OpenShift Container Platform 4.8

OpenShift 的 Windows 容器支持

Red Hat OpenShift for Windows Containers 指南

Last Updated: 2023-04-19

OpenShift Container Platform 4.8 OpenShift 的 Windows 容器支持

Red Hat OpenShift for Windows Containers 指南

法律通告

摘要

Red Hat OpenShift for Windows Containers 为在 OpenShift Container Platform 上运行 Microsoft Windows Server 容器提供了内置的支持。本指南提供所有详细信息。

目录

第 1 章 RED HAT OPENSHIFT 对 WINDOWS CONTAINERS 的支持概述

Red Hat OpenShift for Windows Containers 提供了在 OpenShift Container Platform 集群中运行 Windows 计算节点的功能。这可以通过使用 Red Hat Windows Machine Config Operator（WMCO）来安装和管理 Windows 节点来实现。通过红帽订阅，您可以获得对在 OpenShift Container Platform 中运行 Windows 工作负载的支持。如需更多信息，请参阅 发行注记。

对于 Linux 和 Windows 等工作负载，OpenShift Container Platform 允许您部署在 Windows Server 容器上运行的 Windows 工作负载，同时还提供托管在 Red Hat Enterprise Linux CoreOS(RHCOS)或 Red Hat Enterprise Linux(RHEL)上的传统 Linux 工作负载。如需更多信息，请参阅开始使用 Windows 容器工作负载。

您需要 WMCO 在集群中运行 Windows 工作负载。WMCO 在集群中管理部署和管理 Windows 工作负载的过程。如需更多信息，请参阅如何启用 Windows 容器工作负载。

您可以创建一个 Windows **MachineSet** 对象来创建基础架构 Windows 机器集和相关的机器，以便将支持的 Windows 工作负载移到新的 Windows 机器。您可以在多个平台上创建 Windows **MachineSet** 对象。

您可以将 Windows 工作负载调度到 Windows 计算节点。

您可以执行 Windows Machine Config Operator 升级 以确保 Windows 节点有最新的更新。

您可以通过删除特定机器来删除 Windows 节点。

您可以使用 Bring-Your-Own-Host(BYOH)Windows 实例 重新使用 Windows Server 虚拟机并将其引入 OpenShift Container Platform。BYOH Windows 实例让希望降低 Windows 服务器离线时发生重大中断的用户受益。您可以使用 BYOH Windows 实例作为 OpenShift Container Platform 4.8 及更新的版本的节点。

您可以通过执行以下操作 禁用 Windows 容器工作负载 ：

- 卸载 Windows Machine Config Operator
- 删除 Windows Machine Config Operator 命名空间

第 2 章 WINDOWS CONTAINER SUPPORT FOR RED HAT OPENSHIFT 发行注记

2.1. 关于 WINDOWS CONTAINER SUPPORT FOR RED HAT OPENSHIFT

Red Hat OpenShift support for Windows Containers 允许在 OpenShift Container Platform 集群中运行 Windows 计算节点。通过使用 Red Hat Windows Machine Config Operator (WMCO) 来安装和管理 Windows 节点，可以运行 Windows 工作负载。在 Windows 节点可用后，您可以在 OpenShift Container Platform 中运行 Windows 容器工作负载。

Red Hat OpenShift for Windows Containers 发行注记包括了 WMCO 的开发信息，WMCO 提供了在 OpenShift Container Platform 中运行 Windows 容器工作负载的功能。

2.2. 获取支持

Red Hat OpenShift 对 Windows Containers 提供，它是一个可选的、可安装的组件。Windows Container Support for Red Hat OpenShift 不是 OpenShift Container Platform 订阅的一部分。它需要额外的红帽订阅，并根据覆盖范围和服务等级协议进行支持。

您必须具有此单独的订阅才能获得对 Red Hat OpenShift 的 Windows Container 支持。如果没有此附加红帽订阅，则不支持在生产环境中部署 Windows 容器工作负载。您可以通过 红帽客户门户网站 请求支持。

如需更多信息，请参阅 Red Hat OpenShift Container Platform 生命周期政策文档中关于 Red Hat OpenShift support for Windows Containers 的部分。

如果您没有额外的红帽订阅，可以使用 Community Windows Machine Config Operator，它是一个不被官方支持的版本。

2.3. RED HAT WINDOWS MACHINE CONFIG OPERATOR 3.1.2 发行注记

发布日期： 2022 年 11 月 9 日

WMCO 3.1.2 现已正式发布，带有相关的程序漏洞修复。WMCO 组件在 RHBA-2022:7076 中发布。

2.3.1. 程序错误修复

- 在以前的版本中，如果在集群安装过程中指定了非默认集群 DNS，Windows 节点上的 Windows 容器会被分配错误的 DNS 服务器 IP。因此，DNS 解析无法正常工作。在这个版本中，删除了硬编码的集群 DNS 信息，并使用命令行参数来传递值。(**BZ#2030943**)

2.4. RED HAT WINDOWS MACHINE CONFIG OPERATOR 3.1.1 发行注记

发布日期：2021 年 12 月 7 日

WMCO 3.1.1 现已正式发布，带有相关的程序漏洞修复。WMCO 组件在 RHBA-2021:4710 中发布。

2.4.1. 程序错误修复

- 在以前的版本中，**windows-exporter** 指标端点对象包含对已删除机器的引用。这个不正确的引用会导致 WMCO 忽略具有无效 IP 地址的机器被删除的事件。在这个版本中，从事件过滤中删除了机器对象的验证，允许 **windows-exporter** 指标端点对象在机器仍处于 **Deleting** 阶段时正确更新。(**BZ#2008994**)

- 在以前的版本中，在 Operator 重启后删除与 Windows **Machine** 对象关联的节点会出现一个协调错误。在这个版本中，当集群中找不到为 **Running** 状态的 Windows 机器引用的节点时，不会响应或协调，从而导致 Linux 机器对象出现任何错误循环和标准化功能。(BZ#2009475)

- 在以前的版本中，当虚拟机使用 DNS 对象指定时，WMCO 没有将 BYOH Windows 虚拟机与其 **Node** 对象正确关联。这会导致 WMCO 尝试配置已全面配置的虚拟机。现在，WMCO 在查找关联的节点时可以正确地解析由 DNS 地址指定的虚拟机。(BZ#2020650)

- 在以前的版本中，加密的用户名使用额外的标签生成，这会导致它们无法正确显示。在这个版本中，删除了额外的标签，允许加密的用户名正确显示。(BZ#2023417)

- 在以前的版本中，PowerShell 在 Windows 虚拟机中运行的某些命令不会被 PowerShell 正确解析。这会导致带有 PowerShell 的 Windows VM 作为其默认 SSH shell 无法作为节点加入集群。WMCO 现在可识别 Windows 虚拟机的默认 SSH shell，并相应地运行关联的命令。这一新功能允许将 PowerShell 用作默认 SSH shell 的 Windows 虚拟机配置为集群中的节点。(BZ#2025730)

2.5. RED HAT WINDOWS MACHINE CONFIG OPERATOR 3.1.0 发行注记

发布日期：2021 年 9 月 21 日

WMCO 3.1.0 现在提供了程序错误修正和新功能。WMCO 组件在 RHBA-2021:3215 中发布。

2.5.1. 新功能

2.5.1.1. 使用 Bring-Your-Own-Host(BYOH)Windows 实例

现在，您可以将现有 Windows 实例作为计算节点添加到 OpenShift Container Platform 集群中。这需要在 WMCO 命名空间中创建配置映射。

以下平台的安装程序置备的基础架构支持 BYOH Windows 实例：

- Amazon Web Services (AWS)
- Microsoft Azure
- VMware vSphere

用户置备的基础架构支持 BYOH Windows 实例，只有在 **install-config.yaml** 文件中设置了 **platform: none** 字段时才支持以下平台：

- VMware vSphere
- 裸机

有关如何配置 BYOH Windows 实例的更多信息，请参阅配置 BYOH Windows 实例。

2.5.2. 程序错误修复

- 对于 VMware vSphere 上安装的集群，WMCO 会忽略 **Deleting** 阶段通知事件，在 **windows-exporter** 指标端点中保留不正确的节点信息。这会导致 Prometheus metrics 端点的映射无效。这个程序错误已被解决。WMCO 现在可识别 **Deleting** 阶段通知事件，并相应地映射 Prometheus metrics 端点。(BZ#1995341)

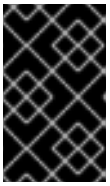
2.5.3. 已知问题

- 当使用配置映射中的 DNS 名称条目安装 BYOH Windows 实例时，WMCO 会在将其标记为 **Ready** 节点前配置实例两次。以后的 WMCO 发行版本中会解决这个问题。(BZ#2005360)

- 当在基于 Linux 的 Pod 的安全上下文中设置 **RunAsUser** 权限时，投射文件具有正确的权限集，包括容器用户所有权。但是，如果在 Windows pod 中设置 Windows 等同的 **RunAsUsername** 权限，kubelet 将无法为投射卷中的文件设置正确的所有权。如果与一个 hostPath 卷 一起使用时，不遵循最佳实践，则此问题可能更加严重。例如，为 pod 授予对 **C:\var\lib\kubelet\pods** 文件夹的访问权限，会导致该 pod 能够从其他 pod 访问服务帐户令牌。
  默认情况下，投射文件具有以下所有权，如本例 Windows projected 卷文件所示：

```
Path   :
Microsoft.PowerShell.Core\FileSystem::C:\var\run\secrets\kubernetes.io\serviceaccount\..2021_
08_31_22_22_18.318230061\ca.crt
Owner  : BUILTIN\Administrators
Group  : NT AUTHORITY\SYSTEM
Access : NT AUTHORITY\SYSTEM Allow  FullControl
         BUILTIN\Administrators Allow  FullControl
         BUILTIN\Users Allow  ReadAndExecute, Synchronize
Audit  :
Sddl   : O:BAG:SYD:AI(A;ID;FA;;;SY)(A;ID;FA;;;BA)(A;ID;0x1200a9;;;BU)
```

  这表示所有管理员用户，如具有 **ContainerAdministrator** 角色的用户，都具有读取、写入和执行访问权限，非管理员用户具有读取和执行访问权限。



重要

OpenShift Container Platform 将 **RunAsUser** 安全上下文应用于所有 pod，而不考虑其操作系统。这意味着 Windows Pod 会自动将 **RunAsUser** 权限应用到其安全上下文。

另外，如果使用设置了默认 **RunAsUser** 权限的投射卷创建了 Windows pod，则 pod 会停留在 **ContainerCreating** 阶段。

为了解决这些问题，OpenShift Container Platform 会在 Pod 安全上下文中设置的服务帐户卷中强制文件权限处理，对于 Windows 上的投射卷，不会满足它们。请注意，Windows pod 的这个行为是 OpenShift Container Platform 4.7 之前用于处理所有 pod 类型的文件权限处理方式。(BZ#1971745)

2.6. RED HAT WINDOWS MACHINE CONFIG OPERATOR 3.0.0 发行注记

此 WMCO 发行版本为在 OpenShift Container Platform 集群中运行的 Windows 计算节点提供程序错误修正和增强。WMCO 3.0.0 组件在 RHSA-2021:3001 中发布。

重要

运行 WMCO 2.0.2 及以下的用户必须卸载 Operator 并重新安装 WMCO 3.0.0，以便解决一个已知问题，防止用户从 WMCO 2.0.2 升级到 WMCO 3.0.0。(BZ#1983153)

2.7. WINDOWS MACHINE CONFIG OPERATOR 的先决条件

以下信息详细介绍了 Windows Machine Config Operator 支持的平台版本、Windows Server 版本和网络配置。有关仅与该平台相关的任何信息请参阅 vSphere 文档。

