OpenShift Container Platform 4.9

更新集群

更新 OpenShift Container Platform 集群

Last Updated: 2023-06-04

OpenShift Container Platform 4.9 更新集群

更新 OpenShift Container Platform 集群

法律通告

Copyright © 2023 Red Hat, Inc.

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

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

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

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

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

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

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

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

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

All other trademarks are the property of their respective owners.

摘要

本文档提供了有关更新和升级 OpenShift Container Platform 集群的信息。更新集群的过程较简单，可以在不需要使集群离线的情况下进行。

目录

第 1 章 更新集群概述

您可以使用 Web 控制台或 OpenShift CLI (**oc**) 使用单一操作来更新 OpenShift Container Platform 4 集群。

1.1. 了解 OPENSHIFT CONTAINER PLATFORM 更新

关于 OpenShift Update Service：对于可访问互联网的集群，红帽通过 OpenShift Container Platform 更新服务提供更新，它作为公共 API 后面的一个托管服务运行。

1.2. 了解升级频道和发行版本

升级频道和发行版本: 使用升级频道，您可以选择一个升级策略。升级频道特定于 OpenShift Container Platform 的次版本。升级频道仅控制发行版本选择，不会影响您安装的集群版本。OpenShift Container Platform 的一个特定版本的 **openshift-install** 二进制文件总会安装该次版本。如需更多信息，请参阅以下：

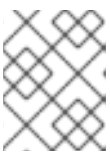
- 升级版本路径
- fast 和 stable 频道的使用和策略
- 了解受限网络集群
- 在频道间切换

1.3. 了解集群 OPERATOR 条件类型

集群 Operator 的状态包括它们的 condition 类型，它告知您 Operator 的健康状况的当前状态。以下定义涵盖了一些常见 ClusterOperator 条件类型的列表。省略了具有额外条件类型和特定 Operator 语言的 Operator。

Cluster Version Operator (CVO) 负责从集群 Operator 收集状态条件，以便集群管理员可以更好地了解 OpenShift Container Platform 集群的状态。

- available: 条件类型 **Available** 表示 Operator 功能且在集群中可用。如果状态是 **False**，则操作对象中的至少一个部分无法正常工作，并且条件要求管理员干预。
- progressing: 条件类型 **Progressing** 表示 Operator 正在主动推出新的代码、传播配置更改，或者从一个稳定状态移到另一个状态。
  当 Operator 协调之前已知状态时，Operator 不会报告条件类型 **Progressing** 为 **True**。如果观察到的集群状态已更改，且 Operator 会响应它，则状态将报告为 **True**，因为它从一个 steady 状态移到另一个状态。
- Degraded：条件类型 **Degraded** 表示 Operator 具有在一段时间内不匹配其所需状态的当前状态。周期可能会因组件而异，但 **Degraded** 状态代表 Operator 条件的持久性观察。因此，Operator 不会波动处于 Degraded 状态和没有处于 **Degraded** 状态。
  如果从一个状态转换到另一个状态的过渡在长时间内没有保留，则可能会有一个不同的条件类型来报告 **Degraded**。Operator 在正常升级过程中不会报告 **Degraded**。Operator 可能会报告 **Degraded**，以响应需要最终管理员干预的持久性基础架构失败。



注意

此条件类型仅表示可能需要调查和调整某项。只要 Operator 可用，**Degraded** 条件就不会造成用户工作负载失败或应用程序停机。

- Upgradeable: 具有条件类型为 **Upgradeable** 的 Operator 指示 Operator 是否根据当前的集群状态安全升级。message 字段包含管理员对集群成功更新需要执行的操作的人类可读描述。当此条件为 **True**、**Unknown** 或缺失时，CVO 允许更新。
  当 **Upgradeable** 状态为 **False** 时，只有次版本更新会受到影响，CVO 会阻止集群执行受影响的更新，除非强制（强制）更新。

1.4. 了解集群版本状况类型

Cluster Version Operator (CVO) 监控集群 Operator 和其他组件，并负责收集集群版本及其 Operator 的状态。此状态包括条件类型，它告知您 OpenShift Container Platform 集群的健康状态和当前状态。

除了 **Available**,**Progressing**, 和 **Upgradeable** 外，还有影响集群版本和 Operator 的条件类型。

- Failing：集群版本状况类型 **Failing** 表示集群无法访问其所需状态，不健康，需要管理员干预。
- Invalid: 集群版本条件类型 **Invalid** 表示集群版本具有阻止服务器执行操作的错误。只要设置了此条件，CVO 仅协调当前状态。
- RetrievedUpdates ：集群版本条件类型 **RetrievedUpdates** 表示已从上游更新服务器检索可用更新。在检索前条件为 **Unknown**，如果更新最近失败或无法检索，则为 **False**；如果 **availableUpdates** 字段是最新且准确的，则为 **True**。

1.5. 准备执行 EUS 更新

准备执行 EUS 到 EUS 更新 ：由于 Kubernetes 的设计，次版本之间的所有 OpenShift Container Platform 升级都必须按顺序进行。您必须从 OpenShift Container Platform 4.8 更新至 4.9，然后更新至 4.10。您无法直接从 OpenShift Container Platform 4.8 更新至 4.10。但是，如果您想要在两个延长更新支持(EUS)版本之间更新，您可以只发生一次重启非控制平面主机。如需更多信息，请参阅以下：

- 更新 EUS 到 EUS

1.6. 使用 WEB 控制台更新集群

使用 Web 控制台将集群更新为一个新的次版本：您可以使用 Web 控制台更新 OpenShift Container Platform 集群。以下步骤在次版本中更新集群。您可以使用相同的说明在次版本之间更新集群。

- 执行 canary rollout 更新
- 暂停 MachineHealthCheck 资源
- 关于在单节点集群中更新 OpenShift Container Platform
- 使用Web控制台更新集群
- 使用 Web 控制台更改更新服务器

1.7. 使用命令行界面(CLI)将集群更新为一个新的次版本

使用 CLI 更新集群的次版本 ：您可以使用 OpenShift CLI (**oc**) 更新 OpenShift Container Platform 集群。以下步骤在次版本中更新集群。您可以使用相同的说明在次版本之间更新集群。

- 暂停 MachineHealthCheck 资源
- 关于在单节点集群中更新 OpenShift Container Platform

- 使用 CLI 更新集群
- 使用 CLI 更改更新服务器

1.8. 执行 CANARY ROLLOUT 更新

执行 Canary 推出部署更新 ： 通过控制对 worker 节点的更新，您可以确保关键任务应用程序在整个更新过程中仍然可用，即使更新过程会导致应用程序失败。根据您的机构需求，您可能需要更新一小部分 worker 节点，在一个时间段内评估集群和工作负载健康状况，然后更新剩余的节点。这称为 *Canary* 更新。或者，您可能还希望将通常需要主机重新引导的 worker 节点更新放入较小的定义的维护窗口（不可能一次使用大型维护窗口来更新整个集群）。您可以执行以下步骤：

- 创建机器配置池来执行 Canary rollout 更新
- 暂停机器配置池
- 执行集群更新
- 取消暂停机器配置池
- 将节点移动到原始机器配置池

1.9. 更新包含使用 RHEL 的计算（COMPUTE）系统的集群

更新包含 RHEL 计算机器的集群 ：您可以更新 OpenShift Container Platform 集群。如果您的集群包含 Red Hat Enterprise Linux（RHEL）系统，则必须执行额外的步骤来更新这些系统。您可以执行以下步骤：

- 使用Web控制台更新集群
- 可选：添加 hook 以在RHEL系统上执行Ansible任务
- 更新集群中的RHEL compute 系统

1.10. 在断开连接的环境中更新集群

在断开连接的环境中的集群更新：如果镜像主机无法同时访问互联网和集群，您可以将镜像镜像到与环境断开连接的文件系统中。然后您可以使用那个主机或可移动介质来实现安装。如果本地容器 registry 和集群连接到镜像 registry 的主机，您可以直接将发行镜像推送到本地 registry。

- 准备您的镜像主机
- 配置允许对容器镜像进行镜像的凭证
- 镜像 OpenShift Container Platform 镜像存储库
- 更新断开连接的集群
- 配置镜像 registry 存储库镜像
- 镜像镜像目录的范围，以减少集群节点重启的频率
- 安装 OpenShift Update Service Operator
- 创建 OpenShift Update Service 应用程序

- 删除 OpenShift Update Service 应用程序
- 卸载 OpenShift Update Service Operator

1.11. 更新在 VSPHERE 上运行的节点上运行的硬件

更新 vSphere 上的硬件 ：您必须确保在 OpenShift Container Platform 支持的硬件版本中运行 vSphere 中运行的节点。目前，硬件版本 13 或更高版本都支持集群中的 vSphere 虚拟机。如需更多信息，请参阅以下信息：

- 更新 vSphere 上的虚拟硬件
- 在 vSphere 上调度虚拟硬件的更新

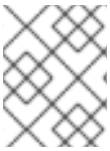
重要

在 vSphere 上运行的集群节点使用硬件版本 13 现已弃用。这个版本仍然被完全支持，但在以后的 OpenShift Container Platform 版本中将会删除支持。现在，硬件版本 15 是 OpenShift Container Platform 中 vSphere 虚拟机的默认版本。

第 2 章 了解 OPENSHIFT CONTAINER PLATFORM 更新

在 OpenShift Container Platform 4 中，您可以使用 Web 控制台或 OpenShift CLI(**oc**)使用单一操作来更新 OpenShift Container Platform 集群。当一个更新可用于其集群时，平台管理员会自动获得通知。

OpenShift Update Service(OSUS)根据 registry 中的发行镜像构建了一个更新可能性图。该图基于特定版本的推荐、经过测试的更新路径。OpenShift Container Platform 集群连接到红帽混合云服务器，并确定用户正在运行的集群以及版本信息。OSUS 使用已知更新目标的信息做出响应。集群管理员或自动更新控制器都会编辑 Cluster Version Operator(CVO)的自定义资源(CR)，以及要升级到的新版本。在 CVO 从 registry 接收更新镜像后，CVO 会应用更改。



注意

之前通过 Operator Lifecycle Manager (OLM) 安装的 Operator 会遵循不同的更新过程。如需更多信息，请参阅更新安装的 Operator。

2.1. 关于 OPENSHIFT UPDATE 服务

OpenShift Update Service (OSUS) 为 OpenShift Container Platform 提供推荐的更新，包括 Red Hat Enterprise Linux CoreOS (RHCOS)。它提供了一个图表，其中包含组件 Operator 的*顶点（vertices）*和连接它们的 *边（edges）*。图中的边代表了您可以安全更新到的版本。顶点是更新的有效负载，用于指定受管集群组件的预期状态。

集群中的 Cluster Version Operator (CVO) 会检查 OpenShift Container Platform 更新服务，并根据当前组件版本和图中的信息决定有效的更新和更新路径。当您请求更新时，CVO 使用该更新的发行镜像来升级您的集群。发行工件 (artifact) 作为容器镜像托管在 Quay 中。

为了让 OpenShift Update Service 仅提供兼容的更新，可以使用一个版本验证管道来驱动自动化过程。每个发行工件都会被验证是否与支持的云平台和系统架构以及其他组件包兼容。在管道确认有适用的版本后，OpenShift Update Service 会通知您它可用。

重要

OpenShift Update Service 显示当前集群的所有推荐更新。如果 OpenShift Update Service 不建议升级路径，这可能是因为更新或目标发行版本存在已知问题。

两个控制器在持续更新模式下运行。第一个控制器持续更新有效负载清单，将清单应用到集群，并输出 Operator 的受控推出的状态，以指示它们是否处于可用、升级或失败状态。第二个控制器轮询 OpenShift Update Service，以确定更新是否可用。

重要

仅支持升级到较新版本。不支持将集群还原或回滚到以前的版本。如果您的更新失败，请联系红帽支持。

在更新过程中，Machine Config Operator(MCO)将新配置应用到集群机器。MCO 会处理由 **maxUnavailable** 字段指定的、协调机器配置池中的节点数量，并将它们标记为不可用。在默认情况下，这个值被设置为 **1**。然后，MCO 会应用新配置并重启机器。

如果您将 Red Hat Enterprise Linux (RHEL) 机器用作 worker，MCO 不会在这些机器上更新 kubelet，因为您必须首先在这些机器上更新 OpenShift API。

当新版本规格应用到旧的 kubelet 时，RHEL 机器无法返回 **Ready** 状态。在机器可用前，您无法完成更新。但是，因为已设置了最大不可用节点数，所以可以在一定机器无法使用的情况下，确保正常的集群操作。

OpenShift Update Service 由 Operator 和一个或多个应用程序实例组成。

2.2. 常见术语

Control plane（控制平面）

control plane 由 control plane 机器组成，负责管理 OpenShift Container Platform 集群。control plane 机器管理计算机器（也被称为 worker）上的工作负载。

Cluster Version Operator

Cluster Version Operator (CVO)启动集群的更新过程。它根据当前的集群版本检查 OSUS，并检索包含可用或可能的更新路径的图形。

Machine Config Operator

Machine Config Operator (MCO)是一个集群级别的 Operator，用于管理操作系统和机器配置。通过 MCO，平台管理员可以配置和更新 worker 节点上的 systemd、CRI-O 和 Kubelet、内核、NetworkManager 和其他系统功能。

