RHCOS）节点的一个关键优点是能够进行通过默认的 OpenShift Container Platform 安装方法无法进行的配置。本节介绍了您可以使用的一些技术来进行配置，其中包括：

- 将内核参数传递给实时安装程序
- 从 live 系统手动运行 **coreos-installer**
- 将 Ignition 配置嵌入 ISO 中

本节详述了与 Red Hat Enterprise Linux CoreOS（RHCOS）手动安装的高级配置相关的内容，如磁盘分区、网络以及使用 Ignition 配置的不同方式相关。

1.3.11.3.1. 使用高级网络选项进行 PXE 和 ISO 安装

OpenShift Container Platform 节点的网络默认使用 DHCP 来收集所有必要配置设置。要设置静态 IP 地址或配置特殊的设置，如绑定,您可以执行以下操作之一：

- 引导 live 安装程序时会传递特殊的内核参数。
- 使用机器配置将网络文件复制到安装的系统中。

- 使用 live installer shell 提示配置网络，然后将那些设置复制到安装的系统上，以便在安装的系统第一次引导时生效。

要配置 PXE 或 iPXE 安装，请使用以下选项之一：

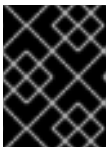
- 请参阅"高级 RHCOS 安装参考"表。
- 使用机器配置将网络文件复制到安装的系统中。

要配置 ISO 安装，请使用以下步骤。

流程

1. 引导 ISO 安装程序。
2. 在 live 系统 shell 提示下，使用可用的 RHEL 工具（如 **nmcli** 或 **nmtui**）为 Live 系统配置网络。
3. 运行 **coreos-installer** 命令来安装系统，添加 **--copy-network** 选项来复制网络配置。例如：

```
$ coreos-installer install --copy-network \
    --ignition-url=http://host/worker.ign /dev/sda
```



重要

copy-network 选项只复制 **/etc/NetworkManager/system-connections** 下的网络配置。特别是，它不会复制系统主机名。

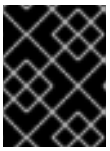
4. 重启安装的系统。

1.3.11.3.2. 磁盘分区

磁盘分区是在 Red Hat Enterprise Linux CoreOS（RHCOS）安装过程中在 OpenShift Container Platform 集群节点上创建的。特定架构的每个 RHCOS 节点都使用相同的分区布局，除非默认分区配置被覆盖。在 RHCOS 安装过程中，根文件系统的大小会增大，以使用目标设备中剩余的可用空间。

但是，在安装 OpenShift Container Platform 节点时，在两种情况下您可能需要覆盖默认分区：

- 创建单独的分区：对于在空磁盘中的 greenfield 安装，您可能想要在分区中添加单独的存储。这只在生成 **/var** 或者一个 **/var** 独立分区的子目录（如 **/var/lib/etcd** ）时被正式支持，但不支持两者。



重要

Kubernetes 只支持两个文件系统分区。如果您在原始配置中添加多个分区，Kubernetes 无法监控所有这些分区。

- 保留现有分区：对于 brownfield 安装，您要在现有节点上重新安装 OpenShift Container Platform，并希望保留从之前的操作系统中安装的数据分区，对于 **coreos-installer** 来说，引导选项和选项都允许您保留现有数据分区。

1.3.11.3.2.1. 创建一个独立的 /var 分区

通常情况下，OpenShift Container Platform 的磁盘分区应该留给安装程序。然而，在有些情况下您可能需要在文件系统的一部分中创建独立分区。

OpenShift Container Platform 支持添加单个分区来将存储附加到 **/var** 分区或 **/var** 的子目录。例如：

- **/var/lib/containers**：保存镜像相关的内容，随着更多镜像和容器添加到系统中，它所占用的存储会增加。
- **/var/lib/etcd**：保存您可能希望保持独立的数据，比如 etcd 存储的性能优化。
- **/var**：保存您希望独立保留的数据，用于特定目的（如审计）。

单独存储 **/var** 目录的内容可方便地根据需要对区域扩展存储，并可以在以后重新安装 OpenShift Container Platform 时保持该数据地完整。使用这个方法，您不必再次拉取所有容器，在更新系统时也无法复制大量日志文件。

因为 **/var** 在进行一个全新的 Red Hat Enterprise Linux CoreOS（RHCOS）安装前必需存在，所以这个流程会在 OpenShift Container Platform 安装过程的 **openshift-install** 准备阶段插入的机器配置来设置独立的 **/var** 分区。

流程

1. 创建存放 OpenShift Container Platform 安装文件的目录：

```
$ mkdir $HOME/clusterconfig
```

2. 运行 **openshift-install** 在 **manifest** 和 **openshift** 子目录中创建一组文件。在出现提示时回答系统问题：

```
$ openshift-install create manifests --dir $HOME/clusterconfig
? SSH Public Key ...
$ ls $HOME/clusterconfig/openshift/
99_kubeadmin-password-secret.yaml
99_openshift-cluster-api_master-machines-0.yaml
99_openshift-cluster-api_master-machines-1.yaml
99_openshift-cluster-api_master-machines-2.yaml
...
```