2.7.1. 基于 OpenShift Container Platform 和 WMCO 版本支持的平台

平台	支持的 OpenShift Container Platform 版本	支持的 WMCO 版本	安装程序置备的基础架构安装支持	用户置备的基础架构安装支持
Amazon Web Services (AWS)	4.6+	WMCO 1.0+	GA	技术预览
Microsoft Azure	4.6+	WMCO 1.0+	GA	技术预览
VMware vSphere	4.7+	WMCO 2.0+	GA	技术预览

2.7.2. 基于 OpenShift Container Platform 和 WMCO 版本的 Bring-Your-Own-Host (BYOH) 实例支持的平台

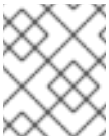
平台	支持的 OpenShift Container Platform 版本	支持的 WMCO 版本	用于安装程序置备的基础架构安装支持的 BYOH	BYOH 用于用户置备的基础架构安装支持
Amazon Web Services (AWS)	4.8+	WMCO 3.1+	GA	技术预览
Microsoft Azure	4.8+	WMCO 3.1+	GA	技术预览
VMware vSphere	4.8+	WMCO 3.1+	GA	GA[1]
裸机	4.8+	WMCO 3.1+		GA[1]

1. 只有在集群安装过程中在 **install-config.yaml** 文件中设置了 **platform: none** 字段时，才会支持此安装类型。

2.7.3. 支持的 Windows Server 版本

下表根据适用平台列出了受支持的 Windows Server 版本。任何未列出的 Windows Server 版本均不受支持，并将导致错误。为防止这些错误，请根据所使用的平台仅使用适当的版本。

平台	支持的 Windows Server 版本
Amazon Web Services (AWS)	Windows Server 2019，版本 1809
Microsoft Azure	Windows Server 2019，版本 1809

平台	支持的 Windows Server 版本
VMware vSphere	Windows Server 2022，OS Build 20348.681 或更高版本  **注意** 不支持 Windows Server 2019，因为没有包括 KB4565351 补丁。
裸机	Windows Server 2019，版本 1809

2.7.4. 支持的网络

与 OVN-Kubernetes 的混合网络是唯一支持的网络配置。有关此功能的更多信息，请参见下面的其他资源。下表概述了根据您的平台使用的网络配置和 Windows Server 版本类型。安装集群时必须指定网络配置。请注意，OpenShift SDN 网络是 OpenShift Container Platform 集群的默认网络。但是，OpenShift SDN 不支持 WMCO。

表 2.1. 平台网络支持

平台	支持的网络
Amazon Web Services (AWS)	使用 OVN-Kubernetes 的混合网络
Microsoft Azure	使用 OVN-Kubernetes 的混合网络
VMware vSphere	带有自定义 VXLAN 端口的 OVN-Kubernetes 的混合网络
裸机	使用 OVN-Kubernetes 的混合网络

表 2.2. 混合 OVN-Kubernetes Windows Server 支持

使用 OVN-Kubernetes 的混合网络	支持的 Windows Server 版本
默认 VXLAN 端口	Windows Server 2019，版本 1809
自定义 VXLAN 端口	Windows Server 2022，OS Build 20348.681 或更高版本

重要

在受限网络或断开连接的环境中的集群不支持运行 Windows 容器工作负载。

WMCO 版本 3.x 仅与 OpenShift Container Platform 4.8 兼容。

2.7.5. 新功能及改进

此发行版本添加了以下新功能及改进。

2.7.5.1. 对自定义 VXLAN 端口选择进行强制限制

在使用最新版本的 Windows 服务器时，用户不得选择自定义 VXLAN 端口。

2.7.6. 程序错误修复

- 在以前的版本中，当后备部署在不同 Windows 节点上调度多个 pod 时，负载均衡器服务会变得不稳定。这个问题已被解决。（BZ#1905950）
- 在以前的版本中，WMCO 将公钥注解 **windowsmachineconfig.openshift.io/pub-key-hash** 添加到 Linux 节点。现在，WMCO 不再为 Linux 节点添加注解。(BZ#1930791)
- 在以前的版本中，当用户提供无效的私钥时，WMCO 将失败。在这个版本中，WMCO 会为无效密钥的用户生成错误警报。（BZ#1929579）
- 在以前的版本中，当后备开发有多个 pod 调度到不同的 Windows 节点上时，kube-proxy 服务会在创建负载均衡器服务时崩溃。这个问题现已解决。（BZ#1939968）
- 在以前的版本中，**windows_exporter** 组件会报告各种指标作为 **windows_***。此错误导致一些节点级指标，它们通过遥测服务了解节点，无法报告。现在，组件会导出正确的显示所有预期指标。(BZ#1948037)

2.7.7. RHSA-2021:3001 - Windows Container 支持 OpenShift Container Platform 安全更新

作为之前记录的程序错误修复 (BZ#1946538) 的一部分，Red Hat Windows Machine Config Operator 2.0.1 现在提供了 Windows kube-proxy 的更新。有关更新的详情请查看 RHSA-2021:2130 公告。

2.8. 已知问题

- Web 控制台中提供的文件系统图不会显示 Windows 节点。这是因为文件系统查询的变化所致。以后的 WMCO 发行版本中会解决这个问题。(BZ#1930347)
- 当使用配置映射中的 DNS 名称条目安装 BYOH Windows 实例时，WMCO 会在将其标记为 **Ready** 节点前配置实例两次。以后的 WMCO 发行版本中会解决这个问题。(BZ#2005360)

2.9. 已知限制

在使用由 WMCO 管理的 Windows 节点（Windows 节点）时，请注意以下限制：

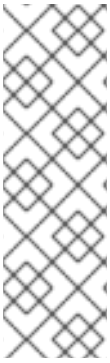
- Windows 节点上不支持以下 OpenShift Container Platform 功能：
  - Red Hat OpenShift Developer CLI (odo)
  - 镜像构建
  - OpenShift Pipelines
  - OpenShift Service Mesh
  - OpenShift 监控用户定义的项目

  - OpenShift Serverless
  - Pod 横向自动扩展
  - Pod 垂直自动扩展

- Windows 节点上不支持以下红帽功能：
  - 红帽成本管理
  - Red Hat OpenShift Local
- Windows 节点不支持从私有 registry 中拉取容器镜像。您可以使用公共 registry 中的镜像，或预拉取 (pull) 镜像。
- Windows 节点不支持使用部署配置创建的工作负载。您可以使用部署或其他方法部署工作负载。
- 使用集群范围代理的集群中不支持 Windows 节点。这是因为 WMCO 无法通过工作负载的代理连接路由流量。
- 在断开连接的环境中，不支持 Windows 节点。
- Red Hat OpenShift support for Windows Containers 不支持通过中继端口在集群中添加 Windows 节点。添加 Windows 节点唯一支持的网络配置是通过为 VLAN 传输流量的访问端口。
- Red Hat OpenShift support for Windows Containers 只支持所有云供应商的树内存储驱动程序。
- Kubernetes 有以下节点功能限制 ：
  - Windows 容器不支持巨页。
  - Windows 容器不支持特权容器。
  - Pod 终止宽限期要求在 Windows 节点上安装容器容器运行时。
- Kubernetes 已发现几个 API 兼容性问题。

第 3 章 了解 WINDOWS 容器工作负载

Red Hat OpenShift for Windows Containers 为在 OpenShift Container Platform 上运行 Microsoft Windows Server 容器提供了内置的支持。对于使用 Linux 和 Windows 工作负载管理异构环境的管理员，OpenShift Container Platform 允许您部署在 Windows Server 容器上运行的 Windows 工作负载，同时也提供托管在 Red Hat Enterprise Linux CoreOS（RHCOS）或 Red Hat Enterprise Linux（RHEL）上的传统 Linux 工作负载。



注意

不支持具有 Windows 节点的集群的多租户。托管多租户用法在所有 Kubernetes 环境中都存在安全性问题。额外的安全功能，如 Pod 安全策略，或节点的更精细的访问控制（RBAC），使利用安全漏洞进行工具更困难。但是，如果您选择运行托管多租户工作负载，则管理程序是唯一应使用的安全选项。Kubernetes 的安全域包括整个集群，而不是只限于单个节点。对于可能会有恶意的多租户工作负载，应该使用物理隔离的集群。

Windows 服务器容器使用共享内核提供资源隔离，但它并不适用于托管可能会有恶意的多租户工作负载。涉及主机多租户的情况应该使用 Hyper-V 隔离容器来严格隔离租户。

3.1. WINDOWS MACHINE CONFIG OPERATOR 的先决条件

以下信息详细介绍了 Windows Machine Config Operator 支持的平台版本、Windows Server 版本和网络配置。有关仅与该平台相关的任何信息请参阅 vSphere 文档。

3.1.1. 基于 OpenShift Container Platform 和 WMCO 版本支持的平台

平台	支持的 OpenShift Container Platform 版本	支持的 WMCO 版本	安装程序置备的基础架构安装支持	用户置备的基础架构安装支持
Amazon Web Services (AWS)	4.6+	WMCO 1.0+	GA	技术预览
Microsoft Azure	4.6+	WMCO 1.0+	GA	技术预览
VMware vSphere	4.7+	WMCO 2.0+	GA	技术预览

3.1.2. 基于 OpenShift Container Platform 和 WMCO 版本的 Bring-Your-Own-Host (BYOH) 实例支持的平台

平台	支持的 OpenShift Container Platform 版本	支持的 WMCO 版本	用于安装程序置备的基础架构安装支持的 BYOH	BYOH 用于用户置备的基础架构安装支持
Amazon Web Services (AWS)	4.8+	WMCO 3.1+	GA	技术预览
Microsoft Azure	4.8+	WMCO 3.1+	GA	技术预览

平台	支持的 OpenShift Container Platform 版本	支持的 WMCO 版本	用于安装程序置备的基础架构安装支持的 BYOH	BYOH 用于用户置备的基础架构安装支持
VMware vSphere	4.8+	WMCO 3.1+	GA	GA[1]
裸机	4.8+	WMCO 3.1+		GA[1]

1. 只有在集群安装过程中在 **install-config.yaml** 文件中设置了 **platform: none** 字段时，才会支持此安装类型。

3.1.3. 支持的 Windows Server 版本

下表根据适用平台列出了受支持的 Windows Server 版本。任何未列出的 Windows Server 版本均不受支持，并将导致错误。为防止这些错误，请根据所使用的平台仅使用适当的版本。

平台	支持的 Windows Server 版本
Amazon Web Services (AWS)	Windows Server 2019，版本 1809
Microsoft Azure	Windows Server 2019，版本 1809
VMware vSphere	Windows Server 2022，OS Build 20348.681 或更高版本 **注意** 不支持 Windows Server 2019，因为没有包括 KB4565351 补丁。
裸机	Windows Server 2019，版本 1809

3.1.4. 支持的网络

与 OVN-Kubernetes 的混合网络是唯一支持的网络配置。有关此功能的更多信息，请参见下面的其他资源。下表概述了根据您的平台使用的网络配置和 Windows Server 版本类型。安装集群时必须指定网络配置。请注意，OpenShift SDN 网络是 OpenShift Container Platform 集群的默认网络。但是，OpenShift SDN 不支持 WMCO。

表 3.1. 平台网络支持

平台	支持的网络
Amazon Web Services (AWS)	使用 OVN-Kubernetes 的混合网络
Microsoft Azure	使用 OVN-Kubernetes 的混合网络

平台	支持的网络
VMware vSphere	带有自定义 VXLAN 端口的 OVN-Kubernetes 的混合网络
裸机	使用 OVN-Kubernetes 的混合网络

表 3.2. 混合 OVN-Kubernetes Windows Server 支持

使用 OVN-Kubernetes 的混合网络	支持的 Windows Server 版本
默认 VXLAN 端口	Windows Server 2019，版本 1809
自定义 VXLAN 端口	Windows Server 2022，OS Build 20348.681 或更高版本