OpenShift 更新服务

OpenShift Update Service (OSUS)为 OpenShift Container Platform 提供无线更新，包括 Red Hat Enterprise Linux CoreOS(RHCOS)。它提供了一个图形或图表，其中包含组件 Operator 的顶点和连接它们的边。

Channels

Channels 声明了一个与 OpenShift Container Platform 次版本相关的更新策略。OSUS 使用这个配置的策略来推荐与该策略一致更新边缘。

推荐的更新边缘

推荐的更新边缘 是 OpenShift Container Platform 发行版本之间的建议更新。建议使用给定的更新，具体取决于集群配置的频道、当前版本、已知的错误和其他信息。OSUS 将建议的边缘与 CVO 通信，后者在每个集群中运行。

延长更新支持

所有 post-4.7 甚至编号的次版本都标记为 *延长更新支持* (EUS)版本。这些版本包括了在 EUS 版本之间更轻松地更新路径，可以简化 worker 节点的更新，并可以通过规划 EUS-to-EUS OpenShift Container Platform 版本的更新策略来减少重启 worker 节点的更新。
如需更多信息，请参阅 Red Hat OpenShift Extended Update Support(EUS)概述 。

其他资源

- 机器配置概述
- 更新频道和发行版本

第 3 章 了解升级频道和发行版本

在 OpenShift Container Platform 4.1 中，红帽引进了频道的概念，用于为集群更新推荐适当的版本。通过控制更新的速度，这些升级频道允许您选择更新策略。升级频道与 OpenShift Container Platform 的次要版本关联。例如，OpenShift Container Platform 4.9 升级频道推荐更新到 4.9 并在 4.9 中更新。它们还推荐在 4.8 内更新以及从 4.8 升级到 4.9，以允许 4.8 上的集群最终更新至 4.9。它们不推荐对 4.10 或更高版本的更新。此策略可确保管理员明确决定升级到下一个 OpenShift Container Platform 次要版本。

升级频道仅控制版本选择，它不会影响您安装的集群版本; 特定版本的 OpenShift Container Platform 的 **openshift-install** 二进制文件始终会安装这个特定版本。

OpenShift Container Platform 4.9 提供了以下升级频道：

- **candidate-4.9**
- **fast-4.9**
- **stable-4.9**
- **eus-4.y**（仅在运行偶数 4.y 集群发行版本时，如 4.8）

如果您不希望 Cluster Version Operator 从升级建议服务获取可用的更新，您可以使用 OpenShift CLI 中的 **oc adm upgrade channel** 命令配置空频道。例如，当集群有受限网络访问且没有本地可访问的升级建议服务时，这个配置很有用。

警告

红帽建议升级到 OpenShift Update Service 建议的版本。对于次版本更新，版本必须相邻。红帽没有测试在非连续地版本间的升级，无法保证与之前版本的兼容性。

3.1. 升级频道和发行路径

集群管理员可以通过 Web 控制台配置升级频道。

3.1.1. candidate-4.9 频道

candidate-4.9 频道包含 z-stream（4.9.z）和之前的次版本的候选构建。发行候选版本包含该产品的所有功能但不被正式支持。发行候选版本可以用来测试新版本的功能以决定下一个 OpenShift Container Platform 版本是否适用于您的系统。发行候选是指候选频道中可用的构建，包括那些不包含 预发布版本（如 **-rc** ）的构建。当一个版本出现在候选频道中后，它仍然会进行更多的质量测试。如果达到质量标准，则会将其推广至 **fast-4.9** 或 **stable-4.9** 频道。因此，如果一个特定的版本同时存在于 **candidate-4.9** 频道以及 **fast-4.9** 或 **stable-4.9** 频道中，则代表红帽会支持这个版本。**candidate-4.9** 频道可能会包括任何频道都不推荐更新的发行版本。

您可以使用 **candidate-4.9** 频道从以前的 OpenShift Container Platform 次版本进行更新。

3.1.2. fast-4.9 频道

当红帽声明某个特定版本成为正式发行版本时，**fast-4.9** 频道被更新为 4.9 的新次版本。这意味着，这些版本被完全支持，且具有符合生产环境的质量，当它们作为发行候选版本出现在 **candidate-4.9** 频道期

间，被证明可以正常工作。当一个发行版本出现在 **fast-4.9** 频道中的一段时间后，会被添加到 **stable-4.9** 频道。如果版本没有出现在 **fast-4.9** 频道中，则这个版本一定不会出现在 **stable-4.9** 频道中。

您可以使用 **fast-4.9** 频道来从以前的 OpenShift Container Platform 次版本进行更新。

3.1.3. stable-4.9 频道

虽然当它们的勘误被发布后马上就会出现在 **fast-4.9** 频道中，但这些内容可能需要一段延迟时间会被添加到 **stable-4.9** 频道中。在此延迟期间，红帽 SRE 团队、红帽支持服务以及参与连接的客户程序的生产前和产品环境中收集有关此发行版本的稳定性数据。

您可以使用 **stable-4.9** 频道来从以前的 OpenShift Container Platform 次版本进行更新。

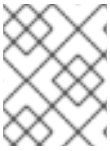
3.1.4. eus-4.y 频道

除了 stable 频道外，所有以数字相等的 OpenShift Container Platform 次版本都提供延长更新支持 (EUS)。对于具有标准和高级订阅的客户，这些 EUS 版本将完全支持和维护支持阶段延长至 18 个月。

虽然 **stable-4.y** 和 **eus-4.y** 频道之间没有区别，直到 OpenShift Container Platform 4.y 过渡到 EUS 阶段，您可以立即切换到 **eus-4.y** 频道。

当提供了升级到下一个 EUS 的频道时，您可以切换到下一个 EUS 频道并升级，直到您升级到下一个 EUS 版本。

此更新过程不适用于 **eus-4.6** 频道。



注意

标准和非 EUS 订阅者都可以访问所有 EUS 软件仓库和所需的 RPM(**rhel-*-eus-rpms**)，它们都能够支持关键目的，如调试和构建驱动程序。

3.1.5. 升级版本路径

OpenShift Container Platform 维护一个升级建议服务，它了解已安装的 OpenShift Container Platform 版本以及您选择用来获取下一版本的频道中的路径。

您可在 **fast-4.9** 频道中看到以下内容：

- 4.9.0
- 4.9.1
- 4.9.3
- 4.9.4

该服务只建议经过测试且不存在严重问题的升级。它不会建议把系统更新到一个包含已知漏洞的 OpenShift Container Platform 版本。例如，如果集群为 4.9.1，OpenShift Container Platform 建议为 4.9.4，那么可以安全地从 4.9.1 更新至 4.9.4。您不需要一定在连续的补丁号间进行升级。在这个示例中，该频道并没有（且重来没有包括） 4.9.2。

更新的稳定性取决于您的频道。在 **candidate-4.9** 频道中存在一个更新建议并不意味着这个更新会被支持。它代表，在更新中还没有发现任何严重问题，这可能是因为此更新还没有足够的使用情况来证明它的稳定性。如果在 **fast-4.9** 或 **stable-4.9** 频道中出现了一个更新建议，则代表这个更新被支持。虽然发行版本永远不会从一个频道中删除，但存在严重问题的更新建议会从所有频道中删除。在更新建议被删除后启动的更新仍被支持。

红帽最终会为 **fast-4.9** 或 **stable-4.9** 频道中支持的发行版本提供到最新的 4.9.z 版本的更新路径，但可能会因为创建并验证解决已知问题的更新路径而有一定的延迟。

3.1.6. fast 和 stable 频道的使用和策略

通过 **fast-4.9** 和 **stable-4.9** 频道，您可以选择在一个发行版本正式发行后马上接收到这个版本，或选择由红帽控制向用户推出更新的过程。如果在推出部署的过程或之后发现问题，到这个版本的更新可能会在 **fast-4.9** 和 **stable-4.9** 频道中被禁止。一个新版本可能会出现，做为新的首选升级目标。

通过在 **fast-4.9** 频道中配置预生产环境的系统、在 **stable-4.9** 频道中配置生产环境的系统，并参与红帽连接的客户项目，用户可以改进更新的过程。红帽使用这个程序观察更新对您特定的硬件和软件配置的影响。将来的版本可能会改进或修改更新从 **fast-4.9** 频道进入 **stable-4.9** 频道的速度。

3.1.7. 受限网络集群

如果您自己为 OpenShift Container Platform 集群管理容器镜像，您必须考虑与产品关联的红帽勘误中的升级信息。在升级过程中，用户界面可能会提醒您在这些版本间进行切换，因此您必须在跳过这些警告前确定选择了正确的版本。

3.1.8. 在频道间切换

可以从 Web 控制台或通过 **adm upgrade channel** 命令来切换频道：

```
$ oc adm upgrade channel <channel>
```

如果您切换到没有包括当前版本的频道，web 控制台将显示警报。在没有当前发行版本的频道中，web 控制台不推荐任何更新。但是，您可以在任何时候返回原始频道。

更改您的频道可能会影响集群的可支持性。可能适用以下条件：

- 如果您从 **stable-4.9** 频道改到 **fast-4.9** 频道，您的集群仍然被支持。
- 您可以随时切换到 **candidate-4.9** 频道，但这个频道的一些发行版本可能不被支持。
- 如果您当前的发行版本是正式发布版本，则可以从 **candidate-4.9** 频道切换到 **fast-4.9** 频道。
- 从 **fast-4.9** 频道切换到 **stable-4.9** 频道始终被允许。如果当前版本最近被提升，该发行版本可能会有最多一天的延迟才会出现在 **stable-4.9** 中。

第 4 章 准备升级到 OPENSHIFT CONTAINER PLATFORM 4.9

OpenShift Container Platform 4.9 使用 Kubernetes 1.22，它删除了大量已弃用的 **v1beta1** API。

OpenShift Container Platform 4.8.14 引入了一项要求，即管理员必须先提供手动确认，然后才能从 OpenShift Container Platform 4.8 升级到 4.9。这有助于防止升级到 OpenShift Container Platform 4.9 后出现问题，其中已删除的 API 仍在由运行或与集群交互的工作负载、工具或其他组件使用。管理员必须针对将要删除的任何 API 评估其集群，并迁移受影响的组件，以使用适当的新 API 版本。完成此评估和迁移后，管理员可以进行确认。

在将 OpenShift Container Platform 4.8 集群更新至 4.9 之前，您必须提供管理员确认。

4.1. 删除的 KUBERNETES API

OpenShift Container Platform 4.9 使用 Kubernetes 1.22，它删除了已弃用的 **v1beta1** API。您必须迁移清单和 API 客户端以使用 **v1** API 版本。有关迁移已删除 API 的更多信息，请参阅 Kubernetes 文档。

表 4.1. v1beta1 API 从 Kubernetes 1.22 中删除

资源	API	主要变化
APIService	**apiregistration.k8s.io/v1beta1**	否
CertificateSigningRequest	**certificates.k8s.io/v1beta1**	是
ClusterRole	**rbac.authorization.k8s.io/v1beta1**	否
ClusterRoleBinding	**rbac.authorization.k8s.io/v1beta1**	否
CSIDriver	**storage.k8s.io/v1beta1**	否
CSINode	**storage.k8s.io/v1beta1**	否
CustomResourceDefinition	**apiextensions.k8s.io/v1beta1**	是
入口	**extensions/v1beta1**	是
入口	**networking.k8s.io/v1beta1**	是
IngressClass	**networking.k8s.io/v1beta1**	否
Lease	**coordination.k8s.io/v1beta1**	否
LocalSubjectAccessReview	**authorization.k8s.io/v1beta1**	是
MutatingWebhookConfiguration	**admissionregistration.k8s.io/v1beta1**	是
PriorityClass	**scheduling.k8s.io/v1beta1**	否

资源	API	主要变化
角色	**rbac.authorization.k8s.io/v1beta1**	否
RoleBinding	**rbac.authorization.k8s.io/v1beta1**	否
SelfSubjectAccessReview	**authorization.k8s.io/v1beta1**	是
StorageClass	**storage.k8s.io/v1beta1**	否
SubjectAccessReview	**authorization.k8s.io/v1beta1**	是
TokenReview	**authentication.k8s.io/v1beta1**	否
ValidatingWebhookConfiguration	**admissionregistration.k8s.io/v1beta1**	是
VolumeAttachment	**storage.k8s.io/v1beta1**	否

4.2. 为删除的 API 评估集群

有多种方法可帮助管理员确定将使用移除的 API 的位置。但是，OpenShift Container Platform 无法识别所有实例，尤其是使用闲置或外部工具的工作负载。管理员负责针对已删除 API 实例正确评估所有工作负载和其他集成。

4.2.1. 查看警报以识别已删除 API 的使用

当使用 API 时会触发两个警报，该 API 将在以后的版本中删除：

- **APIRemovedInNextReleaseInUse** - 针对将在下一个 OpenShift Container Platform 发行版本中删除的 API。
- **APIRemovedInNextEUSReleaseInUse** - 针对将在下一个 OpenShift Container Platform 扩展更新支持（EUS）版本中删除的 API。