3. 创建 **MachineConfig** 对象并将其添加到 **openshift** 目录中的一个文件中。例如，把文件命名为 **98-var-partition.yaml**，将磁盘设备名称改为 **worker** 系统中存储设备的名称，并根据情况设置存储大小。这个示例将 **/var** 目录放在独立分区中：

```
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfig
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
spec:
  config:
    ignition:
      version: 3.1.0
    storage:
      disks:
      - device: /dev/<device_name> 1
        partitions:
        - label: var
          startMiB: <partition_start_offset> 2
```

```
      sizeMiB: <partition_size> 3
  filesystems:
    - device: /dev/disk/by-partlabel/var
      path: /var
      format: xfs
systemd:
  units:
    - name: var.mount 4
      enabled: true
      contents: |
        [Unit]
        Before=local-fs.target
        [Mount]
        What=/dev/disk/by-partlabel/var
        Where=/var
        Options=defaults,prjquota 5
        [Install]
        WantedBy=local-fs.target
```

1. 要分区的磁盘的存储设备名称。
2. 当在引导磁盘中添加数据分区时，推荐最少使用 25000MB。root 文件系统会自动重新定义大小使其占据所有可用空间（最多到指定的偏移值）。如果没有指定值，或者指定的值小于推荐的最小值，则生成的 root 文件系统会太小，而在以后进行的 RHCOS 重新安装可能会覆盖数据分区的开始部分。
3. 数据分区的大小（以兆字节为单位）。
4. 挂载单元的名称必须与 where = 指令中指定的目录匹配。例如，对于挂载到 **/var/lib/containers** 的文件系统，这个单元必须命名为 **var-lib-containers.mount**。
5. 必须针对用于容器存储的文件系统启用 **prjquota** 挂载选项。

注意

在创建独立 **/var** 分区时，如果不同的实例类型没有相同的设备名称，则无法将不同的实例类型用于 worker 节点。

4. 再次运行 **openshift-install**，从 **manifest** 和 **openshift** 子目录中的一组文件创建 Ignition 配置：

```
$ openshift-install create ignition-configs --dir $HOME/clusterconfig
$ ls $HOME/clusterconfig/
auth  bootstrap.ign  master.ign  metadata.json  worker.ign
```

现在，可以使用 Ignition 配置文件作为 ISO 或 PXE 手动安装过程的输入来安装 Red Hat Enterprise Linux CoreOS（RHCOS）系统。

1.3.11.3.2.2. 保留现有分区

对于 ISO 安装，您可以在 **coreos-installer** 命令行中添加可让安装程序维护一个或多个现有分区的选项。对于 PXE 安装，您可以 **APPEND coreos.inst.*** 选项来保留分区。

保存的分区可能是来自现有 OpenShift Container Platform 系统中的分区，其中包括了您希望保留的数据分区。以下是几个提示：

- 如果您保存了现有分区，且这些分区没有为 RHCOS 留下足够空间，则安装将失败但不会损害已保存的分区。
- 通过分区标签或数字识别您要保留的磁盘分区。

对于 ISO 安装

这个示例保留分区标签以**数据（data***）开头的任何分区：

```
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign \
      --save-partlabel 'data*' /dev/sda
```

以下示例演示了在运行 **coreos-installer** 时要保留磁盘上的第 6 个分区：

```
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign \
      --save-partindex 6 /dev/sda
```

这个示例保留了分区 5 及更高分区：

```
# coreos-installer install --ignition-url http://10.0.2.2:8080/user.ign
      --save-partindex 5- /dev/sda
```

在前面已保存分区的示例中，**coreos-installer** 会立即重新创建分区。

对于 PXE 安装

这个 **APPEND** 选项保留分区标签以 'data'('data*')开头的所有分区：

```
coreos.inst.save_partlabel=data*
```

这个 **APPEND** 选项保留分区 5 及其后的分区：

```
coreos.inst.save_partindex=5-
```

这个 **APPEND** 选项保留分区 6:

```
coreos.inst.save_partindex=6
```

1.3.11.3.3. 标识 Ignition 配置

在进行 RHCOS 手动安装时，您可以提供两种 Ignition 配置类型，它们有不同的原因：

- **永久安装 Ignition 配置**：每个手动 RHCOS 安装都需要传递 **openshift-installer** 生成的 Ignition 配置文件之一，如 **bootstrap.ign**、**master.ign** 和 **worker.ign**，才能进行安装。

  重要

  不建议修改这些文件。

  对于 PXE 安装，您可以使用 **coreos.inst.ignition_url=** 选项在 **APPEND** 行上传递 Ignition 配置。对于 ISO 安装，在 ISO 引导至 shell 提示符后，您可以使用 **--ignition-url=** 选项在 **coreos-installer** 命令行上识别 Ignition 配置。在这两种情况下，都只支持 HTTP 和 HTTPS 协议。

