OpenShift Container Platform 4.19

使用基于代理的安装程序安装内部集群

使用基于代理的安装程序安装内部 OpenShift Container Platform 集群

Last Updated: 2025-10-11

OpenShift Container Platform 4.19 使用基于代理的安装程序安装内部集群

使用基于代理的安装程序安装内部 OpenShift Container Platform 集群

Legal Notice

Abstract

本文档论述了如何使用基于代理的安装程序安装内部 OpenShift Container Platform 集群。

Table of Contents

第 1 章 准备使用基于代理的安装程序安装

1.1. 关于基于代理的安装程序

基于代理的安装方法提供了以任何方式引导内部服务器的灵活性。它将辅助安装服务的使用与离线运行的功能相结合，包括在 air-gapped 环境中。基于代理的安装是 OpenShift Container Platform 安装程序的子命令。它生成一个可引导的 ISO 镜像，其中包含使用可用发行镜像部署 OpenShift Container Platform 集群所需的所有信息。

配置的格式与安装程序置备的基础架构和用户置备的基础架构安装方法相同。基于代理的安装程序也可以选择生成或接受 Zero Touch Provisioning (ZTP) 自定义资源。ZTP 允许您使用裸机设备声明配置来置备新的边缘站点。

表 1.1. 基于代理的安装程序支持的构架

CPU 架构	连接安装	断开连接的安装
64-bit x86	✓	✓
64-bit ARM	✓	✓
ppc64le	✓	✓
s390x	✓	✓

1.2. 了解基于代理的安装程序

作为 OpenShift Container Platform 用户，您可以在断开连接的环境中利用 Assisted Installer 托管服务的优势。

基于代理的安装包含一个可引导 ISO，其中包含辅助发现代理和辅助服务。两者都需要执行集群安装，但后者仅在其中一个主机上运行。

注意

目前，IBM Z® (**s390x**) 上的 ISO 引导支持仅适用于 Red Hat Enterprise Linux (RHEL) KVM，这为选择 PXE 或基于 ISO 的安装提供了灵活性。对于使用 z/VM 和逻辑分区 (LPAR) 的安装，只支持 PXE 引导。

openshift-install agent create image 子命令会根据您提供的输入生成一个临时 ISO。您可以选择通过以下清单提供输入：

Preferred:

- **install-config.yaml**
- **agent-config.yaml**

可选：ZTP 清单

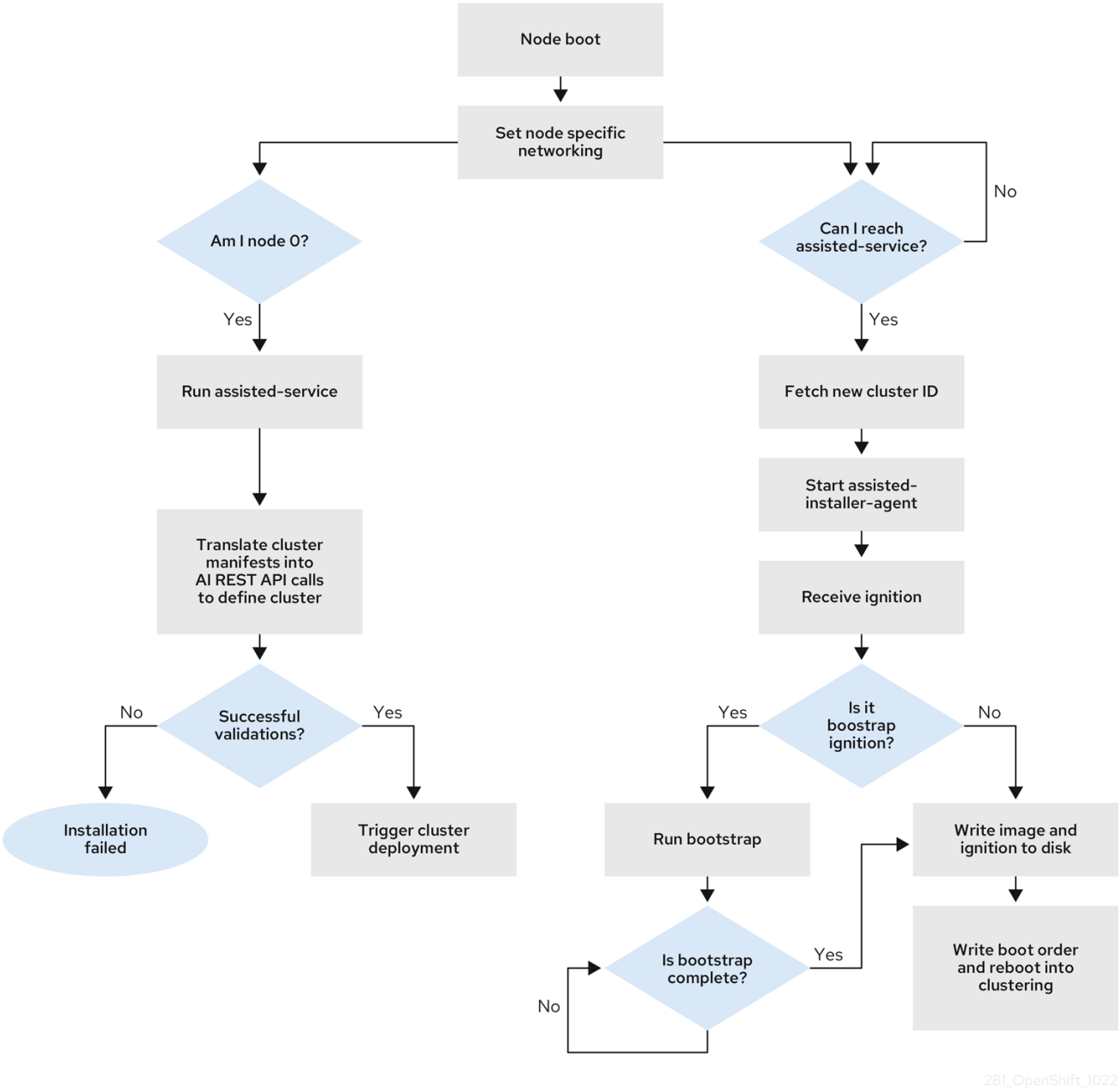
- **cluster-manifests/cluster-deployment.yaml**

- **cluster-manifests/agent-cluster-install.yaml**
- **cluster-manifests/pull-secret.yaml**
- **cluster-manifests/infraenv.yaml**
- **cluster-manifests/cluster-image-set.yaml**
- **cluster-manifests/nmstateconfig.yaml**
- **mirror/registries.conf**
- **mirror/ca-bundle.crt**

1.2.1. 基于代理的安装程序工作流

其中一个 control plane 主机在引导过程中运行 Assisted Service，最终成为 bootstrap 主机。此节点称为 **rendezvous 主机** (node 0)。Assisted Service 确保所有主机都满足要求，并触发 OpenShift Container Platform 集群部署。所有节点都将 Red Hat Enterprise Linux CoreOS (RHCOS) 镜像写入磁盘。非引导节点重新引导并启动集群部署。节点重启后，rendezvous 主机会重启并加入集群。bootstrap 已完成，并部署了集群。

图 1.1. 节点安装工作流



您可以通过 **openshift-install agent create image** 子命令为以下拓扑安装断开连接的 OpenShift Container Platform 集群：

- **单节点 OpenShift Container Platform 集群 (SNO)**：一个 master 和 worker 的节点。
- **三节点 OpenShift Container Platform 集群**：一个紧凑集群，它有三个 master 节点，也是 worker 节点。
- **高可用性 OpenShift Container Platform 集群 (HA)**：具有任意数量的 worker 节点的 master 节点。

1.2.2. 拓扑的建议资源

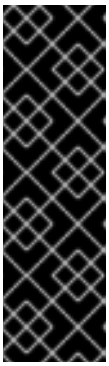
为以下拓扑推荐的集群资源：

表 1.2. 推荐的集群资源

Topology	control plane 节点数量	计算节点数量	vCPU	memory	存储
单节点集群	1	0	8 个 vCPU	16 GB RAM	120 GB
紧凑集群	3	0 或 1	8 个 vCPU	16 GB RAM	120 GB
HA 集群	3 到 5	2 及更高版本	8 个 vCPU	16 GB RAM	120 GB

在 **install-config.yaml** 中，指定在其上执行安装的平台。支持以下平台：

- **baremetal**
- **vsphere**
- **nutanix**
- **external**
- **none**



重要

对于平台，**none**:

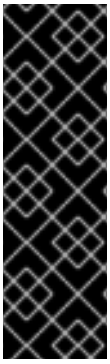
- **none** 选项需要在集群中置备 DNS 名称解析和负载平衡基础架构。如需更多信息，请参阅"Additional resources"部分中的*使用平台"none"选项集群的要求*。
- 在尝试在虚拟化或云环境中安装 OpenShift Container Platform 集群前，请参阅有关在未经测试的平台上部署 OpenShift Container Platform 的指南 中的信息。

其他资源

- 使用平台"none"选项对集群的要求
- 增加网络 MTU
- 将 worker 节点添加到单节点 OpenShift 集群

1.3. 关于 FIPS 合规性

对于许多 OpenShift Container Platform 客户，在将任何系统投入生产前需要达到一定级别的法规就绪状态或合规性。这种法规就绪状态可通过国家标准、行业标准或机构的企业监管框架来施加。
FIPS（Federal Information Processing Standards）合规性是高安全性环境中所需的最重要的组件之一，可确保节点上只允许使用支持的加密技术。



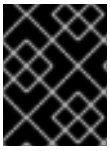
重要

要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 Red Hat Enterprise Linux (RHEL) 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅将 RHEL 切换到 FIPS 模式。

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

1.4. 通过基于代理的安装程序配置 FIPS

在集群部署期间，当在集群中部署 Red Hat Enterprise Linux CoreOS (RHCOS) 机器时，会应用联邦信息处理标准 (FIPS) 更改。在 Red Hat Enterprise Linux(RHEL) 机器中，您必须在计划用作 worker 机器的机器上安装操作系统时启用 FIPS 模式。



重要

OpenShift Container Platform 需要使用支持 FIPS 的安装二进制文件来在 FIPS 模式中安装集群。

您可以通过首选使用 **install-config.yaml** 和 **agent-config.yaml** 来启用 FIPS 模式：

1. 您必须在 **install-config.yaml** 文件中将 **fips** 字段的值设置为 **true** ：

install-config.yaml.file 示例

```
apiVersion: v1
baseDomain: test.example.com
metadata:
  name: sno-cluster
fips: true
```

重要

要在 IBM Z® 集群上启用 FIPS 模式，还需要在 **.parm** 文件中，或使用 **virt-install** 启用 FIPS，如手动添加 IBM Z® 代理的步骤中所述。

2. 可选：如果使用 GitOps ZTP 清单，您必须在 **agent-cluster-install.yaml** 文件的 **agent-install.openshift.io/install-config-overrides** 字段中将 **fips** 的值设置为 **true** ：

agent-cluster-install.yaml 文件示例

```
apiVersion: extensions.hive.openshift.io/v1beta1
kind: AgentClusterInstall
metadata:
  annotations:
    agent-install.openshift.io/install-config-overrides: '{"fips": true}'
  name: sno-cluster
  namespace: sno-cluster-test
```

其他资源

其他资源

- OpenShift 安全指南手册
- 支持 FIPS 加密

1.5. 主机配置

您可以在 **agent-config.yaml** 文件中为集群中的每个主机创建额外的配置，如网络配置和 root 设备提示。

重要

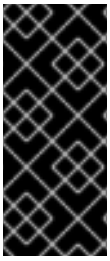
对于您配置的每个主机，您必须提供主机上接口的 MAC 地址，以指定您要配置的主机。

1.5.1. 主机角色

集群中的每个主机都被分配了 **master** 或 **worker** 的角色。您可以使用 **role** 参数为 **agent-config.yaml** 文件中的每个主机定义角色。如果您没有为主机分配角色，则会在安装过程中随机分配角色。

建议为您的主机显式定义角色。

rendezvousIP 必须分配给具有 **master** 角色的主机。这可以手动完成，或者通过允许基于代理的安装程序分配角色。



重要

您不需要为 rendezvous 主机显式定义 **master** 角色，但您无法创建与这个分配冲突的配置。

例如，如果您有 4 个主机明确定义为具有 **master** 角色，则在安装过程中自动分配 **worker** 角色的最后一个主机无法配置为 rendezvous 主机。

agent-config.yaml 文件示例

```
apiVersion: v1beta1
kind: AgentConfig
metadata:
  name: example-cluster
rendezvousIP: 192.168.111.80
hosts:
  - hostname: master-1
    role: master
    interfaces:
      - name: eno1
        macAddress: 00:ef:44:21:e6:a5
  - hostname: master-2
    role: master
    interfaces:
      - name: eno1
        macAddress: 00:ef:44:21:e6:a6
  - hostname: master-3
    role: master
    interfaces:
      - name: eno1
```

```
      macAddress: 00:ef:44:21:e6:a7
  - hostname: worker-1
    role: worker
    interfaces:
      - name: eno1
        macAddress: 00:ef:44:21:e6:a8
```

1.5.2. 关于 root 设备提示

rootDeviceHints 参数可让安装程序将 Red Hat Enterprise Linux CoreOS(RHCOS)镜像置备到特定的设备。安装程序会按照发现设备的顺序检查设备，并将发现的值与 hint 值进行比较。安装程序使用第一个与 hint 值匹配的发现设备。配置可以组合多个 hint，但设备必须与所有提示匹配，以便安装程序进行选择。

表 1.3. 子字段

子字段	描述
deviceName	包含 Linux 设备名称的字符串（如 **/dev/vda** 或 **/dev/disk/by-path/** ）。 **注意** 建议您使用 **/dev/disk/by-path/<device_path>** 链接到存储位置。 hint 必须与实际值完全匹配。
hctl	包含类似 **0:0:0:0:0 的 SCSI 总线地址的字符串**。hint 必须与实际值完全匹配。
model	包含特定厂商的设备标识符的字符串。hint 可以是实际值的子字符串。
vendor	包含该设备厂商或制造商名称的字符串。hint 可以是实际值的子字符串。
serialNumber	包含设备序列号的字符串。hint 必须与实际值完全匹配。
minSizeGigabytes	以 GB 为单位代表设备的最小大小的整数。
wwn	包含唯一存储标识符的字符串。hint 必须与实际值完全匹配。如果使用 **udevadm** 命令检索 **wwn** 值，并且命令会输出 **ID_WWN_WITH_EXTENSION** 的值，则必须使用这个值来指定 **wwn** 子字段。
rotational	指明该设备为旋转磁盘(true)还是非旋转磁盘(false)的布尔值。

用法示例

```
  - name: master-0
    role: master
    rootDeviceHints:
      deviceName: "/dev/sda"
```

1.6. 关于网络

在生成代理 ISO 时，必须知道 **rendezvous IP**，以便在初始引导过程中，所有主机都可以检查辅助服务。如果使用动态主机配置协议 (DHCP) 服务器分配 IP 地址，则必须将 **rendezvousIP** 字段设置为将成为部署 control plane 一部分的主机的 IP 地址。在没有 DHCP 服务器的环境中，您可以静态定义 IP 地址。

除了静态 IP 地址外，您还可以应用任何采用 NMState 格式的网络配置。这包括 VLAN 和 NIC 绑定。

1.6.1. DHCP

首选方法：install-config.yaml 和 agent-config.yaml

您必须为 **rendezvousIP** 字段指定值。**networkConfig** 字段可以留空：

agent-config.yaml.file 示例

```
apiVersion: v1alpha1
kind: AgentConfig
metadata:
  name: sno-cluster
rendezvousIP: 192.168.111.80 (1)
```

(1) rendezvous 主机的 IP 地址。

1.6.2. 静态网络

a. 首选方法：**install-config.yaml** 和 **agent-config.yaml**

agent-config.yaml.file 示例

```
cat > agent-config.yaml << EOF
apiVersion: v1alpha1
kind: AgentConfig
metadata:
  name: sno-cluster
rendezvousIP: 192.168.111.80 (1)
hosts:
  - hostname: master-0
    interfaces:
      - name: eno1
        macAddress: 00:ef:44:21:e6:a5 (2)
    networkConfig:
      interfaces:
        - name: eno1
          type: ethernet
          state: up
          mac-address: 00:ef:44:21:e6:a5
          ipv4:
            enabled: true
            address:
              - ip: 192.168.111.80 (3)
                prefix-length: 23 (4)
            dhcp: false
```

```
    dns-resolver:
      config:
        server:
          - 192.168.111.1 5
    routes:
      config:
        - destination: 0.0.0.0/0
          next-hop-address: 192.168.111.1 6
          next-hop-interface: eno1
          table-id: 254
EOF
```

1 如果没有为 **rendezvousIP** 字段指定值，则会从 **networkConfig** 字段中指定的静态 IP 地址选择一个地址。

2 主机上接口的 MAC 地址，用于决定要将配置应用到的主机。

3 目标裸机主机的静态 IP 地址。

4 目标裸机主机的静态 IP 地址子网前缀。

5 目标裸机主机的 DNS 服务器。

6 节点流量的下一跳地址。这必须与为特定接口设定的 IP 地址位于同一个子网中。

b. 可选方法：GitOps ZTP 清单
GitOps ZTP 自定义资源的可选方法包含 6 个自定义资源；您可以在 **nmstateconfig.yaml** 文件中配置静态 IP。

```
apiVersion: agent-install.openshift.io/v1beta1
kind: NMStateConfig
metadata:
  name: master-0
  namespace: openshift-machine-api
  labels:
    cluster0-nmstate-label-name: cluster0-nmstate-label-value
spec:
  config:
    interfaces:
      - name: eth0
        type: ethernet
        state: up
        mac-address: 52:54:01:aa:aa:a1
        ipv4:
          enabled: true
          address:
            - ip: 192.168.122.2 1
              prefix-length: 23 2
          dhcp: false
    dns-resolver:
      config:
        server:
          - 192.168.122.1 3
    routes:
```

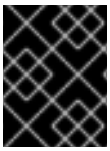
```
    config:
      - destination: 0.0.0.0/0
        next-hop-address: 192.168.122.1 4
        next-hop-interface: eth0
        table-id: 254
  interfaces:
    - name: eth0
      macAddress: 52:54:01:aa:aa:a1 5
```

1. 目标裸机主机的静态 IP 地址。
2. 目标裸机主机的静态 IP 地址子网前缀。
3. 目标裸机主机的 DNS 服务器。
4. 节点流量的下一跳地址。这必须与为特定接口设定的 IP 地址位于同一个子网中。
5. 主机上接口的 MAC 地址，用于决定要将配置应用到的主机。

rendezvous IP 从 **config** 字段中指定的静态 IP 地址中选择。

1.7. 使用平台"NONE"选项对集群的要求

本节介绍了配置为使用平台 **none** 选项的基于 Agent 的 OpenShift Container Platform 安装的要求。



重要

在尝试在虚拟化或云环境中安装 OpenShift Container Platform 集群前，请参阅有关在未经测试的平台上部署 OpenShift Container Platform 的指南 中的信息。

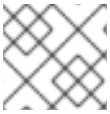
1.7.1. 平台"none" DNS 要求

在 OpenShift Container Platform 部署中，以下组件需要 DNS 名称解析：

- The Kubernetes API
- OpenShift Container Platform 应用程序通配符
- control plane 和计算机器

Kubernetes API、control plane 机器和计算机器还需要反向 DNS 解析。

DNS A/AAAA 或 CNAME 记录用于名称解析，PTR 记录用于反向名称解析。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，除非 DHCP 提供主机名。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)。



注意

建议使用 DHCP 服务器为每个群集节点提供主机名。

使用 platform **none** 选项的 OpenShift Container Platform 集群需要以下 DNS 记录，且必须在安装前就位它们。在每个记录中，**<cluster_name>** 是集群名称，**<base_domain>** 是您在 **install-config.yaml** 文件中指定的基域。完整的 DNS 记录采用以下形式：**<component>.<cluster_name>.**