如果其中任何一个警报在集群中触发，请查看警报并采取措施通过迁移清单和 API 客户端以使用新的 API 版本来清除警报。

使用 **APIRequestCount** API 获取有关使用哪些 API 的更多信息，以及哪些工作负载正在使用删除的 API，因为警报不提供此信息。此外，一些 API 可能不会触发这些警报，但仍然被 **APIRequestCount** 捕获。将警报调优为不太敏感，以避免在生产环境中出现警报。

4.2.2. 使用 APIRequestCount 来识别已删除 API 的使用

您可以使用 **APIRequestCount API** 来跟踪 API 请求，并检查它们中的任何请求是否使用其中一个已删除的 API。

先决条件

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

流程

- 运行以下命令并检查输出的 **REMOVEDINRELEASE** 列以确定当前正在使用的 API:

```
$ oc get apirequestcounts
```

输出示例

```
NAME                                        REMOVEDINRELEASE   REQUESTSINCURRENTHOUR
REQUESTSINLAST24H
cloudcredentials.v1.operator.openshift.io                       32                  111
ingresses.v1.networking.k8s.io                                  28                  110
ingresses.v1beta1.extensions                1.22                16                  66
ingresses.v1beta1.networking.k8s.io         1.22                0                   1
installplans.v1alpha1.operators.coreos.com                      93                  167
...
```

重要

您可以安全地忽略结果中出现的以下条目：

- **system:serviceaccount:kube-system:generic-garbage-collector** 和 **system:serviceaccount:kube-system:namespace-controller** 用户可能会出现在结果中，因为这些服务在搜索要删除的资源时调用所有注册的 API。
- **system:kube-controller-manager** 和 **system:cluster-policy-controller** 用户可能会出现在结果中，因为它们在强制执行各种策略时遍历所有资源。

您还可以使用 **-o jsonpath** 来过滤结果：

```
$ oc get apirequestcounts -o jsonpath='{range .items[?(@.status.removedInRelease!="")]}
{.status.removedInRelease}{"\t"}{.metadata.name}{"\n"}{end}'
```

输出示例

```
1.22    certificatesigningrequests.v1beta1.certificates.k8s.io
1.22    ingresses.v1beta1.extensions
1.22    ingresses.v1beta1.networking.k8s.io
```

4.2.3. 使用 APIRequestCount 来识别哪些工作负载正在使用删除的 API

您可以检查给定 API 版本的 **APIRequestCount** 资源，以帮助识别正在使用 API 的工作负载。

先决条件

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

流程

- 运行以下命令并检查 **username** 和 **userAgent** 字段，以帮助识别使用 API 的工作负载：

```
$ oc get apirequestcounts <resource>.<version>.<group> -o yaml
```

例如：

```
$ oc get apirequestcounts ingresses.v1beta1.networking.k8s.io -o yaml
```

您还可以使用 **-o jsonpath** 从 **APIRequestCount** 资源中提取 **username** 和 **userAgent** 值：

```
$ oc get apirequestcounts ingresses.v1beta1.networking.k8s.io \
  -o jsonpath='{range .status.currentHour..byUser[*]}{..byVerb[*].verb}{","}{.username}{","}
{.userAgent}{"\n"}{end}' \
  | sort -k 2 -t, -u | column -t -s, -NVERBS,USERNAME,USERAGENT
```

输出示例

```
VERBS  USERNAME                    USERAGENT
watch  bob                         oc/v4.8.11
watch  system:kube-controller-manager  cluster-policy-controller/v0.0.0
```

4.3. 迁移已删除 API 实例

有关如何迁移删除 Kubernetes API 的详情，请参阅 Kubernetes 文档中的已弃用 API 迁移指南。

4.4. 管理员确认

在为任何已删除的 API 评估集群并迁移了任何删除 API 后，您可以确认集群已准备好从 OpenShift Container Platform 4.8 升级到 4.9。

警告

请注意，管理员须负责确保已移除 API 的所有使用都已根据需要得到解决和迁移，然后再提供此管理员的确认。OpenShift Container Platform 可以协助评估，但无法识别所有可能移除的 API 的使用，特别是空闲的工作负载或外部工具。

先决条件

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

流程

- 运行以下命令确认您已完成评估，并且集群已准备好升级到 OpenShift Container Platform 4.9：

```
$ oc -n openshift-config patch cm admin-acks --patch '{"data":{"ack-4.8-kube-1.22-api-
removals-in-4.9":"true"}}' --type=merge
```

第 5 章 准备执行 EUS 更新

由于 Kubernetes 的设计，次版本之间的所有 OpenShift Container Platform 升级都必须按顺序进行。您必须从 OpenShift Container Platform 4.8 更新至 4.9，然后升级到 4.10。您无法直接从 OpenShift Container Platform 4.8 更新至 4.10。但是，从 OpenShift Container Platform 4.8 升级到 4.9 再升级到 4.10 开始，希望在两个延长更新支持(EUS)版本间升级的管理员可以进行这样的升级，非 master 主机只需要重启一次。

当尝试进行 EUS 到 EUS 更新时需要考虑很多注意事项。

- EUS 到 EUS 更新仅在所有涉及的版本在 **stable** 频道中提供。
- 如果您在升级到一个奇数次版本后但在升级到下一个偶数次版本前遇到了问题，则需要在继续进行前，把非控制平面主机完成升级到奇数次版本。
- 您可以在多个维护窗口期间通过暂停中间步骤完成升级过程。但是，计划在 60 天内完成整个更新。确保完成正常的集群自动化过程非常重要，包括与证书轮转关联的操作。
- 开始 EUS 到 EUS 升级过程前，您必须至少运行 OpenShift Container Platform 4.8.14。如果您没有满足这个最低要求，在尝试 EUS 升级到 EUS 前，升级到更新的 4.8.z。
- OpenShift Container Platform 4.10 中删除了对 RHEL7 worker 的支持，并使用 RHEL8 worker 替代，因此使用 RHEL7 worker 的集群没有 EUS 到 EUS 的升级。
- 节点组件没有更新至 OpenShift Container Platform 4.9。在完成升级到 OpenShift Container Platform 4.10 前，不要预期 OpenShift Container Platform 4.9 中修复的所有功能和错误都可用，并启用所有 MachineConfigPool 进行更新。
- 所有集群可能会在不需要暂停池的情况下，对常规更新使用 EUS 频道进行更新，但只有具有非控制平面 **MachineConfigPools** 对象的集群可以进行 EUS 到 EUS 的更新，且需要暂停池。

5.1. EUS 到 EUS 更新

以下流程暂停所有非 master MachineConfigPools，并从 OpenShift Container Platform 4.8 升级到 4.9 再升级到 4.10，然后取消暂停之前暂停的 MachineConfigPool。以下过程减少了升级持续时间，以及重启 worker 节点的次数。

先决条件

- 查看 OpenShift Container Platform 4.9 和 4.10 发行注记
- 查阅有关任何层次产品和 OLM Operator 的发行注记和产品生命周期。有些操作可能需要在 EUS 到 EUS 更新之前或进行中进行。
- 确保您熟悉了特定于版本的前提条件，如在从 OpenShift Container Platform 4.8 升级到 4.9 之前需要管理员进行确认。
- 验证集群是否正在运行 OpenShift Container Platform 版本 4.8.14 或更高版本。如果您的集群运行早于 OpenShift Container Platform 4.8.14 的版本，则必须在升级到 4.9 前升级到更新的 4.8.z 版本。需要升级到 4.8.14 或更高版本，才能满足最低版本要求，且无需暂停 MachineConfigPool。
- 验证 MachineConfigPools 是否已取消暂停。

流程

1. 将任何 OLM Operator 升级到与您要升级到的版本兼容的版本。
2. 验证所有 MachineConfigPools 均显示 **UPDATED** 状态，没有 MachineConfigPools 显示 **UPDATING** 状态。要查看所有 MachineConfigPools 的状态，请运行以下命令：

```
$ oc get mcp
```

输出示例

为清晰起见，输出已被修剪：

```
NAME     CONFIG                                             UPDATED   UPDATING
master   rendered-master-ecbb9582781c1091e1c9f19d50cf836c       True     False
worker   rendered-worker-00a3f0c68ae94e747193156b491553d5       True     False
```

3. 暂停您要跳过重启的 MachineConfigPools，运行以下命令：

注意

您无法暂停 master 池。

```
$ oc patch mcp/worker --type merge --patch '{"spec":{"paused":true}}'
```

4. 切换到 **eus-4.10** 频道，运行以下命令：

```
$ oc adm upgrade channel eus-4.10
```

5. 更新至 4.9，运行以下命令：

```
$ oc adm upgrade --to-latest
```

输出示例

```
Updating to latest version 4.9.18
```

6. 运行以下命令，查看集群版本以确保更新已完成：

```
$ oc get clusterversion
```

输出示例

```
NAME     VERSION  AVAILABLE  PROGRESSING  SINCE   STATUS
version  4.9.18   True       False        6m29s   Cluster version is 4.9.18
```

7. 如有必要，使用 web 控制台中的 Administrator 视角升级 OLM operator。
8. 更新至 4.10，运行以下命令：

```
$ oc adm upgrade --to-latest
```

9. 运行以下命令，查看集群版本以确保更新已完成：

```
$ oc get clusterversion
```

输出示例

```
NAME     VERSION  AVAILABLE  PROGRESSING   SINCE   STATUS
version  4.10.1   True       False         6m29s   Cluster version is 4.10.1
```

10. 取消暂停所有之前暂停的 MachineConfigPools，运行以下命令：

```
$ oc patch mcp/worker --type merge --patch '{"spec":{"paused":false}}'
```

注意

如果没有取消暂停池，集群不允许升级到将来的次要维护任务，如证书轮转。这会使集群面临未来降级的风险。

11. 验证之前暂停的池是否已更新，集群完成了更新到 4.10，运行以下命令：

```
$ oc get mcp
```

输出示例

为清晰起见，输出已被修剪：

```
NAME     CONFIG                                             UPDATED     UPDATING
master   rendered-master-52da4d2760807cb2b96a3402179a9a4c    True    False
worker   rendered-worker-4756f60eccae96fb9dcb4c392c69d497    True    False
```

第 6 章 使用 WEB 控制台更新集群

您可以使用 web 控制台对 OpenShift Container Platform 集群进行更新或升级。以下步骤在次版本中更新集群。您可以使用相同的说明在次版本之间更新集群。

注意

使用 Web 控制台或 **oc adm upgrade channel *<channel>*** 更改更新频道。在改到一个 4.9 频道后，按使用 CLI 更新集群中的步骤完成更新。

6.1. 先决条件

- 使用具有 **admin** 权限的用户访问集群。请参阅使用 RBAC 定义和应用权限。
- 具有最新的 etcd 备份，以防因为升级失败需要将集群恢复到以前的状态。
  OpenShift Container Platform 4.9 需要从 etcd 版本 3.4 升级到 3.5。如果 etcd Operator 停止更新，则会触发警报。要清除此警报，请使用以下命令取消更新：

  ```
  $ oc adm upgrade --clear
  ```

- 确保之前通过 Operator Lifecycle Manager（OLM）安装的所有 Operator 均更新至其最新频道中的最新版本。更新 Operator 可确保当默认 OperatorHub 目录在集群升级过程中从当前次要版本切换到下一个次版本时，它们有有效的升级路径。如需更多信息，请参阅更新安装的 Operator。
- 确保所有机器配置池 (MCP) 都正在运行且未暂停。在更新过程中跳过与暂停 MCP 关联的节点。如果要执行 canary rollout 更新策略，可以暂停 MCP。
- 如果您的集群使用手动维护的凭证，请确保 Cloud Credential Operator（CCO）处于可升级状态。如需更多信息，请参阅为 AWS、Azure 或 GCP *手动维护凭证升级集群*。
- 如果您运行 Operator 或您已配置了 pod 中断预算，您可能会在升级过程中遇到中断。如果在 **PodDisruptionBudget** 中将 **minAvailable** 设置为 1，则节点会排空以应用可能会阻止驱除过程的待处理机器配置。如果重启了几个节点，则所有 pod 只能有一个节点上运行，**PodDisruptionBudget** 字段可能会阻止节点排空。
- 如果您的集群通过 AWS Secure Token Service (STS) 使用手动维护的凭证，请从要升级到的发行镜像获取 **ccoctl** 实用程序的副本，并使用它来处理任何更新的凭证。如需更多信息，请参阅 *升级使用 STS 的手动模式配置的 OpenShift Container Platform 集群*。
- 检查 Kubernetes 1.22 中删除的 API 列表，将任何受影响的组件迁移到使用新的 API 版本，并提供管理员确认。如需更多信息，请参阅准备升级到 OpenShift Container Platform 4.9。

重要

- 当更新无法完成时，Cluster Version Operator(CVO)会在尝试协调更新时报告任何阻塞组件的状态。当前还不支持将集群还原到以前的版本。如果您的更新无法完成，请联系红帽支持。
- 使用 **unsupportedConfigOverrides** 部分修改 Operator 配置不受支持，并可能会阻止集群更新。您必须在更新集群前删除此设置。