- **live 安装 Ignition 配置**：此类型必须手动创建，并应该尽可能避免，因为红帽不支持它。使用此方法，Ignition 配置会传递到 live 安装介质，在引导时立即运行，并在 RHCOS 系统安装到磁盘之前和/或之后执行设置任务。这个方法只用于必须执行一次且之后不能再次应用的任务，如不能使用机器配置进行的高级分区。
  对于 PXE 或 ISO 引导，您可以创建 Ignition 配置，**APPEND ignition.config.url=** 选项，以标识 Ignition 配置的位置。您还需要附加 **ignition.firstboot ignition.platform.id=metal** 或者 **ignition.config.url** 选项。

1.3.11.3.3.1. 在 RHCOS ISO 中嵌入 Ignition 配置

您可以直接嵌入 RHCOS ISO 镜像中的 live 安装 Ignition 配置。引导 ISO 镜像后，内嵌的配置将自动应用。

流程

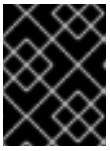
1. 从以下镜像镜像页面下载 **coreos-installer** 二进制文件：
   https://mirror.openshift.com/pub/openshift-v4/clients/coreos-installer/latest/。
2. 检索 RHCOS ISO 镜像和 Ignition 配置文件，并将其复制到可访问的目录中，如 **/mnt**:

```
# cp rhcos-<version>-live.x86_64.iso bootstrap.ign /mnt/
# chmod 644 /mnt/rhcos-<version>-live.x86_64.iso
```

3. 运行以下命令将 Ignition 配置嵌入 ISO 中：

```
# ./coreos-installer iso ignition embed -i /mnt/bootstrap.ign \
    /mnt/rhcos-<version>-live.x86_64.iso
```

   现在，您以使用该 ISO 使用指定的 live 安装 Ignition 配置来安装 RHCOS。



重要

不支持且不推荐使用 **coreos-installer iso ignition embed** 来嵌入由 **openshift-installer** 生成的文件，如 **bootstrap.ign**、**master.ign** 和 **worker.ign**。

4. 要显示嵌入的 Ignition 配置的内容并将其定向到文件中，请运行：

```
# ./coreos-installer iso ignition show /mnt/rhcos-<version>-live.x86_64.iso > mybootstrap.ign
```

```
# diff -s bootstrap.ign mybootstrap.ign
```

   输**出示例**

```
Files bootstrap.ign and mybootstrap.ign are identical
```

5. 要删除 Ignition 配置并将 ISO 返回到其 pristine 状态（因此您可以重复使用它），请运行：

```
# ./coreos-installer iso ignition remove /mnt/rhcos-<version>-live.x86_64.iso
```

   现在，您可以将另一个 Ignition 配置嵌入到 ISO 中，或者在其 pristine 状态下使用 ISO。

1.3.11.3.4. 高级 RHCOS 安装参考

本节演示了网络配置和其他高级选项，允许您修改 Red Hat Enterprise Linux CoreOS（RHCOS）手动安装过程。下表描述了您可以与 RHCOS live installer 和 **coreos-installer** 命令一起使用的内核参数和命令行选项。

RHCOS 启动提示下的路由和绑定选项
如果从 ISO 镜像安装 RHCOS，您可以在引导该镜像时手动添加内核参数以配置节点的网络。如果没有使用网络参数，则安装默认为使用 DHCP。

重要

添加网络参数时，还必须添加 **rd.neednet=1** 内核参数。

下表描述了如何为实时 ISO 安装使用 **ip=** 、 **nameserver=** 和 **bond=** 内核参数。

注意

在添加内核参数时顺序非常重要： **ip=**，**nameserver=**，然后 **bond=**。

ISO 的路由和绑定选项

下表提供了配置 Red Hat Enterprise Linux CoreOS（RHCOS）节点网络的示例。这些是在系统引导过程中传递给 **dracut** 工具的网络选项。有关 **dracut** 支持的网络选项的详情，请参考 **dracut.cmdline** 手册页。

描述	例子
要配置一个 IP 地址，可以使用 DHCP(**ip=dhcp**)或者设置单独的静态 IP 地址(**ip=<host_ip>**)。然后在每个节点上指定 DNS 服务器 IP 地址(**nameserver=<dns_ip>**)。这个示例设置： • 节点的 IP 地址为 **10.10.10.2** • 网关地址为 **10.10.10.254** • 子网掩码为 **255.255.255.0** • 主机名为 **core0.example.com** • DNS 服务器地址为 **4.4.4.41**	ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none nameserver=4.4.4.41
通过指定多个 **ip=** 条目来指定多个网络接口。	ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none ip=10.10.10.3::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none