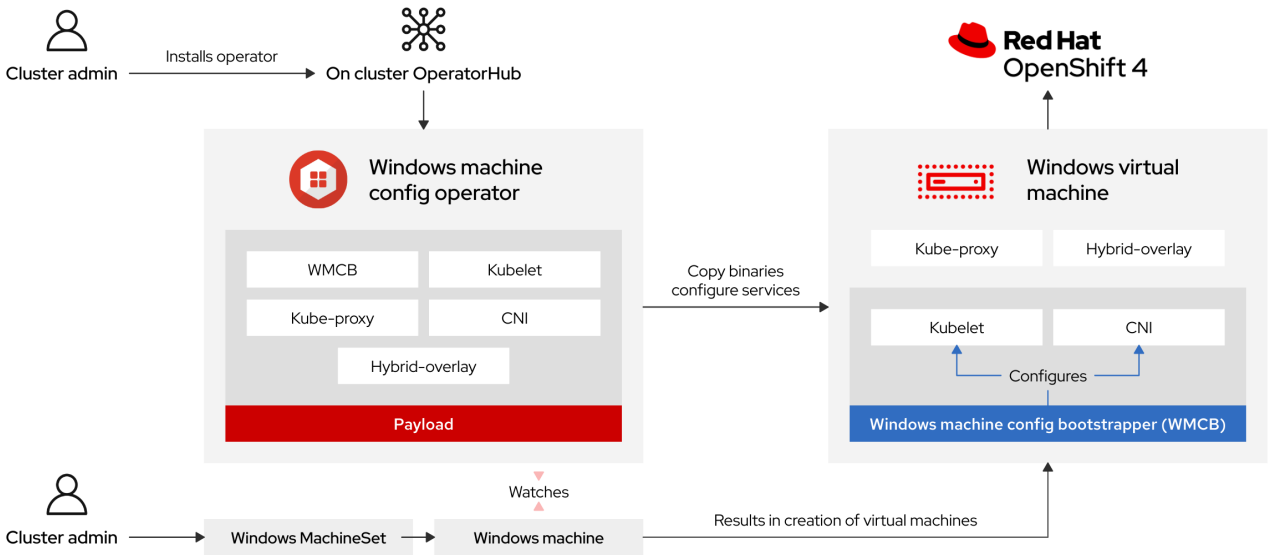
其他资源

- 请参阅使用 OVN-Kubernetes 配置混合网络

3.2. WINDOWS 工作负载管理

要在集群中运行 Windows 工作负载，必须首先安装 Windows Machine Config Operator（WMCO）。WMCO 是一个基于 Linux 的 Operator，它运行在基于 Linux 的 control plane 和计算节点上。WMCO 在集群中管理部署和管理 Windows 工作负载的过程。

图 3.1. WMCO 设计



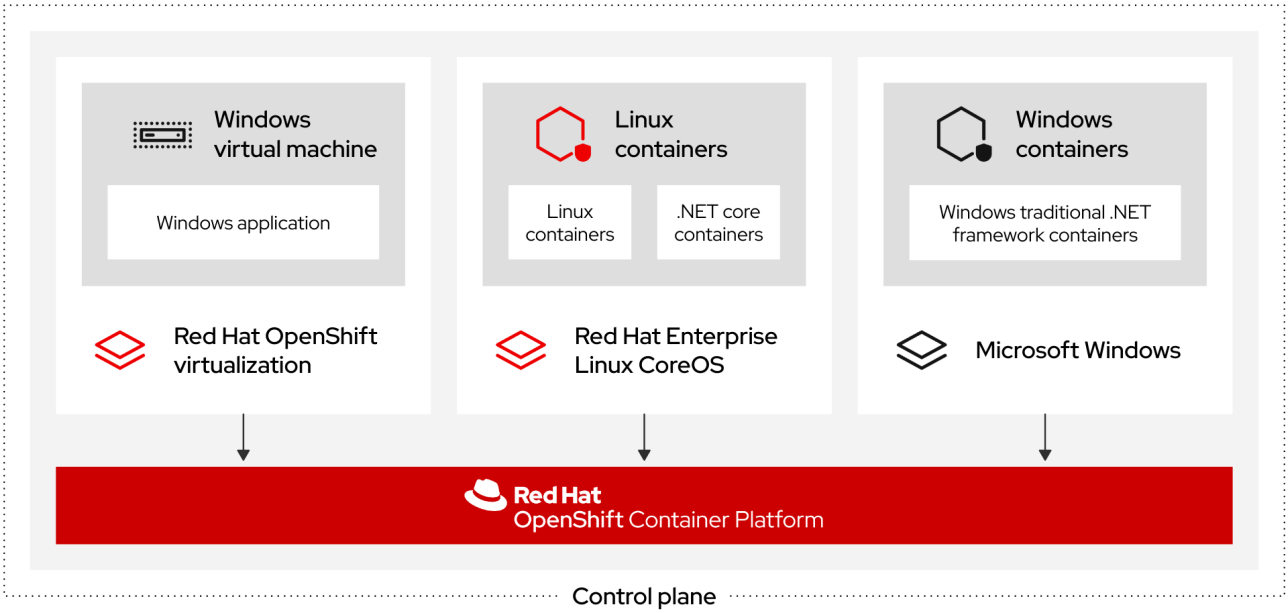
在部署 Windows 工作负载前，您必须创建一个 Windows 计算节点并加入集群。Windows 节点会在集群中托管 Windows 工作负载，并可与其他基于 Linux 的计算节点一起运行。您可以通过创建一个 Windows 机器集来创建一个 Windows 计算节点，以托管 Windows Server 计算机器。您必须对机器集应用特定于 Windows 的标签，该标签指定启用了 Docker 格式的容器运行时附加组件的 Windows OS 镜像。

重要

目前，在 Windows 节点中使用 Docker 格式的容器运行时。Kubernetes 将弃用 Docker 作为容器运行时，详情请参阅 Kubernetes 文档中的 Docker 弃用信息。在未来的 Kubernetes 发行版本中，Containerd 将是 Windows 节点新支持的容器运行时。

WMCO 监视有 Windows 标签的机器。检测到 Windows 机器集并置备相应机器后，WMCO 配置底层 Windows 虚拟机（VM），以便它可以将集群加入为计算节点。

图 3.2. 混合 Windows 和 Linux 工作负载



WMCO 在命名空间中需要一个预先确定的 secret，该 secret 包含一个用于与 Windows 实例交互的私钥。WMCO 在引导时检查此 secret，并创建一个用户数据 secret，您必须在您创建的 Windows **MachineSet** 对象中引用该 secret。然后，WMCO 使用与私钥对应的公钥填充用户数据 secret。通过这些数据，集群可以使用 SSH 连接到 Windows 虚拟机。

当集群与 Windows 虚拟机建立连接后，您可以使用类似管理 Linux 节点一样的方法管理 Windows 节点。

注意

OpenShift Container Platform Web 控制台为 Linux 节点可用的 Windows 节点提供大多数相同的监控功能。但是，目前无法监控 Windows 节点上运行的 pod 的工作负载图形。

将 Windows 工作负载调度到 Windows 节点可使用典型的 pod 调度实践，如污点、容限和节点选择器。或者，您也可以使用 **RuntimeClass** 对象来把 Windows 工作负载与 Linux 工作负载和其他 Windows 版本工作负载进行区分。

3.3. WINDOWS 节点服务

每个 Windows 节点均安装以下与 Windows 相关的服务：

Service	描述
kubelet	注册 Windows 节点并管理其状态。

Service	描述
Container Network Interface (CNI) 插件	为 Windows 节点公开网络。
Windows Machine Config Bootstrapper（WMCB）	配置 kubelet 和 CNI 插件。
Windows Exporter	从 Windows 节点导出 Prometheus 指标
hybrid-overlay	创建 OpenShift Container Platform 主机网络服务（HNS）。
kube-proxy	在节点上维护网络规则，以允许外部通信。

3.4. 已知限制

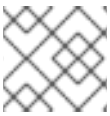
在使用由 WMCO 管理的 Windows 节点（Windows 节点）时，请注意以下限制：

- Windows 节点上不支持以下 OpenShift Container Platform 功能：
  - Red Hat OpenShift Developer CLI (odo)
  - 镜像构建
  - OpenShift Pipelines
  - OpenShift Service Mesh
  - OpenShift 监控用户定义的项目
  - OpenShift Serverless
  - Pod 横向自动扩展
  - Pod 垂直自动扩展
- Windows 节点上不支持以下红帽功能：
  - 红帽成本管理
  - Red Hat OpenShift Local
- Windows 节点不支持从私有 registry 中拉取容器镜像。您可以使用公共 registry 中的镜像，或预拉取 (pull) 镜像。
- Windows 节点不支持使用部署配置创建的工作负载。您可以使用部署或其他方法部署工作负载。
- 使用集群范围代理的集群中不支持 Windows 节点。这是因为 WMCO 无法通过工作负载的代理连接路由流量。
- 在断开连接的环境中，不支持 Windows 节点。

- Red Hat OpenShift support for Windows Containers 不支持通过中继端口在集群中添加 Windows 节点。添加 Windows 节点唯一支持的网络配置是通过为 VLAN 传输流量的访问端口。
- Red Hat OpenShift support for Windows Containers 只支持所有云供应商的树内存储驱动程序。
- Kubernetes 有以下节点功能限制 ：
  - Windows 容器不支持巨页。
  - Windows 容器不支持特权容器。
  - Pod 终止宽限期要求在 Windows 节点上安装容器容器运行时。
- Kubernetes 已发现几个 API 兼容性问题。

第 4 章 启用 WINDOWS 容器工作负载

在向集群中添加 Windows 工作负载前，您必须安装由 OpenShift Container Platform OperatorHub 提供的 Windows Machine Config Operator（WMCO）。WMCO 在集群中管理部署和管理 Windows 工作负载的过程。



注意

WMCO 管理的 Windows 实例不支持双 NIC。

先决条件

- 可以使用具有 **cluster-admin** 权限的账户访问 OpenShift Container Platform 集群。
- 已安装 OpenShift CLI（**oc**）。
- 已使用安装程序置备的基础架构安装集群，或使用用户置备的基础架构在 **install-config.yaml** 文件中设置了 **platform: none** 字段。
- 您已为集群配置了带有 OVN-Kubernetes 的混合网络。这必须在集群安装过程中完成。如需更多信息，请参阅配置混合网络。
- 运行 OpenShift Container Platform 集群版本 4.6.8 或更高版本。

注意

使用 集群范围代理 的集群中不支持 WMCO，因为 WMCO 无法通过工作负载的代理连接路由流量。

其他资源

- 有关 Windows Machine Config Operator 的完整先决条件，请参阅了解 Windows 容器工作负载。

4.1. 安装 WINDOWS MACHINE CONFIG OPERATOR

您可以使用 Web 控制台或 OpenShift CLI（**oc**）安装 Windows Machine Config Operator。

4.1.1. 使用 Web 控制台安装 Windows Machine Config Operator

您可以使用 OpenShift Container Platform Web 控制台安装 Windows Machine Config Operator（WMCO）。

注意

WMCO 管理的 Windows 实例不支持双 NIC。

流程

1. 从 OpenShift Container Platform Web 控制台中的 **Adminstrator** 视角进入 **Operators → OperatorHub** 页面。
2. 使用 **Filter by keyword** 复选框在目录中搜索 **Windows Machine Config Operator**。点 **Windows Machine Config Operator** 标题。

3. 查看 Operator 信息并点 **Install**。
4. 在 **Install Operator** 页面中：
   a. 选择 **stable** 频道作为 **更新频道**。**stable** 频道允许安装 WMCO 的最新稳定版本。
   b. **安装模式** 被预先配置，因为 WMCO 只能在单一命名空间中可用。
   c. 为 WMCO 选择 **Installed Namespace**。默认 Operator 建议命名空间为 **openshift-windows-machine-config-operator**。
   d. 点 **Enable Operator recommended cluster monitoring on the Namespace**为 WMCO 启用集群监控。
   e. 选择一个**批准策略**。
      - **Automatic** 策略允许 Operator Lifecycle Manager（OLM）在有新版本可用时自动更新 Operator。
      - **Manual** 策略需要拥有适当凭证的用户批准 Operator 更新。
1. 点 **Install**。WMCO 现在列在 **Installed Operators** 页中。