其他资源

- 非受管 Operator 的支持策略

6.2. 执行 CANARY ROLLOUT 更新

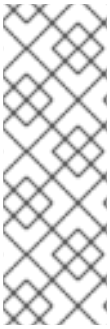
在某些情况下，您可能需要一个受控的更新过程，如您不希望特定节点与集群的其余部分同时更新。这些用例包括但不限于：

- 您有任务关键型应用程序，您希望在更新过程中仍然可以使用这些应用程序。在更新后，您可以慢慢地以小批的形式对应用进行测试。
- 您有一个短的维护窗口，在此期间不允许更新所有节点；或者您有多个维护窗口。

滚动更新过程**不是**典型的更新工作流。对于较大的集群，这可能是一个耗时的过程，需要您执行多个命令。这种复杂性可能会导致出现可能会影响整个集群的错误。建议您仔细考虑您的机构是否希望使用滚动更新，并在开始前仔细规划流程的实施。

本主题中描述的滚动更新过程涉及：

- 创建一个或多个自定义机器配置池 (MCP)。
- 标记您不想立即更新的每个节点，以将这些节点移至自定义 MCP。
- 暂停这些自定义 MCP，这会阻止对这些节点的更新。
- 执行集群更新。
- 取消暂停一个自定义 MCP，它会在这些节点上触发更新。
- 测试这些节点上的应用程序，以确保应用程序在这些新更新的节点上可以正常工作。
- （可选）从小批处理中的其余节点移除自定义标签，并在这些节点上测试应用。



注意

暂停 MCP 可防止 Machine Config Operator 在关联的节点上应用任何配置更改。暂停 MCP 还可以防止任何自动轮转的证书被推送到关联的节点，包括自动轮转 **kube-apiserver-to-kubelet-signer** CA 证书。如果 **kube-apiserver-to-kubelet-signer** CA 证书过期且 MCO 尝试自动更新证书时，MCP 会暂停，但不会在相应机器配置池中的节点中应用新证书。这会导致多个 **oc** 命令失败，包括但不限于 **oc debug**、**oc logs**、**oc exec** 和 **oc attach**。在暂停 MCP 时应该非常小心，需要仔细考虑 **kube-apiserver-to-kubelet-signer** CA 证书过期的问题，且仅在短时间内暂停。

如果要使用 Canary rollout 更新过程，请参阅执行 Canforming rollout 更新。

6.3. 使用手动维护的凭证升级集群

默认情况下，带有手动维护凭证的集群的 Cloud Credential Operator(CCO)U **gradable 状态为 False**。

- 对于次发行版本（例如从 4.8 升级到 4.9），这个状态会阻止升级，直到您解决了任何更新的权限并 **添加了 CloudCredential** 资源，以指示下一版本根据需要更新权限。此注解将 **Upgradable** 状态更改为 **True**。
- 对于 z-stream 版本（例如从 4.9.0 到 4.9.1），没有添加或更改任何权限，因此不会阻止升级。

在使用手动维护凭证升级集群前，您必须为要升级到的发行镜像创建新凭证。另外，您必须检查现有凭证所需的权限，并满足新版本中这些组件的任何新权限要求。

流程

1. 提取并检查 **新版本的 CredentialsRequest** 自定义资源。
详情请参阅您的云供应商的"手动创建 IAM"部分来了解如何获取和使用您的云所需的凭证。

2. 更新集群中手动维护的凭证：

   - 为新发行镜像添加的任何 **CredentialsRequest** 自定义资源创建新 secret。
   - 如果存储在 secret 中的任何现有凭证的 **CredentialsRequest** 自定义资源更改了其权限要求，请根据需要更新权限。

3. 当所有 secret 都对新发行版本正确时，表示集群已准备好升级：

   a. 以具有 **cluster-admin** 角色的用户身份登录 OpenShift Container Platform CLI。

   b. 编辑 **CloudCredential** 资源，以在 **metadata** 字段中添加 **可升级至** 注解：

   ```
   $ oc edit cloudcredential cluster
   ```

   要添加的文本

   ```
   ...
     metadata:
       annotations:
         cloudcredential.openshift.io/upgradeable-to: <version_number>
   ...
   ```

   其中 **<version_number>** 是您要升级到的版本，格式为 **x.y.z**。例如，OpenShift Container Platform **4.8.2** 代表 OpenShift Container Platform 4.8.2。

   添加可升级状态进行更改的注解后，可能需要几分钟时间。

4. 验证 CCO 是否可升级：

   a. 在 Web 控制台的 **Administrator 视角中**，导航到 **Administration → Cluster Settings**。

   b. 要查看 CCO 状态详情，请点击 **Cluster Operators** 列表中的 **cloud-credential**。

   c. 如果 **Conditions** 部分中的 **Upgradeable** 状态为 **False**，请验证 **upgradeable-to** 注解没有拼写错误。

当 **Conditions** 部分中的 **Upgradeable** 状态为 **True** 时，您可以开始 OpenShift Container Platform 升级。

其他资源

- 为 AWS 手动创建 IAM
- 为 Azure 手动创建 IAM
- 为 GCP 手动创建 IAM

6.4. 使用 WEB 控制台暂停 MACHINEHEALTHCHECK 资源

在升级过程中，集群中的节点可能会临时不可用。对于 worker 节点，机器健康检查可能会认为这样的节点不健康，并重新引导它们。为避免重新引导这样的节点，请在更新集群前暂停所有 **MachineHealthCheck** 资源。

先决条件

- 您可以使用 **cluster-admin** 权限访问集群。
- 访问 OpenShift Container Platform web 控制台。

流程

1. 登陆到 OpenShift Container Platform Web 控制台。
2. 进入到 **Compute → MachineHealthChecks**。
3. 要暂停机器健康检查，请在每个 **MachineHealthCheck** 资源中添加 **cluster.x-k8s.io/paused=""** 注解。例如，要将注解添加到 **machine-api-termination-handler** 资源，请完成以下步骤：
   a. 点 **machine-api-termination-handler** 旁边的 Options 菜单  并点 **Edit annotations**。
   b. 在 **Edit annotations** 对话框中，点 **Add more**。
   c. 在 **Key** 和 **Value** 字段中，分别添加 **cluster.x-k8s.io/paused** 和 **""** 值，然后点 **Save**。

6.5. 关于更新单个节点 OPENSHIFT CONTAINER PLATFORM

您可以使用控制台或 CLI 更新或升级单节点 OpenShift Container Platform 集群。

但请注意以下限制：

- 不需要暂停 **MachineHealthCheck** 资源，因为没有其他节点可以执行健康检查。
- 不支持使用 etcd 备份来恢复单节点 OpenShift Container Platform 集群。但是，最好在升级失败时执行 etcd 备份。如果 control plane 健康，您可以使用备份将集群恢复到以前的状态。
- 更新单节点 OpenShift Container Platform 集群需要停机，并可包括自动重启。停机时间取决于更新有效负载，如下例所示：
  - 如果更新有效负载包含操作系统更新（需要重启），则停机时间会非常显著，并影响集群管理和用户工作负载。
  - 如果更新包含不需要重启的机器配置更改，则停机时间会减少，并且对集群管理和用户工作负载的影响会减少。在这种情况下，节点排空步骤会通过单节点 OpenShift Container Platform 跳过，因为集群中没有其他节点可以重新调度工作负载。
  - 如果更新有效负载不包含操作系统更新或机器配置更改，则会出现简短的 API 中断并快速解决。

重要

某些条件（如更新的软件包中的错误）可能会导致单个节点在重启后无法重启。在这种情况下，更新不会自动回滚。

其他资源

- 有关哪些机器配置更改需要重新引导的信息，请参阅 了解 Machine Config Operator 中的备注。

6.6. 使用WEB控制台更新集群

如果有可用更新，您可以从Web控制台更新集群。

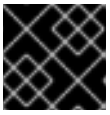
您可以在客户门户网站的勘误部分找到有关可用 OpenShift Container Platform 公告和更新的信息。

先决条件

- 使用具有 **admin** 权限的用户登陆到 web 控制台。
- 暂停所有 **MachineHealthCheck** 资源。

流程

1. 在 web 控制台中点击 **Administration → Cluster Settings** 并查看 **Details** 选项卡中的内容。
2. 对于生产环境中的集群，请确保将 **Channel** 设置为您要升级到的版本的正确频道，如 **stable-4.9**。



重要

对于生产环境中的集群，您必须订阅一个 **stable-***, **eus-*** 或 **fast-*** 频道。

- 如果 **Update 状态** 不是 **Updates available**，则无法升级集群。
- **Select channel** 表示集群正在运行或正在更新的集群版本。

3. 选择要更新到的版本，然后单击 **Save**。
输入频道 **Update Status** 变为**Update to <product-version> in progress**，您可以通过监视 Operator 和节点的进度条来查看集群更新的进度。

注意

如果您要将集群升级到下一个次版本，如从 4.y 升级到 4.(y+1)，建议在部署依赖新功能的工作负载前确认您的节点已升级：任何尚未更新的 worker 节点池都会显示在 **Cluster Settings** 页面。

4. 更新完成后，Cluster Version Operator 会刷新可用更新，检查当前频道中是否有更多可用更新。
    - 如果有可用更新，请继续在当前频道中执行更新，直到您无法再更新为止。
    - 如果没有可用的更新，请将 **Channel** 改为下一个次版本的 **stable-***, **eus-*** 或 **fast-*** 频道，并更新至您在该频道中想要的版本。

    您可能需要执行一些过渡的更新，直到您到达您想要的版本。

6.7. 使用 WEB 控制台更改更新服务器

更改更新服务器是可选的。如果您在本地安装和配置了 OpenShift Update Service (OSUS)，您必须将服务器的 URL 设置为 **upstream**，以便在更新期间使用本地服务器。

流程

1. 导航到 **Administration → Cluster Settings**，点 **version**。
2. 点击 **YAML** 选项卡，然后编辑 **upstream** 参数值：

   输出示例

   ```
   ...
   spec:
     clusterID: db93436d-7b05-42cc-b856-43e11ad2d31a
     upstream: '<update-server-url>' 1
   ...
   ```

   1 **<update-server-url>** 变量指定更新服务器的 URL。

   默认 **upstream** 是 **https://api.openshift.com/api/upgrades_info/v1/graph**。

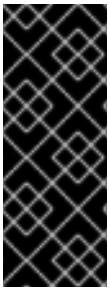
3. 点 **Save**。

第 7 章 使用 CLI 更新集群

您可以使用 OpenShift CLI (**oc**) 将 OpenShift Container Platform 集群更新或升级到一个次版本。您还可以按照相同的说明在次版本间更新集群。

7.1. 先决条件

- 使用具有 **admin** 权限的用户访问集群。请参阅使用 RBAC 定义和应用权限。
- 具有最新的 etcd 备份，以防因为升级失败需要将集群恢复到以前的状态。
- 确保之前通过 Operator Lifecycle Manager（OLM）安装的所有 Operator 均更新至其最新频道中的最新版本。更新 Operator 可确保当默认 OperatorHub 目录在集群升级过程中从当前次要版本切换到下一个次版本时，它们有有效的升级路径。如需更多信息，请参阅更新安装的 Operator。
- 如果您的集群使用手动维护的凭证，请确保 Cloud Credential Operator（CCO）处于可升级状态。如需更多信息，请参阅为 AWS、Azure 或 GCP *手动维护凭证升级集群*。
- 如果您的集群通过 AWS Secure Token Service (STS) 使用手动维护的凭证，请从要升级到的发行镜像获取 **ccoctl** 实用程序的副本，并使用它来处理任何更新的凭证。如需更多信息，请参阅*升级使用 STS 的手动模式配置的 OpenShift Container Platform 集群*。
- 确保解决所有 **Upgradeable=False** 条件，以便集群可以更新到下一个次版本。您可以运行 **oc adm upgrade** 命令以了解所有 **Upgradeable=False** 条件的输出，以及帮助您为次版本更新做准备的条件。
- 如果您运行 Operator 或您已配置了 pod 中断预算，您可能会在升级过程中遇到中断。如果在 **PodDisruptionBudget** 中将 **minAvailable** 设置为 1，则节点会排空以应用可能会阻止驱除过程的待处理机器配置。如果重启了几个节点，则所有 pod 只能有一个节点上运行，**PodDisruptionBudget** 字段可能会阻止节点排空。



重要

- 当更新无法完成时，Cluster Version Operator(CVO)会在尝试协调更新时报告任何阻塞组件的状态。当前还不支持将集群还原到以前的版本。如果您的更新无法完成，请联系红帽支持。
- 使用 **unsupportedConfigOverrides** 部分修改 Operator 配置不受支持，并可能会阻止集群更新。您必须在更新集群前删除此设置。

其他资源

- 非受管 Operator 的支持策略

7.2. 暂停 MACHINEHEALTHCHECK 资源

在升级过程中，集群中的节点可能会临时不可用。对于 worker 节点，机器健康检查可能会认为这样的节点不健康，并重新引导它们。为避免重新引导这样的节点，请在更新集群前暂停所有 **MachineHealthCheck** 资源。

先决条件

- 安装 OpenShift CLI (**oc**) 。

流程

1. 要列出您要暂停的所有可用 **MachineHealthCheck** 资源，请运行以下命令：

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

2. 要暂停机器健康检查，请将 **cluster.x-k8s.io/paused=""** 注解添加到 **MachineHealthCheck** 资源。运行以下命令:

```
$ oc -n openshift-machine-api annotate mhc <mhc-name> cluster.x-k8s.io/paused=""
```