<base_domain>.。

表 1.4. 所需的 DNS 记录

组件	记录	描述
Kubernetes API	**api.<cluster_name>. <base_domain>.**	DNS A/AAAA 或 CNAME 记录，以及用于标识 API 负载均衡器的 DNS PTR 记录。这些记录必须由集群外的客户端和集群中的所有节点解析。
	api-int.<cluster_name>. <base_domain>.	DNS A/AAAA 或 CNAME 记录，以及用于内部标识 API 负载均衡器的 DNS PTR 记录。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 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>.**	通配符 DNS A/AAAA 或 CNAME 记录，指向应用程序入口负载均衡器。应用程序入口负载均衡器以运行 Ingress Controller Pod 的机器为目标。默认情况下，Ingress Controller Pod 在计算机器上运行。这些记录必须由集群外的客户端和集群中的所有节点解析。 例如，console **-openshift-console.apps. <cluster_name>.<base_domain>** 用作到 OpenShift Container Platform 控制台的通配符路由。
control plane 机器	**<master><n>. <cluster_name>. <base_domain>.**	DNS A/AAAA 或 CNAME 记录，以识别 control plane 节点的每台机器。这些记录必须由集群中的节点解析。
计算机器	**<worker><n>. <cluster_name>. <base_domain>.**	DNS A/AAAA 或 CNAME 记录，用于识别 worker 节点的每台机器。这些记录必须由集群中的节点解析。

注意

在 OpenShift Container Platform 4.4 及更新的版本中，您不需要在 DNS 配置中指定 etcd 主机和 SRV 记录。

提示

您可以使用 **dig** 命令验证名称和反向名称解析。

1.7.1.1. 平台 "none" 集群的 DNS 配置示例

本节提供了 A 和 PTR 记录配置示例，它满足使用平台 **none** 选项部署 OpenShift Container Platform 的 DNS 要求。样本不是为选择一个 DNS 解决方案提供建议。

在这个示例中，集群名称为 **ocp4**，基域是 **example.com**。

平台 "none" 集群的 DNS A 记录配置示例

以下示例是 BIND 区域文件，它显示了使用 platform **none** 选项在集群中名称解析的 A 记录示例。

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.example.com.  IN A 192.168.1.5
smtp.example.com.  IN A 192.168.1.5
;
helper.example.com.  IN A 192.168.1.5
helper.ocp4.example.com. IN A 192.168.1.5
;
api.ocp4.example.com.  IN A 192.168.1.5 1
api-int.ocp4.example.com. IN A 192.168.1.5 2
;
*.apps.ocp4.example.com. IN A 192.168.1.5 3
;
master0.ocp4.example.com. IN A 192.168.1.97 4
master1.ocp4.example.com. IN A 192.168.1.98 5
master2.ocp4.example.com. IN A 192.168.1.99 6
;
worker0.ocp4.example.com. IN A 192.168.1.11 7
worker1.ocp4.example.com. IN A 192.168.1.7 8
;
;EOF
```

1. 为 Kubernetes API 提供名称解析。记录引用 API 负载均衡器的 IP 地址。
2. 为 Kubernetes API 提供名称解析。记录引用 API 负载均衡器的 IP 地址，用于内部集群通信。
3. 为通配符路由提供名称解析。记录引用应用程序入口负载均衡器的 IP 地址。应用程序入口负载均衡器以运行 Ingress Controller Pod 的机器为目标。默认情况下，Ingress Controller Pod 在计算机器上运行。

注意

在这个示例中，将相同的负载均衡器用于 Kubernetes API 和应用入口流量。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。

4 5 6 为 control plane 机器提供名称解析。

7 8 为计算机器提供名称解析。

平台 "none" 集群的 DNS PTR 记录配置示例

下面的 BIND 区文件示例显示集群中使用 platform **none** 选项反向名称解析的 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.
;
5.1.168.192.in-addr.arpa. IN PTR api.ocp4.example.com. 1
5.1.168.192.in-addr.arpa. IN PTR api-int.ocp4.example.com. 2
;
97.1.168.192.in-addr.arpa. IN PTR master0.ocp4.example.com. 3
98.1.168.192.in-addr.arpa. IN PTR master1.ocp4.example.com. 4
99.1.168.192.in-addr.arpa. IN PTR master2.ocp4.example.com. 5
;
11.1.168.192.in-addr.arpa. IN PTR worker0.ocp4.example.com. 6
7.1.168.192.in-addr.arpa. IN PTR worker1.ocp4.example.com. 7
;
;EOF
```

1 为 Kubernetes API 提供反向 DNS 解析。PTR 记录引用 API 负载均衡器的记录名称。

2 为 Kubernetes API 提供反向 DNS 解析。PTR 记录引用 API 负载均衡器的记录名称，用于内部集群通信。

3 4 5 为 control plane 机器提供反向 DNS 解析。

6 7 为计算机器提供反向 DNS 解析。

注意

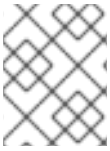
OpenShift Container Platform 应用程序通配符不需要 PTR 记录。

1.7.2. 平台"none"负载均衡要求

在安装 OpenShift Container Platform 前，您必须置备 API 和应用程序入口负载均衡基础架构。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。

注意

这些要求不适用于使用 platform **none** 选项的单节点 OpenShift 集群。



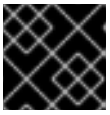
注意

如果要使用 Red Hat Enterprise Linux (RHEL) 实例部署 API 和应用程序入口负载均衡器，您必须单独购买 RHEL 订阅。

负载平衡基础架构必须满足以下要求：

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

   - 仅第 4 层负载均衡.这可称为 Raw TCP、SSL Passthrough 或 SSL 网桥模式。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施而有所不同。

   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端配置以下端口：

   表 1.5. API 负载均衡器

port	后端机器（池成员）	internal	外部	描述
6443	控制平面。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	控制平面。	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** 端点到从池中移除 API 服务器实例时最多需要 30 秒。在 **/readyz** 返回错误或健康后的时间范围内，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是经过良好测试的值。

2. **应用程序入口负载均衡器** ：为应用程序流量从集群外部流提供入口点。OpenShift Container Platform 集群需要正确配置入口路由器。
   配置以下条件：

   - 仅第 4 层负载均衡.这可称为 Raw TCP、SSL Passthrough 或 SSL 网桥模式。如果使用 SSL Bridge 模式，您必须为入口路由启用 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，使用基于连接的或基于会话的持久性。

提示

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

在负载均衡器的前端和后端配置以下端口：

表 1.6. 应用程序入口负载均衡器

port	后端机器（池成员）	internal	外部	描述
443	默认情况下，运行 Ingress Controller Pod、计算或 worker 的机器。	X	X	HTTPS 流量
80	默认情况下，运行 Ingress Controller Pod、计算或 worker 的机器。	X	X	HTTP 流量

注意

如果要部署一个带有零计算节点的三节点集群，Ingress Controller Pod 在 control plane 节点上运行。在三节点集群部署中，您必须配置应用程序入口负载均衡器，将 HTTP 和 HTTPS 流量路由到 control plane 节点。

1.7.2.1. 平台 "none" 集群的负载均衡器配置示例

本节提供了一个满足平台 **none** 选项的集群的负载均衡要求的 API 和应用程序 Ingress 负载均衡器配置示例。示例是 HAProxy 负载均衡器的 **/etc/haproxy/haproxy.cfg** 配置。这个示例不是为选择一个负载平衡解决方案提供建议。

在这个示例中，将相同的负载均衡器用于 Kubernetes API 和应用入口流量。在生产环境中，您可以单独部署 API 和应用程序入口负载均衡器，以便可以隔离扩展每个负载均衡器基础架构。

注意

如果您使用 HAProxy 作为负载均衡器，并且 SELinux 设置为 **enforcing**，您必须通过运行 **setsebool -P haproxy_connect_any=1** 来确保 HAProxy 服务可以绑定到配置的 TCP 端口。

例 1.3. API 和应用程序入口负载均衡器配置示例

```
global
  log         127.0.0.1 local2
  pidfile     /var/run/haproxy.pid
  maxconn     4000
  daemon
defaults
  mode                    http
  log                     global
  option                  dontlognull
  option http-server-close
  option                  redispatch
  retries                 3
```

```
  timeout http-request    10s
  timeout queue           1m
  timeout connect         10s
  timeout client          1m
  timeout server          1m
  timeout http-keep-alive 10s
  timeout check           10s
  maxconn                 3000
listen api-server-6443 (1)
  bind *:6443
  mode tcp
  server master0 master0.ocp4.example.com:6443 check inter 1s
  server master1 master1.ocp4.example.com:6443 check inter 1s
  server master2 master2.ocp4.example.com:6443 check inter 1s
listen machine-config-server-22623 (2)
  bind *:22623
  mode tcp
  server master0 master0.ocp4.example.com:22623 check inter 1s
  server master1 master1.ocp4.example.com:22623 check inter 1s
  server master2 master2.ocp4.example.com:22623 check inter 1s
listen ingress-router-443 (3)
  bind *:443
  mode tcp
  balance source
  server worker0 worker0.ocp4.example.com:443 check inter 1s
  server worker1 worker1.ocp4.example.com:443 check inter 1s
listen ingress-router-80 (4)
  bind *:80
  mode tcp
  balance source
  server worker0 worker0.ocp4.example.com:80 check inter 1s
  server worker1 worker1.ocp4.example.com:80 check inter 1s
```

(1) 端口 **6443** 处理 Kubernetes API 流量并指向 control plane 机器。

(2) 端口 **22623** 处理机器配置服务器流量并指向 control plane 机器。

(3) 端口 **443** 处理 HTTPS 流量，并指向运行 Ingress Controller pod 的机器。默认情况下，Ingress Controller Pod 在计算机器上运行。

(4) 端口 **80** 处理 HTTP 流量，并指向运行 Ingress Controller pod 的机器。默认情况下，Ingress Controller Pod 在计算机器上运行。

注意

如果要部署一个带有零计算节点的三节点集群，Ingress Controller Pod 在 control plane 节点上运行。在三节点集群部署中，您必须配置应用程序入口负载均衡器，将 HTTP 和 HTTPS 流量路由到 control plane 节点。

提示

如果您使用 HAProxy 作为负载均衡器，您可以通过在 HAProxy 节点上运行 **netstat -nltupe** 来检查 **haproxy** 进程是否在侦听端口 **6443**、**22623**、**443** 和 **80**。

1.8. 示例：绑定和 VLAN 接口节点网络配置

以下 **agent-config.yaml** 文件是绑定和 VLAN 接口的清单示例。

```
apiVersion: v1alpha1
kind: AgentConfig
rendezvousIP: 10.10.10.14
hosts:
  - hostname: master0
    role: master
    interfaces:
     - name: enp0s4
       macAddress: 00:21:50:90:c0:10
     - name: enp0s5
       macAddress: 00:21:50:90:c0:20
    networkConfig:
      interfaces:
        - name: bond0.300 (1)
          type: vlan (2)
          state: up
          vlan:
            base-iface: bond0
            id: 300
          ipv4:
            enabled: true
            address:
              - ip: 10.10.10.14
                prefix-length: 24
            dhcp: false
        - name: bond0 (3)
          type: bond (4)
          state: up
          mac-address: 00:21:50:90:c0:10 (5)
          ipv4:
            enabled: false
          ipv6:
            enabled: false
          link-aggregation:
            mode: active-backup (6)
            options:
              miimon: "150" (7)
            port:
             - enp0s4
             - enp0s5
      dns-resolver: (8)
        config:
          server:
            - 10.10.10.11
            - 10.10.10.12
```

```
    routes:
      config:
        - destination: 0.0.0.0/0
          next-hop-address: 10.10.10.10 9
          next-hop-interface: bond0.300 10
          table-id: 254
```

1 3 接口的名称。

2 接口的类型。这个示例创建了一个 VLAN。

4 接口的类型。这个示例创建了一个绑定。

5 接口的 mac 地址。

6 **mode** 属性指定绑定模式。

7 以毫秒为单位指定 MII 链接监控频率。这个示例每 150 毫秒检查绑定链接。

8 可选：指定 DNS 服务器的搜索和服务器设置。

9 节点流量的下一跳地址。这必须与为特定接口设定的 IP 地址位于同一个子网中。

10 节点流量的下一跳接口。

1.9. 示例：绑定和 SR-IOV 双 NIC 节点网络配置

以下 **agent-config.yaml** 文件是一个带有绑定和 SR-IOV 接口的双端口网络接口控制器(NIC)的清单示例：

```
apiVersion: v1alpha1
kind: AgentConfig
rendezvousIP: 10.10.10.14
hosts:
  - hostname: worker-1
    interfaces:
      - name: eno1
        macAddress: 0c:42:a1:55:f3:06
      - name: eno2
        macAddress: 0c:42:a1:55:f3:07
    networkConfig: 1
      interfaces: 2
        - name: eno1 3
          type: ethernet 4
          state: up
          mac-address: 0c:42:a1:55:f3:06
          ipv4:
            enabled: true
            dhcp: false 5
          ethernet:
            sr-iov:
              total-vfs: 2 6
          ipv6:
            enabled: false
```

```
- name: sriov:eno1:0
  type: ethernet
  state: up 7
  ipv4:
    enabled: false 8
  ipv6:
    enabled: false
    dhcp: false
- name: sriov:eno1:1
  type: ethernet
  state: down
- name: eno2
  type: ethernet
  state: up
  mac-address: 0c:42:a1:55:f3:07
  ipv4:
    enabled: true
  ethernet:
    sr-iov:
      total-vfs: 2
  ipv6:
    enabled: false
- name: sriov:eno2:0
  type: ethernet
  state: up
  ipv4:
    enabled: false
  ipv6:
    enabled: false
- name: sriov:eno2:1
  type: ethernet
  state: down
- name: bond0
  type: bond
  state: up
  min-tx-rate: 100 9
  max-tx-rate: 200 10
  link-aggregation:
    mode: active-backup 11
    options:
      primary: sriov:eno1:0 12
    port:
      - sriov:eno1:0
      - sriov:eno2:0
  ipv4:
    address:
      - ip: 10.19.16.57 13
        prefix-length: 23
    dhcp: false
    enabled: true
  ipv6:
    enabled: false
  dns-resolver:
    config:
      server:
```

```
        - 10.11.5.160
        - 10.2.70.215
    routes:
      config:
        - destination: 0.0.0.0/0
          next-hop-address: 10.19.17.254
          next-hop-interface: bond0 14
          table-id: 254
```

1. **networkConfig** 字段包含主机的网络配置的信息，子字段包括 **接口**、**dns-resolver** 和 **routes**。
2. **interfaces** 字段是为主机定义的网络接口数组。
3. 接口的名称。
4. 接口的类型。这个示例创建了一个以太网接口。
5. 如果物理功能 (PF) 没有严格要求，则将其设置为 **false** 以禁用 DHCP。
6. 把它设置为要实例化的 SR-IOV 虚拟功能 (VF) 的数量。
7. 把它设置为 **up**。
8. 把它设置为 **false**，以禁用附加到绑定的 VF 的 IPv4 寻址。
9. 为 VF 设置最小传输率（以 Mbps 为单位）。这个示例值设置 100 Mbps 的速度。
    - 这个值必须小于或等于最大传输率。
    - Intel NIC 不支持 **min-tx-rate** 参数。如需更多信息，请参阅 BZ#1772847。
10. 为 VF 设置最大传输率（以 Mbps 为单位）。此示例值设置 200 Mbps 的速度。
11. 设置所需的绑定模式。
12. 设置绑定接口的首选端口。主设备是要使用的绑定接口的第一个，除非失败，否则不会被取消。当绑定接口中的一个 NIC 速度更快时，此设置特别有用，因此可以处理较大的负载。只有在绑定接口处于 **active-backup** 模式（模式 1）时，此设置才有效。
13. 为绑定接口设置静态 IP 地址。这是节点 IP 地址。
14. 将 **bond0** 设置为默认路由的网关。

其他资源

- 配置网络绑定

1.10. 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 集群平台的更多详情，或修改所需参数的值。

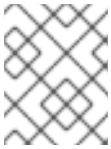
```
apiVersion: v1
baseDomain: example.com 1
```

```
compute: 2
- name: worker
  replicas: 0 3
  architecture: amd64
controlPlane: 4
  name: master
  replicas: 1 5
  architecture: amd64
metadata:
  name: sno-cluster 6
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14 7
    hostPrefix: 23 8
  networkType: OVNKubernetes 9
  serviceNetwork: 10
  - 172.30.0.0/16
platform:
  none: {} 11
fips: false 12
pullSecret: '{"auths": ...}' 13
sshKey: 'ssh-ed25519 AAAA...' 14
```

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 4 **controlPlane** 部分是一个单个映射，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** 的第一行则不以连字符开头。仅使用一个 control plane 池。

3 此参数控制基于代理的安装在触发安装过程前等待发现的计算机器数量。它是必须使用生成的 ISO 引导的计算机器数量。



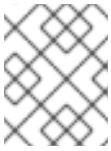
注意

如果要安装一个三节点集群，在安装 Red Hat Enterprise Linux CoreOS(RHCOS)机器时不要部署任何计算机器。

5 您添加到集群的 control plane 机器数量。由于集群使用这些值作为集群中的 etcd 端点数量，所以该值必须与您部署的 control plane 机器数量匹配。

6 您在 DNS 记录中指定的集群名称。

7 从中分配 Pod IP 地址的 IP 地址块。此块不得与现有物理网络重叠。这些 IP 地址用于 pod 网络。如果需要从外部网络访问 pod，您必须配置负载均衡器和路由器来管理流量。



注意

类 E CIDR 范围被保留以供以后使用。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。

8 分配给每个节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从 given **cidr** 中分配 a **/23** 子网，这样就能有 510（2^(32 - 23)- 2）个 pod IP 地址。如果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。

9 要安装的集群网络插件。默认值 **OVNKubernetes** 是唯一支持的值。

10 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。

11 对于单节点集群，您必须将平台设置为 **none**。对于多节点集群，您可以将平台设置为 **vsphere**、**baremetal** 或 **none**。

注意

如果将平台设置为 **vsphere** 或 **baremetal**，您可以以三种方式为集群节点配置 IP 地址端点：

- IPv4
- IPv6
- IPv4 和 IPv6 并行 (dual-stack)

双栈网络示例

```
networking:
  clusterNetwork:
    - cidr: 172.21.0.0/16
      hostPrefix: 23
    - cidr: fd02::/48
      hostPrefix: 64
  machineNetwork:
    - cidr: 192.168.11.0/16
    - cidr: 2001:DB8::/32
  serviceNetwork:
    - 172.22.0.0/16
    - fd03::/112
  networkType: OVNKubernetes
platform:
  baremetal:
    apiVIPs:
    - 192.168.11.3
    - 2001:DB8::4
    ingressVIPs:
    - 192.168.11.4
    - 2001:DB8::5
```

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

重要

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

13 此 pull secret 允许您与所含授权机构提供的服务进行身份验证，这些服务包括为 OpenShift Container Platform 组件提供容器镜像的 Quay.io。

14 Red Hat Enterprise Linux CoreOS(RHCOS)中 **core** 用户的 SSH 公钥。

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。

1.11. 在代理 ISO 创建前验证检查

基于代理的安装程序在创建 ISO 之前对用户定义的 YAML 文件执行验证检查。验证成功后，会创建代理 ISO。

install-config.yaml

- 支持**裸机**, **vsphere** 和 **none** 平台。
- 如果平台为 **none**，则 **networkType** 参数需要为 **OVNKubernetes**。
- 如果是裸机和 vSphere 平台，则需要设置 **apiVIPs** 和 **ingressVIPs** 参数。
- 在 **agent-config.yaml** 文件中，裸机平台配置中有一些特定于主机的字段将被忽略。如果设置了这些字段，则会记录警告消息。

agent-config.yaml

- 每个接口都必须有一个定义的 MAC 地址。另外，所有接口都必须具有不同的 MAC 地址。
- 每个主机必须至少定义一个接口。
- root 设备提示不支持全局名称 (WWN) 供应商扩展。
- **host** 对象中的 **role** 参数的值必须是 **master** 或 **worker**。

1.11.1. ZTP 清单

agent-cluster-install.yaml

- 对于 IPv6，**networkType** 参数唯一支持的值是 **OVNKubernetes**。**OpenshiftSDN** 值只能用于 IPv4。

cluster-image-set.yaml

- **ReleaseImage** 参数必须与安装程序中定义的发行版本匹配。

1.12. 后续步骤

- 安装集群

- 使用自定义安装集群

第 2 章 了解断开连接的安装镜像

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

2.1. 通过基于代理的安装程序为断开连接的安装镜像镜像

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

- 为断开连接的安装 mirror 镜像
- 使用 oc-mirror 插件 v2 为断开连接的安装 mirror 镜像

2.2. 关于为断开连接的 REGISTRY 镜像 OPENSHIFT CONTAINER PLATFORM 镜像存储库

要将镜像镜像用于基于代理的安装程序的断开连接的安装，您必须修改 **install-config.yaml** 文件。

您可以使用 **oc adm release mirror** 或 **oc mirror** 命令的输出来镜像发行镜像。这取决于您用来设置镜像 registry 的命令。