注意

WMCO 会自动安装到您定义的命名空间中，如 **openshift-windows-machine-config-operator**。

2. 验证 **Status** 显示为 **Succeeded** 以确认成功安装了 WMCO。

4.1.2. 使用 CLI 安装 Windows Machine Config Operator

您可以使用 OpenShift CLI（**oc**）安装 Windows Machine Config Operator（WMCO）。

注意

WMCO 管理的 Windows 实例不支持双 NIC。

流程

1. 为 WMCO 创建命名空间。
   a. 为 WMCO 创建一个 **Namespace** 对象 YAML 文件。例如，**wmco-namespace.yaml**：

```
apiVersion: v1
kind: Namespace
metadata:
  name: openshift-windows-machine-config-operator 1
  labels:
    openshift.io/cluster-monitoring: "true" 2
```

1. 建议您在 **openshift-windows-machine-config-operator** 命名空间中部署 WMCO。
2. 为 WMCO 启用集群监控需要此标签。

b. 创建命名空间：

```
$ oc create -f <file-name>.yaml
```

例如：

```
$ oc create -f wmco-namespace.yaml
```

2. 为 WMCO 创建 Operator 组。

a. 创建 **OperatorGroup** 对象 YAML 文件。例如，**wmco-og.yaml**：

```
apiVersion: operators.coreos.com/v1
kind: OperatorGroup
metadata:
  name: windows-machine-config-operator
  namespace: openshift-windows-machine-config-operator
spec:
  targetNamespaces:
  - openshift-windows-machine-config-operator
```

b. 创建 Operator 组：

```
$ oc create -f <file-name>.yaml
```

例如：

```
$ oc create -f wmco-og.yaml
```

3. 为 WMCO 订阅命名空间。

a. 创建 **Subscription** 对象 YAML 文件。例如，**wmco-sub.yaml**：

```
apiVersion: operators.coreos.com/v1alpha1
kind: Subscription
metadata:
  name: windows-machine-config-operator
  namespace: openshift-windows-machine-config-operator
spec:
  channel: "stable" (1)
  installPlanApproval: "Automatic" (2)
  name: "windows-machine-config-operator"
  source: "redhat-operators" (3)
  sourceNamespace: "openshift-marketplace" (4)
```

(1) 指定 **stable** 作为频道。

(2) 设置批准策略。可设置 **Automatic** 或 **Manual**。

(3) 指定包含 **windows-machine-config-operator** 软件包清单的 **redhat-operators** 目录源。如果 OpenShift Container Platform 集群安装在受限网络中（也称为断开连接的集群），请指定配置 Operator LifeCycle Manager (OLM) 时创建的 **CatalogSource** 对象的名称。

4 目录源的命名空间。将 **openshift-marketplace** 用于默认的 OperatorHub 目录源。

b. 创建订阅：

```
$ oc create -f <file-name>.yaml
```

例如：

```
$ oc create -f wmco-sub.yaml
```

WMCO 现已安装到 **openshift-windows-machine-config-operator** 中。

4. 验证 WMCO 安装：

```
$ oc get csv -n openshift-windows-machine-config-operator
```

输出示例

```
NAME                                 DISPLAY                          VERSION   REPLACES   PHASE
windows-machine-config-operator.2.0.0   Windows Machine Config Operator   2.0.0
Succeeded
```

4.2. 为 WINDOWS MACHINE CONFIG OPERATOR 配置 SECRET

要运行 Windows Machine Config Operator（WMCO），您必须在包含私钥的 WMCO 命名空间中创建一个 secret。这需要允许 WMCO 与 Windows 虚拟机（VM）进行通信。

先决条件

- 已使用 Operator Lifecycle Manager（OLM）安装 Windows Machine Config Operator（WMCO）。
- 您创建包含 RSA 密钥的 PEM 编码文件。

流程

- 定义访问 Windows 虚拟机所需的 secret:

```
$ oc create secret generic cloud-private-key --from-file=private-
key.pem=${HOME}/.ssh/<key> \
    -n openshift-windows-machine-config-operator 1
```

1 您必须在 WMCO 命名空间中创建私钥，如 **openshift-windows-machine-config-operator**。

建议您使用与安装集群时所用的私钥不同的私钥。

4.3. 其他资源

- 为集群节点 SSH 访问生成密钥对
- 将 Operator 添加到集群.

第 5 章 创建 WINDOWS MACHINESET 对象

5.1. 在 AWS 上创建 WINDOWS MACHINESET 对象

您可以在 Amazon Web Services（AWS）上的 OpenShift Container Platform 集群中创建 Windows **MachineSet** 对象来满足特定目的。例如，您可以创建基础架构 Windows 机器集和相关机器，以便将支持的 Windows 工作负载转移到新的 Windows 机器上。

先决条件

- 已使用 Operator Lifecycle Manager（OLM）安装 Windows Machine Config Operator（WMCO）。
- 您可以使用受支持的 Windows 服务器作为操作系统镜像，并启用了 Docker 格式的容器运行时附加组件。
  使用以下 **aws** 命令查询有效的 AMI 镜像：

```
$ aws ec2 describe-images --region <aws region name> --filters
"Name=name,Values=Windows_Server-2019*English*Full*Containers*" "Name=is-
public,Values=true" --query "reverse(sort_by(Images, &CreationDate))[*].{name: Name, id:
ImageId}" --output table
```

重要

目前，在 Windows 节点中使用 Docker 格式的容器运行时。Kubernetes 将弃用 Docker 作为容器运行时，详情请参阅 Kubernetes 文档中的 Docker 弃用信息。在未来的 Kubernetes 发行版本中，Containerd 将是 Windows 节点新支持的容器运行时。

5.1.1. Machine API 概述

Machine API 将基于上游 Cluster API 项目的主要资源与自定义 OpenShift Container Platform 资源相结合。

对于 OpenShift Container Platform 4.8 集群，Machine API 在集群安装完成后执行所有节点主机置备管理操作。由于此系统的缘故，OpenShift Container Platform 4.8 在公有或私有云基础架构之上提供了一种弹性动态置备方法。

两种主要资源分别是：

Machines

描述节点主机的基本单元。机器具有 **providerSpec** 规格，用于描述为不同云平台提供的计算节点的类型。例如，Amazon Web Services (AWS) 上的 worker 节点的机器类型可能会定义特定的机器类型和所需的元数据。

机器集

MachineSet 资源是机器组。机器集适用于机器，复制集则适用于 pod。如果需要更多机器或必须缩减规模，则可以更改机器集的 **replicas** 字段来满足您的计算需求。

警告

control plane 机器不能由机器集管理。

以下自定义资源可为集群添加更多功能：

机器自动扩展

MachineAutoscaler 资源自动扩展云中的机器。您可以为指定机器集中的节点设置最小和最大扩展界限，机器自动扩展就会维护此范围内的节点。**ClusterAutoscaler** 对象存在后，**MachineAutoscaler** 对象生效。**ClusterAutoscaler** 和 **MachineAutoscaler** 资源都由 **ClusterAutoscalerOperator** 对象提供。

集群自动扩展

此资源基于上游集群自动扩展项目。在 OpenShift Container Platform 实现中，它通过扩展机器集 API 来与 Machine API 集成。您可以为核心、节点、内存和 GPU 等资源设置集群范围的扩展限制。您可以设置优先级，使集群对 Pod 进行优先级排序，以便不针对不太重要的 Pod 使新节点上线。您还可以设置扩展策略，以便可以扩展节点，但不会缩减节点。

机器健康检查

MachineHealthCheck 资源可检测机器何时处于不健康状态并将其删除，然后在支持的平台上生成新的机器。

在 OpenShift Container Platform 版本 3.11 中，您无法轻松地推出多区架构，因为集群不负责管理机器置备。自 OpenShift Container Platform 版本 4.1 起，此过程变得更加容易。每个机器集限定在一个区域，因此安装程序可以代表您将机器集分发到多个可用区。然后，由于您的计算是动态的，因此在面对区域故障时，您始终都有一个区域来应对必须重新平衡机器的情况。自动扩展器在集群生命周期内尽可能提供平衡。

5.1.2. AWS 上 Windows MachineSet 对象的 YAML 示例

此 YAML 示例定义了一个在 Amazon Web Services（AWS）上运行的 Windows **MachineSet** 对象，它可响应 Windows Machine Config Operator（WMCO）。

```
apiVersion: machine.openshift.io/v1beta1
kind: MachineSet
metadata:
  labels:
    machine.openshift.io/cluster-api-cluster: <infrastructure_id> 1
  name: <infrastructure_id>-windows-worker-<zone> 2
  namespace: openshift-machine-api
spec:
  replicas: 1
  selector:
    matchLabels:
      machine.openshift.io/cluster-api-cluster: <infrastructure_id> 3
      machine.openshift.io/cluster-api-machineset: <infrastructure_id>-windows-worker-<zone> 4
  template:
    metadata:
      labels:
```

```
    machine.openshift.io/cluster-api-cluster: <infrastructure_id> 5
    machine.openshift.io/cluster-api-machine-role: worker
    machine.openshift.io/cluster-api-machine-type: worker
    machine.openshift.io/cluster-api-machineset: <infrastructure_id>-windows-worker-<zone> 6
    machine.openshift.io/os-id: Windows 7
spec:
  metadata:
    labels:
      node-role.kubernetes.io/worker: "" 8
  providerSpec:
    value:
      ami:
        id: <windows_container_ami> 9
      apiVersion: awsproviderconfig.openshift.io/v1beta1
      blockDevices:
        - ebs:
            iops: 0
            volumeSize: 120
            volumeType: gp2
      credentialsSecret:
        name: aws-cloud-credentials
      deviceIndex: 0
      iamInstanceProfile:
        id: <infrastructure_id>-worker-profile 10
      instanceType: m5a.large
      kind: AWSMachineProviderConfig
      placement:
        availabilityZone: <zone> 11
        region: <region> 12
      securityGroups:
        - filters:
            - name: tag:Name
              values:
                - <infrastructure_id>-worker-sg 13
      subnet:
        filters:
          - name: tag:Name
            values:
              - <infrastructure_id>-private-<zone> 14
      tags:
        - name: kubernetes.io/cluster/<infrastructure_id> 15
          value: owned
      userDataSecret:
        name: windows-user-data 16
        namespace: openshift-machine-api
```

1 3 5 10 13 14 15 指定基于置备集群时所设置的集群 ID 的基础架构 ID。您可以运行以下命令来获取基础架构 ID:

```
$ oc get -o jsonpath='{.status.infrastructureName}{"\n"}' infrastructure cluster
```

2 4 6 指定基础架构 ID、worker 标签和区。

7 将机器配置为 Windows 机器。

8 将 Windows 节点配置为计算机器。

9 指定安装的容器运行时的 Windows 镜像的 AMI ID。您必须使用 Windows Server 2019。

11 指定 AWS 区域，如 **us-east-1a**。

12 指定 AWS 区域，如 **us-east-1**。

16 由 WMCO 配置第一个 Windows 机器时创建的。之后，所有后续机器组都可以使用 **windows-user-data**。

5.1.3. 创建机器集

除了安装程序创建的机器集之外，还可创建自己的机器集来动态管理您选择的特定工作负载的机器计算资源。

先决条件

- 部署一个 OpenShift Container Platform 集群。
- 安装 OpenShift CLI（**oc**）。
- 以具有 **cluster-admin** 权限的用户身份登录 **oc**。