描述	例子
可选：您可以通过设置一个 **rd.route=** 值来配置到额外网络的路由。 如果额外网络网关与主要网络网关不同，则默认网关必须是主要网络网关。	配置默认网关： `ip=::10.10.10.254::::` 为额外网络配置路由： `rd.route=20.20.20.0/24:20.20.20.254:enp2s0`
在单一接口中禁用 DHCP，比如当有两个或者多个网络接口时，且只有一个接口被使用。	`ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp1s0:none` `ip=::::core0.example.com:enp2s0:none`
您可以将系统中 DHCP 和静态 IP 配置与多个网络接口结合在一起。	`ip=enp1s0:dhcp` `ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0:none`
可选： 您可以使用 **vlan=** 参数在单独的接口上配置 VLAN。	在网络接口中配置 VLAN 并使用静态 IP 地址： `ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:enp2s0.100:none` `vlan=enp2s0.100:enp2s0` 在网络接口中配置 VLAN 并使用 DHCP： `ip=enp2s0.100:dhcp` `vlan=enp2s0.100:enp2s0`
您可以为每个服务器添加一个 **nameserver=** 条目来提供多个 DNS 服务器。	`nameserver=1.1.1.1` `nameserver=8.8.8.8`
可选：使用 **bond=** 选项支持将多个网络接口绑定到一个接口。在这两个示例中： ● 配置绑定接口的语法为：**bond=name[:network_interfaces][:options]** ● *name* 是绑定设备名称（**bond0**），*network_interfaces* 代表用逗号分开的物理（以太网）接口(**em1,em2**)的列表，*options* 是用逗号分开的绑定选项列表。输入 **modinfo bonding** 查看可用选项。 ● 当使用 **bond=** 创建绑定接口时，您必须指定如何分配 IP 地址以及绑定接口的其他信息。	要将绑定的接口配置为使用 DHCP，请将绑定的 IP 地址设置为 **dhcp**。例如： `bond=bond0:em1,em2:mode=active-backup` `ip=bond0:dhcp` 要将绑定接口配置为使用静态 IP 地址，请输入您需要的特定 IP 地址以及相关信息。例如： `bond=bond0:em1,em2:mode=active-backup` `ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0:none`

描述	例子
可选： 您可以使用 **vlan=** 参数在绑定接口上配置 VLAN。	使用 VLAN 配置绑定接口并使用 DHCP： `ip=bond0.100:dhcp` `bond=bond0:em1,em2:mode=active-backup` `vlan=bond0.100:bond0` 使用 VLAN 配置绑定接口，并使用静态 IP 地址： `ip=10.10.10.2::10.10.10.254:255.255.255.0:core0.example.com:bond0.100:none` `bond=bond0:em1,em2:mode=active-backup` `vlan=bond0.100:bond0`
可选：使用 **team=** 参数将网络团队用作绑定的替代选择。在本例中： • 配置组接口的语法为：**team=name[:network_interfaces]** *name* 是团队设备名称（**team0**），*network_interfaces* 代表用逗号分开的物理（以太网）接口（**em1、em2**）。 **注意** 当 RHCOS 切换到即将发布的 RHEL 版本时，团队计划被弃用。如需更多信息，请参阅 Red Hat 知识库文章。	配置网络团队： `team=team0:em1,em2` `ip=team0:dhcp`

coreos.inst 引导选项用于 ISO 或 PXE 安装

虽然您可以将大多数标准安装引导参数传递给 live 安装程序，但也有一些特定于 RHCOS live 安装程序的参数。

- 对于 ISO，可以通过中断 RHCOS 安装程序来添加这些选项。
- 对于 PXE 或 iPXE，这些选项必须在启动 PXE 内核前添加到 **APPEND** 行中。您无法中断实时 PXE 安装。

下表显示了用于 ISO 和 PXE 安装的 RHCOS live installer 引导选项。

表 1.37. coreos.inst 引导选项

参数	描述
coreos.inst.install_dev	必需。要安装的系统中的块设备。虽然可以使用 **sda** 这样的相对路径，但建议使用完整路径，如 **/dev/sda**。

参数	描述
coreos.inst.ignition_url	可选：嵌入到已安装系统中的 Ignition 配置的 UR如果没有指定 URL，则不会嵌入 Ignition 配置。
coreos.inst.save_partlabel	可选：在安装过程中要保留的分区压缩标签。允许使用 glob 风格的通配符。指定分区不需要存在。
coreos.inst.save_partindex	可选：在安装过程中完成要保留的分区的分离索引。可以使用 **m**-**n** 指定范围，**m** 或 **n** 可以被省略。指定分区不需要存在。
coreos.inst.insecure	可选：将 **coreos.inst.image_url** 指定的 OS 镜像提交取消签名。
coreos.inst.image_url	可选：下载并安装指定的 RHCOS 镜像。 ● 这个参数不应该在生产环境中使用，而是只用于调试目的。 ● 虽然在 RHCOS 的安装版本与 live 介质的版本不匹配时可以使用这个参数，但建议使用与您要安装版本匹配的介质。 ● 如果您使用的是 **coreos.inst.image_url**。还必须使用 **coreos.inst.insecure**。这是因为，裸机介质没有为 OpenShift Container Platform 进行 GPG 签名。 ● 只支持 HTTP 和 HTTPS 协议。
coreos.inst.skip_reboot	可选：安装后该系统不会重启。安装完成后，您会收到提示，提示您检查在安装过程中发生的情况。这个参数不应该在生产环境中使用，而是只用于调试目的。
coreos.inst.platform_id	可选：安装 RHCOS 镜像的平台的 Ignition 平台 ID。默认为 **metal**。这个选项决定是否从云供应商（如 VMware）请求 Ignition 配置。例如：**coreos.inst.platform_id=vmware**。
ignition.config.url	可选：用于实时启动的 Ignition 配置的 URL。例如，它可以用来定制调用 **coreos-installer** 的方式，或者用来在安装前或安装后运行代码。这与 **coreos.inst.ignition_url**（这是已安装系统的 Ignition 配置）不同。