注解的 **MachineHealthCheck** 资源类似以下 YAML 文件：

```
apiVersion: machine.openshift.io/v1beta1
kind: MachineHealthCheck
metadata:
  name: example
  namespace: openshift-machine-api
  annotations:
    cluster.x-k8s.io/paused: ""
spec:
  selector:
    matchLabels:
      role: worker
  unhealthyConditions:
  - type:    "Ready"
    status:  "Unknown"
    timeout: "300s"
  - type:    "Ready"
    status:  "False"
    timeout: "300s"
  maxUnhealthy: "40%"
status:
  currentHealthy: 5
  expectedMachines: 5
```

重要

更新集群后恢复机器健康检查。要恢复检查，请运行以下命令从 **MachineHealthCheck** 资源中删除暂停注解：

```
$ oc -n openshift-machine-api annotate mhc <mhc-name> cluster.x-k8s.io/paused-
```

7.3. 关于更新单个节点 OPENSHIFT CONTAINER PLATFORM

您可以使用控制台或 CLI 更新或升级单节点 OpenShift Container Platform 集群。

但请注意以下限制：

- 不需要暂停 **MachineHealthCheck** 资源，因为没有其他节点可以执行健康检查。
- 不支持使用 etcd 备份来恢复单节点 OpenShift Container Platform 集群。但是，最好在升级失败时执行 etcd 备份。如果 control plane 健康，您可以使用备份将集群恢复到以前的状态。

- 更新单节点 OpenShift Container Platform 集群需要停机，并可包括自动重启。停机时间取决于更新有效负载，如下例所示：
  - 如果更新有效负载包含操作系统更新（需要重启），则停机时间会非常显著，并影响集群管理和用户工作负载。
  - 如果更新包含不需要重启的机器配置更改，则停机时间会减少，并且对集群管理和用户工作负载的影响会减少。在这种情况下，节点排空步骤会通过单节点 OpenShift Container Platform 跳过，因为集群中没有其他节点可以重新调度工作负载。
  - 如果更新有效负载不包含操作系统更新或机器配置更改，则会出现简短的 API 中断并快速解决。

重要

某些条件（如更新的软件包中的错误）可能会导致单个节点在重启后无法重启。在这种情况下，更新不会自动回滚。

其他资源

- 有关哪些机器配置更改需要重新引导的信息，请参阅 了解 Machine Config Operator 中的备注。

7.4. 使用 CLI 更新集群

如果有可用更新，您可以使用OpenShift CLI (**oc**)更新集群。

您可以在客户门户网站的勘误部分找到有关可用 OpenShift Container Platform 公告和更新的信息。

先决条件

- 安装与更新版本的版本匹配的 OpenShift CLI(**oc**)。
- 使用具有 **cluster-admin** 权限的用户登陆到集群。
- 安装 **jq** 软件包。
- 暂停所有 **MachineHealthCheck** 资源。

流程

1. 确认集群可用：

```
$ oc get clusterversion
```

输出示例

```
NAME     VERSION   AVAILABLE   PROGRESSING   SINCE   STATUS
version  4.8.13    True        False         158m    Cluster version is 4.8.13
```

2. 查看当前的更新频道信息，并确认您的频道已设置为 **stable-4.9**:

```
$ oc get clusterversion -o json|jq ".items[0].spec"
```

输出示例

```
{
  "channel": "stable-4.9",
  "clusterID": "990f7ab8-109b-4c95-8480-2bd1deec55ff"
}
```

重要

对于生产环境中的集群，您必须订阅一个 **stable-***, **eus-*** 或 **fast-*** 频道。

3. 查看可用更新，记录下要应用的更新的版本号：

```
$ oc adm upgrade
```

输出示例

```
Cluster version is 4.8.13

Updates:

VERSION IMAGE
4.9.0   quay.io/openshift-release-dev/ocp-release@sha256:9c5f0df8b192a0d7b46cd5f6a4da2289c155fd5302dec7954f8f06c878160b8b
```

4. 应用更新：

- 要更新到最新版本：

```
$ oc adm upgrade --to-latest=true (1)
```

- 要更新到一个特定版本：

```
$ oc adm upgrade --to=<version> (1)
```

(1) **<version>**是从上一个命令输出中获取的更新版本。

5. 查看 Cluster Version Operator 的状态：

```
$ oc get clusterversion -o json|jq ".items[0].spec"
```

输出示例

```
{
  "channel": "stable-4.9",
  "clusterID": "990f7ab8-109b-4c95-8480-2bd1deec55ff",
  "desiredUpdate": {
    "force": false,
    "image": "quay.io/openshift-release-dev/ocp-release@sha256:9c5f0df8b192a0d7b46cd5f6a4da2289c155fd5302dec7954f8f06c878160b8b",
```

```
    "version": "4.9.0" (1)
  }
}
```

(1) 如果**desiredUpdate**中的**version**值与您指定的值匹配，则更新正在进行中。

6. 查看集群版本状态历史记录以监控更新的状态。这可能需要一些时间才能完成对所有对象的更新。

```
$ oc get clusterversion -o json|jq ".items[0].status.history"
```

输出示例

```
[
  {
    "completionTime": null,
    "image": "quay.io/openshift-release-dev/ocp-release@sha256:b8fa13e09d869089fc5957c32b02b7d3792a0b6f36693432acc0409615ab23b7",
    "startedTime": "2021-01-28T20:30:50Z",
    "state": "Partial",
    "verified": true,
    "version": "4.9.0"
  },
  {
    "completionTime": "2021-01-28T20:30:50Z",
    "image": "quay.io/openshift-release-dev/ocp-release@sha256:b8fa13e09d869089fc5957c32b02b7d3792a0b6f36693432acc0409615ab23b7",
    "startedTime": "2021-01-28T17:38:10Z",
    "state": "Completed",
    "verified": false,
    "version": "4.8.13"
  }
]
```

历史记录包含了应用于集群的最新版本的列表。当CVO应用更新时，此值将会被相应更新。该列表按日期排序，最新的更新会在列表中第一个显示。如果历史信息中的更新状态为 **Completed**，则表示部署已完成；如果状态为 **Partial**，则表示更新失败或还未完成。

7. 更新完成后，可以通过以下方法确认集群已更新为新版本：

```
$ oc get clusterversion
```

输出示例

```
NAME      VERSION   AVAILABLE   PROGRESSING   SINCE     STATUS
version   4.9.0     True        False         2m        Cluster version is 4.9.0
```

注意

如果 **oc get clusterversion** 命令在 **PROGRESSING** 状态为 **True** 时显示以下错误，您可以忽略这个错误。

```
NAME    VERSION AVAILABLE PROGRESSING SINCE STATUS
version 4.10.26 True      True        24m   Unable to apply 4.11.0-rc.7: an
unknown error has occurred: MultipleErrors
```

8. 如果您要将集群升级到下一个次版本，如从 4.y 升级到 4.(y+1)，建议在部署依赖新功能的工作负载前确认您的节点已升级：

```
$ oc get nodes
```

输出示例

```
NAME                         STATUS   ROLES    AGE   VERSION
ip-10-0-168-251.ec2.internal   Ready    master   82m   v1.22.1
ip-10-0-170-223.ec2.internal   Ready    master   82m   v1.22.1
ip-10-0-179-95.ec2.internal    Ready    worker   70m   v1.22.1
ip-10-0-182-134.ec2.internal   Ready    worker   70m   v1.22.1
ip-10-0-211-16.ec2.internal    Ready    master   82m   v1.22.1
ip-10-0-250-100.ec2.internal   Ready    worker   69m   v1.22.1
```

7.5. 使用 CLI 更改更新服务器

更改更新服务器是可选的。如果您在本地安装和配置了 OpenShift Update Service (OSUS)，您必须将服务器的 URL 设置为 **upstream**，以便在更新期间使用本地服务器。**upstream** 的默认值是 **https://api.openshift.com/api/upgrades_info/v1/graph**。

流程

- 更改集群版本中的 **upstream** 参数值：

```
$ oc patch clusterversion/version --patch '{"spec":{"upstream":"<update-server-url>"}}' --
type=merge
```

<update-server-url> 变量指定更新服务器的 URL。

输出示例

```
clusterversion.config.openshift.io/version patched
```

第 8 章 执行 CANARY ROLLOUT 更新

有些情况下，您可能希望对 worker 节点进行更多受控的更新推出，以确保任务关键型应用程序在更新过程中仍然可用，即使更新过程导致您的应用程序失败。根据您的机构需求，您可能需要更新一小部分 worker 节点，在一个时间段内评估集群和工作负载健康状况，然后更新剩余的节点。这通常被称为 *Canary* 更新。或者，您可能还希望将通常需要主机重新引导的 worker 节点更新放入较小的定义的维护窗口（不可能一次使用大型维护窗口来更新整个集群）。

在这些情况下，您可以创建多个自定义机器配置池 (MCP)，以防止某些 worker 节点在更新集群时进行更新。在更新剩余的集群后，您可以在适当的时间批量更新这些 worker 节点。

例如，如果您的集群有 100 个节点，且有 10% 过量的容量，维护窗口不能超过 4 小时，并且您知道排空和重新引导 worker 节点的时间不超过 8 分钟，您可以利用 MCP 来实现您的目标。例如，您可以定义四个 MCP，分别名为 **workerpool-canary**、**workerpool-A**、**workerpool-B** 和 **workerpool-C**，分别有 10、30、30 和 30 个节点。

在第一次维护窗口中，您将暂停 **workerpool-A**、**workerpool-B** 和 **workerpool-C** 的 MCP，然后启动集群更新。这会更新在 OpenShift Container Platform 上运行的组件，以及属于 **workerpool-canary MCP** 成员的 10 个节点，因为这个池没有暂停。其他三个 MCP 不会更新，因为它们已暂停。如果出于某种原因，您确定集群或工作负载健康受到 **workerpool-canary** 更新的负面影响，那么在分析完问题前，您会在保持足够容量的同时，对那个池中的所有节点进行 cordon 和 drain 操作。当一切正常时，您将在决定取消暂停前评估集群和工作负载健康状况，从而在每个额外的维护窗口中持续更新 **workerpool-A**、**workerpool-B** 和 **workerpool-C**。

使用自定义 MCP 管理 worker 节点更新提供了灵活性，这可能是一个耗时的过程，需要您执行多个命令。这种复杂性可能会导致出现可能会影响整个集群的错误。建议您仔细考虑您的组织需求，并在开始之前仔细规划流程的实施。

注意

不建议将 MCP 更新至不同的 OpenShift Container Platform 版本。例如，请勿将一个 MCP 从 4.y.10 更新至 4.y.11，另一个更新为 4.y.12。这个场景还没有被测试，可能会导致未定义的集群状态。

重要

暂停机器配置池可防止 Machine Config Operator 在关联的节点上应用任何配置更改。暂停 MCP 还可以防止任何自动轮转的证书被推送到关联的节点，包括自动轮转 **kube-apiserver-to-kubelet-signer** CA 证书。如果 **kube-apiserver-to-kubelet-signer** CA 证书过期且 MCO 尝试自动更新证书时，MCP 会暂停，但不会在相应机器配置池中的节点中应用新证书。这会导致多个 **oc** 命令失败，包括但不限于 **oc debug**、**oc logs**、**oc exec** 和 **oc attach**。在暂停 MCP 时应该非常小心，需要仔细考虑 **kube-apiserver-to-kubelet-signer** CA 证书过期的问题，且仅在短时间内暂停。

8.1. 关于 CANARY ROLLOUT 更新过程和 MCP

在 OpenShift Container Platform 中，节点不会被单独考虑。节点分组到机器配置池中 (MCP)。默认 OpenShift Container Platform 集群中有两个 MCP：一个用于 control plane 节点，一个用于 worker 节点。OpenShift Container Platform 更新会同时影响所有 MCP。

在更新过程中，Machine Config Operator (MCO) 会排空并记录 MCP 中的所有节点，直至指定的 **maxUnavailable** 数量（如果指定），默认为 **1**。排空节点取消调度节点上的所有 pod，并将该节点标记为不可调度。节点排空后，Machine Config Daemon 应用一个新的机器配置，其中包括更新操作系统 (OS)。更新操作系统需要主机重新引导。

要防止特定节点被更新，且不会排空、进行 cordoned 和更新，您可以创建自定义 MCP。然后，暂停这些 MCP，以确保不更新与这些 MCP 关联的节点。MCO 不会更新任何暂停的 MCP。您可以创建一个或多个自定义 MCP，这样可以让您更好地控制您更新这些节点的顺序。更新第一个 MCP 中的节点后，您可以验证应用兼容性，然后逐步将其余节点更新至新版本。

注意

为确保 control plane 的稳定性，不支持从 control plane 节点创建自定义 MCP。Machine Config Operator (MCO) 会忽略为 control plane 节点创建的任何自定义 MCP。

根据工作负载部署拓扑，您应该仔细考虑您创建的 MCP 数以及每个 MCP 中的节点数量。例如，如果需要将更新融入特定的维护窗口，您需要知道 OpenShift Container Platform 可在窗口中更新的节点数量。这个数字取决于您具体的集群和工作负载特性。