流程

1. 创建一个包含机器集自定义资源（CR）示例的新 YAML 文件，并将其命名为 **<file_name>.yaml**。
   确保设置 **<clusterID>** 和 **<role>** 参数值。

   a. 如果您不确定要为特定字段设置哪个值，您可以从集群中检查现有机器集：

   ```
   $ oc get machinesets -n openshift-machine-api
   ```

   输出示例

   ```
   NAME                                DESIRED   CURRENT   READY   AVAILABLE   AGE
   agl030519-vplxk-worker-us-east-1a   1         1         1       1           55m
   agl030519-vplxk-worker-us-east-1b   1         1         1       1           55m
   agl030519-vplxk-worker-us-east-1c   1         1         1       1           55m
   agl030519-vplxk-worker-us-east-1d   0         0                             55m
   agl030519-vplxk-worker-us-east-1e   0         0                             55m
   agl030519-vplxk-worker-us-east-1f   0         0                             55m
   ```

   b. 检查特定机器集的值：

   ```
   $ oc get machineset <machineset_name> -n \
       openshift-machine-api -o yaml
   ```

   输出示例

   ```
   ...
   template:
   ```

```
metadata:
  labels:
    machine.openshift.io/cluster-api-cluster: agl030519-vplxk (1)
    machine.openshift.io/cluster-api-machine-role: worker (2)
    machine.openshift.io/cluster-api-machine-type: worker
    machine.openshift.io/cluster-api-machineset: agl030519-vplxk-worker-us-east-1a
```

1. 集群 ID。
2. 默认节点标签。

2. 创建新的 **MachineSet** CR:

```
$ oc create -f <file_name>.yaml
```

3. 查看机器集列表：

```
$ oc get machineset -n openshift-machine-api
```

输出示例

```
NAME                                  DESIRED   CURRENT   READY   AVAILABLE   AGE
agl030519-vplxk-windows-worker-us-east-1a   1         1         1       1           11m
agl030519-vplxk-worker-us-east-1a           1         1         1       1           55m
agl030519-vplxk-worker-us-east-1b           1         1         1       1           55m
agl030519-vplxk-worker-us-east-1c           1         1         1       1           55m
agl030519-vplxk-worker-us-east-1d           0         0                             55m
agl030519-vplxk-worker-us-east-1e           0         0                             55m
agl030519-vplxk-worker-us-east-1f           0         0                             55m
```

当新机器集可用时，**DESIRED** 和 **CURRENT** 的值会匹配。如果机器集不可用，请等待几分钟，然后再次运行命令。

5.1.4. 其他资源

- 有关管理机器集的更多信息，请参阅 *机器管理* 部分。

5.2. 在 AZURE 上创建 WINDOWSMACHINESET 对象

您可以在 Microsoft Azure 上的 OpenShift Container Platform 集群中创建 Windows **MachineSet** 对象来满足特定目的。例如，您可以创建基础架构 Windows 机器集和相关机器，以便将支持的 Windows 工作负载转移到新的 Windows 机器上。

先决条件

- 已使用 Operator Lifecycle Manager（OLM）安装 Windows Machine Config Operator（WMCO）。
- 您可以使用受支持的 Windows 服务器作为操作系统镜像，并启用了 Docker 格式的容器运行时附加组件。

重要

目前，在 Windows 节点中使用 Docker 格式的容器运行时。Kubernetes 将弃用 Docker 作为容器运行时，详情请参阅 Kubernetes 文档中的 Docker 弃用信息。在未来的 Kubernetes 发行版本中，Containerd 将是 Windows 节点新支持的容器运行时。

5.2.1. Machine API 概述

Machine API 将基于上游 Cluster API 项目的主要资源与自定义 OpenShift Container Platform 资源相结合。

对于 OpenShift Container Platform 4.8 集群，Machine API 在集群安装完成后执行所有节点主机置备管理操作。由于此系统的缘故，OpenShift Container Platform 4.8 在公有或私有云基础架构之上提供了一种弹性动态置备方法。

两种主要资源分别是：

Machines

描述节点主机的基本单元。机器具有 **providerSpec** 规格，用于描述为不同云平台提供的计算节点的类型。例如，Amazon Web Services (AWS) 上的 worker 节点的机器类型可能会定义特定的机器类型和所需的元数据。

机器集

MachineSet 资源是机器组。机器集适用于机器，复制集则适用于 pod。如果需要更多机器或必须缩减规模，则可以更改机器集的 **replicas** 字段来满足您的计算需求。

警告

control plane 机器不能由机器集管理。

以下自定义资源可为集群添加更多功能：

机器自动扩展

MachineAutoscaler 资源自动扩展云中的机器。您可以为指定机器集中的节点设置最小和最大扩展界限，机器自动扩展就会维护此范围内的节点。**ClusterAutoscaler** 对象存在后，**MachineAutoscaler** 对象生效。**ClusterAutoscaler** 和 **MachineAutoscaler** 资源都由 **ClusterAutoscalerOperator** 对象提供。

集群自动扩展

此资源基于上游集群自动扩展项目。在 OpenShift Container Platform 实现中，它通过扩展机器集 API 来与 Machine API 集成。您可以为核心、节点、内存和 GPU 等资源设置集群范围的扩展限制。您可以设置优先级，使集群对 Pod 进行优先级排序，以便不针对不太重要的 Pod 使新节点上线。您还可以设置扩展策略，以便可以扩展节点，但不会缩减节点。

机器健康检查

MachineHealthCheck 资源可检测机器何时处于不健康状态并将其删除，然后在支持的平台上生成新的机器。

在 OpenShift Container Platform 版本 3.11 中，您无法轻松地推出多区架构，因为集群不负责管理机器置备。自 OpenShift Container Platform 版本 4.1 起，此过程变得更加容易。每个机器集限定在一个区域，

因此安装程序可以代表您将机器集分发到多个可用区。然后，由于您的计算是动态的，因此在面对区域故障时，您始终都有一个区域来应对必须重新平衡机器的情况。自动扩展器在集群生命周期内尽可能提供平衡。

5.2.2. Azure 上 Windows MachineSet 对象的 YAML 示例

此 YAML 示例定义了一个在 Microsoft Azure 上运行的 Windows **MachineSet** 对象，Windows Machine Config Operator（WMCO）可以响应。

```
apiVersion: machine.openshift.io/v1beta1
kind: MachineSet
metadata:
  labels:
    machine.openshift.io/cluster-api-cluster: <infrastructure_id> 1
  name: <windows_machine_set_name> 2
  namespace: openshift-machine-api
spec:
  replicas: 1
  selector:
    matchLabels:
      machine.openshift.io/cluster-api-cluster: <infrastructure_id> 3
      machine.openshift.io/cluster-api-machineset: <windows_machine_set_name> 4
  template:
    metadata:
      labels:
        machine.openshift.io/cluster-api-cluster: <infrastructure_id> 5
        machine.openshift.io/cluster-api-machine-role: worker
        machine.openshift.io/cluster-api-machine-type: worker
        machine.openshift.io/cluster-api-machineset: <windows_machine_set_name> 6
        machine.openshift.io/os-id: Windows 7
    spec:
      metadata:
        labels:
          node-role.kubernetes.io/worker: "" 8
      providerSpec:
        value:
          apiVersion: azureproviderconfig.openshift.io/v1beta1
          credentialsSecret:
            name: azure-cloud-credentials
            namespace: openshift-machine-api
          image: 9
            offer: WindowsServer
            publisher: MicrosoftWindowsServer
            resourceID: ""
            sku: 2019-Datacenter-with-Containers
            version: latest
          kind: AzureMachineProviderSpec
          location: <location> 10
          managedIdentity: <infrastructure_id>-identity 11
          networkResourceGroup: <infrastructure_id>-rg 12
          osDisk:
            diskSizeGB: 128
            managedDisk:
              storageAccountType: Premium_LRS
```

```
    osType: Windows
  publicIP: false
  resourceGroup: <infrastructure_id>-rg (13)
  subnet: <infrastructure_id>-worker-subnet
  userDataSecret:
    name: windows-user-data (14)
    namespace: openshift-machine-api
  vmSize: Standard_D2s_v3
  vnet: <infrastructure_id>-vnet (15)
  zone: "<zone>" (16)
```

(1) (3) (5) (11) (12) (13) (15) 指定基于置备集群时所设置的集群 ID 的基础架构 ID。您可以运行以下命令来获取基础架构 ID:

```
$ oc get -o jsonpath='{.status.infrastructureName}{"\n"}' infrastructure cluster
```

(2) (4) (6) 指定 Windows 机器集名称。Azure 上的 Windows 机器名称不能超过 15 个字符。因此，由于机器名称的生成方式，机器集名称不能超过 9 个字符。

(7) 将机器配置为 Windows 机器。

(8) 将 Windows 节点配置为计算机器。

(9) 指定一个 **WindowsServer** 镜像，它定义了 **2019-Datacenter-with-Containers** SKU。

(10) 指定 Azure 区域，如 **centralus**。

(14) 由 WMCO 配置第一个 Windows 机器时创建的。之后，所有后续机器组都可以使用 **windows-user-data**。

(16) 指定您所在地区（region）内要放置机器的区域 (zone)。确保您的地区支持您指定的区域。

5.2.3. 创建机器集

除了安装程序创建的机器集之外，还可创建自己的机器集来动态管理您选择的特定工作负载的机器计算资源。

先决条件

- 部署一个 OpenShift Container Platform 集群。
- 安装 OpenShift CLI（**oc**）。
- 以具有 **cluster-admin** 权限的用户身份登录 **oc**。

流程

1. 创建一个包含机器集自定义资源（CR）示例的新 YAML 文件，并将其命名为 **<file_name>.yaml**。
   确保设置 **<clusterID>** 和 **<role>** 参数值。

   a. 如果您不确定要为特定字段设置哪个值，您可以从集群中检查现有机器集：

   ```
   $ oc get machinesets -n openshift-machine-api
   ```

-

输出示例

```
NAME                                DESIRED   CURRENT   READY   AVAILABLE   AGE
agl030519-vplxk-worker-us-east-1a   1         1         1       1           55m
agl030519-vplxk-worker-us-east-1b   1         1         1       1           55m
agl030519-vplxk-worker-us-east-1c   1         1         1       1           55m
agl030519-vplxk-worker-us-east-1d   0         0                             55m
agl030519-vplxk-worker-us-east-1e   0         0                             55m
agl030519-vplxk-worker-us-east-1f   0         0                             55m
```

b. 检查特定机器集的值：

```
$ oc get machineset <machineset_name> -n \
    openshift-machine-api -o yaml
```

输出示例

```
...
template:
  metadata:
    labels:
      machine.openshift.io/cluster-api-cluster: agl030519-vplxk (1)
      machine.openshift.io/cluster-api-machine-role: worker (2)
      machine.openshift.io/cluster-api-machine-type: worker
      machine.openshift.io/cluster-api-machineset: agl030519-vplxk-worker-us-east-1a
```

(1) 集群 ID。

(2) 默认节点标签。

2. 创建新的 **MachineSet** CR:

```
$ oc create -f <file_name>.yaml
```

3. 查看机器集列表：

```
$ oc get machineset -n openshift-machine-api
```

输出示例

```
NAME                                DESIRED   CURRENT   READY   AVAILABLE   AGE
agl030519-vplxk-windows-worker-us-east-1a   1         1         1       1           11m
agl030519-vplxk-worker-us-east-1a   1         1         1       1           55m
agl030519-vplxk-worker-us-east-1b   1         1         1       1           55m
agl030519-vplxk-worker-us-east-1c   1         1         1       1           55m
agl030519-vplxk-worker-us-east-1d   0         0                             55m
agl030519-vplxk-worker-us-east-1e   0         0                             55m
agl030519-vplxk-worker-us-east-1f   0         0                             55m
```