以下示例显示了 **oc adm release mirror** 命令的输出。

```
$ oc adm release mirror
```

输出示例

```
To use the new mirrored repository to install, add the following
section to the install-config.yaml:

imageContentSources:

mirrors:
virthost.ostest.test.metalkube.org:5000/localimages/local-release-image
source: quay.io/openshift-release-dev/ocp-v4.0-art-dev
mirrors:
virthost.ostest.test.metalkube.org:5000/localimages/local-release-image
source: registry.ci.openshift.org/ocp/release
```

以下示例显示了 oc-mirror 插件生成的 **imageContentSourcePolicy.yaml** 文件的一部分。该文件可以在结果目录中找到，如 **oc-mirror-workspace/results-1682697932/**。

imageContentSourcePolicy.yaml 文件示例

```
spec:
  repositoryDigestMirrors:
  - mirrors:
    - virthost.ostest.test.metalkube.org:5000/openshift/release
    source: quay.io/openshift-release-dev/ocp-v4.0-art-dev
  - mirrors:
    - virthost.ostest.test.metalkube.org:5000/openshift/release-images
    source: quay.io/openshift-release-dev/ocp-release
```

-

2.2.1. 配置基于代理的安装程序以使用镜像的镜像

您必须使用 **oc adm release mirror** 命令的输出或 oc-mirror 插件来配置基于 Agent 的安装程序以使用镜像镜像。

流程

1. 如果您使用 oc-mirror 插件来镜像发行镜像：
   a. 打开位于结果目录中的 **imageContentSourcePolicy.yaml**，如 **oc-mirror-workspace/results-1682697932/**。
   b. 在 yaml 文件的 **repositoryDigestMirrors** 部分中复制文本。
2. 如果使用 **oc adm release mirror** 命令镜像发行镜像：
   - 在命令输出的 **imageContentSources** 部分中复制文本。
3. 将复制的文本粘贴到 **install-config.yaml** 文件的 **imageContentSources** 字段中。
4. 将用于镜像 registry 的证书文件添加到 yaml 文件的 **additionalTrustBundle** 字段中。

重要

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

install-config.yaml 文件示例

```
additionalTrustBundle: |
  -----BEGIN CERTIFICATE-----
  ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
  -----END CERTIFICATE-----
```

5. 如果使用 GitOps ZTP 清单：将 **registry.conf** 和 **ca-bundle.crt** 文件添加到 **mirror** 路径中，以便在代理 ISO 镜像中添加镜像配置。

注意

您可以从 **oc adm release mirror** 命令的输出或 **oc mirror** 插件创建 **registries.conf** 文件。**/etc/containers/registries.conf** 文件的格式已更改。现在它是第 2 版，采用 TOML 格式。

registry.conf 文件示例

```
[[registry]]
location = "registry.ci.openshift.org/ocp/release" mirror-by-digest-only = true

[[registry.mirror]] location = "virthost.ostest.test.metalkube.org:5000/localimages/local-release-image"

[[registry]]
```

```
location = "quay.io/openshift-release-dev/ocp-v4.0-art-dev" mirror-by-digest-only = true

[[registry.mirror]] location = "virthost.ostest.test.metalkube.org:5000/localimages/local-release-image"
```

2.3. 其他资源

- 使用基于代理的安装程序安装 OpenShift Container Platform 集群

第 3 章 安装集群

您可以使用基于代理的安装程序安装基本 OpenShift Container Platform 集群。

有关包括使用基于代理的安装程序时可以进行的可选自定义的步骤，请参阅使用自定义安装集群。

3.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读选择集群安装方法并为用户准备它的文档。
- 如果使用防火墙或代理，将其配置为允许集群需要访问的站点。

3.2. 使用基于代理的安装程序安装 OPENSHIFT CONTAINER PLATFORM

以下流程在断开连接的环境中部署单节点 OpenShift Container Platform。您可以使用这些步骤作为基础，并根据您的要求进行修改。

3.2.1. 下载基于代理的安装程序

使用这个流程下载安装所需的基于代理的安装程序和 CLI。

流程

1. 使用您的登录凭证登录到 OpenShift Container Platform web 控制台。
2. 进入到 Datacenter。
3. **在本地点 Run Agent-based Installer。**
4. 为 **OpenShift Installer 和命令行界面**选择操作系统和架构。
5. 点 **Download Installer** 下载并提取安装程序。
6. 通过点 **Download pull secret** 或 **Copy pull secret** 下载或复制 pull secret。
7. 点 **Download command-line tools**，将 **openshift-install** 二进制文件放在 **PATH** 中的目录中。

3.2.2. 创建配置输入

您必须创建安装程序用来创建代理镜像的配置文件。

流程

1. 将 **openshift-install** 二进制文件放到 PATH 中的目录中。
2. 运行以下命令，创建一个目录来存储安装配置：

```
$ mkdir ~/<directory_name>
```

3. 运行以下命令来创建 **install-config.yaml** 文件：

```
$ cat << EOF > ./my-cluster/install-config.yaml
```

```
apiVersion: v1
baseDomain: test.example.com
compute:
- architecture: amd64 (1)
  hyperthreading: Enabled
  name: worker
  replicas: 0
controlPlane:
  architecture: amd64
  hyperthreading: Enabled
  name: master
  replicas: 1
metadata:
  name: sno-cluster (2)
networking:
  clusterNetwork:
  - cidr: fd01::/48
    hostPrefix: 64
  machineNetwork:
  - cidr: fd2e:6f44:5dd8:c956::/120
  networkType: OVNKubernetes (3)
  serviceNetwork:
  - fd02::/112
platform: (4)
  none: {}
pullSecret: '<pull_secret>' (5)
sshKey: '<ssh_pub_key>' (6)
additionalTrustBundle: | (7)
  -----BEGIN CERTIFICATE-----
  ZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZZ
  -----END CERTIFICATE-----
imageContentSources: (8)
- 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
EOF
```

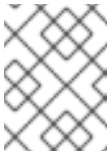
(1) 指定系统架构。有效值为 **amd64**,**arm64**,**ppc64le**, 和 **s390x**。

如果使用带有 **multi** 有效负载的发行镜像，则可以在不同的架构上安装集群，如 **arm64**, **amd64**, **s390x**, 和 **ppc64le**。否则，您只能在 **openshift-install version** 命令的输出中所显示的发**行构架**上安装集群。如需更多信息，请参阅"验证安装基于代理的安装程序集群的受支持架构"。

(2) 必需。指定集群名称。

(3) 要安装的集群网络插件。默认值 **OVNKubernetes** 是唯一支持的值。

(4) 指定您的平台。



注意

对于裸机平台，默认使用 **install-config.yaml** 文件的 platform 部分中进行的主机设置，除非它们被 **agent-config.yaml** 文件中的配置覆盖。

5 指定 pull secret。

6 指定 SSH 公钥。

7 提供用于镜像 registry 的证书文件内容。证书文件可以是现有的可信证书颁发机构，也可以是您为镜像 registry 生成的自签名证书。如果使用断开连接的镜像 registry，则必须指定此参数。

8 根据您用来镜像存储库的命令输出提供 **imageContentSources** 部分。如果使用断开连接的镜像 registry，则必须指定此参数。

重要

- 使用 **oc adm release mirror** 命令时，请使用 **imageContentSources** 部分中的输出。
- 使用 **oc mirror** 命令时，请使用 **ImageContentSourcePolicy** 文件的 **repositoryDigestMirrors** 部分，该文件源自运行命令。
- **ImageContentSourcePolicy** 资源已弃用。

4. 运行以下命令来创建 **agent-config.yaml** 文件：

```
$ cat > agent-config.yaml << EOF
apiVersion: v1beta1
kind: AgentConfig
metadata:
  name: sno-cluster
rendezvousIP: fd2e:6f44:5dd8:c956::50 1
EOF
```

1 此 IP 地址用于确定哪些节点执行 bootstrap 过程，以及运行 **assisted-service** 组件。当您没有在 **networkConfig** 参数中指定至少一个主机 IP 地址时，您必须提供 rendezvous IP 地址。如果没有提供这个地址，则会从提供的主机 **networkConfig** 参数中选择一个 IP 地址。

3.2.3. 创建并引导代理镜像

使用这个流程在机器上引导代理镜像。

流程

1. 运行以下命令来创建代理镜像：

```
$ openshift-install --dir <install_directory> agent create image
```

注意

Red Hat Enterprise Linux CoreOS (RHCOS) 支持主磁盘上的多路径，允许对硬件故障进行更强大的弹性，以实现更高的主机可用性。在代理 ISO 镜像中默认启用多路径，默认 **/etc/multipath.conf** 配置。

2. 在裸机机器上引导 **agent.x86_64.iso**,**agent.aarch64.iso**, 或 **agent.s390x.iso** 镜像。

3.2.4. 验证当前安装主机是否可以拉取发行镜像

引导代理镜像和网络服务可用于主机后，代理控制台应用会执行拉取检查，以验证当前主机是否可以检索发行镜像。

如果主拉取检查通过，您可以退出应用程序以继续安装。如果拉取检查失败，应用程序会执行额外的检查，如 TUI 的**额外检查**部分中所示，以帮助您对问题进行故障排除。只要主拉取检查成功，则对任何其他检查失败不一定至关重要。

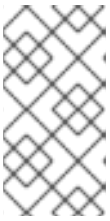
如果有可能导致安装失败的主机网络配置问题，您可以使用控制台应用程序调整网络配置。

重要

如果代理控制台应用程序检测到主机网络配置问题，则安装工作流将停止，直到用户手动停止控制台应用程序并信号继续操作。

流程

1. 等待代理控制台应用程序检查是否可以从 registry 中拉取配置的发行镜像。
2. 如果代理控制台应用程序指出安装程序连接检查已通过，请等待提示符超时。

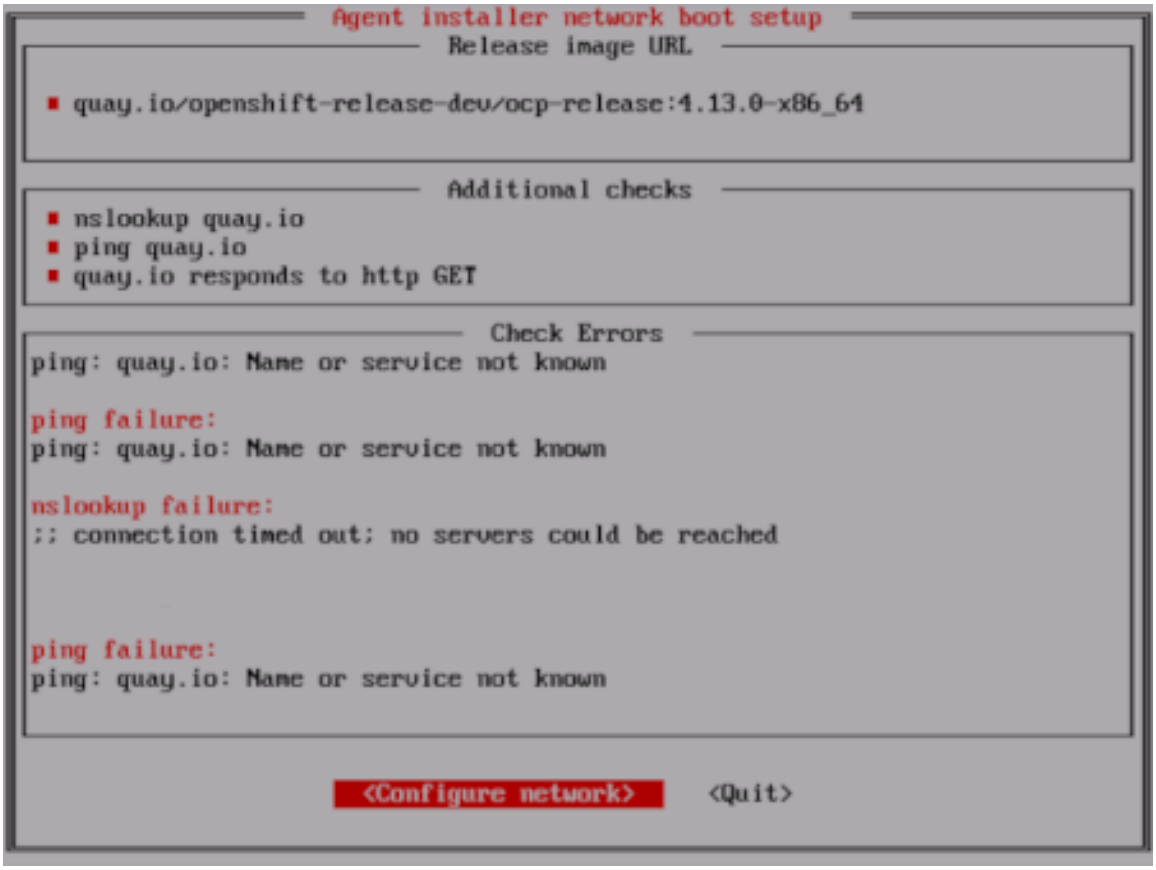


注意

您仍然可以选择查看或更改网络配置设置，即使连接检查已通过了。

但是，如果您选择与代理控制台应用程序交互，而不是让其超时，您必须手动退出 TUI 才能继续安装。

3. 如果代理控制台应用程序检查失败（由 **发行镜像 URL** pull 检查旁的红色图标表示），请按照以下步骤重新配置主机的网络设置：
   a. 阅读 TUI 的**检查错误**部分。本节显示特定于失败检查的错误消息。



b. 选择 **Configure network** 以启动 NetworkManager TUI。

c. 选择 **Edit a connection** 并选择您要重新配置的连接。

d. 编辑配置并选择 **OK** 保存您的更改。

e. 选择 **Back** 返回到 NetworkManager TUI 的主屏幕。

f. 选择 **Activate a Connection**。

g. 选择重新配置的网络来取消激活它。

h. 再次选择重新配置的网络来重新激活它。

i. 选择 **Back**，然后选择 **Quit** 以返回到代理控制台应用程序。

j. 至少等待五秒，以便持续网络检查使用新的网络配置重新启动。

k. 如果 **Release image URL** pull 检查成功，并显示 URL 旁边的绿色图标，请选择 **Quit** 退出代理控制台应用程序并继续安装。

3.2.5. 跟踪并验证安装进度

使用以下步骤跟踪安装进度并验证安装是否成功。

先决条件

- 您已为 Kubernetes API 服务器配置了 DNS 记录。

流程

1. 可选：要了解 bootstrap 主机（渲染主机）何时重启，请运行以下命令：

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

1 对于 **<install_directory>**，请指定到生成代理 ISO 的目录的路径。

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

输出示例

```
.....
.....
INFO Bootstrap configMap status is complete
INFO cluster bootstrap is complete
```

当 Kubernetes API 服务器提示已在 control plane 机器上引导它时，该命令会成功。

2. 要跟踪进度并验证安装是否成功，请运行以下命令：

```
$ openshift-install --dir <install_directory> agent wait-for install-complete 1
```

1 对于 **<install_directory>** 目录，请指定到生成代理 ISO 的目录的路径。

输出示例

```
.....
.....
INFO Cluster is installed
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run
INFO     export KUBECONFIG=/home/core/installer/auth/kubeconfig
INFO Access the OpenShift web-console here: https://console-openshift-console.apps.sno-cluster.test.example.com
```

3.3. 从基于代理的安装收集日志数据

使用以下步骤收集有关基于代理的安装失败的日志数据，以便为支持问题单提供。

先决条件

- 您已为 Kubernetes API 服务器配置了 DNS 记录。

流程

1. 运行以下命令并收集输出：

```
$ ./openshift-install --dir <installation_directory> agent wait-for bootstrap-complete --log-level=debug
```

错误信息示例

```
...
ERROR Bootstrap failed to complete: : bootstrap process timed out: context deadline
exceeded
```

2. 如果上一命令的输出显示失败，或者 bootstrap 没有进展，请运行以下命令连接到 rendezvous 主机并收集输出：

```
$ ssh core@<node-ip> agent-gather -O >agent-gather.tar.xz
```

注意

红帽支持可以使用 rendezvous 主机收集的数据诊断大多数问题，但如果某些主机无法注册，从每个主机收集这些数据可能会很有用。

3. 如果 bootstrap 完成且集群节点重启，请运行以下命令并收集输出：

```
$ ./openshift-install --dir <install_directory> agent wait-for install-complete --log-level=debug
```

4. 如果上一命令的输出显示失败，请执行以下步骤：

   a. 运行以下命令，将 **kubeconfig** 文件导出到您的环境：

   ```
   $ export KUBECONFIG=<install_directory>/auth/kubeconfig
   ```

   b. 运行以下命令为调试收集信息：

   ```
   $ oc adm must-gather
   ```

   c. 运行以下命令，从工作目录中刚刚创建的 **must-gather** 目录创建一个压缩文件：

   ```
   $ tar cvaf must-gather.tar.gz <must_gather_directory>
   ```

5. 排除 **/auth** 子目录，将部署期间使用的安装目录附加到红帽客户门户上的支持问题单中。

6. 将从此流程收集的所有其他数据添加到您的支持问题单中。

第 4 章 使用自定义安装集群

使用以下步骤使用基于代理的安装程序使用自定义的 OpenShift Container Platform 集群。

4.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读选择集群安装方法并为用户准备它的文档。
- 如果使用防火墙或代理，将其配置为允许集群需要访问的站点。

4.2. 使用基于代理的安装程序安装 OPENSHIFT CONTAINER PLATFORM

以下流程在断开连接的环境中部署单节点 OpenShift Container Platform。您可以使用这些步骤作为基础，并根据您的要求进行修改。

4.2.1. 下载基于代理的安装程序

使用这个流程下载安装所需的基于代理的安装程序和 CLI。

流程

1. 使用您的登录凭证登录到 OpenShift Container Platform web 控制台。
2. 进入到 Datacenter。
3. **在本地点 Run Agent-based Installer。**
4. 为 **OpenShift Installer** 和**命令行界面**选择操作系统和架构。
5. 点 **Download Installer** 下载并提取安装程序。
6. 通过点 **Download pull secret** 或 **Copy pull secret** 下载或复制 pull secret。
7. 点 **Download command-line tools**，将 **openshift-install** 二进制文件放在 **PATH** 中的目录中。

4.2.2. 验证基于代理的安装支持的构架

在使用基于代理的安装程序安装 OpenShift Container Platform 集群前，您可以验证要在其上安装集群的支持的架构。这个过程是可选的。

先决条件

- 已安装 OpenShift CLI（**oc**）。
- 您已下载了安装程序。

流程

1. 登录 OpenShift CLI (**oc**)。
2. 运行以下命令检查您的发行镜像有效负载：

```
$ ./openshift-install version
```

输出示例

```
./openshift-install 4.19.0
built from commit abc123def456
release image quay.io/openshift-release-dev/ocp-
release@sha256:123abc456def789ghi012jkl345mno678pqr901stu234vwx567yz0
release architecture amd64
```

如果您使用带有 **multi** 有效负载的发行镜像，这个命令的输出中显示的发**行架构**是默认的架构。

3. 要检查有效负载的架构，请运行以下命令：

```
$ oc adm release info <release_image> -o jsonpath="{ .metadata.metadata}"
```
(1)

(1) 将 **<release_image>** 替换为发行镜像。例如：**quay.io/openshift-release-dev/ocp-release@sha256:123abc456def789ghi012jkl345mno678pqr901stu234vwx567yz0**。

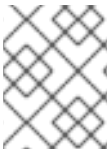
当发行镜像使用 **multi** 有效负载时的输出示例：

```
{"release.openshift.io architecture":"multi"}
```

如果使用带有 **multi** 有效负载的发行镜像，则可以在不同的架构上安装集群，如 **arm64**, **amd64**, **s390x**, 和 **ppc64le**。否则，您只能在 **openshift-install version** 命令的输出中所显示的发**行构架**上安装集群。

4.2.3. 创建首选配置输入

使用这个流程创建用于创建代理镜像的首选配置输入。



注意

配置 **install-config.yaml** 和 **agent-config.yaml** 文件是使用基于代理的安装程序的首选方法。使用 GitOps ZTP 清单是可选的。

流程

1. 运行以下命令来安装 **nmstate** 依赖项：

```
$ sudo dnf install /usr/bin/nmstatectl -y
```

2. 将 **openshift-install** 二进制文件放到 PATH 中的目录中。

3. 运行以下命令，创建一个目录来存储安装配置：

```
$ mkdir ~/<directory_name>
```

4. 运行以下命令来创建 **install-config.yaml** 文件：

```
$ cat << EOF > ./<directory_name>/install-config.yaml
```

```
apiVersion: v1
baseDomain: test.example.com
compute:
- architecture: amd64 (1)
  hyperthreading: Enabled
  name: worker
  replicas: 0
controlPlane:
  architecture: amd64
  hyperthreading: Enabled
  name: master
  replicas: 1
metadata:
  name: sno-cluster (2)
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 192.168.0.0/16
  networkType: OVNKubernetes (3)
  serviceNetwork:
  - 172.30.0.0/16
platform: (4)
  none: {}
pullSecret: '<pull_secret>' (5)
sshKey: '<ssh_pub_key>' (6)
EOF
```