另外，您需要考虑集群中可用的额外容量量。例如，当应用程序无法在更新的节点上按预期工作时，您可以对池中那些节点进行 cordon 和 drain 操作，这会将应用 pod 移到其他节点上。您需要考虑您可用的额外容量数量，以确定您需要的自定义 MCP 数量以及每个 MCP 中有多少节点。例如，如果您使用两个自定义 MCP 和 50% 的节点在每个池中，则需要确定运行 50% 的节点是否能为您的应用程序提供足够的服务质量(QoS)。

您可以将这个更新过程与所有记录的 OpenShift Container Platform 更新过程一起使用。但是，该过程不适用于使用 Ansible playbook 进行更新的 Red Hat Enterprise Linux (RHEL) 机器。

8.2. 关于执行 CANARY ROLLOUT 更新

本主题描述了此 canary rollout 更新过程的一般工作流。以下小节介绍了工作流中执行每项任务的步骤。

1. 根据 worker 池创建 MCP。每个 MCP 中的节点数量取决于几个因素，如每个 MCP 的维护窗口持续时间以及集群中可用的保留容量（即额外的 worker 节点）。

   注意

   您可以更改 MCP 中的 **maxUnavailable** 设置，以指定在任意给定时间可以更新的机器的百分比或数量。默认值为 1。

2. 将节点选择器添加到自定义 MCP。对于您不想与剩余的集群同时更新的每个节点，请向节点添加匹配的标签。该标签将节点与 MCP 相关联。

   注意

   不要从节点中删除默认 worker 标签。节点 **必须具有** role 标签才能在集群中正常工作。

3. 在更新过程中暂停您不想更新的 MCP。

   注意

   暂停 MCP 还会暂停 kube-apiserver-to-kubelet-signer 自动 CA 证书轮转。在自安装日期起的 292 天生成新 CA 证书，旧证书将从安装日期的 365 天后删除。请参阅红帽 OpenShift 4 中的了解 CA 证书自动续订，以了解您在下一次自动 CA 证书轮转前的时间。确保发生 CA 证书轮转时，池已被取消暂停。如果 MCP 暂停，则不会发生证书轮转，这会导致集群降级，并导致多个 **oc** 命令失败，包括但不限于 **oc debug**、**oc logs**、**oc exec** 和 **oc attach**。

4. 执行集群更新。更新过程更新没有暂停的 MCP，包括 control plane 节点。

5. 在更新的节点上测试应用，以确保它们按预期工作。

6. 逐一取消暂停剩余的 MCP，并在这些节点上测试应用程序，直到所有 worker 节点都已更新。取消暂停 MCP 会启动与该 MCP 关联的节点的更新过程。您可以通过点击 **Administration → Cluster settings** 从 web 控制台检查更新的进度。或者，使用 **oc get machineconfigpools** CLI 命令。

7. （可选）从更新的节点中删除自定义标签并删除自定义 MCP。

8.3. 创建机器配置池来执行 CANARY ROLLOUT 更新

执行此 canary rollout 更新的第一个任务是创建一个或多个机器配置池 (MCP)。

1. 从 worker 节点创建 MCP。

   a. 列出集群中的 worker 节点。

   ```
   $ oc get -l 'node-role.kubernetes.io/master!=' -o 'jsonpath={range .items[*]}
   {.metadata.name}{"\n"}{end}' nodes
   ```

   输出示例

   ```
   ci-ln-pwnll6b-f76d1-s8t9n-worker-a-s75z4
   ci-ln-pwnll6b-f76d1-s8t9n-worker-b-dglj2
   ci-ln-pwnll6b-f76d1-s8t9n-worker-c-lldbm
   ```

   b. 对于您要延迟的节点，在节点中添加自定义标签：

   ```
   $ oc label node <node name> node-role.kubernetes.io/<custom-label>=
   ```

   例如：

   ```
   $ oc label node ci-ln-0qv1yp2-f76d1-kl2tq-worker-a-j2ssz node-
   role.kubernetes.io/workerpool-canary=
   ```

   输出示例

   ```
   node/ci-ln-gtrwm8t-f76d1-spbl7-worker-a-xk76k labeled
   ```

   c. 创建新的 MCP：

   ```
   apiVersion: machineconfiguration.openshift.io/v1
   kind: MachineConfigPool
   metadata:
     name: workerpool-canary 1
   spec:
     machineConfigSelector:
       matchExpressions: 2
         - {
           key: machineconfiguration.openshift.io/role,
           operator: In,
   ```

```
        values: [worker,workerpool-canary]
      }
  nodeSelector:
    matchLabels:
      node-role.kubernetes.io/workerpool-canary: "" (3)
```

(1) 为 MCP 指定名称。

(2) 指定 **worker** 和自定义 MCP 名称。

(3) 指定添加到此池中的自定义标签。

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

输出示例

```
machineconfigpool.machineconfiguration.openshift.io/workerpool-canary created
```

d. 查看集群中的 MCP 列表及其当前状态：

```
$ oc get machineconfigpool
```

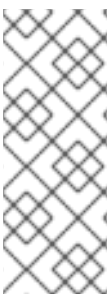
输出示例

```
NAME              CONFIG                                             UPDATED   UPDATING
DEGRADED   MACHINECOUNT   READYMACHINECOUNT
UPDATEDMACHINECOUNT   DEGRADEDMACHINECOUNT   AGE
master            rendered-master-b0bb90c4921860f2a5d8a2f8137c1867            True
False      False      3            3                3                    0                    97m
workerpool-canary rendered-workerpool-canary-87ba3dec1ad78cb6aecebf7fbb476a36
True       False      False      1            1                1                    0                    2m42s
worker            rendered-worker-87ba3dec1ad78cb6aecebf7fbb476a36            True
False      False      2            2                2                    0                    97m
```

创建新的机器配置池 **workerpool-canary**，机器计数中会显示您添加自定义标签的节点数量。worker MCP 机器数会减少相同的数字。更新机器数可能需要几分钟时间。在本例中，一个节点已从 **worker** MCP 移到 worker **pool-canary** MCP。

8.4. 暂停机器配置池

在这个 Canary rollout 更新过程中，在使用 OpenShift Container Platform 集群的其余集群标记了节点并创建机器配置池 (MCP) 后，您会暂停这些 MCP。暂停 MCP 可防止 Machine Config Operator (MCO) 更新与该 MCP 关联的节点。



注意

暂停 MCP 还会暂停 kube-apiserver-to-kubelet-signer 自动 CA 证书轮转。在自安装日期起的 292 天生成新 CA 证书，旧证书将从安装日期的 365 天后删除。请参阅红帽 OpenShift 4 中的了解 CA 证书自动续订，以了解您在下一次自动 CA 证书轮转前的时间。确保发生 CA 证书轮转时，池已被取消暂停。如果 MCP 暂停，则不会发生证书轮转，这会导致集群降级，并导致多个 **oc** 命令失败，包括但不限于 **oc debug**、**oc logs**、**oc exec** 和 **oc attach**。

暂停 MCP：

1. 对您要暂停的 MCP 进行补丁：

```
$ oc patch mcp/<mcp_name> --patch '{"spec":{"paused":true}}' --type=merge
```

例如：

```
$  oc patch mcp/workerpool-canary --patch '{"spec":{"paused":true}}' --type=merge
```

输出示例

```
machineconfigpool.machineconfiguration.openshift.io/workerpool-canary patched
```

8.5. 执行集群更新

当 MCP 进入 ready 状态时，您可以执行集群更新。根据您的集群，请查看以下更新方法之一：

- 使用 Web 控制台更新集群
- 使用 CLI 更新集群

更新完成后，您可以开始逐一取消暂停 MCP。

8.6. 取消暂停机器配置池

在这个 Canary rollout 更新过程中，OpenShift Container Platform 更新完成后，取消逐一暂停自定义 MCP。取消暂停 MCP 允许 Machine Config Operator (MCO) 更新与该 MCP 关联的节点。

取消暂停 MCP：

1. 对您要取消暂停的 MCP 进行补丁：

```
$ oc patch mcp/<mcp_name> --patch '{"spec":{"paused":false}}' --type=merge
```

例如：

```
$  oc patch mcp/workerpool-canary --patch '{"spec":{"paused":false}}' --type=merge
```

输出示例

```
machineconfigpool.machineconfiguration.openshift.io/workerpool-canary patched
```

您可以使用 **oc get machineconfigpools** 命令检查更新的进度。

2. 在更新的节点上测试您的应用，以确保它们按预期工作。
3. 逐一取消暂停任何其他暂停的 MCP，并验证您的应用程序是否正常工作。

8.6.1. 如果应用程序失败

如果应用程序出现故障，如应用程序未在更新的节点上工作，您可以对池中的节点进行 cordon 和 drain 操作，这会将应用 pod 移到其他节点，以帮助维护应用程序的服务质量。第一个 MCP 不应大于过量容量。

8.7. 将节点移到原始机器配置池中

在这个 Canary rollout 更新过程中，在取消暂停自定义机器配置池 (MCP) 并验证与该 MCP 关联的节点上的应用程序是否按预期工作后，您应该删除添加到节点的自定义标签，将节点移回原始 MCP。

重要

节点必须具有角色才能在集群中正常工作。

将节点移动到其原始 MCP 中：

1. 从节点移除自定义标签。

```
$ oc label node <node_name> node-role.kubernetes.io/<custom-label>-
```

例如：

```
$ oc label node ci-ln-0qv1yp2-f76d1-kl2tq-worker-a-j2ssz node-
role.kubernetes.io/workerpool-canary-
```

输出示例

```
node/ci-ln-0qv1yp2-f76d1-kl2tq-worker-a-j2ssz labeled
```

MCO 将节点移回到原始 MCP，并将节点与 MCP 配置协调。

2. 查看集群中的 MCP 列表及其当前状态：

```
$oc get mcp
```

```
NAME              CONFIG                                             UPDATED   UPDATING
DEGRADED   MACHINECOUNT   READYMACHINECOUNT   UPDATEDMACHINECOUNT
DEGRADEDMACHINECOUNT   AGE
master            rendered-master-1203f157d053fd987c7cbd91e3fbc0ed        True      False
False      3              3                   3                   0                  61m
workerpool-canary   rendered-mcp-noupdate-5ad4791166c468f3a35cd16e734c9028   True
False      False      0              0                   0                   0                  21m
worker            rendered-worker-5ad4791166c468f3a35cd16e734c9028        True      False
False      3              3                   3                   0                  61m
```

该节点从自定义 MCP 中删除，并移回原始 MCP。更新机器数可能需要几分钟时间。在这个示例中，将一个节点从删除的 **workerpool-canary MCP** 移到 'worker'MCP。

3. 可选：删除自定义 MCP：

```
$ oc delete mcp <mcp_name>
```

第 9 章 更新包含使用 RHEL 的计算（COMPUTE）系统的集群

您可以更新或升级OpenShift Container Platform集群。如果您的集群包含Red Hat Enterprise Linux（RHEL）系统，则必须执行额外的步骤来更新这些系统。

9.1. 先决条件

- 使用具有 **admin** 权限的用户访问集群。请参阅使用 RBAC 定义和应用权限。
- 具有最新的 etcd 备份，以防因为升级失败需要将集群恢复到以前的状态。
- 如果您的集群使用手动维护的凭证，请确保 Cloud Credential Operator（CCO）处于可升级状态。如需更多信息，请参阅为 AWS、Azure 或 GCP *手动维护凭证升级集群*。
- 如果您的集群通过 AWS Secure Token Service (STS) 使用手动维护的凭证，请从要升级到的发行镜像获取 **ccoctl** 实用程序的副本，并使用它来处理任何更新的凭证。如需更多信息，请参阅*升级使用 STS 的手动模式配置的 OpenShift Container Platform 集群*。
- 如果您运行 Operator 或您已配置了 pod 中断预算，您可能会在升级过程中遇到中断。如果在 **PodDisruptionBudget** 中将 **minAvailable** 设置为 1，则节点会排空以应用可能会阻止驱除过程的待处理机器配置。如果重启了几个节点，则所有 pod 只能有一个节点上运行，**PodDisruptionBudget** 字段可能会阻止节点排空。

其他资源

- 非受管 Operator 的支持策略

9.2. 使用WEB控制台更新集群

如果有可用更新，您可以从Web控制台更新集群。

您可以在客户门户网站的勘误部分找到有关可用 OpenShift Container Platform 公告和更新的信息。

先决条件

- 使用具有 **admin** 权限的用户登陆到 web 控制台。
- 暂停所有 **MachineHealthCheck** 资源。

流程

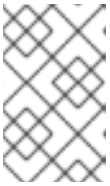
1. 在 web 控制台中点击 **Administration → Cluster Settings** 并查看 **Details** 选项卡中的内容。
2. 对于生产环境中的集群，请确保将 **Channel** 设置为您要升级到的版本的正确频道，如 **stable-4.9**。

重要

对于生产环境中的集群，您必须订阅一个 **stable-***, **eus-*** 或 **fast-*** 频道。

- 如果 **Update 状态** 不是 **Updates available**，则无法升级集群。
- **Select channel** 表示集群正在运行或正在更新的集群版本。