当新机器集可用时，**DESIRED** 和 **CURRENT** 的值会匹配。如果机器集不可用，请等待几分钟，然后再次运行命令。

5.2.4. 其他资源

- 有关管理机器集的更多信息，请参阅 *机器管理* 部分。

5.3. 在 VSPHERE 上创建 WINDOWS MACHINESET 对象

您可以在 VMware vSphere 上的 OpenShift Container Platform 集群中创建 Windows **MachineSet** 对象来满足特定目的。例如，您可以创建基础架构 Windows 机器集和相关机器，以便将支持的 Windows 工作负载转移到新的 Windows 机器上。

先决条件

- 已使用 Operator Lifecycle Manager（OLM）安装 Windows Machine Config Operator（WMCO）。
- 您可以使用受支持的 Windows 服务器作为操作系统镜像，并启用了 Docker 格式的容器运行时附加组件。

重要

目前，在 Windows 节点中使用 Docker 格式的容器运行时。Kubernetes 将弃用 Docker 作为容器运行时，详情请参阅 Kubernetes 文档中的 Docker 弃用信息。在未来的 Kubernetes 发行版本中，Containerd 将是 Windows 节点新支持的容器运行时。

5.3.1. Machine API 概述

Machine API 将基于上游 Cluster API 项目的主要资源与自定义 OpenShift Container Platform 资源相结合。

对于 OpenShift Container Platform 4.8 集群，Machine API 在集群安装完成后执行所有节点主机置备管理操作。由于此系统的缘故，OpenShift Container Platform 4.8 在公有或私有云基础架构之上提供了一种弹性动态置备方法。

两种主要资源分别是：

Machines

描述节点主机的基本单元。机器具有 **providerSpec** 规格，用于描述为不同云平台提供的计算节点的类型。例如，Amazon Web Services (AWS) 上的 worker 节点的机器类型可能会定义特定的机器类型和所需的元数据。

机器集

MachineSet 资源是机器组。机器集适用于机器，复制集则适用于 pod。如果需要更多机器或必须缩减规模，则可以更改机器集的 **replicas** 字段来满足您的计算需求。

警告

control plane 机器不能由机器集管理。

以下自定义资源可为集群添加更多功能：

机器自动扩展

MachineAutoscaler 资源自动扩展云中的机器。您可以为指定机器集中的节点设置最小和最大扩展界限，机器自动扩展就会维护此范围内的节点。**ClusterAutoscaler** 对象存在后，**MachineAutoscaler** 对象生效。**ClusterAutoscaler** 和 **MachineAutoscaler** 资源都由 **ClusterAutoscalerOperator** 对象提供。

集群自动扩展

此资源基于上游集群自动扩展项目。在 OpenShift Container Platform 实现中，它通过扩展机器集 API 来与 Machine API 集成。您可以为核心、节点、内存和 GPU 等资源设置集群范围的扩展限制。您可以设置优先级，使集群对 Pod 进行优先级排序，以便不针对不太重要的 Pod 使新节点上线。您还可以设置扩展策略，以便可以扩展节点，但不会缩减节点。

机器健康检查

MachineHealthCheck 资源可检测机器何时处于不健康状态并将其删除，然后在支持的平台上生成新的机器。

在 OpenShift Container Platform 版本 3.11 中，您无法轻松地推出多区架构，因为集群不负责管理机器置备。自 OpenShift Container Platform 版本 4.1 起，此过程变得更加容易。每个机器集限定在一个区域，因此安装程序可以代表您将机器集分发到多个可用区。然后，由于您的计算是动态的，因此在面对区域故障时，您始终都有一个区域来应对必须重新平衡机器的情况。自动扩展器在集群生命周期内尽可能提供平衡。

5.3.2. 为 Windows 容器工作负载准备 vSphere 环境

您必须通过创建 vSphere Windows VM 金级镜像并启用与 WMCO 的内部 API 服务器通信来为 Windows 容器工作负载准备 vSphere 环境。

5.3.2.1. 创建 vSphere Windows 虚拟机金级镜像

创建 vSphere Windows 虚拟机（VM）金级镜像。

先决条件

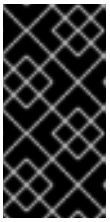
- 您已创建了私钥/公钥对，用于在 OpenSSH 服务器中配置基于密钥的身份验证。还必须在 Windows Machine Config Operator (WMCO) 命名空间中配置私钥。这需要允许 WMCO 与 Windows 虚拟机进行通信。如需了解更多详细信息，请参阅"为 Windows Machine Config Operator 配置 secret" 部分。

注意

在创建 Windows 虚拟机的多个情形中，您必须使用 Microsoft PowerShell 命令。本指南中的 PowerShell 命令由 **PS C:\>** 前缀区分。

流程

1. 使用 Windows Server 2022、OS Build 20348.681 或更高版本在 vSphere 客户端中创建新虚拟机。



重要

您的虚拟机的虚拟硬件版本必须满足 OpenShift Container Platform 的基础架构要求。如需更多信息，请参阅 OpenShift Container Platform 文档中的 "VMware vSphere infrastructure requirements" 部分。另外，您还可以参考 VMware 有关虚拟机硬件版本的文档。

2. 在 Windows 虚拟机上安装和配置 VMware Tools 版本 11.0.6 或更高版本。如需更多信息，参阅 VMware Tools 文档。

3. 在 Windows 虚拟机上安装了 VMware Tools 后，请验证以下内容：

   a. The **C:\ProgramData\VMware\VMware Tools\tools.conf** 文件存在，并带有以下条目：

   ```
   exclude-nics=
   ```

   如果 **tools.conf** 文件不存在，则使用 **exclude-nics** 选项取消注释并设置为空值创建该文件。

   此条目确保通过混合覆盖在 Windows 虚拟机上生成的克隆 vNIC 不被忽略。

   b. Windows 虚拟机在 vCenter 中有一个有效的 IP 地址：

   ```
   C:\> ipconfig
   ```

   c. VMTools Windows 服务正在运行：

   ```
   PS C:\> Get-Service -Name VMTools | Select Status, StartType
   ```

4. 在 Windows 虚拟机上安装和配置 OpenSSH 服务器。如需了解更多详细信息，请参阅 Microsoft 的有关安装 OpenSSH 的文档。

5. 为管理用户设置 SSH 访问权限。详情请参阅 Microsoft 文档的管理员用户。

重要

指令中使用的公钥必须与稍后在 WMCO 命名空间中创建的私钥对应，后者包含您的 secret。如需了解更多详细信息，请参阅"为 Windows Machine Config Operator 配置 secret" 部分。

6. 按照 Microsoft 文档 在 Windows 虚拟机上安装 **docker** 容器运行时。

7. 您必须在 Windows 虚拟机中创建新的防火墙规则，以允许进入连接的容器日志。运行以下命令，在 TCP 端口 10250 中创建防火墙规则：

```
PS C:\> New-NetFirewallRule -DisplayName "ContainerLogsPort" -LocalPort 10250 -
Enabled True -Direction Inbound -Protocol TCP -Action Allow -EdgeTraversalPolicy Allow
```

8. 克隆 Windows 虚拟机，使其成为可重复利用的镜像。遵循 VMware 文档来了解如何克隆现有虚拟机以了解更多详细信息。

9. 在克隆的 Windows 虚拟机上，运行 Windows Sysprep 工具 ：

```
C:\> C:\Windows\System32\Sysprep\sysprep.exe /generalize /oobe /shutdown /unattend:
<path_to_unattend.xml> 1
```

1 指定 **unattend.xml** 文件的路径。

注意

对于您可以在 Windows 镜像中运行 **sysprep** 命令的次数有一个限制。如需更多信息，请参阅 Microsoft 文档。

提供了 **unattend.xml** 示例，它维护 WMCO 所需的所有更改。您必须修改这个示例，它不能直接使用。

例 5.1. unattend.xml 示例

```
<?xml version="1.0" encoding="UTF-8"?>
<unattend xmlns="urn:schemas-microsoft-com:unattend">
  <settings pass="specialize">
    <component xmlns:wcm="http://schemas.microsoft.com/WMIConfig/2002/State"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" name="Microsoft-Windows-
International-Core" processorArchitecture="amd64"
publicKeyToken="31bf3856ad364e35" language="neutral" versionScope="nonSxS">
      <InputLocale>0409:00000409</InputLocale>
      <SystemLocale>en-US</SystemLocale>
      <UILanguage>en-US</UILanguage>
      <UILanguageFallback>en-US</UILanguageFallback>
      <UserLocale>en-US</UserLocale>
    </component>
    <component xmlns:wcm="http://schemas.microsoft.com/WMIConfig/2002/State"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" name="Microsoft-Windows-
Security-SPP-UX" processorArchitecture="amd64" publicKeyToken="31bf3856ad364e35"
language="neutral" versionScope="nonSxS">
      <SkipAutoActivation>true</SkipAutoActivation>
    </component>
    <component xmlns:wcm="http://schemas.microsoft.com/WMIConfig/2002/State"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" name="Microsoft-Windows-
SQMApi" processorArchitecture="amd64" publicKeyToken="31bf3856ad364e35"
language="neutral" versionScope="nonSxS">
      <CEIPEnabled>0</CEIPEnabled>
    </component>
    <component xmlns:wcm="http://schemas.microsoft.com/WMIConfig/2002/State"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" name="Microsoft-Windows-
Shell-Setup" processorArchitecture="amd64" publicKeyToken="31bf3856ad364e35"
language="neutral" versionScope="nonSxS">
      <ComputerName>winhost</ComputerName> (1)
    </component>
  </settings>
  <settings pass="oobeSystem">
    <component xmlns:wcm="http://schemas.microsoft.com/WMIConfig/2002/State"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" name="Microsoft-Windows-
Shell-Setup" processorArchitecture="amd64" publicKeyToken="31bf3856ad364e35"
language="neutral" versionScope="nonSxS">
      <AutoLogon>
        <Enabled>false</Enabled> (2)
      </AutoLogon>
      <OOBE>
        <HideEULAPage>true</HideEULAPage>
        <HideLocalAccountScreen>true</HideLocalAccountScreen>
        <HideOEMRegistrationScreen>true</HideOEMRegistrationScreen>
        <HideOnlineAccountScreens>true</HideOnlineAccountScreens>
```

```
        <HideWirelessSetupInOOBE>true</HideWirelessSetupInOOBE>
        <NetworkLocation>Work</NetworkLocation>
        <ProtectYourPC>1</ProtectYourPC>
        <SkipMachineOOBE>true</SkipMachineOOBE>
        <SkipUserOOBE>true</SkipUserOOBE>
      </OOBE>
      <RegisteredOrganization>Organization</RegisteredOrganization>
      <RegisteredOwner>Owner</RegisteredOwner>
      <DisableAutoDaylightTimeSet>false</DisableAutoDaylightTimeSet>
      <TimeZone>Eastern Standard Time</TimeZone>
      <UserAccounts>
        <AdministratorPassword>
          <Value>MyPassword</Value> 3
          <PlainText>true</PlainText>
        </AdministratorPassword>
      </UserAccounts>
    </component>
  </settings>
</unattend>
```

1. 指定 **ComputerName**，它必须遵循 Kubernetes 名称规格。这些规格也适用于在创建新虚拟机时在生成的模板中执行的 Guest OS 自定义。
2. 禁用自动登录，以避免在启动时使用管理员特权打开终端的安全问题。这是默认值，不得更改。
3. 将 **MyPassword** 占位符替换为 Administrator 帐户的密码。这可防止内置的 Administrator 帐户默认拥有空白密码。遵循 Microsoft 选择密码的最佳实践。