(1) 指定系统架构。有效值为 **amd64**,**arm64**,**ppc64le**, 和 **s390x**。

如果使用带有 **multi** 有效负载的发行镜像，则可以在不同的架构上安装集群，如 **arm64**, **amd64**, **s390x**, 和 **ppc64le**。否则，您只能在 **openshift-install version** 命令的输出中所显示的发**行构架**上安装集群。如需更多信息，请参阅"验证安装基于代理的安装程序集群的受支持架构"。

(2) 必需。指定集群名称。

(3) 要安装的集群网络插件。默认值 **OVNKubernetes** 是唯一支持的值。

(4) 指定您的平台。

注意

对于裸机平台，默认使用 **install-config.yaml** 文件的 platform 部分中进行的主机设置，除非它们被 **agent-config.yaml** 文件中的配置覆盖。

(5) 指定 pull secret。

(6) 指定 SSH 公钥。

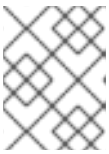
注意

如果将平台设置为 **vSphere**、**baremetal** 或 **none**，您可以以三种方式为集群节点配置 IP 地址端点：

- IPv4
- IPv6
- IPv4 和 IPv6 并行 (dual-stack)

双栈网络示例

```
networking:
  clusterNetwork:
    - cidr: 172.21.0.0/16
      hostPrefix: 23
    - cidr: fd02::/48
      hostPrefix: 64
  machineNetwork:
    - cidr: 192.168.11.0/16
    - cidr: 2001:DB8::/32
  serviceNetwork:
    - 172.22.0.0/16
    - fd03::/112
  networkType: OVNKubernetes
platform:
  baremetal:
    apiVIPs:
    - 192.168.11.3
    - 2001:DB8::4
    ingressVIPs:
    - 192.168.11.4
    - 2001:DB8::5
```



注意

使用断开连接的镜像 registry 时，您必须将之前为镜像 registry 创建的证书文件添加到 **install-config.yaml** 文件的 **additionalTrustBundle** 字段中。

5. 运行以下命令来创建 **agent-config.yaml** 文件：

```
$ cat > agent-config.yaml << EOF
apiVersion: v1beta1
kind: AgentConfig
metadata:
  name: sno-cluster
rendezvousIP: 192.168.111.80 (1)
hosts: (2)
  - hostname: master-0 (3)
    interfaces:
      - name: eno1
        macAddress: 00:ef:44:21:e6:a5
    rootDeviceHints: (4)
```

```
    deviceName: /dev/sdb
  networkConfig: 5
    interfaces:
      - name: eno1
        type: ethernet
        state: up
        mac-address: 00:ef:44:21:e6:a5
        ipv4:
          enabled: true
          address:
            - ip: 192.168.111.80
              prefix-length: 23
          dhcp: false
    dns-resolver:
      config:
        server:
          - 192.168.111.1
    routes:
      config:
        - destination: 0.0.0.0/0
          next-hop-address: 192.168.111.2
          next-hop-interface: eno1
          table-id: 254
EOF
```

1. 此 IP 地址用于确定哪些节点执行 bootstrap 过程，以及运行 **assisted-service** 组件。当您没有在 **networkConfig** 参数中指定至少一个主机的 IP 地址时，您必须提供 rendezvous IP 地址。如果没有提供此地址，则会从提供的主机的 **networkConfig** 中选择一个 IP 地址。

2. 可选：主机配置。定义的主机数量不能超过 **install-config.yaml** 文件中定义的主机总数，这是 **compute.replicas** 和 **controlPlane.replicas** 参数的值的总和。

3. 可选：覆盖从动态主机配置协议(DHCP)或反向 DNS 查找中获取的主机名。每个主机必须具有由这些方法提供的唯一主机名。

4. 启用将 Red Hat Enterprise Linux CoreOS (RHCOS)镜像置备到特定设备。安装程序会按照发现设备的顺序检查设备，并将发现的值与 hint 值进行比较。它使用第一个与 hint 值匹配的发现设备。

   注意

   对于 IBM Z 上的 FCP 多路径配置，此参数是必须的。

5. 可选：以 NMState 格式配置主机的网络接口。

其他资源

- 为 VMware vCenter 配置区域和区域
- 验证安装基于代理的安装程序集群的支持的架构
- 配置基于代理的安装程序以使用镜像的镜像

4.2.4. 创建其他清单文件

作为可选任务，您可以创建额外的清单，以便在 **install-config.yaml** 和 **agent-config.yaml** 文件中可用的配置外进一步配置集群。

重要

对由额外清单进行的集群自定义不会被验证，无法保证正常工作，并可能导致集群无法正常工作。

4.2.4.1. 创建包含额外清单的目录

如果创建额外的清单，以在 **install-config.yaml** 和 **agent-config.yaml** 文件外配置基于 Agent 的安装，则必须在安装目录中创建 **openshift** 子目录。所有附加机器配置都必须位于此子目录中。

注意

您可以添加的附加清单的最常见类型是 **MachineConfig** 对象。有关您可以在基于代理的安装过程中添加的 **MachineConfig** 对象示例，请参阅"添加资源"部分中的"使用 MachineConfig 对象来配置节点"。

流程

- 在安装主机上，运行以下命令在安装目录中创建一个 **openshift** 子目录：

```
$ mkdir <installation_directory>/openshift
```

其他资源

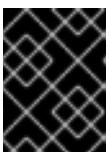
- 使用 MachineConfig 对象配置节点

4.2.4.2. 磁盘分区

通常，您应该使用在 RHCOS 安装过程中创建的默认磁盘分区。然而，在有些情况下您可能需要为预期增长的目录创建独立分区。

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

- **/var/lib/containers** ：保存随着系统中添加更多镜像和容器而增长的容器相关内容。
- **/var/lib/etcd** ：保存您可能希望独立保留的数据，比如 etcd 存储的性能优化。
- **/var** ：保存您可能希望独立保留的数据，以满足审计等目的。



重要

对于大于 100GB 的磁盘大小，特别是磁盘大小大于 1TB，请创建一个独立的 **/var** 分区。

通过单独存储 **/var** 目录的内容，可以更轻松地根据需要为区域扩展存储，并在以后重新安装 OpenShift Container Platform，并保持该数据的完整性。使用这个方法，您不必再次拉取所有容器，在更新系统时也不必复制大量日志文件。

将独立分区用于 **/var** 目录或 **/var** 的子目录也会防止分区目录中的数据增加填充根文件系统。

以下流程通过添加机器配置清单来设置独立的 **/var** 分区，该清单会在安装准备阶段封装到节点类型的 Ignition 配置文件中。

先决条件

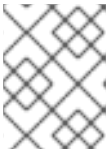
- 您已在安装目录中创建了 **openshift** 子目录。

流程

1. 创建用于配置额外分区的 Butane 配置。例如，将文件命名为 **$HOME/clusterconfig/98-var-partition.bu**，将磁盘设备名称改为 **worker** 系统上存储设备的名称，并根据情况设置存储大小。这个示例将 **/var** 目录放在一个单独的分区中：

```
variant: openshift
version: 4.19.0
metadata:
  labels:
    machineconfiguration.openshift.io/role: worker
  name: 98-var-partition
storage:
  disks:
  - device: /dev/disk/by-id/<device_name> 1
    partitions:
    - label: var
      start_mib: <partition_start_offset> 2
      size_mib: <partition_size> 3
      number: 5
  filesystems:
    - device: /dev/disk/by-partlabel/var
      path: /var
      format: xfs
      mount_options: [defaults, prjquota] 4
      with_mount_unit: true
```

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，推荐最少使用偏移值 25000 兆字节。root 文件系统会自动调整大小以填充所有可用空间（最多到指定的偏移值）。如果没有指定偏移值，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. 以兆字节为单位的数据分区大小。
4. 对于用于容器存储的文件系统，必须启用 **prjquota** 挂载选项。



注意

在创建单独的 **/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计算节点。

2. 从 Butane 配置创建一个清单，并将它保存到 **clusterconfig/openshift** 目录中。例如，运行以下命令：

```
$ butane $HOME/clusterconfig/98-var-partition.bu -o $HOME/clusterconfig/openshift/98-var-partition.yaml
```

4.2.5. 使用 ZTP 清单

作为可选任务，您可以使用 GitOps Zero Touch Provisioning (ZTP) 清单在通过 **install-config.yaml** 和 **agent-config.yaml** 文件提供的选项之外配置安装。

注意

GitOps ZTP 清单可以使用或不提前配置 **install-config.yaml** 和 **agent-config.yaml** 文件生成。如果您选择配置 **install-config.yaml** 和 **agent-config.yaml** 文件，则配置会在生成时导入到 ZTP 集群清单中。

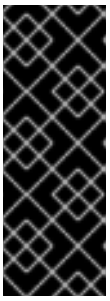
先决条件

- 您已将 **openshift-install** 二进制文件放在 **PATH** 中的目录中。
- 可选：您已创建并配置了 **install-config.yaml** 和 **agent-config.yaml** 文件。

流程

1. 运行以下命令来生成 ZTP 集群清单：

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



重要

如果您已创建了 **install-config.yaml** 和 **agent-config.yaml** 文件，则这些文件将被删除，并替换为通过这个命令生成的集群清单。

在运行 **openshift-install agent create cluster-manifests** 命令时，对 **install-config.yaml** 和 **agent-config.yaml** 文件所做的任何配置都会导入到 ZTP 集群清单中。

2. 运行以下命令，进入 **cluster-manifests** 目录：

```
$ cd <installation_directory>/cluster-manifests
```

3. 在 **cluster-manifests** 目录中配置清单文件。如需示例文件，请参阅 "Sample GitOps ZTP 自定义资源" 部分。

4. 断开连接的集群： 如果您在生成 ZTP 清单前没有在 **install-config.yaml** 文件中定义镜像配置，请执行以下步骤：

   a. 运行以下命令来进入 **mirror** 目录：

```
$ cd ../mirror
```

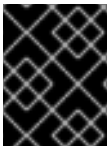
b. 在 **mirror** 目录中配置清单文件。

其他资源

- GitOps ZTP 自定义资源示例。
- 请参阅网络边缘的挑战，以了解更多有关 GitOps 零接触置备 (ZTP) 的信息。

4.2.6. 加密磁盘

作为一个可选任务，在使用基于代理的安装程序安装 OpenShift Container Platform 时，您可以使用此流程加密磁盘或分区。



重要

如果在裸机主机上有之前操作系统中剩余的 TPM 加密密钥，集群部署可能会卡住。为避免这种情况，强烈建议在引导 ISO 前在 BIOS 中重置 TPM 芯片。

先决条件

- 您已创建并配置了 **install-config.yaml** 和 **agent-config.yaml** 文件，除非您使用 ZTP 清单。
- 您已将 **openshift-install** 二进制文件放在 **PATH** 中的目录中。

流程

1. 运行以下命令来生成 ZTP 集群清单：

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

重要

如果您已创建了 **install-config.yaml** 和 **agent-config.yaml** 文件，则这些文件将被删除，并替换为通过这个命令生成的集群清单。

在运行 **openshift-install agent create cluster-manifests** 命令时，对 **install-config.yaml** 和 **agent-config.yaml** 文件所做的任何配置都会导入到 ZTP 集群清单中。

注意

如果您已经生成了 ZTP 清单，请跳过这一步。

2. 运行以下命令，进入 **cluster-manifests** 目录：

```
$ cd <installation_directory>/cluster-manifests
```

3. 在 **agent-cluster-install.yaml** 文件中添加以下部分：

```
diskEncryption:
  enableOn: all 1
  mode: tang 2
```

```
tangServers: "server1": "http://tang-server-1.example.com:7500" 3
```

1. 指定要启用磁盘加密的节点。有效值为 **none**, **all**, **masters**, 和 **workers**。
2. 指定要使用的磁盘加密模式。有效值为 **tpmv2** 和 **tang**。
3. 可选：如果您使用 Tang，请指定 Tang 服务器。

其他资源

- 关于磁盘加密

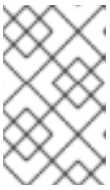
4.2.7. 创建并引导代理镜像

使用这个流程在机器上引导代理镜像。

流程

1. 运行以下命令来创建代理镜像：

```
$ openshift-install --dir <install_directory> agent create image
```



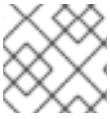
注意

Red Hat Enterprise Linux CoreOS (RHCOS) 支持主磁盘上的多路径，允许对硬件故障进行更强大的弹性，以实现更高的主机可用性。在代理 ISO 镜像中默认启用多路径，默认 **/etc/multipath.conf** 配置。

2. 在裸机机器上引导 **agent.x86_64.iso**,**agent.aarch64.iso**, 或 **agent.s390x.iso** 镜像。

4.2.8. 使用 RHEL KVM 添加 IBM Z 代理

使用以下步骤使用 RHEL KVM 手动添加 IBM Z® 代理。仅将此流程用于带有 RHEL KVM 的 IBM Z® 集群。



注意

必须为 KVM 引导配置 **nmstateconfig** 参数。

流程

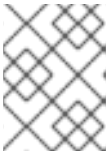
1. 引导 RHEL KVM 机器。
2. 要部署虚拟服务器，请使用以下参数运行 **virt-install** 命令：

ISO 引导

```
$ virt-install
    --name <vm_name> \
    --autostart \
    --memory=<memory> \
    --cpu host \
    --vcpus=<vcpus> \
```

```
--cdrom \<path_to_image>/<agent_iso_image> \ 1
--disk pool=default,size=<disk_pool_size> \
--network network:default,mac=<mac_address> \
--graphics none \
--noautoconsole \
--os-variant rhel9.0 \
--wait=-1
```

1 对于 **--cdrom** 参数，指定本地服务器上的 ISO 镜像的位置，例如 **<path_to_image>/home/<image>.iso**。



注意

对于在 IBM Z 上使用 DASD 设备的基于 KVM 的安装，必须使用 **fdasd** 分区工具创建一个分区（例如 **/dev/dasdb1**）。

3. 可选：启用 FIPS 模式。
要使用 RHEL KVM 在 IBM Z® 集群上启用 FIPS 模式，您必须使用 PXE 引导，并使用以下参数运行 **virt-install** 命令：

PXE 引导

```
$ virt-install \
  --name <vm_name> \
  --autostart \
  --ram=16384 \
  --cpu host \
  --vcpus=8 \
  --location <path_to_kernel_initrd_image>,kernel=kernel.img,initrd=initrd.img \ 1
  --disk <qcow_image_path> \
  --network network:macvtap ,mac=<mac_address> \
  --graphics none \
  --noautoconsole \
  --wait=-1 \
  --extra-args "rd.neednet=1 nameserver=<nameserver>" \
  --extra-args "ip=<IP>::<nameserver>::<hostname>:enc1:none" \
  --extra-args "coreos.live.rootfs_url=http://<http_server>:8080/agent.s390x-rootfs.img" \
  --extra-args "random.trust_cpu=on rd.luks.options=discard" \
  --extra-args "ignition.firstboot ignition.platform.id=metal" \
  --extra-args "console=tty1 console=ttyS1,115200n8" \
  --extra-args "coreos.inst.persistent-kargs=console=tty1 console=ttyS1,115200n8" \
  --extra-args "fips=1" \ 2
  --osinfo detect=on,require=off
```

注意

对于在 IBM Z 上使用 DASD 设备的基于 KVM 的安装，必须使用 **fdasd** 分区工具创建一个分区（例如 **/dev/dasdb1**）。

1 对于 **--location** 参数，指定 HTTP 或 HTTPS 服务器中的 kernel/initrd 的位置。

2

要启用 FIPS 模式，请指定 **fips=1**。除了 **install-config.yaml** 文件中将 **fips** 参数设置为 **true** 外，还需要此条目。

注意

目前，只支持 PXE 引导在 IBM Z® 上启用 FIPS 模式。

4.2.9. 验证当前安装主机是否可以拉取发行镜像

引导代理镜像和网络服务可用于主机后，代理控制台应用会执行拉取检查，以验证当前主机是否可以检索发行镜像。

如果主拉取检查通过，您可以退出应用程序以继续安装。如果拉取检查失败，应用程序会执行额外的检查，如 TUI 的**额外检查**部分中所示，以帮助您对问题进行故障排除。只要主拉取检查成功，则对任何其他检查失败不一定至关重要。

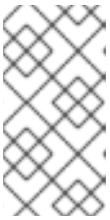
如果有可能导致安装失败的主机网络配置问题，您可以使用控制台应用程序调整网络配置。

重要

如果代理控制台应用程序检测到主机网络配置问题，则安装工作流将停止，直到用户手动停止控制台应用程序并信号继续操作。

流程

1. 等待代理控制台应用程序检查是否可以从 registry 中拉取配置的发行镜像。
2. 如果代理控制台应用程序指出安装程序连接检查已通过，请等待提示符超时。

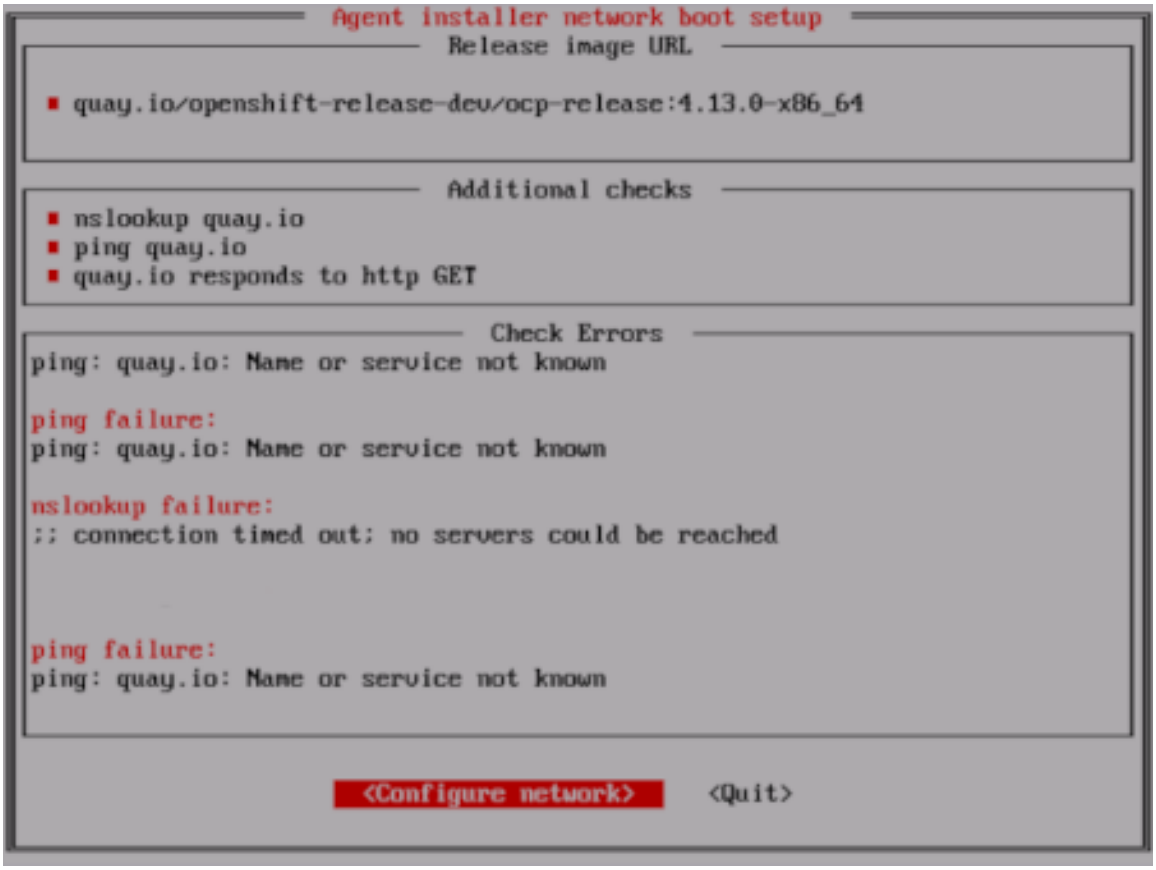


注意

您仍然可以选择查看或更改网络配置设置，即使连接检查已通过了。

但是，如果您选择与代理控制台应用程序交互，而不是让其超时，您必须手动退出 TUI 才能继续安装。

3. 如果代理控制台应用程序检查失败（由 **发行镜像 URL** pull 检查旁的红色图标表示），请按照以下步骤重新配置主机的网络设置：
   a. 阅读 TUI 的**检查错误**部分。本节显示特定于失败检查的错误消息。



b. 选择 **Configure network** 以启动 NetworkManager TUI。

c. 选择 **Edit a connection** 并选择您要重新配置的连接。

d. 编辑配置并选择 **OK** 保存您的更改。

e. 选择 **Back** 返回到 NetworkManager TUI 的主屏幕。

f. 选择 **Activate a Connection**。

g. 选择重新配置的网络来取消激活它。

h. 再次选择重新配置的网络来重新激活它。

i. 选择 **Back**，然后选择 **Quit** 以返回到代理控制台应用程序。

j. 至少等待五秒，以便持续网络检查使用新的网络配置重新启动。

k. 如果 **Release image URL** pull 检查成功，并显示 URL 旁边的绿色图标，请选择 **Quit** 退出代理控制台应用程序并继续安装。