3. 选择要更新到的版本，然后单击 **Save**。
输入频道 **Update Status** 变为**Update to <product-version> in progress**，您可以通过监视 Operator 和节点的进度条来查看集群更新的进度。

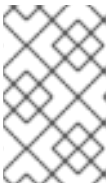


注意

如果您要将集群升级到下一个次版本，如从 4.y 升级到 4.(y+1)，建议在部署依赖新功能的工作负载前确认您的节点已升级：任何尚未更新的 worker 节点池都会显示在 **Cluster Settings** 页面。

4. 更新完成后，Cluster Version Operator 会刷新可用更新，检查当前频道中是否有更多可用更新。
   - 如果有可用更新，请继续在当前频道中执行更新，直到您无法再更新为止。
   - 如果没有可用的更新，请将 **Channel** 改为下一个次版本的 **stable-***, **eus-*** 或 **fast-*** 频道，并更新至您在该频道中想要的版本。

   您可能需要执行一些过渡的更新，直到您到达您想要的版本。



注意

当您更新包含有 Red Hat Enterprise Linux (RHEL) worker 机器的集群时，这些 worker 会在更新过程中暂时不可用。当集群进入 **NotReady** 状态时，您需要针对每个 RHEL 机器运行升级 playbook 以完成更新。

9.3. 可选：添加 HOOK 以在RHEL系统上执行ANSIBLE任务

在OpenShift Container Platform更新期间，您可以使用*hook*在RHEL计算系统上运行Ansible任务。

9.3.1. 在升级过程中使用 Ansible hook

更新OpenShift Container Platform时，可以使用*hook*在执行特定操作时在Red Hat Enterprise Linux（RHEL）节点上运行自定义的任务。您可以使用 hook 提供定义了在执行特定任务之前或之后要运行的任务的文件。在OpenShift Container Platform集群中更新RHEL计算节点时，可以使用 hook 来验证或修改自定义的基础架构。

因为当 hook 失败时，这个操作将会失败，所以您必须把 hook 设计为可以多次运行，并且获得相同的结果。

hook 有以下限制： - hook 没有已定义或版本化的界面。它们可以使用内部的**openshift-ansible**变量，但这些变量可能会在将来的OpenShift Container Platform版本被修改或删除。 - hook 本身没有错误处理机制，因此 hook 中的错误会暂停更新过程。如果出现错误，则需要解决相关的问题，然后再次进行升级。

9.3.2. 配置Ansible inventory文件以使用 hook

您可以在 **hosts** inventory 文件的**all:vars** 部分中定义 Red Hat Enterprise Linux（RHEL）compute 机器（也称为 worker 机器）更新时使用的 hook 。

先决条件

- 您可以访问用于添加RHEL compute 系统集群的计算机。您必须有访问定义RHEL系统的**hosts** Ansible 清单文件的权限。

流程

1. 在设计了 hook 后，创建一个YAML文件，为其定义Ansible任务。此文件必须是一组任务，不能是一个 playbook，如以下示例所示：

```
---
# Trivial example forcing an operator to acknowledge the start of an upgrade
# file=/home/user/openshift-ansible/hooks/pre_compute.yml

- name: note the start of a compute machine update
  debug:
    msg: "Compute machine upgrade of {{ inventory_hostname }} is about to start"

- name: require the user agree to start an upgrade
  pause:
    prompt: "Press Enter to start the compute machine update"
```

2. 修改 **hosts** Ansible inventory 文件来指定 hook 文件。hook 文件作为参数值在 **[all:vars]** 部分指定。如下所示：

清单文件中的 hook 定义示例

```
[all:vars]
openshift_node_pre_upgrade_hook=/home/user/openshift-ansible/hooks/pre_node.yml
openshift_node_post_upgrade_hook=/home/user/openshift-ansible/hooks/post_node.yml
```

为了避免歧义，请在其定义中使用 hook 文件的绝对路径而不要使用相对路径。

9.3.3. RHEL计算系统可用的 hook

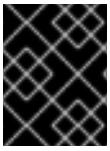
在更新OpenShift Container Platform集群中的Red Hat Enterprise Linux（RHEL）compute 系统时，可以使用以下 hook。

Hook 名	描述
openshift_node_pre_cordon_hook	• 在每个节点被封锁（cordon）**之前**运行。 • 此 hook 以串行方式针对**每个节点**运行。 • 如果某个任务必须针对其他主机运行，则该任务必须使用**delegate_to**或**local_action**。
openshift_node_pre_upgrade_hook	• 在每个节点被封锁**后**，且被更新**前**运行。 • 此 hook 以串行方式针对**每个节点**运行。 • 如果某个任务必须针对其他主机运行，则该任务必须使用**delegate_to**或**local_action**。

Hook 名	描述
openshift_node_pre_uncordon_hook	• 在每个节点被更新**后**，且被取消封锁（uncordon）**前**运行。 • 此 hook 以串行方式针对**每个节点**运行。 • 如果某个任务必须针对其他主机运行，则该任务必须使用**delegate_to**或**local_action**。
openshift_node_post_upgrade_hook	• 每个节点未被取消封锁**后**运行。这是**最后一个**节点更新操作。 • 此 hook 以串行方式针对**每个节点**运行。 • 如果某个任务必须针对其他主机运行，则该任务必须使用**delegate_to**或**local_action**。

9.4. 更新集群中的RHEL COMPUTE 系统

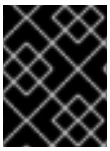
在对集群进行更新后，必须更新集群中的Red Hat Enterprise Linux（RHEL）compute 系统。



重要

RHEL worker (compute) 集群支持使用 Red Hat Enterprise Linux (RHEL) 版本 8.4 和 8.5。

如果您使用 RHEL 作为操作系统，您还可以将计算机器更新至 OpenShift Container Platform 的另一个次要版本。当执行次版本更新时，您不需要排除 RHEL 中的任何 RPM 软件包。

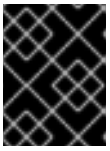


重要

您无法将 RHEL 7 计算机器升级到 RHEL 8。您必须部署新的 RHEL 8 主机，并且应该删除旧的 RHEL 7 主机。

先决条件

- 已更新了集群。



重要

因为 RHEL 机器需要集群生成的资产才能完成更新过程，所以您必须在更新其中的 RHEL worker 机器前更新集群。

- 您可以访问用于将 RHEL 计算机器添加到集群的本地机器。您必须有权访问定义了 RHEL 系统及 **upgrade** playbook 的 **hosts** Ansible 清单文件。
- 对于次版本的更新，RPM 存储库使用的是集群上运行的相同版本的 OpenShift Container Platform。

流程

1. 停止并禁用主机上的防火墙：

```
# systemctl disable --now firewalld.service
```

注意

默认情况下，使用 "Minimal" 安装选项的基础操作系统 RHEL 启用 firewalld 保护。在主机上启用了 firewalld 服务会阻止您访问 worker 上的 OpenShift Container Platform 日志。如果您希望继续访问 worker 上的 OpenShift Container Platform 日志，以后不要启用 firewalld。

2. 启用 OpenShift Container Platform 4.9 所需的存储库：

   a. 在运行 Ansible playbook 的机器上，更新所需的存储库：

```
# subscription-manager repos --disable=rhel-7-server-ose-4.8-rpms \
                           --enable=rhel-7-server-ansible-2.9-rpms \
                           --enable=rhel-7-server-ose-4.9-rpms
```

   b. 在运行 Ansible playbook 的机器上，更新所需的软件包，包括 **openshift-ansible**：

```
# yum update openshift-ansible openshift-clients
```

   c. 在每个 RHEL 计算节点上，更新所需的软件仓库：

```
# subscription-manager repos --disable=rhel-7-server-ose-4.8-rpms \
                           --enable=rhel-7-server-ose-4.9-rpms  \
                           --enable=rhel-7-fast-datapath-rpms   \
                           --enable=rhel-7-server-optional-rpms
```

3. 更新 RHEL worker 机器：

   a. 查看当前节点状态，以确定要更新哪个 RHEL worker：

```
# oc get node
```

   输出示例

```
NAME                 STATUS                     ROLES    AGE    VERSION
mycluster-control-plane-0   Ready                      master   145m   v1.22.1
mycluster-control-plane-1   Ready                      master   145m   v1.22.1
mycluster-control-plane-2   Ready                      master   145m   v1.22.1
mycluster-rhel7-0           NotReady,SchedulingDisabled   worker   98m    v1.22.1
mycluster-rhel7-1           Ready                      worker   98m    v1.22.1
mycluster-rhel7-2           Ready                      worker   98m    v1.22.1
mycluster-rhel7-3           Ready                      worker   98m    v1.22.1
```

   记录下哪个机器具有 **NotReady，SchedulingDisabled** 状态。

   b. 查看位于 **/<path>/inventory/hosts** 中的 Ansible 清单文件，并更新其内容，以便只有具有 **NotReady,SchedulingDisabled** 状态的机器才列在 **[workers]** 部分中，如下例所示：

```
[all:vars]
ansible_user=root
#ansible_become=True

openshift_kubeconfig_path="~/.kube/config"

[workers]
mycluster-rhel7-0.example.com
```

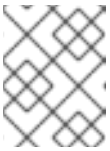
c. 进入 **openshift-ansible** 目录：

```
$ cd /usr/share/ansible/openshift-ansible
```

d. 运行 **upgrade** playbook:

```
$ ansible-playbook -i /<path>/inventory/hosts playbooks/upgrade.yml 1
```

1 对于**<path>**，指定您创建的Ansible库存文件的路径。



注意

upgrade playbook 仅升级 OpenShift Container Platform 软件包。它不会更新操作系统软件包。

4. 按照上一步中的流程更新集群中的每个 RHEL worker 机器。

5. 更新完所有 worker 后，确认所有集群节点已更新至新版本：

```
# oc get node
```

输出示例

```
NAME                 STATUS                     ROLES    AGE    VERSION
mycluster-control-plane-0   Ready                      master   145m   v1.22.1
mycluster-control-plane-1   Ready                      master   145m   v1.22.1
mycluster-control-plane-2   Ready                      master   145m   v1.22.1
mycluster-rhel7-0        NotReady,SchedulingDisabled   worker   98m    v1.22.1
mycluster-rhel7-1        Ready                      worker   98m    v1.22.1
mycluster-rhel7-2        Ready                      worker   98m    v1.22.1
mycluster-rhel7-3        Ready                      worker   98m    v1.22.1
```

6. 可选：更新 **upgrade** playbook 没有更新的操作系统软件包。要更新不在 4.9 中的软件包，请使用以下命令：

```
# yum update
```

注意

如果您使用安装 4.9 时使用的相同 RPM 存储库，则不需要排除 RPM 软件包。

第 10 章 在断开连接的环境中更新集群

10.1. 关于在断开连接的环境中的集群更新

断开连接的环境是集群节点无法访问互联网的环境。因此，您必须在 registry 中填充安装镜像。如果您的 registry 主机无法同时访问互联网和集群，您可以将镜像镜像到与这个环境断开连接的文件系统中，然后使用主机或可移动介质填补该空白。如果本地容器 registry 和集群连接到镜像 registry 的主机，您可以直接将发行镜像推送到本地 registry。

一个独立的容器镜像 registry 足以为断开连接的网络中的多个集群托管 mirror 的镜像。

10.1.1. 镜像 OpenShift Container Platform 镜像存储库

要在断开连接的环境中更新集群，您的集群环境必须有权访问具有目标更新所需镜像和资源的镜像 registry。以下页提供了将镜像镜像到断开连接的集群中的存储库的说明：

- 镜像 OpenShift Container Platform 镜像存储库

10.1.2. 在断开连接的环境中执行集群更新

您可以使用以下步骤之一更新断开连接的 OpenShift Container Platform 集群：

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

10.1.3. 从集群中删除 OpenShift Update Service

您可以使用以下步骤从集群中卸载 OpenShift Update Service (OSUS) 的本地副本：

- 从集群中删除 OpenShift Update Service

10.2. 镜像 OPENSHIFT CONTAINER PLATFORM 镜像存储库

您必须将容器镜像镜像到镜像 registry 中，然后才能在受限网络环境中更新集群。您还可以在连接的环境中使用此流程来确保集群只运行满足您机构对外部内容控制的批准的容器镜像。

10.2.1. 先决条件

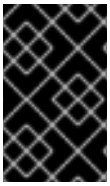
- 您必须在托管 OpenShift Container Platform 集群的位置（如 Red Hat Quay）中有一个支持 Docker v2-2 的容器镜像 registry。

10.2.2. 准备您的镜像主机

执行镜像步骤前，必须准备主机以检索内容并将其推送到远程位置。

10.2.2.1. 通过下载二进制文件安装 OpenShift CLI

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



重要

如果安装了旧版本的 **oc**，则无法使用 OpenShift Container Platform 4.9 中的所有命令。下载并安装新版本的 **oc**。如果要在断开连接的环境中升级集群，请安装您要升级到的 **oc** 版本。

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

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.9 Linux 客户端** 条目旁边的 **Download Now**，再保存文件。
4. 解包存档：