Sysprep 工具完成后，Windows 虚拟机将关闭。您不得再使用此虚拟机或打开此虚拟机。

10. 将 Windows 虚拟机转换为 vCenter 中的模板。

5.3.2.1.1. 其他资源

- 为 Windows Machine Config Operator 配置 secret
- VMware vSphere 基础架构要求

5.3.2.2. 在 vSphere 上启用与 WMCO 的内部 API 服务器通信

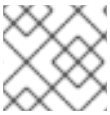
Windows Machine Config Operator（WMCO）从内部 API 服务器端点下载 Ignition 配置文件。您必须启用与内部 API 服务器通信，以便您的 Windows 虚拟机可以下载 Ignition 配置文件，而配置的虚拟机上的 kubelet 只能与内部 API 服务器通信。

先决条件

- 您已在 vSphere 上安装了集群。

流程

- 为 **api-int.<cluster_name>.<base_domain>** 添加一个新的 DNS 项，它指向外部 API 服务器 URL **api.<cluster_name>.<base_domain>**。这可以是一个 CNAME 或一个 A 记录。



注意

外部 API 端点已创建，作为 vSphere 上初始集群安装的一部分。

5.3.3. vSphere 上 Windows MachineSet 对象的 YAML 示例

此 YAML 示例定义了一个在 VMware vSphere 上运行的 Windows **MachineSet** 对象，Windows Machine Config Operator（WMCO）可响应。

```
apiVersion: machine.openshift.io/v1beta1
kind: MachineSet
metadata:
  labels:
    machine.openshift.io/cluster-api-cluster: <infrastructure_id> 1
  name: <windows_machine_set_name> 2
  namespace: openshift-machine-api
spec:
  replicas: 1
  selector:
    matchLabels:
      machine.openshift.io/cluster-api-cluster: <infrastructure_id> 3
      machine.openshift.io/cluster-api-machineset: <windows_machine_set_name> 4
  template:
    metadata:
      labels:
        machine.openshift.io/cluster-api-cluster: <infrastructure_id> 5
        machine.openshift.io/cluster-api-machine-role: worker
        machine.openshift.io/cluster-api-machine-type: worker
        machine.openshift.io/cluster-api-machineset: <windows_machine_set_name> 6
        machine.openshift.io/os-id: Windows 7
    spec:
      metadata:
        labels:
          node-role.kubernetes.io/worker: "" 8
      providerSpec:
        value:
          apiVersion: vsphereprovider.openshift.io/v1beta1
          credentialsSecret:
            name: vsphere-cloud-credentials
          diskGiB: 128 9
          kind: VSphereMachineProviderSpec
          memoryMiB: 16384
          network:
            devices:
            - networkName: "<vm_network_name>" 10
          numCPUs: 4
          numCoresPerSocket: 1
          snapshot: ""
          template: <windows_vm_template_name> 11
          userDataSecret:
            name: windows-user-data 12
          workspace:
            datacenter: <vcenter_datacenter_name> 13
            datastore: <vcenter_datastore_name> 14
```

```
folder: <vcenter_vm_folder_path> 15
resourcePool: <vsphere_resource_pool> 16
server: <vcenter_server_ip> 17
```

1 3 5 指定基于置备集群时所设置的集群 ID 的基础架构 ID。您可以运行以下命令来获取基础架构 ID:

```
$ oc get -o jsonpath='{.status.infrastructureName}{"\n"}' infrastructure cluster
```

2 4 6 指定 Windows 机器集名称。由于 vSphere 中生成机器名称的方式，机器设置的名称不能超过 9 个字符。

7 将机器配置为 Windows 机器。

8 将 Windows 节点配置为计算机器。

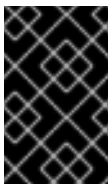
9 指定 vSphere Virtual Machine Disk (VMDK) 的大小。

注意

这个参数不会设置 Windows 分区的大小。您可以使用 **unattend.xml** 文件或创建带有所需磁盘大小的 vSphere Windows 虚拟机 (VM) 金级镜像来重新定义 Windows 分区大小。

10 指定要将机器集部署到的 vSphere VM 网络。此虚拟机网络必须是集群中其他 Linux 计算机器所处的位置。

11 指定要使用的 Windows vSphere 虚拟机模板的完整路径，如 **golden-images/windows-server-template**。名称必须是唯一的。



重要

不要指定原始虚拟机模板。VM 模板必须保持关闭，必须为新的 Windows 机器克隆。启动 VM 模板会将 VM 模板配置为平台上的虚拟机，这会阻止它用作机器集可以应用到的模板。

12 当配置了第一个 Windows 机器时，**windows-user-data** 由 WMCO 创建。之后，所有后续机器组都可以使用 **windows-user-data**。

13 指定要将机器集部署到的 vCenter Datacenter。

14 指定要部署机器集的 vCenter Datastore。

15 指定 vCenter 中 vSphere 虚拟机文件夹的路径，如 **/dc1/vm/user-inst-5ddjd**。

16 可选：为您的 Windows 虚拟机指定 vSphere 资源池。

17 指定 vCenter 服务器 IP 或完全限定域名。

5.3.4. 创建机器集

除了安装程序创建的机器集之外，还可创建自己的机器集来动态管理您选择的特定工作负载的机器计算资源。

先决条件

- 部署一个 OpenShift Container Platform 集群。
- 安装 OpenShift CLI（**oc**）。
- 以具有 **cluster-admin** 权限的用户身份登录 **oc**。

流程

1. 创建一个包含机器集自定义资源（CR）示例的新 YAML 文件，并将其命名为 **<file_name>.yaml**。
   确保设置 **<clusterID>** 和 **<role>** 参数值。

   a. 如果您不确定要为特定字段设置哪个值，您可以从集群中检查现有机器集：

   ```
   $ oc get machinesets -n openshift-machine-api
   ```

   输出示例

   ```
   NAME                                DESIRED   CURRENT   READY   AVAILABLE   AGE
   agl030519-vplxk-worker-us-east-1a   1         1         1       1           55m
   agl030519-vplxk-worker-us-east-1b   1         1         1       1           55m
   agl030519-vplxk-worker-us-east-1c   1         1         1       1           55m
   agl030519-vplxk-worker-us-east-1d   0         0                             55m
   agl030519-vplxk-worker-us-east-1e   0         0                             55m
   agl030519-vplxk-worker-us-east-1f   0         0                             55m
   ```

   b. 检查特定机器集的值：

   ```
   $ oc get machineset <machineset_name> -n \
       openshift-machine-api -o yaml
   ```

   输出示例

   ```
   ...
   template:
     metadata:
       labels:
         machine.openshift.io/cluster-api-cluster: agl030519-vplxk 1
         machine.openshift.io/cluster-api-machine-role: worker 2
         machine.openshift.io/cluster-api-machine-type: worker
         machine.openshift.io/cluster-api-machineset: agl030519-vplxk-worker-us-east-1a
   ```

   1 集群 ID。

   2 默认节点标签。

2. 创建新的 **MachineSet** CR:

   ```
   $ oc create -f <file_name>.yaml
   ```

3. 查看机器集列表：

```
$ oc get machineset -n openshift-machine-api
```

输出示例

```
NAME                                  DESIRED   CURRENT   READY   AVAILABLE   AGE
agl030519-vplxk-windows-worker-us-east-1a   1         1         1       1           11m
agl030519-vplxk-worker-us-east-1a   1         1         1       1           55m
agl030519-vplxk-worker-us-east-1b   1         1         1       1           55m
agl030519-vplxk-worker-us-east-1c   1         1         1       1           55m
agl030519-vplxk-worker-us-east-1d   0         0                             55m
agl030519-vplxk-worker-us-east-1e   0         0                             55m
agl030519-vplxk-worker-us-east-1f   0         0                             55m
```

当新机器集可用时，**DESIRED** 和 **CURRENT** 的值会匹配。如果机器集不可用，请等待几分钟，然后再次运行命令。

5.3.5. 其他资源

- 有关管理机器集的更多信息，请参阅 *机器管理* 部分。

第 6 章 调度 WINDOWS 容器工作负载

您可以将 Windows 工作负载调度到 Windows 计算节点。

注意

使用 集群范围代理 的集群中不支持 WMCO，因为 WMCO 无法通过工作负载的代理连接路由流量。

先决条件

- 已使用 Operator Lifecycle Manager（OLM）安装 Windows Machine Config Operator（WMCO）。
- 您可以使用 Windows 容器作为 OS 镜像，并启用了 Docker 格式的容器运行时附加组件。
- 您已创建了 Windows 机器集。

重要

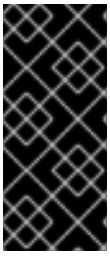
目前，在 Windows 节点中使用 Docker 格式的容器运行时。Kubernetes 将弃用 Docker 作为容器运行时，详情请参阅 Kubernetes 文档中的 Docker 弃用信息。在未来的 Kubernetes 发行版本中，Containerd 将是 Windows 节点新支持的容器运行时。

6.1. WINDOWS POD 放置

在集群中部署 Windows 工作负载前，您必须配置 Windows 节点调度，以便正确分配 pod。在有了托管 Windows 节点的机器后，就可以使用管理 Linux 节点相同的方法进行管理。同样，将 Windows pod 调度到适当的 Windows 节点会以相似的方式完成，可以使用污点、容限和节点选择器等机制。

如果在一个集群中，有多个操作系统以及运行多个 Windows OS 变体，您必须使用 **RuntimeClass** 对象将 Windows pod 映射到基本 Windows OS 变体。例如，如果您在不同 Windows Server 容器版本中运行多个 Windows 节点，集群可将 Windows pod 调度到不兼容的 Windows OS 变体。您必须为集群中的每个 Windows OS 变体配置 **RuntimeClass** 对象。如果集群中只有一个 Windows OS 变体，则建议使用 **RuntimeClass** 对象。

如需更多信息，请参阅微软有关主机和容器版本兼容性的文档。



重要

容器基础镜像与容器要调度到的节点的 Windows OS 版本和构建号需要相同。

另外，如果您将 Windows 节点从一个版本升级到另一个版本，例如从 20H2 升级到 2022，则需要升级容器基础镜像以匹配新版本。如需更多信息，请参阅 Windows 容器版本兼容性。

其他资源

- 使用调度程序控制 pod 放置
- 使用节点污点控制 pod 放置
- 使用节点选择器将 pod 放置到特定节点

6.2. 创建 RUNTIMECLASS 对象来封装调度机制

使用 **RuntimeClass** 对象简化了调度机制的使用，如污点和容限；您可以部署一个运行时类来封装您的污点和容限，然后将其应用到 pod，再将它们调度到适当的节点。在支持多个操作系统变体的集群中，还需要创建运行时类。

流程

1. 创建 **RuntimeClass** 对象 YAML 文件。例如，**runtime-class.yaml**：

```
apiVersion: node.k8s.io/v1beta1
kind: RuntimeClass
metadata:
  name: <runtime_class_name> 1
handler: 'docker'
scheduling:
  nodeSelector: 2
    kubernetes.io/os: 'windows'
    kubernetes.io/arch: 'amd64'
    node.kubernetes.io/windows-build: '10.0.17763'
  tolerations: 3
  - effect: NoSchedule
    key: os
    operator: Equal
    value: "Windows"
```

1. 指定 **RuntimeClass** 对象名称，该名称在您要由此运行时类管理的 pod 中定义。
2. 指定支持这个运行时类的节点必须存在的标签。使用此运行时类的 Pod 只能调度到与此选择器匹配的节点。运行时类的节点选择器与 pod 的现有节点选择器合并。任何冲突都会阻止 pod 调度到节点。
3. 指定要在 pod 附加的容限（不包括重复），在准入过程中使用此运行时类运行。这将合并 pod 和运行时类容许的节点集合。

2. 创建 **RuntimeClass** 对象：

```
$ oc create -f <file-name>.yaml
```

例如：

```
$ oc create -f runtime-class.yaml
```

3. 将 **RuntimeClass** 对象应用到您的 pod，以确保其被调度到适当的操作系统变体：

```
apiVersion: v1
kind: Pod
metadata:
  name: my-windows-pod
spec:
  runtimeClassName: <runtime_class_name> 1
  ...
```