4.2.10. 跟踪并验证安装进度

使用以下步骤跟踪安装进度并验证安装是否成功。

先决条件

- 您已为 Kubernetes API 服务器配置了 DNS 记录。

流程

1. 可选：要了解 bootstrap 主机（渲染主机）何时重启，请运行以下命令：

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

1 对于 **<install_directory>**，请指定到生成代理 ISO 的目录的路径。

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

输出示例

```
.....
.....
INFO Bootstrap configMap status is complete
INFO cluster bootstrap is complete
```

当 Kubernetes API 服务器提示已在 control plane 机器上引导它时，该命令会成功。

2. 要跟踪进度并验证安装是否成功，请运行以下命令：

```
$ openshift-install --dir <install_directory> agent wait-for install-complete 1
```

1 对于 **<install_directory>** 目录，请指定到生成代理 ISO 的目录的路径。

输出示例

```
.....
.....
INFO Cluster is installed
INFO Install complete!
INFO To access the cluster as the system:admin user when using 'oc', run
INFO     export KUBECONFIG=/home/core/installer/auth/kubeconfig
INFO Access the OpenShift web-console here: https://console-openshift-console.apps.sno-cluster.test.example.com
```

注意

如果使用可选的 GitOps ZTP 清单方法，您可以以三种方式通过 **AgentClusterInstall.yaml** 文件为集群节点配置 IP 地址端点：

- IPv4
- IPv6
- IPv4 和 IPv6 并行 (dual-stack)

IPv6 仅在裸机平台上被支持。

双栈网络示例

```
apiVIP: 192.168.11.3
```

```
ingressVIP: 192.168.11.4
clusterDeploymentRef:
  name: mycluster
imageSetRef:
  name: openshift-4.19
networking:
  clusterNetwork:
  - cidr: 172.21.0.0/16
    hostPrefix: 23
  - cidr: fd02::/48
    hostPrefix: 64
  machineNetwork:
  - cidr: 192.168.11.0/16
  - cidr: 2001:DB8::/32
  serviceNetwork:
  - 172.22.0.0/16
  - fd03::/112
  networkType: OVNKubernetes
```

其他资源

- 请参阅使用双栈网络进行部署。
- 请参阅配置 install-config yaml 文件。
- 请参阅配置三节点集群以在裸机环境中部署三节点集群。
- 请参阅关于 root 设备提示。
- 请参阅 NMState 状态示例。

4.3. GITOPS ZTP 自定义资源示例

您可以使用 GitOps Zero Touch Provisioning (ZTP) 自定义资源 (CR) 对象来使用基于代理的安装程序安装 OpenShift Container Platform 集群。

您可以自定义以下 GitOps ZTP 自定义资源，以指定有关 OpenShift Container Platform 集群的更多详情。以下 GitOps ZTP 自定义资源示例是单节点集群。

agent-cluster-install.yaml 文件示例

```
apiVersion: extensions.hive.openshift.io/v1beta1
kind: AgentClusterInstall
metadata:
  name: test-agent-cluster-install
  namespace: cluster0
spec:
  clusterDeploymentRef:
    name: ostest
  imageSetRef:
    name: openshift-4.19
  networking:
    clusterNetwork:
    - cidr: 10.128.0.0/14
      hostPrefix: 23
```

```
    serviceNetwork:
    - 172.30.0.0/16
  provisionRequirements:
    controlPlaneAgents: 1
    workerAgents: 0
  sshPublicKey: <ssh_public_key>
```

cluster-deployment.yaml 文件示例

```
apiVersion: hive.openshift.io/v1
kind: ClusterDeployment
metadata:
  name: ostest
  namespace: cluster0
spec:
  baseDomain: test.metalkube.org
  clusterInstallRef:
    group: extensions.hive.openshift.io
    kind: AgentClusterInstall
    name: test-agent-cluster-install
    version: v1beta1
  clusterName: ostest
  controlPlaneConfig:
    servingCertificates: {}
  platform:
    agentBareMetal:
      agentSelector:
        matchLabels:
          bla: aaa
  pullSecretRef:
    name: pull-secret
```

cluster-image-set.yaml 文件示例

```
apiVersion: hive.openshift.io/v1
kind: ClusterImageSet
metadata:
  name: openshift-4.19
spec:
  releaseImage: registry.ci.openshift.org/ocp/release:4.19.0-0.nightly-2022-06-06-025509
```

infra-env.yaml 文件示例

```
apiVersion: agent-install.openshift.io/v1beta1
kind: InfraEnv
metadata:
  name: myinfraenv
  namespace: cluster0
spec:
  clusterRef:
    name: ostest
    namespace: cluster0
  cpuArchitecture: aarch64
  pullSecretRef:
```

```
    name: pull-secret
  sshAuthorizedKey: <ssh_public_key>
  nmStateConfigLabelSelector:
    matchLabels:
      cluster0-nmstate-label-name: cluster0-nmstate-label-value
```

nmstateconfig.yaml 文件示例

```
apiVersion: agent-install.openshift.io/v1beta1
kind: NMStateConfig
metadata:
  name: master-0
  namespace: openshift-machine-api
  labels:
    cluster0-nmstate-label-name: cluster0-nmstate-label-value
spec:
  config:
    interfaces:
      - name: eth0
        type: ethernet
        state: up
        mac-address: 52:54:01:aa:aa:a1
        ipv4:
          enabled: true
          address:
            - ip: 192.168.122.2
              prefix-length: 23
          dhcp: false
    dns-resolver:
      config:
        server:
          - 192.168.122.1
    routes:
      config:
        - destination: 0.0.0.0/0
          next-hop-address: 192.168.122.1
          next-hop-interface: eth0
          table-id: 254
  interfaces:
    - name: "eth0"
      macAddress: 52:54:01:aa:aa:a1
```

pull-secret.yaml 文件示例

```
apiVersion: v1
kind: Secret
type: kubernetes.io/dockerconfigjson
metadata:
  name: pull-secret
  namespace: cluster0
stringData:
  .dockerconfigjson: <pull_secret>
```

其他资源

- 请参阅网络边缘的挑战，以了解更多有关 GitOps 零接触置备 (ZTP) 的信息。

4.4. 从基于代理的安装收集日志数据

使用以下步骤收集有关基于代理的安装失败的日志数据，以便为支持问题单提供。

先决条件

- 您已为 Kubernetes API 服务器配置了 DNS 记录。

流程

1. 运行以下命令并收集输出：

```
$ ./openshift-install --dir <installation_directory> agent wait-for bootstrap-complete --log-level=debug
```

错误信息示例

```
...
ERROR Bootstrap failed to complete: : bootstrap process timed out: context deadline exceeded
```

2. 如果上一命令的输出显示失败，或者 bootstrap 没有进展，请运行以下命令连接到 rendezvous 主机并收集输出：

```
$ ssh core@<node-ip> agent-gather -O >agent-gather.tar.xz
```

注意

红帽支持可以使用 rendezvous 主机收集的数据诊断大多数问题，但如果某些主机无法注册，从每个主机收集这些数据可能会很有用。

3. 如果 bootstrap 完成且集群节点重启，请运行以下命令并收集输出：

```
$ ./openshift-install --dir <install_directory> agent wait-for install-complete --log-level=debug
```

4. 如果上一命令的输出显示失败，请执行以下步骤：

   a. 运行以下命令，将 **kubeconfig** 文件导出到您的环境：

   ```
   $ export KUBECONFIG=<install_directory>/auth/kubeconfig
   ```

   b. 运行以下命令为调试收集信息：

   ```
   $ oc adm must-gather
   ```

   c. 运行以下命令，从工作目录中刚刚创建的 **must-gather** 目录创建一个压缩文件：

   ```
   $ tar cvaf must-gather.tar.gz <must_gather_directory>
   ```

5. 排除 **/auth** 子目录，将部署期间使用的安装目录附加到红帽客户门户上的支持问题单中。
6. 将从此流程收集的所有其他数据添加到您的支持问题单中。

第 5 章 为 OPENSHIFT CONTAINER PLATFORM 准备 PXE 资产

使用以下步骤创建基于代理的安装程序，创建 PXE 引导 OpenShift Container Platform 集群所需的资产。

您在这些步骤中创建的资产将部署单节点 OpenShift Container Platform 安装。您可以使用这些步骤作为基础并根据您的要求修改配置。

请参阅使用基于代理的安装程序安装 OpenShift Container Platform 集群，以了解更多有关基于代理的安装程序可用的配置。

5.1. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。

5.2. 下载基于代理的安装程序

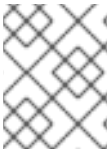
使用这个流程下载安装所需的基于代理的安装程序和 CLI。

流程

1. 使用您的登录凭证登录到 OpenShift Container Platform web 控制台。
2. 进入到 Datacenter。
3. **在本地点 Run Agent-based Installer。**
4. 为 **OpenShift Installer** 和**命令行界面**选择操作系统和架构。
5. 点 **Download Installer** 下载并提取安装程序。
6. 通过点 **Download pull secret** 或 **Copy pull secret** 下载或复制 pull secret。
7. 点 **Download command-line tools**，将 **openshift-install** 二进制文件放在 **PATH** 中的目录中。

5.3. 创建首选配置输入

使用这个流程创建用于创建 PXE 文件的首选配置输入。



注意

配置 **install-config.yaml** 和 **agent-config.yaml** 文件是使用基于代理的安装程序的首选方法。使用 GitOps ZTP 清单是可选的。

流程

1. 运行以下命令来安装 **nmstate** 依赖项：

```
$ sudo dnf install /usr/bin/nmstatectl -y
```

2. 将 **openshift-install** 二进制文件放到 PATH 中的目录中。
3. 运行以下命令，创建一个目录来存储安装配置：

```
$ mkdir ~/<directory_name>
```

4. 运行以下命令来创建 **install-config.yaml** 文件：

```
$ cat << EOF > ./<directory_name>/install-config.yaml
apiVersion: v1
baseDomain: test.example.com
compute:
- architecture: amd64 (1)
  hyperthreading: Enabled
  name: worker
  replicas: 0
controlPlane:
  architecture: amd64
  hyperthreading: Enabled
  name: master
  replicas: 1
metadata:
  name: sno-cluster (2)
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 192.168.0.0/16
  networkType: OVNKubernetes (3)
  serviceNetwork:
  - 172.30.0.0/16
platform: (4)
  none: {}
pullSecret: '<pull_secret>' (5)
sshKey: '<ssh_pub_key>' (6)
EOF
```

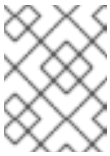
(1) 指定系统架构。有效值为 **amd64**,**arm64**,**ppc64le**, 和 **s390x**。

如果使用带有 **multi** 有效负载的发行镜像，则可以在不同的架构上安装集群，如 **arm64**, **amd64**, **s390x**, 和 **ppc64le**。否则，您只能在 **openshift-install version** 命令的输出中所显示的发**行构架**上安装集群。如需更多信息，请参阅"验证安装基于代理的安装程序集群的受支持架构"。

(2) 必需。指定集群名称。

(3) 要安装的集群网络插件。默认值 **OVNKubernetes** 是唯一支持的值。

(4) 指定您的平台。



注意

对于裸机平台，默认使用 **install-config.yaml** 文件的 platform 部分中进行的主机设置，除非它们被 **agent-config.yaml** 文件中的配置覆盖。

(5) 指定 pull secret。

6 指定 SSH 公钥。

注意

如果将平台设置为 **vSphere**、**baremetal** 或 **none**，您可以以三种方式为集群节点配置 IP 地址端点：

- IPv4
- IPv6
- IPv4 和 IPv6 并行 (dual-stack)

双栈网络示例

```
networking:
  clusterNetwork:
    - cidr: 172.21.0.0/16
      hostPrefix: 23
    - cidr: fd02::/48
      hostPrefix: 64
  machineNetwork:
    - cidr: 192.168.11.0/16
    - cidr: 2001:DB8::/32
  serviceNetwork:
    - 172.22.0.0/16
    - fd03::/112
  networkType: OVNKubernetes
platform:
  baremetal:
    apiVIPs:
    - 192.168.11.3
    - 2001:DB8::4
    ingressVIPs:
    - 192.168.11.4
    - 2001:DB8::5
```

注意

使用断开连接的镜像 registry 时，您必须将之前为镜像 registry 创建的证书文件添加到 **install-config.yaml** 文件的 **additionalTrustBundle** 字段中。

5. 运行以下命令来创建 **agent-config.yaml** 文件：

```
$ cat > agent-config.yaml << EOF
apiVersion: v1beta1
kind: AgentConfig
metadata:
  name: sno-cluster
rendezvousIP: 192.168.111.80 1
hosts: 2
  - hostname: master-0 3
```

```
    interfaces:
      - name: eno1
        macAddress: 00:ef:44:21:e6:a5
    rootDeviceHints: 4
      deviceName: /dev/sdb
    networkConfig: 5
      interfaces:
        - name: eno1
          type: ethernet
          state: up
          mac-address: 00:ef:44:21:e6:a5
          ipv4:
            enabled: true
            address:
              - ip: 192.168.111.80
                prefix-length: 23
            dhcp: false
      dns-resolver:
        config:
          server:
            - 192.168.111.1
      routes:
        config:
          - destination: 0.0.0.0/0
            next-hop-address: 192.168.111.2
            next-hop-interface: eno1
            table-id: 254
EOF
```

1 此 IP 地址用于确定哪些节点执行 bootstrap 过程，以及运行 **assisted-service** 组件。当您没有在 **networkConfig** 参数中指定至少一个主机的 IP 地址时，您必须提供 rendezvous IP 地址。如果没有提供此地址，则会从提供的主机的 **networkConfig** 中选择一个 IP 地址。

2 可选：主机配置。定义的主机数量不能超过 **install-config.yaml** 文件中定义的主机总数，这是 **compute.replicas** 和 **controlPlane.replicas** 参数的值的总和。

3 可选：覆盖从动态主机配置协议(DHCP)或反向 DNS 查找中获取的主机名。每个主机必须具有由这些方法提供的唯一主机名。

4 启用将 Red Hat Enterprise Linux CoreOS (RHCOS)镜像置备到特定设备。安装程序会按照发现设备的顺序检查设备，并将发现的值与 hint 值进行比较。它使用第一个与 hint 值匹配的发现设备。

注意

对于 IBM Z 上的 FCP 多路径配置，此参数是必须的。

5 可选：以 NMState 格式配置主机的网络接口。

6. 可选： 要创建 iPXE 脚本，请将 **bootArtifactsBaseURL** 添加到 **agent-config.yaml** 文件中：

```
apiVersion: v1beta1
kind: AgentConfig
metadata:
```

```
  name: sno-cluster
rendezvousIP: 192.168.111.80
bootArtifactsBaseURL: <asset_server_URL>
```

其中 **<asset_server_URL>** 是要将 PXE 资产上传到的服务器的 URL。

其他资源

- 使用双栈网络进行部署。
- 配置 install-config yaml 文件
- 请参阅配置三节点集群以在裸机环境中部署三节点集群。
- 关于 root 设备提示.
- nmstate 状态示例.
- 可选：创建额外的清单文件

5.4. 创建 PXE 资产

使用以下步骤创建资产和可选脚本，以便在 PXE 基础架构中实现。

流程

1. 运行以下命令来创建 PXE 资产：

```
$ openshift-install agent create pxe-files
```

生成的 PXE 资产和可选 iPXE 脚本可在 **boot-artifacts** 目录中找到。

带有 PXE 资产和可选 iPXE 脚本的文件系统示例

```
boot-artifacts
    ├── agent.x86_64-initrd.img
    ├── agent.x86_64.ipxe
    ├── agent.x86_64-rootfs.img
    └── agent.x86_64-vmlinuz
```

重要

boot-artifacts 目录的内容因指定的架构而异。

注意

Red Hat Enterprise Linux CoreOS (RHCOS) 支持主磁盘上的多路径，允许对硬件故障进行更强大的弹性，以实现更高的主机可用性。在代理 ISO 镜像中默认启用多路径，默认 **/etc/multipath.conf** 配置。

2. 将 PXE 资产和可选脚本上传到您的基础架构，以便在引导过程中访问它们。

注意

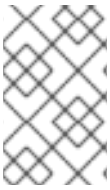
如果您生成 iPXE 脚本，资产的位置必须与添加到 **agent-config.yaml** 文件中的 **bootArtifactsBaseURL** 匹配。

5.5. 手动添加 IBM Z 代理

创建 PXE 资产后，您可以添加 IBM Z® 代理。仅将此流程用于 IBM Z® 集群。

根据您的 IBM Z® 环境，您可以从以下选项中选择：

- 使用 z/VM 添加 IBM Z® 代理
- 使用 RHEL KVM 添加 IBM Z® 代理
- 使用逻辑分区 (LPAR) 添加 IBM Z® 代理



注意

目前，IBM Z® (**s390x**) 上的 ISO 引导支持仅适用于 Red Hat Enterprise Linux (RHEL) KVM，这为选择 PXE 或基于 ISO 的安装提供了灵活性。对于使用 z/VM 和逻辑分区 (LPAR) 的安装，只支持 PXE 引导。

5.5.1. IBM Z 的网络要求

在 IBM Z 环境中，Open Systems Adapter (OSA)、HiperSockets 和 Remote Direct Memory Access (RDMA) 等高级联网技术需要通过融合以太网(RoCE)进行特定的配置，这些配置需要保留基于代理的安装的多个引导场景。

要在引导过程中保留这些参数，在 **.parm** 文件中需要 **ai.ip_cfg_override=1** 参数。这个参数与配置的网卡一起使用，以确保在 IBM Z 上成功且高效的部署。

下表列出了每个 hypervisor 支持的网络设备用于网络配置覆盖功能：

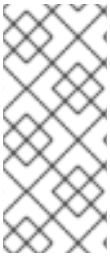
网络设备	z/VM	KVM	LPAR Classic	LPAR Dynamic Partition Manager (DPM)
虚拟交换机	支持 [1]	不适用 [2]	Not applicable	Not applicable
直接附加的 Open Systems Adapter (OSA)	支持	不是必需的 [3]	支持	不是必需的
RDMA over Converged Ethernet (RoCE)	不是必需的	不是必需的	不是必需的	不是必需的
HiperSockets	支持	不是必需的	支持	不是必需的

1. 支持 ：当安装过程需要 **ai.ip_cfg_override** 参数时。
2. 不适用：当网卡不适用于在 hypervisor 上使用时。

3. 不需要：当安装过程不需要 **ai.ip_cfg_override** 参数时。

5.5.2. 在 IBM Z 环境中配置网络覆盖

您可以在使用逻辑分区(LPAR)和 z/VM 的 IBM Z 机器中指定静态 IP 地址。当网络设备没有为其分配静态 MAC 地址时，这非常有用。



注意

如果您在 Processor Resource/Systems Manager (PR/SM) 模式下使用 OSA 网络设备，则缺少持久性 MAC 地址可能会导致为节点动态分配角色。这意味着，单个节点的角色没有被修复，且可以更改，因为系统无法可靠地将特定的 MAC 地址与指定的节点角色关联。如果 MAC 地址不对任何接口具有持久性，则节点的角色会在基于代理的安装过程中随机分配。