ISO 安装的 coreos-installer 选项

您还可以直接从命令行调用 **coreos-installer** 命令来安装 RHCOS。上表中的内核参数提供了在引导时自动调用 **coreos-installer** 的快捷方式，但您可以在 shell 提示符运行时将类似的参数直接传递给 **coreos-installer**。

下表显示了您可以在实时安装过程中从 shell 提示符传递给 **coreos-installer** 命令的选项和子命令。

表 1.38. CoreOS-installer 命令行选项、参数和子命令

命令行选项	
选项	**描述**
-u, **--image-url <url>**	手动指定镜像 URL。
-f, **--image-file <path>**	手动指定本地镜像文件。
-i, --ignition-file <path>	从文件中嵌入 Ignition 配置。
-I, **--ignition-url <URL>**	从 URL 嵌入 Ignition 配置。
--ignition-hash <digest>	Ignition config 的 **type-value** 的文摘值。
-p, **--platform <name>**	覆盖 Ignition 平台 ID。
--append-karg <arg>…	附加默认内核参数。
--delete-karg <arg>…	删除默认内核参数。
-n, **--copy-network**	从安装环境中复制网络配置。 **重要** **copy-network** 选项只复制 **/etc/NetworkManager/system-connections** 下的网络配置。特别是，它不会复制系统主机名。
--network-dir <path>	使用 **-n**。默认为 **/etc/NetworkManager/system-connections/**。
--save-partlabel <lx>..	使用这个标签 glob 保存分区。
--save-partindex <id>…	使用这个数值或者范围保存分区。
--offline	强制离线安装。
--insecure	跳过签名验证。
--insecure-ignition	允许没有 HTTPS 或 hash 的 Ignition URL。
--architecture <name>	目标 CPU 架构。默认为 **x86_64**。
--preserve-on-error	出现错误时不清除分区表。

-h，--help	打印帮助信息。
命令行参数	
参数	**描述**
<device>	目的设备。
CoreOS-installer 嵌入的 Ignition 命令	
命令	**描述**
$ coreos-installer iso ignition embed <options> --ignition-file <file_path> <ISO_image>	在 ISO 镜像中嵌入 Ignition 配置。
coreos-installer iso ignition show <options> <ISO_image>	显示来自 ISO 镜像的内嵌 Ignition 配置。
coreos-installer iso ignition remove <options> <ISO_image>	从 ISO 镜像中删除嵌入的 Ignition 配置。
coreos-installer ISO Ignition 选项	
选项	**描述**
-f, --force	覆盖现有的 Ignition 配置。
-i, --ignition-file <path>	要使用的 Ignition 配置。默认为 **stdin**。
-o, --output <path>	将 ISO 写入到一个新输出文件。
-h，--help	打印帮助信息。
coreos-installer PXE Ignition 命令	
命令	**描述**
请注意，不是所有子命令都接受这些选项。	
coreos-installer pxe ignition wrap <options>	在镜像中嵌套 Ignition 配置。
coreos-installer pxe ignition unwrap <options> <image_name>	显示在镜像中嵌套的 Ignition 配置。
coreos-installer pxe ignition unwrap <options> <initrd_name>	在 **initrd** 镜像中显示嵌套的 Ignition 配置。

coreos-installer PXE Ignition 选项	
选项	**描述**
-i, --ignition-file <path>	要使用的 Ignition 配置。默认为 **stdin**。
-o, --output <path>	将 ISO 写入到一个新输出文件。
-h, --help	打印帮助信息。

1.3.12. 创建集群

要创建 OpenShift Container Platform 集群，请等待您通过安装程序生成的 Ignition 配置文件所置备的机器上完成 bootstrap 过程。

先决条件

- 为集群创建所需的基础架构。
- 已获得安装程序并为集群生成了 Ignition 配置文件。
- 已使用 Ignition 配置文件为集群创建 RHCOS 机器。

流程

1. 监控 bootstrap 过程：

```
$ ./openshift-install --dir <installation_directory> wait-for bootstrap-complete \ 1
    --log-level=info 2
```

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

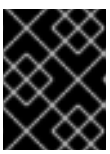
2 要查看不同的安装详情，请指定 **warn**、**debug** 或 **error**，而不要指定 **info**。

输出示例

```
INFO Waiting up to 30m0s for the Kubernetes API at https://api.test.example.com:6443...
INFO API v1.19.0 up
INFO Waiting up to 30m0s for bootstrapping to complete...
INFO It is now safe to remove the bootstrap resources
```

Kubernetes API 服务器提示已在 control plane 机器上完成 bootstrap 时，命令运行成功。

2. bootstrap 过程完成后，请从负载均衡器中删除 bootstrap 机器。



重要

此时您必须从负载均衡器中删除 bootstrap 机器。您还可以删除或重新格式化机器本身。

1.3.13. 使用 CLI 登录到集群

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

先决条件

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

流程

1. 导出 **kubeadmin** 凭证：

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

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

2. 使用导出的配置，验证能否成功运行 **oc** 命令：

```
$ oc whoami
```

输**出示例**

```
system:admin
```

1.3.14. 批准机器的证书签名请求

将机器添加到集群时，会为您添加的每台机器生成两个待处理证书签名请求（CSR）。您必须确认这些 CSR 已获得批准，或根据需要自行批准。客户端请求必须首先被批准，然后是服务器请求。