1 指定管理 pod 调度的运行时类。

6.3. WINDOWS 容器工作负载部署示例

在有 Windows 计算节点可用后，您可以将 Windows 容器工作负载部署到集群中。

注意

这个示例部署仅供参考。

Service 对象示例

```
apiVersion: v1
kind: Service
metadata:
  name: win-webserver
  labels:
    app: win-webserver
spec:
  ports:
    # the port that this service should serve on
  - port: 80
    targetPort: 80
  selector:
    app: win-webserver
  type: LoadBalancer
```

Deployment 对象示例

```
apiVersion: apps/v1
kind: Deployment
metadata:
  labels:
    app: win-webserver
  name: win-webserver
spec:
  selector:
    matchLabels:
      app: win-webserver
  replicas: 1
  template:
    metadata:
      labels:
        app: win-webserver
      name: win-webserver
    spec:
      tolerations:
      - key: "os"
        value: "Windows"
        Effect: "NoSchedule"
      containers:
      - name: windowswebserver
        image: mcr.microsoft.com/windows/servercore:ltsc2019
```

```
      imagePullPolicy: IfNotPresent
      command:
      - powershell.exe
      - -command
      - $listener = New-Object System.Net.HttpListener; $listener.Prefixes.Add('http://*:80/');
$listener.Start();Write-Host('Listening at http://*:80/'); while ($listener.IsListening) { $context =
$listener.GetContext(); $response = $context.Response; $content='<html><body><H1>Red Hat
OpenShift + Windows Container Workloads</H1></body></html>'; $buffer =
[System.Text.Encoding]::UTF8.GetBytes($content); $response.ContentLength64 = $buffer.Length;
$response.OutputStream.Write($buffer, 0, $buffer.Length); $response.Close(); };
      securityContext:
        runAsNonRoot: false
        windowsOptions:
          runAsUserName: "ContainerAdministrator"
    nodeSelector:
      kubernetes.io/os: windows
```

注意

当使用 **mcr.microsoft.com/powershell:<tag>** 容器镜像时，您必须将命令定义为 **pwsh.exe**。如果使用 **mcr.microsoft.com/windows/servercore:<tag>** 容器镜像，则必须将该命令定义为 **powershell.exe**。如需更多信息，请参阅微软的文档。

6.4. 手动扩展机器集

要在机器集中添加或删除机器实例，您可以手动扩展机器集。

这个指南与全自动的、安装程序置备的基础架构安装相关。自定义的、用户置备的基础架构安装没有机器集。

先决条件

- 安装 OpenShift Container Platform 集群和 **oc** 命令行。
- 以具有 **cluster-admin** 权限的用户身份登录 **oc**。

流程

1. 查看集群中的机器集：

   ```
   $ oc get machinesets -n openshift-machine-api
   ```

   机器集以 **<clusterid>-worker-<aws-region-az>** 的形式列出。

2. 查看集群中的机器：

   ```
   $ oc get machine -n openshift-machine-api
   ```

3. 在您要删除的机器上设置注解：

   ```
   $ oc annotate machine/<machine_name> -n openshift-machine-api
   machine.openshift.io/cluster-api-delete-machine="true"
   ```

4. 进行 cordon 操作，排空您要删除的节点：

```
$ oc adm cordon <node_name>
$ oc adm drain <node_name>
```

5. 扩展机器集：

```
$ oc scale --replicas=2 machineset <machineset> -n openshift-machine-api
```

或者：

```
$ oc edit machineset <machineset> -n openshift-machine-api
```

提示

您还可以应用以下 YAML 来扩展机器集：

```
apiVersion: machine.openshift.io/v1beta1
kind: MachineSet
metadata:
  name: <machineset>
  namespace: openshift-machine-api
spec:
  replicas: 2
```

您可以扩展或缩减机器集。需要过几分钟以后新机器才可用。

验证

- 验证删除预期的机器：

```
$ oc get machines
```

第 7 章 WINDOWS 节点升级

您可以通过升级 Windows Machine Config Operator（WMCO），以确保 Windows 节点具有最新的更新。

7.1. WINDOWS MACHINE CONFIG OPERATOR 升级

当发布与当前集群版本兼容的 Windows Machine Config Operator（WMCO）的新版本时，Operator 会根据升级频道和订阅批准策略升级，在使用 Operator Lifecycle Manager（OLM）时会安装它。WMCO 升级会在 Windows 机器中产生 Kubernetes 组件升级。

注意

如果要升级到 WMCO 的新版本并希望使用集群监控，则必须在 WMCO 命名空间中具有 **openshift.io/cluster-monitoring=true** 标签。如果将该标签添加到已存在的 WMCO 命名空间，并且已经配置了 Windows 节点，重启 WMCO pod 以允许显示监控图形。

对于非破坏性升级，WMCO 会终止之前 WMCO 版本配置的 Windows 机器，并使用当前版本重新创建它们。这可以通过删除 **Machine** 对象来完成，这会导致 Windows 节点排空和删除。为便于升级，WMCO 会为所有配置的节点添加版本注解。在升级过程中，版本注解中的不匹配会导致删除和重新创建 Windows 机器。要在升级过程中减少服务的中断，WMCO 一次只更新一个 Windows 机器。

重要

WMCO 仅负责更新 Kubernetes 组件，而不负责 Windows 操作系统更新。您在创建虚拟机时提供 Windows 镜像，因此您需要提供更新的镜像。您可以通过更改 **MachineSet** spec 中的镜像配置来提供更新的 Windows 镜像。

有关使用 Operator Lifecycle Manager (OLM) 升级 Operator 的更多信息，请参阅升级已安装的 Operator。

第 8 章 使用 BRING-YOUR-OWN-HOST (BYOH) WINDOWS 实例作为节点

通过自带主机 (BYOH)，用户可以重新指定 Windows Server 虚拟机的用途，并将它们带入 OpenShift Container Platform。BYOH Windows 实例让希望在 Windows 服务器脱机时缓解重大中断的用户受益。

8.1. 配置 BYOH WINDOWS 实例

创建 BYOH Windows 实例需要在 Windows Machine Config Operator (WMCO) 命名空间中创建配置映射。

先决条件

要作为节点附加到集群的任何 Windows 实例都必须满足以下要求：

- 实例上必须安装 Docker 容器运行时。
- 实例必须与集群中的 Linux worker 节点位于同一个网络中。
- 端口 22 必须处于打开状态，并且正在运行 SSH 服务器。
- SSH 服务器的默认 shell 必须是 Windows 命令 shell 或 **cmd.exe**。
- 端口 10250 必须处于打开状态才能收集日志。
- 管理员用户会看到一个密钥，它在 secret 集中使用作为一个授权 SSH 密钥。
- 如果您要为安装程序置备的基础架构 (IPI) AWS 集群创建 BYOH Windows 实例，您必须在与 worker 节点的机器集中的 **spec.template.spec.value.tag** 值匹配的 AWS 实例中添加标签。例如 **kubernetes.io/cluster/<cluster_id>: owned** 或 **kubernetes.io/cluster/<cluster_id>: shared**。
- 如果您要在 vSphere 上创建 BYOH Windows 实例，则必须启用与内部 API 服务器的通信。
- 实例的主机名必须遵循 RFC 1123 DNS 标签要求，其中包括以下标准：
  - 仅包含小写字母数字字符或 '-'。
  - 以字母数字字符开头。
  - 以字母数字字符结尾。

流程

1. 在 WMCO 命名空间中创建一个名为 **windows-instances** 的 ConfigMap，用于描述要添加的 Windows 实例。

注意

格式化配置映射的 data 部分的每个条目，使用地址作为关键字，值为 **username=<username>**。

配置映射示例

```
kind: ConfigMap
apiVersion: v1
```

```
metadata:
  name: windows-instances
  namespace: openshift-windows-machine-config-operator
data:
  10.1.42.1: |- 1
    username=Administrator 2
  instance.example.com: |-
    username=core
```

1 WMCO 通过 SSH 访问实例的地址，可以是 DNS 名称或 IPv4 地址。这个地址必须存在 DNS PTR 记录。如果您的组织使用 DHCP 来分配 IP 地址，建议您将 DNS 名称与 BYOH 实例搭配使用。如果没有，则需要在实例分配新 IP 地址时更新 **windows-instances** ConfigMap。

2 先决条件中创建的管理员用户的名称。

8.2. 删除 BYOH WINDOWS 实例

您可以通过删除配置映射中实例条目来删除附加到集群的 BYOH 实例。删除实例会在添加到集群之前将该实例恢复到其状态。任何日志和容器运行时工件都不会添加到这些实例中。

要完全删除实例，必须使用提供给 WMCO 的当前私钥访问该实例。例如，要从上例中删除 **10.1.42.1** 实例，配置映射将更改为以下内容：

```
kind: ConfigMap
apiVersion: v1
metadata:
  name: windows-instances
  namespace: openshift-windows-machine-config-operator
data:
  instance.example.com: |-
    username=core
```

删除 **windows-instances** 被视为取消构建添加为节点的所有 Windows 实例的请求。

第 9 章 删除 WINDOWS 节点

您可以通过删除其主机 Windows 机器来删除 Windows 节点。

9.1. 删除一个特定的机器

您可以删除特定的机器。

注意

您无法删除控制平面机器。

先决条件

- 安装 OpenShift Container Platform 集群。
- 安装 OpenShift CLI (**oc**) 。
- 以具有 **cluster-admin** 权限的用户身份登录 **oc**。

流程

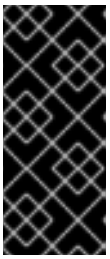
1. 查看集群中的机器，找到要删除的机器：

```
$ oc get machine -n openshift-machine-api
```

命令输出包含 **<clusterid>-worker-<cloud_region>** 格式的机器列表。

2. 删除机器：

```
$ oc delete machine <machine> -n openshift-machine-api
```



重要

默认情况下，机器控制器会尝试排空在机器上运行的节点，直到成功为止。在某些情况下，如错误配置了 pod 的中断预算，节点排空操作可能无法成功完成，从而导致机器无法被删除。您可以在特定机器上使用 "machine.openshift.io/exclude-node-draining" 注解来跳过排空节点的过程。如果要删除的机器属于机器集，则会立即创建一个新机器来满足指定的副本数要求。

第 10 章 禁用 WINDOWS 容器工作负载

您可以通过卸载 Windows Machine Config Operator（WMCO），并删除安装 WMCO 时默认添加的命名空间，来禁用运行 Windows 容器工作负载的能力。

10.1. 卸载 WINDOWS MACHINE CONFIG OPERATOR

您可以从集群卸载 Windows Machine Config Operator（WMCO）。

先决条件

- 删除托管 Windows 工作负载的 Windows **Machine** 对象。

流程

1. 在 **Operators → OperatorHub** 页面中，使用 **Filter by keyword** 复选框来搜索 **Red Hat Windows Machine Config Operator**。
2. 点 **Red Hat Windows Machine Config Operator**标题。Operator 标题表示已安装该 Operator。
3. 在 **Windows Machine Config Operator**描述符页面中，点 **Uninstall**。

10.2. 删除 WINDOWS MACHINE CONFIG OPERATOR 命名空间

您可以删除默认为 Windows Machine Config Operator（WMCO）生成的命名空间。

先决条件

- WMCO 已从集群中移除。

流程

1. 删除 **openshift-windows-machine-config-operator** 命名空间中创建的所有 Windows 工作负载：

```
$ oc delete --all pods --namespace=openshift-windows-machine-config-operator
```

2. 验证 **openshift-windows-machine-config-operator** 命名空间中的所有 pod 是否已删除，或处于终止状态：

```
$ oc get pods --namespace openshift-windows-machine-config-operator
```

3. 删除 **openshift-windows-machine-config-operator** 命名空间：

```
$ oc delete namespace openshift-windows-machine-config-operator
```

其他资源

- 从集群中删除 Operator
- 删除 Windows 节点