流程

- 如果您有一个现有的 **.parm** 文件，请编辑该文件使其包含以下条目：

```
ai.ip_cfg_override=1
```

这个参数允许文件在 Red Hat Enterprise Linux CoreOS (RHCOS) 安装程序中添加网络设置。

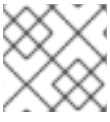
.parm 文件示例

```
rd.neednet=1 cio_ignore=all,!condev
console=ttysclp0
coreos.live.rootfs_url=<coreos_url> 1
ip=<ip>::<gateway>:<netmask>:<hostname>::none nameserver=<dns>
rd.znet=qeth,<network_adaptor_range>,layer2=1
rd.<disk_type>=<adapter> 2
rd.zfcp=<adapter>,<wwpn>,<lun> random.trust_cpu=on 3
zfcp.allow_lun_scan=0
ai.ip_cfg_override=1
ignition.firstboot ignition.platform.id=metal
random.trust_cpu=on
```

1. 对于 **coreos.live.rootfs_url** 工件，请为您要引导的**内核** 和 **initramfs** 指定匹配的 **rootfs** 工件。仅支持 HTTP 和 HTTPS 协议。

2. 对于在直接访问存储设备 (DASD) 类型磁盘中的安装，使用 **rd.** 指定要安装的 Red Hat Enterprise Linux CoreOS (RHCOS) 的 DASD。对于在光纤通道协议(FCP)磁盘中安装，请使用 **rd.zfcp=<adapter>,<wwpn>,<lun>** 指定要安装 {rhel} 的 FCP 磁盘。

3. 为 **adapter**, **wwpn**, 和 **lun** 指定值，如以下示例：**rd.zfcp=0.0.8002,0x500507630400d1e3,0x4000404600000000**。



注意

override 参数覆盖主机的网络配置设置。

5.5.3. 使用 z/VM 添加 IBM Z 代理

使用以下步骤使用 z/VM 手动添加 IBM Z® 代理。仅对带有 z/VM 的 IBM Z® 集群使用此流程。

先决条件

- 可访问客户虚拟机的运行文件服务器。

流程

1. 为 z/VM 客户机创建一个参数文件：

   参数文件示例

   +

```
rd.neednet=1 \
console=ttysclp0 \
coreos.live.rootfs_url=<rootfs_url> \ 1
ip=172.18.78.2::172.18.78.1:255.255.255.0:::none nameserver=172.18.78.1 \ 2
zfcp.allow_lun_scan=0 \ 3
ai.ip_cfg_override=1 \
rd.znet=qeth,0.0.bdd0,0.0.bdd1,0.0.bdd2,layer2=1 \
rd.dasd=0.0.4411 \ 4
rd.zfcp=0.0.8001,0x50050763040051e3,0x4000406300000000 \ 5
fips=1 \ 6
random.trust_cpu=on rd.luks.options=discard \
ignition.firstboot ignition.platform.id=metal \
console=tty1 console=ttyS1,115200n8 \
coreos.inst.persistent-kargs="console=tty1 console=ttyS1,115200n8"
```

   1. 对于 **coreos.live.rootfs_url** 工件，请为您要引导的**内核** 和 **initramfs** 指定匹配的 **rootfs** 工件。仅支持 HTTP 和 HTTPS 协议。

   2. 对于 **ip** 参数，使用 DHCP 自动分配 IP 地址，或者手动分配 IP 地址，如"在 IBM Z® 和 IBM® LinuxONE 上使用 z/VM 安装集群"中所述。

   3. 默认值为 **1**。使用 OSA 网络适配器时省略此条目。

   4. 对于在 DASD 类型磁盘中安装，请使用 **rd.dasd** 指定要安装 Red Hat Enterprise Linux CoreOS (RHCOS) 的 DASD。为 FCP 类型磁盘省略此条目。

   5. 对于在 FCP 类型磁盘中安装，请使用 **rd.zfcp=<adapter>,<wwpn>,<lun>** 指定要安装 RHCOS 的 FCP 磁盘。为 DASD 类型磁盘省略这个条目。

      注意

      对于 FCP 多路径配置，请提供两个磁盘而不是一个。

      Example

```
rd.zfcp=<adapter1>,<wwpn1>,<lun1> \
rd.zfcp=<adapter2>,<wwpn2>,<lun2>
```

6 要启用 FIPS 模式，请指定 **fips=1**。除了 **install-config.yaml** 文件中将 **fips** 参数设置为 **true** 外，还需要此条目。

所有其他参数保持不变。

2. 将 **kernel.img**、**common.parm** 和 **initrd.img** 文件与 z/VM 客户机虚拟机的虚拟读取器中解放。如需更多信息，请参阅 PUNCH (IBM 文档)。

提示

您可以使用 **CP PUNCH** 命令，或者使用 Linux (**vmur** 命令) 在两个 z/VM 虚拟机之间传输文件。

3. 登录到 bootstrap 机器上的对话监控系统 (CMS)。
4. 运行以下命令，从 reader IPL bootstrap 机器：

```
$ ipl c
```

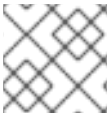
如需更多信息，请参阅 IPL (IBM 文档)。

其他资源

- 在 IBM Z 和 IBM LinuxONE 中使用 z/VM 安装集群

5.5.4. 使用 RHEL KVM 添加 IBM Z 代理

使用以下步骤使用 RHEL KVM 手动添加 IBM Z® 代理。仅将此流程用于带有 RHEL KVM 的 IBM Z® 集群。



注意

必须为 KVM 引导配置 **nmstateconfig** 参数。

流程

1. 引导 RHEL KVM 机器。
2. 要部署虚拟服务器，请使用以下参数运行 **virt-install** 命令：

```
$ virt-install \
  --name <vm_name> \
  --autostart \
  --ram=16384 \
  --cpu host \
  --vcpus=8 \
  --location <path_to_kernel_initrd_image>,kernel=kernel.img,initrd=initrd.img \ 1
  --disk <qcow_image_path> \
  --network network:macvtap ,mac=<mac_address> \
  --graphics none \
  --noautoconsole \
  --wait=-1 \
  --extra-args "rd.neednet=1 nameserver=<nameserver>" \
  --extra-args "ip=<IP>::<nameserver>::<hostname>:enc1:none" \
  --extra-args "coreos.live.rootfs_url=http://<http_server>:8080/agent.s390x-rootfs.img" \
```

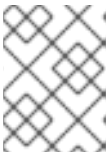
```
--extra-args "random.trust_cpu=on rd.luks.options=discard" \
--extra-args "ignition.firstboot ignition.platform.id=metal" \
--extra-args "console=tty1 console=ttyS1,115200n8" \
--extra-args "coreos.inst.persistent-kargs=console=tty1 console=ttyS1,115200n8" \
--osinfo detect=on,require=off
```

1 对于 **-location** 参数，指定 **kernel** 和 **initrd** 文件的位置。位置可以是本地服务器路径或使用 HTTP 或 HTTPS 的 URL。

3. 可选：启用 FIPS 模式。
要使用 RHEL KVM 在 IBM Z® 集群上启用 FIPS 模式，您必须使用 PXE 引导，并使用以下参数运行 **virt-install** 命令：

PXE 引导

```
$ virt-install \
  --name <vm_name> \
  --autostart \
  --ram=16384 \
  --cpu host \
  --vcpus=8 \
  --location <path_to_kernel_initrd_image>,kernel=kernel.img,initrd=initrd.img \ 1
  --disk <qcow_image_path> \
  --network network:macvtap ,mac=<mac_address> \
  --graphics none \
  --noautoconsole \
  --wait=-1 \
  --extra-args "rd.neednet=1 nameserver=<nameserver>" \
  --extra-args "ip=<IP>::<nameserver>::<hostname>:enc1:none" \
  --extra-args "coreos.live.rootfs_url=http://<http_server>:8080/agent.s390x-rootfs.img" \
  --extra-args "random.trust_cpu=on rd.luks.options=discard" \
  --extra-args "ignition.firstboot ignition.platform.id=metal" \
  --extra-args "console=tty1 console=ttyS1,115200n8" \
  --extra-args "coreos.inst.persistent-kargs=console=tty1 console=ttyS1,115200n8" \
  --extra-args "fips=1" \ 2
  --osinfo detect=on,require=off
```



注意

对于在 IBM Z 上使用 DASD 设备的基于 KVM 的安装，必须使用 **fdasd** 分区工具创建一个分区（例如 **/dev/dasdb1**）。

1 对于 **--location** 参数，指定 HTTP 或 HTTPS 服务器中的 kernel/initrd 的位置。

2 要启用 FIPS 模式，请指定 **fips=1**。除了 **install-config.yaml** 文件中将 **fips** 参数设置为 **true** 外，还需要此条目。

注意

目前，只支持 PXE 引导在 IBM Z® 上启用 FIPS 模式。

其他资源

- 在 IBM Z 和 IBM LinuxONE 中使用 RHEL KVM 安装集群

5.5.5. 在逻辑分区 (LPAR) 中添加 IBM Z 代理

使用以下步骤将 IBM Z® 代理手动添加到在 LPAR 环境中运行的集群。这个过程只适用于在 LPAR 中运行的 IBM Z® 集群。

先决条件

- 已安装 Python 3。
- 可访问逻辑分区(LPAR)的运行文件服务器。

流程

1. 为代理创建引导参数文件。

参数文件示例

```
rd.neednet=1 cio_ignore=all,!condev \
console=ttysclp0 \
ignition.firstboot ignition.platform.id=metal
coreos.live.rootfs_url=http://<http_server>/rhcos-<version>-live-rootfs.<architecture>.img \ 1
coreos.inst.persistent-kargs=console=ttysclp0 \
ip=<ip>::<gateway>:<netmask>:<hostname>::none nameserver=<dns> \ 2
rd.znet=qeth,<network_adaptor_range>,layer2=1
rd.<disk_type>=<adapter> \ 3
fips=1 \ 4
zfcp.allow_lun_scan=0 \
ai.ip_cfg_override=1 \
random.trust_cpu=on rd.luks.options=discard
```

1. 对于 **coreos.live.rootfs_url** 工件，请为您要启动的 **kernel** 和 **initramfs** 指定匹配的 **rootfs** 工件。仅支持 HTTP 和 HTTPS 协议。
2. 对于 **ip** 参数，请手动分配 IP 地址，如 *在 IBM Z 和 IBM LinuxONE 中使用 z/VM 安装集群* 中所述。
3. 对于在 DASD 类型磁盘中安装，请使用 **rd.dasd** 指定要安装 Red Hat Enterprise Linux CoreOS (RHCOS) 的 DASD。对于在 FCP 类型磁盘中安装，请使用 **rd.zfcp=<adapter>,<wwpn>,<lun>** 指定要安装 RHCOS 的 FCP 磁盘。

注意

对于 FCP 多路径配置，请提供两个磁盘而不是一个。

Example

```
rd.zfcp=<adapter1>,<wwpn1>,<lun1> \
rd.zfcp=<adapter2>,<wwpn2>,<lun2>
```

4 要启用 FIPS 模式，请指定 **fips=1**。除了 **install-config.yaml** 文件中将 **fips** 参数设置为 **true** 外，还需要此条目。

注意

.ins 和 **initrd.img.addrsize** 文件会自动为 **s390x** 架构生成，作为安装程序的 boot-artifacts 的一部分，且仅在 LPAR 环境中引导时使用。

带有 LPAR 引导的文件系统示例

```
boot-artifacts
    ├── agent.s390x-generic.ins
    ├── agent.s390x-initrd.addrsize
    ├── agent.s390x-rootfs.img
    └── agent.s390x-kernel.img
    └── agent.s390x-rootfs.img
```

2. 重命名 **generic.ins** 参数文件中的 **boot-artifacts** 文件，以匹配安装程序生成的 **boot-artifacts** 文件的名称。
3. 将 **initrd**、**kernel**、**common.ins** 和 **initrd.img.addrsize** 参数文件传输到文件服务器。如需更多信息，请参阅 在 LPAR 模式中引导 Linux (IBM 文档)。
4. 启动机器。
5. 对集群中的所有其他机器重复这个过程。

其他资源

- 在 IBM Z 和 IBM LinuxONE 的 LPAR 上安装集群

第 6 章 为 ISCSI 引导准备安装资产

您可以使用基于代理的安装程序生成的 ISO 镜像，通过互联网小型计算机系统接口(iSCSI)引导 OpenShift Container Platform 集群。以下流程描述了如何准备从 iSCSI 目标引导所需的安装资源。

您在这些步骤中创建的资产会部署单节点 OpenShift Container Platform 安装。您可以使用这些步骤作为基础并根据您的要求修改配置。

6.1. ISCSI 引导的要求

在使用基于代理的安装程序时，需要以下配置才能启用 iSCSI 引导：

- 必须配置动态主机配置协议(DHCP)。不支持静态网络。
- 您必须为与集群机器网络分开的 iSCSI 创建额外网络。机器网络会在集群安装过程中重启，不能用于 iSCSI 会话。
- 如果您要引导代理 ISO 镜像的主机也安装了磁盘，则可能需要在 **rootDeviceHints** 参数中指定 iSCSI 磁盘名称，以确保它被选为最终 Red Hat Enterprise Linux CoreOS (RHCOS) 镜像的引导磁盘。您还可以将无盘环境用于 iSCSI 引导，在这种情况下，您不需要设置 **rootDeviceHints** 参数。

其他资源

- DHCP
- 关于 root 设备提示

6.2. 先决条件

- 您可以参阅有关 OpenShift Container Platform 安装和更新 流程的详细信息。
- 您可以阅读选择集群安装方法并为用户准备它的文档。
- 如果使用防火墙或代理，将其配置为允许集群需要访问的站点。

6.3. 下载基于代理的安装程序

使用这个流程下载安装所需的基于代理的安装程序和 CLI。

流程

1. 使用您的登录凭证登录到 OpenShift Container Platform web 控制台。
2. 进入到 Datacenter。
3. **在本地点 Run Agent-based Installer。**
4. 为 **OpenShift Installer** 和**命令行界面**选择操作系统和架构。
5. 点 **Download Installer** 下载并提取安装程序。
6. 通过点 **Download pull secret** 或 **Copy pull secret** 下载或复制 pull secret。

7. 点 Download command-line tools，将 **openshift-install** 二进制文件放在 **PATH** 中的目录中。

6.4. 创建首选配置输入

使用这个流程创建用于创建代理镜像的首选配置输入。

注意

配置 **install-config.yaml** 和 **agent-config.yaml** 文件是使用基于代理的安装程序的首选方法。使用 GitOps ZTP 清单是可选的。

流程

1. 运行以下命令来安装 **nmstate** 依赖项：

```
$ sudo dnf install /usr/bin/nmstatectl -y
```

2. 将 **openshift-install** 二进制文件放到 PATH 中的目录中。

3. 运行以下命令，创建一个目录来存储安装配置：

```
$ mkdir ~/<directory_name>
```

4. 运行以下命令来创建 **install-config.yaml** 文件：

```
$ cat << EOF > ./<directory_name>/install-config.yaml
apiVersion: v1
baseDomain: test.example.com
compute:
- architecture: amd64 (1)
  hyperthreading: Enabled
  name: worker
  replicas: 0
controlPlane:
  architecture: amd64
  hyperthreading: Enabled
  name: master
  replicas: 1
metadata:
  name: sno-cluster (2)
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  machineNetwork:
  - cidr: 192.168.0.0/16
  networkType: OVNKubernetes (3)
  serviceNetwork:
  - 172.30.0.0/16
platform: (4)
  none: {}
```

```
pullSecret: '<pull_secret>' 5
sshKey: '<ssh_pub_key>' 6
EOF
```

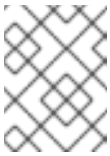
1 指定系统架构。有效值为 **amd64**,**arm64**,**ppc64le**, 和 **s390x**。

如果使用带有 **multi** 有效负载的发行镜像，则可以在不同的架构上安装集群，如 **arm64**, **amd64**, **s390x**, 和 **ppc64le**。否则，您只能在 **openshift-install version** 命令的输出中所显示的发**行构架**上安装集群。如需更多信息，请参阅"验证安装基于代理的安装程序集群的受支持架构"。

2 必需。指定集群名称。

3 要安装的集群网络插件。默认值 **OVNKubernetes** 是唯一支持的值。

4 指定您的平台。

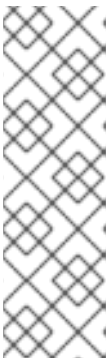


注意

对于裸机平台，默认使用 **install-config.yaml** 文件的 platform 部分中进行的主机设置，除非它们被 **agent-config.yaml** 文件中的配置覆盖。

5 指定 pull secret。

6 指定 SSH 公钥。



注意

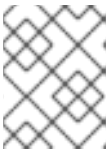
如果将平台设置为 **vSphere**、**baremetal** 或 **none**，您可以以三种方式为集群节点配置 IP 地址端点：

- IPv4
- IPv6
- IPv4 和 IPv6 并行 (dual-stack)

双栈网络示例

```
networking:
  clusterNetwork:
    - cidr: 172.21.0.0/16
      hostPrefix: 23
    - cidr: fd02::/48
      hostPrefix: 64
  machineNetwork:
    - cidr: 192.168.11.0/16
    - cidr: 2001:DB8::/32
  serviceNetwork:
    - 172.22.0.0/16
    - fd03::/112
  networkType: OVNKubernetes
platform:
```

```
baremetal:
  apiVIPs:
  - 192.168.11.3
  - 2001:DB8::4
  ingressVIPs:
  - 192.168.11.4
  - 2001:DB8::5
```



注意

使用断开连接的镜像 registry 时，您必须将之前为镜像 registry 创建的证书文件添加到 **install-config.yaml** 文件的 **additionalTrustBundle** 字段中。

5. 运行以下命令来创建 **agent-config.yaml** 文件：

```
$ cat > agent-config.yaml << EOF
apiVersion: v1beta1
kind: AgentConfig
metadata:
  name: sno-cluster
rendezvousIP: 192.168.111.80 (1)
hosts: (2)
  - hostname: master-0 (3)
    interfaces:
      - name: eno1
        macAddress: 00:ef:44:21:e6:a5
    rootDeviceHints: (4)
      deviceName: /dev/sdb
    networkConfig: (5)
      interfaces:
        - name: eno1
          type: ethernet
          state: up
          mac-address: 00:ef:44:21:e6:a5
          ipv4:
            enabled: true
            address:
              - ip: 192.168.111.80
                prefix-length: 23
            dhcp: false
      dns-resolver:
        config:
          server:
            - 192.168.111.1
      routes:
        config:
          - destination: 0.0.0.0/0
            next-hop-address: 192.168.111.2
            next-hop-interface: eno1
            table-id: 254
minimalISO: true (6)
EOF
```

(1) 此 IP 地址用于确定哪些节点执行 bootstrap 过程，以及运行 **assisted-service** 组件。当您

没有在 **networkConfig** 参数中指定至少一个主机的 IP 地址时，您必须提供 rendezvous IP 地址。如果没有提供此地址，则会从提供的主机的 **networkConfig** 中选择一个 IP 地址。

2. 可选：主机配置。定义的主机数量不能超过 **install-config.yaml** 文件中定义的主机总数，这是 **compute.replicas** 和 **controlPlane.replicas** 参数的值的总和。

3. 可选：覆盖从动态主机配置协议(DHCP)或反向 DNS 查找中获取的主机名。每个主机必须具有由这些方法提供的唯一主机名。

4. 启用将 Red Hat Enterprise Linux CoreOS (RHCOS)镜像置备到特定设备。安装程序会按照发现设备的顺序检查设备，并将发现的值与 hint 值进行比较。它使用第一个与 hint 值匹配的发现设备。

注意

对于 IBM Z 上的 FCP 多路径配置，此参数是必须的。

5. 可选：以 NMState 格式配置主机的网络接口。

6. 生成没有 rootfs 镜像文件的 ISO 镜像，并提供有关从哪里拉取 rootfs 文件的详情。您必须将此参数设置为 **true** 才能启用 iSCSI 引导。

其他资源

- 使用双栈网络进行部署
- 配置 install-config yaml 文件
- 配置三节点集群
- 关于 root 设备提示
- nmstate 状态示例 (NMState 文档)
- 可选：创建额外的清单文件
- 验证基于代理的安装支持的构架

6.5. 创建安装文件

使用以下步骤生成 ISO 镜像，并创建 iPXE 脚本以上传到 iSCSI 目标。

流程

1. 运行以下命令来创建代理镜像：

```
$ openshift-install --dir <install_directory> agent create image
```

2. 运行以下命令来创建 iPXE 脚本：

```
$ cat << EOF > agent.ipxe
!ipxe
set initiator-iqn <iscsi_initiator_base>:\${hostname}
```

```
sanboot --keep iscsi:<iscsi_network_subnet>.1::::<iscsi_target_base>:\${hostname}
EOF
```