先决条件

- 您已将机器添加到集群中。

流程

1. 确认集群可以识别这些机器：

```
$ oc get nodes
```

输**出示例**

```
NAME     STATUS   ROLES   AGE  VERSION
master-0 Ready    master  63m  v1.19.0
master-1 Ready    master  63m  v1.19.0
master-2 Ready    master  64m  v1.19.0
```

输出将列出您创建的所有机器。

注意

在一些 CSR 被批准前，以上输出可能不包括计算节点（也称为 worker 节点）。

2. 检查待处理的 CSR，并确保可以看到添加到集群中的每台机器都有 **Pending** 或 **Approved** 状态的客户端请求：

```
$ oc get csr
```

输**出示例**

```
NAME        AGE    REQUESTOR                                                   CONDITION
csr-8b2br   15m    system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
csr-8vnps   15m    system:serviceaccount:openshift-machine-config-operator:node-bootstrapper   Pending
...
```

在本例中，两台机器加入了集群。您可能会在列表中看到更多已批准的 CSR。

3. 如果 CSR 没有获得批准，请在所添加机器的所有待处理 CSR 都处于 **Pending** 状态后，为您的集群机器批准这些 CSR：

注意

由于 CSR 会自动轮转，因此请在将机器添加到集群后一小时内批准您的 CSR。如果没有在一小时内批准，证书将会轮转，每个节点将会存在多个证书。您必须批准所有这些证书。批准客户端 CSR 后， Kubelet 为服务证书创建辅助 CSR，这需要手动批准。然后，如果 Kubelet 请求具有相同参数的新证书，则 **machine-approver** 会自动批准后续服务证书续订请求。

注意

对于在未启用机器 API 的平台中运行的集群，如裸机和其他用户置备的基础架构，必须采用一种方法自动批准 kubelet 提供证书请求（CSR）。如果没有批准请求，则 **oc exec**、**oc rsh** 和 **oc logs** 命令将无法成功，因为 API 服务器连接到 kubelet 时需要服务证书。与 Kubelet 端点联系的任何操作都需要此证书批准。这个方法必须监视新的 CSR，确认 CSR 由 **system:node** 或 **system:admin** 组中的 **node-bootstrapper** 服务帐户提交，并确认节点的身份。

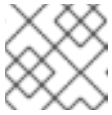
- 若要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}{{end}}{{end}}' | xargs --no-run-if-empty oc adm certificate approve
```



注意

在有些 CSR 被批准前，一些 Operator 可能无法使用。

4. 现在，您的客户端请求已被批准，您必须查看添加到集群中的每台机器的服务器请求：

```
$ oc get csr
```

输**出示例**

```
NAME        AGE     REQUESTOR                                                   CONDITION
csr-bfd72   5m26s   system:node:ip-10-0-50-126.us-east-2.compute.internal       Pending
csr-c57lv   5m26s   system:node:ip-10-0-95-157.us-east-2.compute.internal       Pending
...
```

5. 如果剩余的 CSR 没有被批准，且处于 **Pending** 状态，请批准集群机器的 CSR：

- 若要单独批准，请对每个有效的 CSR 运行以下命令：

```
$ oc adm certificate approve <csr_name> 1
```

1 **<csr_name>** 是当前 CSR 列表中 CSR 的名称。

- 要批准所有待处理的 CSR，请运行以下命令：

```
$ oc get csr -o go-template='{{range .items}}{{if not .status}}{{.metadata.name}}{{"\n"}}
{{end}}{{end}}' | xargs oc adm certificate approve
```

6. 批准所有客户端和服务器 CSR 后，器将处于 **Ready** 状态。运行以下命令验证：

```
$ oc get nodes
```

输**出示例**

```
NAME      STATUS    ROLES   AGE  VERSION
master-0  Ready     master  73m  v1.20.0
master-1  Ready     master  73m  v1.20.0
master-2  Ready     master  74m  v1.20.0
worker-0  Ready     worker  11m  v1.20.0
worker-1  Ready     worker  11m  v1.20.0
```

注意

批准服务器 CSR 后可能需要几分钟时间让机器转换为 **Ready** 状态。

其他信息

- 如需有关 CSR 的更多信息，请参阅证书签名请求。

1.3.15. 初始 Operator 配置

在 control plane 初始化后，您必须立即配置一些 Operator 以便它们都可用。

先决条件

- 您的 control plane 已初始化。

流程

1. 观察集群组件上线：