其中：

<iscsi_initiator_base>

指定启动 ISO 的主机上 iSCSI initiator 名称。此名称也可以供 iSCSI 目标使用。

<iscsi_network_subnet>

指定 iSCSI 目标的 IP 地址。

<iscsi_target_base>

指定 iSCSI 目标名称。此名称可以与启动器名称相同。

命令示例

```
$ cat << EOF > agent.ipxe
!ipxe
set initiator-iqn iqn.2023-01.com.example:\${hostname}
sanboot --keep iscsi:192.168.45.1::::iqn.2023-01.com.example:\${hostname}
EOF
```

第 7 章 为 KUBERNETES OPERATOR 的多集群引擎准备基于 AGENT 的集群

您可以安装多集群引擎 Operator，并使用基于 Agent 的 OpenShift Container Platform 安装程序部署 hub 集群。以下流程部分自动化，在部署初始集群后需要手动步骤。

7.1. 先决条件

- 您已阅读了以下文档：
  - 带有多集群引擎 operator 的集群生命周期概述。
  - 使用本地卷的持久性存储。
  - 使用 GitOps ZTP 在网络边缘置备集群。
  - 准备使用基于代理的安装程序进行安装。
  - 关于断开连接的安装镜像。
- 您可以访问互联网来获取所需的容器镜像。
- 已安装 OpenShift CLI(**oc**)。
- 如果要在断开连接的环境中安装，则必须为断开连接的安装镜像配置了本地镜像 registry。

7.2. 断开连接时为 KUBERNETES OPERATOR 准备基于代理的集群部署

您可以在断开连接的环境中将所需的 OpenShift Container Platform 容器镜像、多集群引擎 Operator 和 Local Storage Operator (LSO) 镜像 (LSO) 到断开连接的环境中的本地镜像 registry 中。确保您注意了镜像 registry 的本地 DNS 主机名和端口。

注意

要将 OpenShift Container Platform 镜像存储库镜像到您的镜像 registry，您可以使用 **oc adm release image** 或 **oc mirror** 命令。在此过程中，**oc mirror** 命令被用作示例。

流程

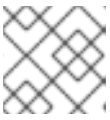
1. 创建一个 **<assets_directory>** 文件夹，使其包含有效的 **install-config.yaml** 和 **agent-config.yaml** 文件。此目录用于存储所有资产。
2. 要镜像 OpenShift Container Platform 镜像存储库、多集群引擎和 LSO，请使用以下设置创建一个 **ImageSetConfiguration.yaml** 文件：

   示例 ImageSetConfiguration.yaml

```
kind: ImageSetConfiguration
apiVersion: mirror.openshift.io/v1alpha2
archiveSize: 4 1
storageConfig: 2
  imageURL: <your-local-registry-dns-name>:<your-local-registry-port>/mirror/oc-mirror-
metadata 3
```

```
  skipTLS: true
mirror:
  platform:
    architectures:
      - "amd64"
    channels:
      - name: stable-4.19 4
        type: ocp
  additionalImages:
    - name: registry.redhat.io/ubi9/ubi:latest
  operators:
    - catalog: registry.redhat.io/redhat/redhat-operator-index:v4.19 5
      packages: 6
        - name: multicluster-engine 7
        - name: local-storage-operator 8
```

1. 指定镜像集合中每个文件的最大大小（以 GiB 为单位）。
2. 设置后端位置，以接收镜像设置元数据。此位置可以是 registry 或本地目录。必须指定 **storageConfig** 值。
3. 设置存储后端的 registry URL。
4. 设置包含您要安装的版本的 OpenShift Container Platform 镜像的频道。
5. 设置包含您要安装的 OpenShift Container Platform 镜像的 Operator 目录。
6. 仅指定要包含在镜像集中的特定 Operator 软件包和频道。删除此字段以检索目录中的所有软件包。
7. 多集群引擎软件包和频道。
8. LSO 软件包和频道。



注意

在镜像内容时，**oc mirror** 命令需要此文件。

3. 要镜像特定的 OpenShift Container Platform 镜像存储库、多集群引擎和 LSO，请运行以下命令：

```
$ oc mirror --dest-skip-tls --config ocp-mce-imageset.yaml docker://<your-local-registry-dns-name>:<your-local-registry-port>
```

4. 更新 **install-config.yaml** 文件中的 registry 和证书：

示例 imageContentSources.yaml

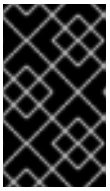
```
imageContentSources:
  - source: "quay.io/openshift-release-dev/ocp-release"
    mirrors:
      - "<your-local-registry-dns-name>:<your-local-registry-port>/openshift/release-images"
  - source: "quay.io/openshift-release-dev/ocp-v4.0-art-dev"
```

```
    mirrors:
      - "<your-local-registry-dns-name>:<your-local-registry-port>/openshift/release"
  - source: "registry.redhat.io/ubi9"
    mirrors:
      - "<your-local-registry-dns-name>:<your-local-registry-port>/ubi9"
  - source: "registry.redhat.io/multicluster-engine"
    mirrors:
      - "<your-local-registry-dns-name>:<your-local-registry-port>/multicluster-engine"
  - source: "registry.redhat.io/rhel8"
    mirrors:
      - "<your-local-registry-dns-name>:<your-local-registry-port>/rhel8"
  - source: "registry.redhat.io/redhat"
    mirrors:
      - "<your-local-registry-dns-name>:<your-local-registry-port>/redhat"
```

另外，请确保您的证书存在于 **install-config.yaml** 的 **additionalTrustBundle** 字段中。

示例 install-config.yaml

```
additionalTrustBundle: |
  -----BEGIN CERTIFICATE-----
  zzzzzzzzzzz
  -----END CERTIFICATE-------
```



重要

oc mirror 命令会创建一个名为 **oc-mirror-workspace** 的文件夹，其中包含几个输出。这包括 **imageContentSourcePolicy.yaml** 文件，用于标识 OpenShift Container Platform 和所选 Operator 所需的所有镜像。

5. 运行以下命令来生成集群清单：

```
$ openshift-install agent create cluster-manifests
```

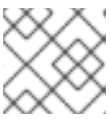
此命令更新集群 manifests 文件夹，使其包含包含您的镜像配置的 **mirror** 文件夹。

7.3. 连接时为 KUBERNETES OPERATOR 准备基于代理的集群部署

为多集群引擎 Operator、Local Storage Operator (LSO) 创建所需的清单，并将基于代理的 OpenShift Container Platform 集群部署为 hub 集群。

流程

1. 在 **<assets_directory>** 文件夹中创建一个名为 **openshift** 的子文件夹。此子文件夹用于存储在安装过程中应用的额外清单，以进一步自定义部署的集群。**<assets_directory>** 文件夹包含所有资产，包括 **install-config.yaml** 和 **agent-config.yaml** 文件。



注意

安装程序不会验证额外的清单。

2. 对于多集群引擎，创建以下清单并将其保存到 **<assets_directory>/openshift** 文件夹中：

示例 mce_namespace.yaml

```
apiVersion: v1
kind: Namespace
metadata:
  labels:
    openshift.io/cluster-monitoring: "true"
  name: multicluster-engine
```

示例 mce_operatorgroup.yaml

```
apiVersion: operators.coreos.com/v1
kind: OperatorGroup
metadata:
  name: multicluster-engine-operatorgroup
  namespace: multicluster-engine
spec:
  targetNamespaces:
  - multicluster-engine
```

示例 mce_subscription.yaml

```
apiVersion: operators.coreos.com/v1alpha1
kind: Subscription
metadata:
  name: multicluster-engine
  namespace: multicluster-engine
spec:
  channel: "stable-2.3"
  name: multicluster-engine
  source: redhat-operators
  sourceNamespace: openshift-marketplace
```

注意

您可以使用支持的安装程序 (AI) 使用 Red Hat Advanced Cluster Management (RHACM) 大规模安装分布式单元 (DU)。这些分布式单元必须在 hub 集群中启用。AI 服务需要手动创建的持久性卷 (PV)。

3. 对于 AI 服务，请创建以下清单并将其保存到 **<assets_directory>/openshift** 文件夹中：

示例 lso_namespace.yaml

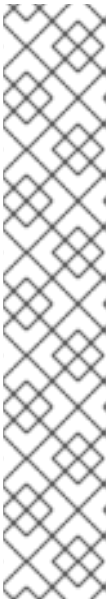
```
apiVersion: v1
kind: Namespace
metadata:
  annotations:
    openshift.io/cluster-monitoring: "true"
  name: openshift-local-storage
```

示例 lso_operatorgroup.yaml

```
apiVersion: operators.coreos.com/v1
kind: OperatorGroup
metadata:
  name: local-operator-group
  namespace: openshift-local-storage
spec:
  targetNamespaces:
    - openshift-local-storage
```

示例 lso_subscription.yaml

```
apiVersion: operators.coreos.com/v1alpha1
kind: Subscription
metadata:
  name: local-storage-operator
  namespace: openshift-local-storage
spec:
  installPlanApproval: Automatic
  name: local-storage-operator
  source: redhat-operators
  sourceNamespace: openshift-marketplace
```



注意

创建所有清单后，文件系统必须显示如下：

文件系统示例

```
<assets_directory>
    ├── install-config.yaml
    ├── agent-config.yaml
    └── /openshift
        ├── mce_namespace.yaml
        ├── mce_operatorgroup.yaml
        ├── mce_subscription.yaml
        ├── lso_namespace.yaml
        ├── lso_operatorgroup.yaml
        └── lso_subscription.yaml
```

4. 运行以下命令来创建代理 ISO 镜像：

```
$ openshift-install agent create image --dir <assets_directory>
```

5. 镜像就绪后，引导目标机器并等待安装完成。

6. 要监控安装，请运行以下命令：

```
$ openshift-install agent wait-for install-complete --dir <assets_directory>
```

注意

要配置功能齐全的 hub 集群，必须创建以下清单，并通过运行 **$ oc apply -f <manifest-name>** 命令手动应用它们。清单创建的顺序非常重要，如果需要，会显示等待条件。

7. 对于 AI 服务所需的 PV，请创建以下清单：

```
apiVersion: local.storage.openshift.io/v1
kind: LocalVolume
metadata:
 name: assisted-service
 namespace: openshift-local-storage
spec:
 logLevel: Normal
 managementState: Managed
 storageClassDevices:
   - devicePaths:
       - /dev/vda
       - /dev/vdb
     storageClassName: assisted-service
     volumeMode: Filesystem
```

8. 在应用后续清单前，使用以下命令等待 PV 的可用性：

```
$ oc wait localvolume -n openshift-local-storage assisted-service --for condition=Available --
timeout 10m
```

注意

> The `devicePath` is an example and may vary depending on the actual hardware configuration used.

9. 为多集群引擎实例创建清单。

示例 MultiClusterEngine.yaml

```
apiVersion: multicluster.openshift.io/v1
kind: MultiClusterEngine
metadata:
  name: multiclusterengine
spec: {}
```

10. 创建清单以启用 AI 服务。

示例 agentserviceconfig.yaml

```
apiVersion: agent-install.openshift.io/v1beta1
kind: AgentServiceConfig
metadata:
  name: agent
  namespace: assisted-installer
```

```
spec:
 databaseStorage:
  storageClassName: assisted-service
  accessModes:
  - ReadWriteOnce
  resources:
   requests:
    storage: 10Gi
 filesystemStorage:
  storageClassName: assisted-service
  accessModes:
  - ReadWriteOnce
  resources:
   requests:
    storage: 10Gi
```

11. 创建清单来部署后续 spoke 集群。

示例 clusterimageset.yaml

```
apiVersion: hive.openshift.io/v1
kind: ClusterImageSet
metadata:
  name: "4.19"
spec:
  releaseImage: quay.io/openshift-release-dev/ocp-release:4.19.0-x86_64
```

12. 创建一个清单来导入安装代理（托管多集群引擎和 Assisted Service）作为 hub 集群。

示例 autoimport.yaml

```
apiVersion: cluster.open-cluster-management.io/v1
kind: ManagedCluster
metadata:
 labels:
   local-cluster: "true"
   cloud: auto-detect
   vendor: auto-detect
 name: local-cluster
spec:
 hubAcceptsClient: true
```

13. 等待受管集群创建。

```
$ oc wait -n multicluster-engine managedclusters local-cluster --for
condition=ManagedClusterJoined=True --timeout 10m
```

验证

- 要确认受管集群安装成功，请运行以下命令：

```
$ oc get managedcluster
NAME            HUB ACCEPTED   MANAGED CLUSTER URLS            JOINED   AVAILABLE
AGE
```

```
local-cluster   true    https://<your cluster url>:6443   True   True   77m
```

其他资源

- Local Storage Operator

第 8 章 基于代理的安装程序的安装配置参数

在使用基于代理的安装程序部署 OpenShift Container Platform 集群前，您可以提供参数来自定义集群以及托管它的平台。在创建 **install-config.yaml** 和 **agent-config.yaml** 文件时，必须为所需参数提供值，您可以使用可选参数进一步自定义集群。

8.1. 可用的安装配置参数

下表指定您可以在基于 Agent 的安装过程中设置的必要和可选安装配置参数。

这些值在 **install-config.yaml** 文件中指定。

重要

这些设置仅用于安装，在安装后无法更改。

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 8.1. 所需的参数

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

参数	描述	值
metadata: name:	集群的名称。集群的 DNS 记录是 **{{.metadata.name}}.{{.baseDomain}}** 的子域。如果没有通过 **install-config.yaml** 或 **agent-config.yaml** 文件提供 **metadata.name** 参数，则集群名称被设置为 **agent-cluster**。例如，只使用 ZTP 清单的安装不提供 **metadata.name** 参数。	小写字母、连字符(**-**)和句点(**.**)字符串，如 **dev**。
platform:	根据执行安装的平台，对特定平台的配置：**baremetal**、**external**、**none**、**vsphere** 或 **nutanix**。	对象

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

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，或者配置与默认值不同的 IP 地址块。

在为集群配置网络参数前，请考虑以下信息：

- 如果使用 Red Hat OpenShift Networking OVN-Kubernetes 网络插件，则支持 IPv4 和 IPv6 地址系列。
- 如果您在带有支持 IPv4 和非本地 IPv6 地址的网络的 OpenShift Container Platform 集群中部署了节点，请将集群配置为使用双栈网络。
  - 对于为双栈网络配置的集群，IPv4 和 IPv6 流量都必须使用与默认网关相同的网络接口。这样可确保在一个多个网络接口控制器(NIC)环境中，集群可以根据可用的网络接口检测要使用的 NIC。如需更多信息，请参阅 *关于 OVN-Kubernetes 网络插件* 中的"OVN-Kubernetes IPv6 和双栈限制"。

- 为防止网络连接问题，请不要在支持双栈网络的主机上安装单堆栈 IPv4 集群。

如果将集群配置为使用两个 IP 地址系列，请查看以下要求：

- 两个 IP 系列都必须将相同的网络接口用于默认网关。
- 两个 IP 系列都必须具有默认网关。
- 您必须为所有网络配置参数指定 IPv4 和 IPv6 地址。例如，在以下配置中，IPv4 地址会在 IPv6 地址之前列出：

```
networking:
  clusterNetwork:
  - cidr: 10.128.0.0/14
    hostPrefix: 23
  - cidr: fd00:10:128::/56
    hostPrefix: 64
  serviceNetwork:
  - 172.30.0.0/16
  - fd00:172:16::/112
```

表 8.2. 网络参数

参数	描述	值
networking:	集群网络的配置。	对象 **注意** 您不能在安装后更改 **networking** 对象指定的参数。
networking: networkType:	要安装的 Red Hat OpenShift Networking 网络插件。	**OVNKubernetes**。**OVNKubernetes** 是 Linux 网络和包含 Linux 和 Windows 服务器的混合网络的 Container Network Interface (CNI)插件。默认值为 **OVNKubernetes**。
networking: clusterNetwork:	pod 的 IP 地址块。 默认值为 **10.128.0.0/14**，主机前缀为 **/23**。 如果您指定了多个 IP 地址块，块不得重叠。	对象数组。例如： networking: clusterNetwork: - cidr: 10.128.0.0/14 hostPrefix: 23 - cidr: fd01::/48 hostPrefix: 64

参数	描述	值
networking: clusterNetwork: cidr:	使用 **networking.clusterNetwork** 时需要此项。IP 地址块。 如果使用 OVN-Kubernetes 网络插件，您可以指定 IPv4 和 IPv6 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。IPv6 块的前缀长度介于 **0** 到 **128** 之间。例如，**10.128.0.0/14** 或 **fd01::/48**。
networking: clusterNetwork: hostPrefix:	分配给每个节点的子网前缀长度。例如，如果 **hostPrefix** 设为 **23**，则每个节点从 given **cidr** 中分配 a/**23** 子网。**hostPrefix** 值 **23** 提供 510（2^(32 - 23)- 2）pod IP 地址。	子网前缀。 对于 IPv4 网络，默认值为 **23**。对于 IPv6 网络，默认值为 **64**。默认值也是 IPv6 的最小值。
networking: serviceNetwork:	服务的 IP 地址块。默认值为 **172.30.0.0/16**。 OVN-Kubernetes 网络插件只支持服务网络的一个 IP 地址块。 如果使用 OVN-Kubernetes 网络插件，您可以为 IPv4 和 IPv6 地址系列指定一个 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 - fd02::/112
networking: machineNetwork:	机器的 IP 地址块。 如果您指定了多个 IP 地址块，块不得重叠。	对象数组。例如： networking: machineNetwork: - cidr: 10.0.0.0/16
networking: machineNetwork: cidr:	使用 **networking.machineNetwork** 时需要此项。IP 地址块。对于 libvirt 和 IBM Power® Virtual Server 以外的所有平台，默认值为 **10.0.0.0/16**。对于 libvirt，默认值 为 **192.168.126.0/24**。对于 IBM Power® Virtual Server，默认值为 **192.168.0.0/24**。	CIDR 表示法中的 IP 网络块。 例如。**10.0.0.0/16** 或 **fd00::/48**。 **注意** 将 **networking.machineNetwork** 设置为与首选 NIC 所在的 CIDR 匹配。
networking: ovnKubernetesConfig: ipv4: internalJoinSubnet:	配置 **ovn-kubernetes** 内部使用的 IPv4 加入子网。此子网不得与 OpenShift Container Platform 使用的任何其他子网重叠，包括节点网络。子网的大小必须大于节点的数量。您不能在安装后更改值。	CIDR 表示法中的 IP 网络块。默认值为 **100.64.0.0/16**。

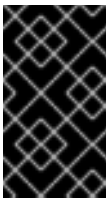
8.1.3. 可选的配置参数

下表描述了可选的安装配置参数：

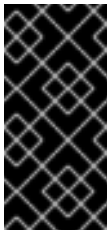
表 8.3. 可选参数

参数	描述	值
additionalTrustBundle:	添加到节点可信证书存储中的 PEM 编码 X.509 证书捆绑包。配置代理时，也可以使用此信任捆绑包。	字符串
capabilities:	控制可选核心组件的安装。您可以通过禁用可选组件来减少 OpenShift Container Platform 集群的空间。如需更多信息，请参阅*安装*中的"集群功能"页面。	字符串数组
capabilities: baselineCapabilitySet:	选择要启用的一组初始可选功能。有效值为 **None**、**v4.11**、**v4.12** 和 **vCurrent**。默认值为 **vCurrent**。	字符串
capabilities: additionalEnabledCapabilities:	将可选功能集合扩展到您在 **baselineCapabilitySet** 中指定的范围。您可以在此参数中指定多个功能。	字符串数组
cpuPartitioningMode:	启用工作负载分区，它会隔离 OpenShift Container Platform 服务、集群管理工作负载和基础架构 pod，以便在保留的一组 CPU 上运行。您只能在安装过程中启用工作负载分区。您不能在安装后禁用它。虽然此字段启用工作负载分区，但它不会将工作负载配置为使用特定的 CPU。如需更多信息，请参阅 *Scalability and Performance* 部分中的 *Workload partitioning* 页面。	**None** 或 **AllNodes**.**None** 是默认值。
compute:	组成计算节点的机器的配置。	**MachinePool** 对象的数组。
compute: architecture:	决定池中机器的指令集合架构。目前，不支持具有不同架构的集群。所有池都必须指定相同的架构。有效值为 **amd64**,**arm64**,**ppc64le**, 和 **s390x**。	字符串