```
$ watch -n5 oc get clusteroperators
```

输出示例

```
NAME                                       VERSION AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.6.0   True        False         False      3h56m
cloud-credential                           4.6.0   True        False         False      29h
cluster-autoscaler                         4.6.0   True        False         False      29h
config-operator                            4.6.0   True        False         False      6h39m
console                                    4.6.0   True        False         False      3h59m
csi-snapshot-controller                    4.6.0   True        False         False      4h12m
dns                                        4.6.0   True        False         False      4h15m
etcd                                       4.6.0   True        False         False      29h
image-registry                             4.6.0   True        False         False      3h59m
ingress                                    4.6.0   True        False         False      4h30m
insights                                   4.6.0   True        False         False      29h
kube-apiserver                             4.6.0   True        False         False      29h
kube-controller-manager                    4.6.0   True        False         False      29h
kube-scheduler                             4.6.0   True        False         False      29h
kube-storage-version-migrator              4.6.0   True        False         False      4h2m
machine-api                                4.6.0   True        False         False      29h
machine-approver                           4.6.0   True        False         False      6h34m
machine-config                             4.6.0   True        False         False      3h56m
marketplace                                4.6.0   True        False         False      4h2m
monitoring                                 4.6.0   True        False         False      6h31m
network                                    4.6.0   True        False         False      29h
node-tuning                                4.6.0   True        False         False      4h30m
openshift-apiserver                        4.6.0   True        False         False      3h56m
openshift-controller-manager               4.6.0   True        False         False      4h36m
openshift-samples                          4.6.0   True        False         False      4h30m
operator-lifecycle-manager                 4.6.0   True        False         False      29h
operator-lifecycle-manager-catalog         4.6.0   True        False         False      29h
operator-lifecycle-manager-packageserver   4.6.0   True        False         False      3h59m
service-ca                                 4.6.0   True        False         False      29h
storage                                    4.6.0   True        False         False      4h30m
```

2. 配置不可用的 Operator。

1.3.15.1. 禁用默认的 OperatorHub 源

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

流程

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

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

提示

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

1.3.15.2. 镜像 registry 存储配置

对于不提供默认存储的平台，Image Registry Operator 最初将不可用。安装后，您必须配置 registry 使用的存储，这样 Registry Operator 才可用。

示配置生产集群所需的持久性卷的说明。如果适用，显示有关将空目录配置为存储位置的说明，该位置只可用于非生产集群。

另外还提供了在升级过程中使用 **Recreate** rollout 策略来允许镜像 registry 使用块存储类型的说明。

1.3.15.2.1. 更改镜像 registry 的管理状态

要启动镜像 registry，需要把 Image Registry Operator 配置的 **managementState** 从 **Removed** 改为 **Managed**。

流程

- 将 **managementState** Image Registry Operator 配置从 **Removed** 改为 **Managed**。例如：

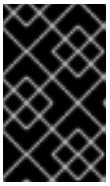
```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec":
{"managementState":"Managed"}}'
```

1.3.15.2.2. 为裸机和其他手动安装配置 registry 存储

作为集群管理员，在安装后需要配置 registry 来使用存储。

先决条件

- 具有 Cluster Administrator 权限
- 使用手动置备的 Red Hat Enterprise Linux CoreOS（RHCOS）节点（如裸机）的集群。
- 为集群置备的持久性存储，如 Red Hat OpenShift Container Storage。



重要

如果您只有一个副本，OpenShift Container Platform 支持对镜像 registry 存储的 **ReadWriteOnce** 访问。要部署支持高可用性的、带有两个或多个副本的镜像 registry，需要 **ReadWriteMany** 访问设置。

- 必须具有 100Gi 容量。

流程

1. 为了配置 registry 使用存储，需要修改 **configs.imageregistry/cluster** 资源中的 **spec.storage.pvc** 。

注意

使用共享存储时，请查看您的安全设置以防止被外部访问。

2. 验证您没有 registry pod:

```
$ oc get pod -n openshift-image-registry
```

注意

如果存储类型为 **emptyDIR**，则副本数不能超过 **1**。

3. 检查 registry 配置：

```
$ oc edit configs.imageregistry.operator.openshift.io
```

输**出示例**

```
storage:
  pvc:
    claim:
```

将 **claim** 字段留空以允许自动创建一个 **image-registry-storage** PVC。

4. 检查 **clusteroperator** 的状态：

```
$ oc get clusteroperator image-registry
```

5. 确保您的 registry 设置为 manage，以启用镜像的构建和推送。

- 运行：

```
$ oc edit configs.imageregistry/cluster
```

然后将行改

```
managementState: Removed
```

为

```
managementState: Managed
```

1.3.15.2.3. 在非生产集群中配置镜像 registry 存储

您必须为 Image Registry Operator 配置存储。对于非生产集群，您可以将镜像 registry 设置为空目录。如果您这样做，重启 registry 后会丢失所有镜像。

流程

- 将镜像 registry 存储设置为空目录：

```
$ oc patch configs.imageregistry.operator.openshift.io cluster --type merge --patch '{"spec":{"storage":{"emptyDir":{}}}}'
```

警告

仅可为非生产集群配置这个选项。

如果在 Image Registry Operator 初始化其组件前运行此命令，**oc patch** 命令会失败并显示以下错误：