参数	描述	值
compute: hyperthreading:	是否在计算机器上启用或禁用并发多**线程或超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。	**enabled** 或 **Disabled**
compute: name:	使用 **compute** 时需要此项。机器池的名称。	**worker**
compute: platform:	使用 **compute** 时需要此项。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。	**baremetal**、**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**。
featureSet:	为功能集启用集群。功能集是 OpenShift Container Platform 功能的集合，默认情况下不启用。有关在安装过程中启用功能集的更多信息，请参阅"使用功能门启用功能"。	字符串.要启用的功能集的名称，如 **TechPreviewNoUpgrade**。
controlPlane:	组成 control plane 的机器的配置。	**MachinePool** 对象的数组。
controlPlane: architecture:	决定池中机器的指令集合架构。目前，不支持具有不同架构的集群。所有池都必须指定相同的架构。有效值为 **amd64**,**arm64**,**ppc64le**, 和 **s390x**。	字符串

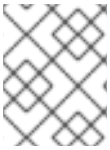
参数	描述	值
controlPlane: hyperthreading:	是否在 control plane 机器上启用或禁用并发多 **线程或超线程**。默认情况下，启用并发多线程以提高机器内核的性能。  **重要** 如果您禁用并发多线程，请确保您的容量规划考虑机器性能显著降低的情况。	**enabled** 或 **Disabled**
controlPlane: name:	使用 **controlPlane** 时需要此项。机器池的名称。	**master**
controlPlane: platform:	使用 **controlPlane** 时需要此项。使用此参数指定托管 control plane 机器的云供应商。此参数值必须与 **compute.platform** 参数值匹配。	**baremetal**、**vsphere** 或 **{}**
controlPlane: replicas:	要置备的 control plane 机器数量。	在部署单节点 OpenShift 时，支持的值包括 **3**、**4**、**5** 或 **1**。
credentialsMode:	Cloud Credential Operator(CCO)模式。如果没有指定模式，CCO 会动态尝试决定提供的凭证的功能，在支持多个模式的平台上首选 mint 模式。  **注意** 不是所有 CCO 模式都支持所有云供应商。有关 CCO 模式的更多信息，请参阅 *身份验证和授权*内容中的"管理云供应商凭证"条目。	**Mint**、**Passthrough**、**Manual** 或空字符串(**""**)。
fips:	启用或禁用 FIPS 模式。默认值为 **false**（禁用）。如果启用 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 RHCOS 提供的加密模块。	**false** 或 **true**

参数	描述	值
	重要 要为集群启用 FIPS 模式，您必须从配置为以 FIPS 模式操作的 Red Hat Enterprise Linux (RHEL) 计算机运行安装程序。有关在 RHEL 中配置 FIPS 模式的更多信息，请参阅将 RHEL 切换到 FIPS 模式。 当以 FIPS 模式运行 Red Hat Enterprise Linux (RHEL) 或 Red Hat Enterprise Linux CoreOS (RHCOS) 时，OpenShift Container Platform 核心组件使用 RHEL 加密库，只有在 x86_64, ppc64le, 和 s390x 架构上的库被提交到 NIST 进行 FIPS 140-2/140-3 Validation。 **重要** 如果使用 Azure File 存储，则无法启用 FIPS 模式。	
imageContentSources:	release-image 内容的源和存储库。	对象数组。包括一个 **source** 以及可选的 **mirrors**，如本表的以下行所述。

参数	描述	值
imageContentSources: source:	使用 **imageContentSources** 时需要此项。指定用户在镜像拉取规格中引用的存储库。	字符串
imageContentSources: mirrors:	指定可能还包含同一镜像的一个或多个存储库。	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。	**内部或外部 .**默认值为 **External**。 在非云平台上不支持将此字段设置为 **Internal**。  **重要** 如果字段的值设为 **Internal**，集群就会变得无法正常工作。如需更多信息，请参阅 BZ#1953035。
sshKey:	用于验证对集群机器的访问的 SSH 密钥。  **注意** 对于您要在其上执行安装调试或灾难恢复的生产环境 OpenShift Container Platform 集群，请指定 **ssh-agent** 进程使用的 SSH 密钥。	例如，**sshKey: ssh-ed25519 AAAA..**。

8.1.4. 基于代理的安装程序的其他裸机配置参数

下表描述了基于代理的安装程序的其他裸机安装配置参数：



注意

这些字段不会在集群的初始置备过程中使用，但它们可在安装集群后使用。在安装时配置这些字段，无需将它们设置为第 2 天操作。

表 8.4. 其他裸机参数

参数	描述	值
platform: baremetal: clusterProvisioningIP:	运行置备服务的集群中的 IP 地址。默认为 provisioning 子网的第三个 IP 地址。例如，**172.22.0.3** 或 **2620:52:0:1307::3**。	IPv4 或 IPv6 地址。
platform: baremetal: provisioningNetwork:	**provisioningNetwork** 配置设置决定了集群是否使用 provisioning 网络。如果存在，配置设置也会决定集群是否管理网络。 **Managed**: 默认。将此参数设置为 **Managed** 以完全管理置备网络，包括 DHCP、TFTP 等等。 **disabled** ：将此参数设置为 **Disabled**，以禁用对 provisioning 网络的要求。当设置为 **Disabled** 时，您只能根据第 2 天使用基于虚拟介质的置备。如果 **Disabled** 并使用电源管理，则必须可以从 bare-metal 网络访问 BMC。如果为 Disabled，则必须在裸机网络上提供两个用于置备服务的 IP 地址。	**Managed** 或 **Disabled**。
platform: baremetal: provisioningMACAddress:	运行置备服务的集群中的 MAC 地址。	MAC 地址。
platform: baremetal: provisioningNetworkCIDR:	用于置备的网络的 CIDR。在 provisioning 网络中不使用默认地址范围时需要这个选项。	有效的 CIDR，如 **10.0.0.0/16**。
platform: baremetal: provisioningNetworkInterface:	连接到 provisioning 网络的节点上的网络接口名称。使用 **bootMACAddress** 配置设置来启用 Ironic 标识 NIC 的 IP 地址，而不是使用 **provisioningNetworkInterface** 配置设置来标识 NIC 的名称。	字符串.

参数	描述	值
platform: baremetal: provisioningDHCPRange:	定义 provisioning 网络上节点的 IP 范围，如 **172.22.0.10,172.22.0.254**。	IP 地址范围。
platform: baremetal: hosts:	配置裸机主机。	主机配置对象的数组。
platform: baremetal: hosts: name:	主机的名称。	字符串.
platform: baremetal: hosts: bootMACAddress:	用于置备主机的 NIC 的 MAC 地址。	MAC 地址。
platform: baremetal: hosts: bmc:	主机的配置，以连接到基板管理控制器 (BMC)。	BMC 配置对象字典。
platform: baremetal: hosts: bmc: username:	BMC 的用户名。	字符串.
platform: baremetal: hosts: bmc: password:	BMC 的密码。	字符串.

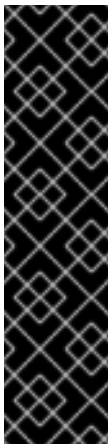
参数	描述	值
platform: baremetal: hosts: bmc: address:	与主机的 BMC 控制器通信的 URL。地址配置设置指定协议。例如，**redfish+http://10.10.10.1:8000/redfish/v1/Systems/1234** 启用 Redfish。如需更多信息，请参阅裸机上的"部署安装程序置备的集群中的"BMC 寻址"。	URL。
platform: baremetal: hosts: bmc: disableCertificateVerification:	**RedFish** 和 **redfish-virtualmedia** 需要这个参数来管理 BMC 地址。当 BMC 地址使用自签名证书时，这个值应该是 **True**。	布尔值.

8.1.5. 其他 VMware vSphere 配置参数

下表描述了其他 VMware vSphere 配置参数：

表 8.5. 其他 VMware vSphere 集群参数

参数	描述	值
platform: vsphere:	描述托管集群的云平台中的帐户。您可以使用参数来自定义平台。如果您为机器池中的 compute 和 control plane 机器提供额外的配置设置，则不需要该参数。	vSphere 配置对象的字典
platform: vsphere: failureDomains:	建立地区和区域之间的关系。您可以使用 vCenter 对象（如 **datastore** 对象）定义故障域。故障域定义 OpenShift Container Platform 集群节点的 vCenter 位置。	故障域配置对象的数组。
platform: vsphere: failureDomains: name:	故障域的名称。	字符串

参数	描述	值
platform: vsphere: failureDomains: region:	如果为集群定义多个故障域，则必须将标签附加到每个 vCenter 数据中心。要定义区域，请使用 **openshift-region** 标签类别中的标签。对于单个 vSphere 数据中心环境，您不需要附加标签，但必须为参数输入一个字母数字值，如 **datacenter**。如果您的故障域基于主机组，请将这些标签附加到 vSphere 集群而不是数据中心。  **重要** OpenShift zones 支持 vSphere 主机组只是一个技术预览功能。技术预览功能不受红帽产品服务等级协议（SLA）支持，且功能可能并不完整。红帽不推荐在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。 有关红帽技术预览功能支持范围的更多信息，请参阅技术预览功能支持范围。	字符串
platform: vsphere: failureDomains: regionType:	指定启用主机组的 **ComputeCluster** 区域类型。  **重要** OpenShift zones 支持 vSphere 主机组只是一个技术预览功能。技术预览功能不受红帽产品服务等级协议（SLA）支持，且功能可能并不完整。红帽不推荐在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。 有关红帽技术预览功能支持范围的更多信息，请参阅技术预览功能支持范围。	字符串
platform: vsphere: failureDomains: server:	指定 VMware vCenter 服务器的完全限定主机名或 IP 地址，以便客户端可以访问故障域资源。您必须将 **server** 器角色应用到 vSphere vCenter 服务器位置。	字符串

参数	描述	值
platform: vsphere: failureDomains: zone:	如果为集群定义多个故障域，则必须为每个 vCenter 集群附加标签。要定义一个区，请使用 **openshift-zone** 标签类别中的标签。对于单个 vSphere 数据中心环境，您不需要附加标签，但必须为参数输入一个字母数字值，如 **cluster**。如果要在主机组上基于故障域，请定义与您的主机组对应的区域，而不是集群。使用这些标签将每个 ESXi 主机与其主机组关联。 **重要** OpenShift zones 支持 vSphere 主机组只是一个技术预览功能。技术预览功能不受红帽产品服务等级协议（SLA）支持，且功能可能并不完整。红帽不推荐在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。 有关红帽技术预览功能支持范围的更多信息，请参阅技术预览功能支持范围。	字符串
platform: vsphere: failureDomains: zoneType:	指定要启用主机组的 **HostGroup** 区域类型。 **重要** OpenShift zones 支持 vSphere 主机组只是一个技术预览功能。技术预览功能不受红帽产品服务等级协议（SLA）支持，且功能可能并不完整。红帽不推荐在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。 有关红帽技术预览功能支持范围的更多信息，请参阅技术预览功能支持范围。	字符串

参数	描述	值
platform: vsphere: failureDomains: topology: computeCluster:	vSphere 计算集群的路径。	字符串
platform: vsphere: failureDomains: topology: datacenter:	列出并定义 OpenShift Container Platform 虚拟机 (VM) 操作的数据中心。数据中心列表必须与 **vcenters** 字段中指定的数据中心列表匹配。	字符串
platform: vsphere: failureDomains: topology: datastore:	保存虚拟机文件、模板和 ISO 镜像的 vSphere 数据存储路径。  **重要** 您可以指定数据存储集群中存在的任何数据存储路径。默认情况下，Storage vMotion 会自动为数据存储集群启用。红帽不支持 Storage vMotion，因此您必须禁用 Storage vMotion 以避免 OpenShift Container Platform 集群的数据丢失问题。 如果需要在多个数据存储间指定虚拟机，请使用 **数据存储** 对象在集群 **install-config.yaml** 配置文件中指定故障域。如需更多信息，请参阅"VMware vSphere 区域和区启用"。	字符串

参数	描述	值
platform: vsphere: failureDomains: topology: folder:	可选：用户创建虚拟机的现有文件夹的绝对路径，例如 **/<data_center_name>/vm/<folder_name>/<subfolder_name>**。	字符串
platform: vsphere: failureDomains: topology: hostGroup:	指定要与故障域关联的 vSphere 主机组。  **重要** OpenShift zones 支持 vSphere 主机组只是一个技术预览功能。技术预览功能不受红帽产品服务等级协议（SLA）支持，且功能可能并不完整。红帽不推荐在生产环境中使用它们。这些技术预览功能可以使用户提早试用新的功能，并有机会在开发阶段提供反馈意见。 有关红帽技术预览功能支持范围的更多信息，请参阅技术预览功能支持范围。	字符串
platform: vsphere: failureDomains: topology: networks:	列出 vCenter 实例中包含您配置的虚拟 IP 地址和 DNS 记录的任何网络。	字符串
platform: vsphere: failureDomains: topology: resourcePool:	可选：安装程序创建虚拟机的现有资源池的绝对路径，例如 **/<data_center_name>/host/<cluster_name>/Resources/<resource_pool_name>/<optional_nested_resource_pool_name>**。	字符串
platform: vsphere: failureDomains: topology template:	指定到预先存在的 Red Hat Enterprise Linux CoreOS (RHCOS)镜像模板或虚拟机的绝对路径。安装程序可以使用镜像模板或虚拟机在 vSphere 主机上快速安装 RHCOS。考虑使用此参数作为在 vSphere 主机上上传 RHCOS 镜像的替代选择。此参数仅适用于安装程序置备的基础架构。	字符串
platform: vsphere: vcenters:	配置连接详情，以便服务可以与 vCenter 服务器通信。	vCenter 配置对象的数组。

参数	描述	值
platform: vsphere: vcenters: datacenters:	列出并定义 OpenShift Container Platform 虚拟机 (VM) 操作的数据中心。数据中心列表必须与 **failureDomains** 字段中指定的数据中心列表匹配。	字符串
platform: vsphere: vcenters: password:	与 vSphere 用户关联的密码。	字符串
platform: vsphere: vcenters: port:	用于与 vCenter 服务器通信的端口号。	整数
platform: vsphere: vcenters: server:	vCenter 服务器的完全限定主机名(FQHN)或 IP 地址。	字符串
platform: vsphere: vcenters: user:	与 vSphere 用户关联的用户名。	字符串

8.1.6. 弃用的 VMware vSphere 配置参数

在 OpenShift Container Platform 4.13 中，以下 vSphere 配置参数已弃用。您可以继续使用这些参数，但安装程序不会在 **install-config.yaml** 文件中自动指定这些参数。

下表列出了每个已弃用的 vSphere 配置参数：

表 8.6. 弃用的 VMware vSphere 集群参数

参数	描述	值
platform: vsphere: cluster:	安装 OpenShift Container Platform 集群的 vCenter 集群。	字符串

参数	描述	值

platform:
  vsphere:
  datacenter:
	定义 OpenShift Container Platform 虚拟机 (VM) 操作的数据中心。	字符串

platform: vsphere: defaultDatastore:	用于调配卷的默认数据存储名称。	字符串

platform:
  vsphere:
  folder:
	可选：安装程序创建虚拟机的现有文件夹的绝对路径。如果没有提供这个值，安装程序会创建一个文件夹，它的名称为 datacenter 虚拟机文件夹中的基础架构 ID。	字符串，如 **/<data_center_name>/vm/<folder_name>/<subfolder_name>**。

platform:
  vsphere:
  password:
	vCenter 用户名的密码。	字符串

platform:
  vsphere:
  resourcePool:
	可选：安装程序创建虚拟机的现有资源池的绝对路径。如果没有指定值，安装程序会在 **/<data_center_name>/host/<cluster_name>/Resources** 下的集群的 root 中安装资源。	字符串，例如 **/<data_center_name>/host/<cluster_name>/Resources/<resource_pool_name>/<optional_nested_resource_pool_name>**。

platform:
  vsphere:
  username:
	用于连接 vCenter 实例的用户名。此用户必须至少具有 vSphere 中 静态或动态持久性卷置备 所需的角色和权限。	字符串

platform:
  vsphere:
  vCenter:
	vCenter 服务器的完全限定主机名或 IP 地址。	字符串

其他资源

- BMC 地址
- 为 VMware vCenter 配置区域和区域
- 所需的 vCenter 帐户权限

8.2. 可用的代理配置配置参数

下表指定您可以在基于 Agent 的安装过程中设置的必要和可选代理配置参数。

这些值在 **agent-config.yaml** 文件中指定。

注意

这些设置仅用于安装，无法在安装后修改。

8.2.1. 所需的配置参数

下表描述了所需的代理配置参数：

表 8.7. 所需的参数

参数	描述	值
apiVersion:	**agent-config.yaml** 内容的 API 版本。当前版本是 **v1beta1**。安装程序可能还支持旧的 API 版本。	字符串
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。在 **agent-config.yaml** 文件中输入的值将被忽略，而是使用 **install-config.yaml** 文件中指定的值。当您没有通过 **install-config.yaml** 或 **agent-config.yaml** 文件提供 **metadata.name** 时，例如只使用 ZTP 清单时，集群名称被设置为 **agent-cluster**。	小写字母和连字符 (**-**) 的字符串，如 **dev**。

8.2.2. 可选的配置参数

下表描述了可选的 Agent 配置参数：

表 8.8. 可选参数

参数	描述	值

参数	描述	值
rendezvousIP:	执行 bootstrap 进程的节点的 IP 地址，以及运行 **assisted-service** 组件。当您没有在 **networkConfig** 参数中指定至少一个主机的 IP 地址时，您必须提供 rendezvous IP 地址。如果没有提供此地址，则会从提供的主机的 **networkConfig** 中选择一个 IP 地址。	IPv4 或 IPv6 地址。
bootArtifactsBaseURL:	当您使用基于代理的安装程序生成最小 ISO 镜像时，此参数指定可在集群安装过程中从中检索 rootfs 镜像文件的 URL。对于连接环境中的引导最小 ISO 镜像，这个参数是可选的。 当您使用基于代理的安装程序生成 iPXE 脚本时，此参数指定要将 Preboot Execution Environment (PXE)资产上传到的服务器 URL。如需更多信息，请参阅"为 OpenShift Container Platform 准备 PXE 资产"。	字符串.
additionalNTPSources:	要添加到所有集群主机的网络时间协议 (NTP) 源列表，这些源被添加到通过其他方法配置的任何 NTP 源中。	主机名或 IP 地址列表。
hosts:	主机配置。可选的主机列表。定义的主机数量不能超过 **install-config.yaml** 文件中定义的主机总数，这是 **compute.replicas** 和 **controlPlane.replicas** 参数的值的总和。	主机配置对象的数组。
hosts: hostname:	主机名.覆盖从动态主机配置协议(DHCP)或反向 DNS 查找中获取的主机名。每个主机必须具有由这些方法提供的唯一主机名，尽管通过此参数配置主机名是可选的。	字符串.
hosts: interfaces:	为主机上的接口提供名称和 MAC 地址映射表。如果在 **agent-config.yaml** 文件中提供了 **NetworkConfig** 部分，则必须包含此表，值必须与 **NetworkConfig** 部分中提供的映射匹配。	主机配置对象的数组。
hosts: interfaces: name:	主机上接口名称。	字符串.

参数	描述	值
hosts: interfaces: macAddress:	主机上接口的 MAC 地址。	一个 MAC 地址，如以下示例：**00-B0-D0-63-C2-26**。
hosts: role:	定义主机是 **master** 节点还是 **worker** 节点。如果在 **agent-config.yaml** 文件中没有定义角色，则会在集群安装过程中随机分配角色。	**master** 或 **worker**。
hosts: rootDeviceHints:	启用将 Red Hat Enterprise Linux CoreOS (RHCOS)镜像置备到特定设备。安装程序会按照发现设备的顺序检查设备，并将发现的值与 hint 值进行比较。它使用第一个与 hint 值匹配的发现设备。这是操作系统在安装过程中写入的设备。	键值对字典。如需更多信息，请参阅"为 OpenShift 安装设置环境"页面中的"Root 设备提示"。
hosts: rootDeviceHints: deviceName:	RHCOS 镜像置备为的设备的名称。	字符串.
hosts: networkConfig:	主机网络定义。配置必须与 nmstate 文档中定义的 Host Network Management API 匹配。	主机网络配置对象的字典。
minimalISO:	定义基于代理的安装程序是否生成完整的 ISO 镜像或最小 ISO 镜像。当此参数设为 **True** 时，基于代理的安装程序会生成一个没有 rootfs 镜像文件的 ISO，并包含有关从中拉取 rootfs 文件的详情。 当您生成最小 ISO 时，如果您没有通过 **bootArtifactsBaseURL** 参数指定 rootfs URL，则基于代理的安装程序会嵌入一个默认 URL，该 URL 可通过互联网连接的环境中访问。 默认值为 **False**。	布尔值.

其他资源

- 为 OpenShift Container Platform 准备 PXE 资产

- Root 设备提示