```
Error from server (NotFound): configs.imageregistry.operator.openshift.io "cluster" not found
```

等待几分钟，然后再次运行该命令。

1.3.15.2.4. 配置块 registry 存储

要允许镜像 registry 在作为集群管理员升级过程中使用块存储类型，您可以使用 **Recreate** rollout 策略。

重要

支持块存储卷，但不建议将其与生产环境中的镜像 registry 一起使用。在块存储上配置 registry 的安装不具有高可用性，因为 registry 无法拥有多个副本。

流程

1. 要将镜像 registry 存储设置为块存储类型，对 registry 进行补丁，使其使用 **Recreate** rollout 策略，并只使用一个（**1**）副本运行：

```
$ oc patch config.imageregistry.operator.openshift.io/cluster --type=merge -p '{"spec":{"rolloutStrategy":"Recreate","replicas":1}}'
```

2. 为块存储设备置备 PV，并为该卷创建 PVC。请求的块卷使用 ReadWriteOnce（RWO）访问模式。

3. 编辑 registry 配置，使其引用正确的 PVC。

1.3.16. 在用户置备的基础架构上完成安装

完成 Operator 配置后，可以在您提供的基础架构上完成集群安装。

先决条件

- 您的 control plane 已初始化。
- 已完成初始 Operator 配置。

流程

1. 使用以下命令确认所有集群组件都已在线：

```
$ watch -n5 oc get clusteroperators
```

输**出示例**

```
NAME                                       VERSION AVAILABLE   PROGRESSING   DEGRADED
SINCE
authentication                             4.6.0   True        False         False      3h56m
cloud-credential                           4.6.0   True        False         False      29h
cluster-autoscaler                         4.6.0   True        False         False      29h
config-operator                            4.6.0   True        False         False      6h39m
console                                    4.6.0   True        False         False      3h59m
csi-snapshot-controller                    4.6.0   True        False         False      4h12m
dns                                        4.6.0   True        False         False      4h15m
etcd                                       4.6.0   True        False         False      29h
image-registry                             4.6.0   True        False         False      3h59m
ingress                                    4.6.0   True        False         False      4h30m
insights                                   4.6.0   True        False         False      29h
kube-apiserver                             4.6.0   True        False         False      29h
kube-controller-manager                    4.6.0   True        False         False      29h
kube-scheduler                             4.6.0   True        False         False      29h
kube-storage-version-migrator              4.6.0   True        False         False      4h2m
machine-api                                4.6.0   True        False         False      29h
machine-approver                           4.6.0   True        False         False      6h34m
machine-config                             4.6.0   True        False         False      3h56m
marketplace                                4.6.0   True        False         False      4h2m
monitoring                                 4.6.0   True        False         False      6h31m
network                                    4.6.0   True        False         False      29h
node-tuning                                4.6.0   True        False         False      4h30m
openshift-apiserver                        4.6.0   True        False         False      3h56m
openshift-controller-manager               4.6.0   True        False         False      4h36m
openshift-samples                          4.6.0   True        False         False      4h30m
operator-lifecycle-manager                 4.6.0   True        False         False      29h
operator-lifecycle-manager-catalog         4.6.0   True        False         False      29h
operator-lifecycle-manager-packageserver   4.6.0   True        False         False      3h59m
service-ca                                 4.6.0   True        False         False      29h
storage                                    4.6.0   True        False         False      4h30m
```

或者，通过以下命令，如果所有集群都可用您会接到通知。它还检索并显示凭证：

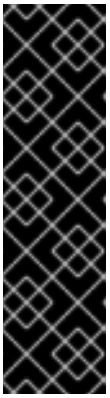
```
$ ./openshift-install --dir <installation_directory> wait-for install-complete 1
```

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

输出示例

```
INFO Waiting up to 30m0s for the cluster to initialize...
```

Cluster Version Operator 完成从 Kubernetes API 服务器部署 OpenShift Container Platform 集群时，命令运行成功。



重要

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

2. 确认 Kubernetes API 服务器正在与 pod 通信。
   a. 要查看所有 pod 的列表，请使用以下命令：

```
$ oc get pods --all-namespaces
```

输出示例

```
NAMESPACE                       NAME                                        READY   STATUS
RESTARTS   AGE
openshift-apiserver-operator    openshift-apiserver-operator-85cb746d55-zqhs8   1/1
Running    1          9m
openshift-apiserver             apiserver-67b9g                             1/1     Running     0
3m
openshift-apiserver             apiserver-ljcmx                             1/1     Running     0
1m
openshift-apiserver             apiserver-z25h4                             1/1     Running     0
2m
openshift-authentication-operator authentication-operator-69d5d8bf84-vh2n8      1/1
Running    0          5m
...
```

   b. 使用以下命令，查看上一命令的输出中所列 pod 的日志：

```
$ oc logs <pod_name> -n <namespace>
```
1

1 指定 pod 名称和命名空间，如上一命令的输出中所示。

如果 pod 日志显示，Kubernetes API 服务器可以与集群机器通信。

3. 在 Cluster registration 页面注册您的集群。

1.3.17. OpenShift Container Platform 的 Telemetry 访问

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

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

其他资源

- 有关 Telemetry 服务的更多信息，请参阅关于远程健康监控。

1.3.18. 后续步骤

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