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

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

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

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

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.9 Windows 客户端** 条目旁边的 **Download Now**，再保存文件。
4. 使用 ZIP 程序解压存档。
5. 将 **oc** 二进制文件移到 **PATH 中的目录中**。
   要查看您的 **PATH**，请打开命令提示并执行以下命令：

```
C:\> path
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

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

在 macOS 上安装 OpenShift CLI

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

流程

1. 导航到红帽客户门户网站上的 OpenShift Container Platform 下载页面。
2. 在 **Version** 下拉菜单中选择相应的版本。
3. 单击 **OpenShift v4.9 MacOSX Client** 条目旁边的 **Download Now**，再保存文件。
4. 解包和解压存档。
5. 将 **oc** 二进制文件移到 PATH 的目录中。
   要查看您的 **PATH**，请打开终端并执行以下命令：

```
$ echo $PATH
```

安装 OpenShift CLI 后，可以使用 **oc** 命令：

```
$ oc <command>
```

10.2.2.2. 配置允许对容器镜像进行镜像的凭证

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

警告

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

警告

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

先决条件

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

流程

在安装主机上完成以下步骤：

1. 从 Red Hat OpenShift Cluster Manager 下载 **registry.redhat.io** 的 pull secret，并将它保存到 **.json** 文件中。

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

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

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

3. 以 JSON 格式创建您的 pull secret 副本：

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

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

4. 将文件保存为 **~/.docker/config.json** 或 **$XDG_RUNTIME_DIR/containers/auth.json**。该文件类似于以下示例：

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

5. 编辑新文件并添加描述 registry 的部分：

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

1. 对于 **<mirror_registry>**，指定 registry 域名，以及您的镜像 registry 用来提供内容的可选端口。例如：**registry.example.com** 或 **registry.example.com:8443**
2. 使用 **<credentials>** 为您的镜像 registry 指定 base64 编码的用户名和密码。

该文件类似于以下示例：

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

10.2.3. 镜像 OpenShift Container Platform 镜像存储库

重要

为了避免 OpenShift Update Service 应用程序过量内存用量，您必须将发行镜像镜像到单独的存储库，如以下步骤所述。

先决条件

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

流程

1. 使用 Red Hat OpenShift Container Platform Upgrade Graph visualizer 和 update planner 计划从一个版本升级到另一个版本。OpenShift Upgrade Graph 提供频道图形，并演示了如何确认您的当前和预定集群版本之间有更新路径。

2. 设置所需的环境变量：

   a. 导出发行版本信息：

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

   对于 **<release_version>**，请指定与升级到的 OpenShift Container Platform 版本对应的标签，如 **4.5.4**。

   b. 导出本地 registry 名称和主机端口：

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

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

   c. 导出本地存储库名称：

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

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

   d. 如果使用 OpenShift Update Service，请导出一个额外的本地存储库名称，使其包含发行镜像：

   ```
   $
   LOCAL_RELEASE_IMAGES_REPOSITORY='<local_release_images_repository_name>'
   ```

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

   e. 导出要进行镜像的存储库名称：

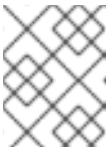
   ```
   $ PRODUCT_REPO='openshift-release-dev'
   ```

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

   f. 导出 registry pull secret 的路径：

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

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

   注意

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

   g. 导出发行版本镜像：

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

对于生产环境版本，您必须指定 **ocp-release**。

h. 为您的服务器导出构架类型，如 **x86_64**。

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

i. 导出托管镜像的目录的路径：

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

1 指定完整路径，包括开始的前斜杠(/)字符。

3. 查看要镜像的镜像和配置清单：

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

4. 将版本镜像镜像(mirror)到镜像 registry。

- 如果您的镜像主机无法访问互联网，请执行以下操作：

i. 将可移动介质连接到连接到互联网的系统。

ii. 将镜像和配置清单镜像到可移动介质的目录中：

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

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

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

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

iv. 使用 **oc** 命令行界面(CLI)登录到您要升级的集群。

v. 将镜像发行镜像签名配置映射应用到连接的集群：

```
$ oc apply -f ${REMOVABLE_MEDIA_PATH}/mirror/config/<image_signature_file>
1
```

1 对于 **<image_signature_file>**，指定文件的路径和名称，例如 **signature-sha256-81154f5c03294534.yaml**。

vi. 如果使用 OpenShift Update Service，请将发行镜像镜像到单独的存储库：

```
$ oc image mirror -a ${LOCAL_SECRET_JSON}
${LOCAL_REGISTRY}/${LOCAL_REPOSITORY}:${OCP_RELEASE}-
${ARCHITECTURE}
${LOCAL_REGISTRY}/${LOCAL_RELEASE_IMAGES_REPOSITORY}:${OCP_REL
EASE}-${ARCHITECTURE}
```

- 如果本地容器 registry 和集群连接到镜像主机，请执行以下操作：

  i. 将发行镜像直接推送到本地 registry，并使用以下命令将配置映射应用到集群：

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

注意

如果包含 **--apply-release-image-signature** 选项，不要为镜像签名验证创建配置映射。

  ii. 如果使用 OpenShift Update Service，请将发行镜像镜像到单独的存储库：

```
$ oc image mirror -a ${LOCAL_SECRET_JSON}
${LOCAL_REGISTRY}/${LOCAL_REPOSITORY}:${OCP_RELEASE}-
${ARCHITECTURE}
${LOCAL_REGISTRY}/${LOCAL_RELEASE_IMAGES_REPOSITORY}:${OCP_REL
EASE}-${ARCHITECTURE}
```

10.3. 使用 OPENSHIFT UPDATE SERVICE 在断开连接的环境中更新集群

要获得与连接的集群类似的更新体验，您可以使用以下步骤在断开连接的环境中安装和配置 OpenShift Update Service。

10.3.1. 在断开连接的环境中使用 OpenShift Update Service

OpenShift Update Service (OSUS) 为 OpenShift Container Platform 集群提供更新建议。红帽公开托管 OpenShift Update Service，连接的环境中的集群可以通过公共 API 连接到该服务，以检索更新建议。

但是，在断开连接的环境中的集群无法访问这些公共 API 来检索更新信息。要在断开连接的环境中提供类似的更新体验，您可以在本地安装和配置 OpenShift Update Service，使其在断开连接的环境中可用。

以下小节介绍了如何安装本地 OSUS 实例，并将其配置为为集群提供更新建议。

其他资源

- 关于 OpenShift Update 服务
- 了解升级频道和发行版本

10.3.2. 先决条件

- 您必须安装了 **oc** 命令行界面（CLI）工具。

- 您必须使用容器镜像置备本地容器镜像 registry，如镜像 OpenShift Container Platform 镜像存储库 中所述。

10.3.3. 为 OpenShift Update Service 配置对安全 registry 的访问

如果发行镜像包含在由自定义证书颁发机构签名的 HTTPS X.509 证书的 registry 中，请完成 为镜像 registry 访问配置额外信任存储 的步骤，以及对更新服务进行以下更改。

OpenShift Update Service Operator 需要 registry CA 证书中的配置映射键名称为 **updateservice-registry**。

更新服务的镜像 registry CA 配置映射示例

```
apiVersion: v1
kind: ConfigMap
metadata:
  name: my-registry-ca
data:
  updateservice-registry: | 1
    -----BEGIN CERTIFICATE-----
    ...
    -----END CERTIFICATE-----
  registry-with-port.example.com..5000: | 2
    -----BEGIN CERTIFICATE-----
    ...
    -----END CERTIFICATE-----
```

1. OpenShift Update Service Operator 需要 registry CA 证书中的配置映射键名称 updateservice-registry。
2. 如果 registry 带有端口，如 **registry-with-port.example.com:5000**，**:** 需要被 **..** 替换。

10.3.4. 更新全局集群 pull secret

您可以通过替换当前的 pull secret 或附加新的 pull secret 来更新集群的全局 pull secret。

当用户使用单独的 registry 存储镜像而不使用安装过程中的 registry时，需要这个过程。

先决条件

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

流程

1. 可选：要将新的 pull secret 附加到现有 pull secret 中，请完成以下步骤：
   a. 输入以下命令下载 pull secret：

```
$ oc get secret/pull-secret -n openshift-config --template='{{index .data
".dockerconfigjson" | base64decode}}' ><pull_secret_location> 1
```

1. 提供 pull secret 文件的路径。

b. 输入以下命令来添加新 pull secret：

```
$ oc registry login --registry="<registry>" \ 1
--auth-basic="<username>:<password>" \ 2
--to=<pull_secret_location> 3
```

1. 提供新的 registry。您可以在同一个 registry 中包含多个软件仓库，例如：**--registry="<registry/my-namespace/my-repository>"**。
2. 提供新 registry 的凭据。
3. 提供 pull secret 文件的路径。

另外，您可以对 pull secret 文件执行手动更新。

2. 输入以下命令为您的集群更新全局 pull secret：

```
$ oc set data secret/pull-secret -n openshift-config --from-file=.dockerconfigjson=
<pull_secret_location> 1
```

1. 提供新 pull secret 文件的路径。

该更新将推广至所有节点，可能需要一些时间，具体取决于集群大小。

注意

从 OpenShift Container Platform 4.7.4 开始，对全局 pull secret 的更改不再触发节点排空或重启。

10.3.5. 安装 OpenShift Update Service Operator

要安装 OpenShift Update Service，您必须首先使用 OpenShift Container Platform Web 控制台或 CLI 安装 OpenShift Update Service Operator。

注意

对于在断开连接的环境中安装的集群（也称为断开连接的集群），Operator Lifecycle Manager 默认无法访问托管在远程 registry 上的红帽提供的 OperatorHub 源，因为这些远程源需要有互联网连接。如需更多信息，请参阅在受限网络中使用 Operator Lifecycle Manager。

10.3.5.1. 使用 Web 控制台安装 OpenShift Update Service Operator

您可以使用 Web 控制台安装 OpenShift Update Service Operator。

流程

1. 在 Web 控制台中，点 **Operators → OperatorHub**。

注意

在 **Filter by keyword...** 字段中输入 **Update Service**，以更快地查找 Operator。

2. 从可用的 Operator 列表中选择 **OpenShift Update Service**，然后点 **Install**。

   a. 频道 **v1** 被选为 **Update Channel**，因为它是这个版本中唯一可用的频道。

   b. 在 **Installation Mode** 下选择 **A specific namespace on the cluster**。

   c. 为 **Installed Namespace** 选择一个命名空间，或接受推荐的命名空间 **openshift-update-service**。

   d. 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** 策略要求集群管理员批准 Operator 更新。

   e. 点 **Install**。

3. 通过切换到 **Operators → Installed Operators** 页来验证 OpenShift Update Service Operator 是否已安装。

4. 确保 **OpenShift Update Service** 列在所选命名空间中，**Status** 为 **Succeeded**。

10.3.5.2. 使用 CLI 安装 OpenShift Update Service Operator

您可以使用 OpenShift CLI（**oc**）安装 OpenShift Update Service Operator。

流程

1. 为 OpenShift Update Service Operator 创建命名空间：

   a. 为 OpenShift Update Service Operator 创建一个 **Namespace** 对象 YAML 文件，如 **update-service-namespace.yaml**：

```
apiVersion: v1
kind: Namespace
metadata:
  name: openshift-update-service
  annotations:
    openshift.io/node-selector: ""
  labels:
    openshift.io/cluster-monitoring: "true" 1
```

   1 将 **openshift.io/cluster-monitoring** 标签设置为在该命名空间中启用 Operator-recommended 集群监控。

   b. 创建命名空间：

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

   例如：

```
$ oc create -f update-service-namespace.yaml
```

2. 通过创建以下对象来安装 OpenShift Update Service Operator：

a. 创建一个 **OperatorGroup** 对象 YAML 文件，如 **update-service-operator-group.yaml**：

```
apiVersion: operators.coreos.com/v1
kind: OperatorGroup
metadata:
  name: update-service-operator-group
spec:
  targetNamespaces:
  - openshift-update-service
```

b. 创建一个 **OperatorGroup** 对象：

```
$ oc -n openshift-update-service create -f <filename>.yaml
```

例如：

```
$ oc -n openshift-update-service create -f update-service-operator-group.yaml
```

c. 创建一个 **Subscription** 对象 YAML 文件，如 **update-service-subscription.yaml**：

订阅示例

```
apiVersion: operators.coreos.com/v1alpha1
kind: Subscription
metadata:
  name: update-service-subscription
spec:
  channel: v1
  installPlanApproval: "Automatic"
  source: "redhat-operators" (1)
  sourceNamespace: "openshift-marketplace"
  name: "cincinnati-operator"
```

1. 指定提供 Operator 的目录源的名称。对于不使用自定义 Operator Lifecycle Manager（OLM）的集群，指定 **redhat-operators**。如果 OpenShift Container Platform 集群安装在断开连接的环境中，请指定配置 Operator Lifecycle Manager (OLM) 时创建的 **CatalogSource** 对象的名称。

d. 创建 **Subscription** 对象：

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

例如：

```
$ oc -n openshift-update-service create -f update-service-subscription.yaml
```

OpenShift Update Service Operator 被安装到 **openshift-update-service** 命名空间，并以 **openshift-update-service** 命名空间为目标。

3. 验证 Operator 安装：

```
$ oc -n openshift-update-service get clusterserviceversions
```

输出示例

```
NAME                         DISPLAY                VERSION   REPLACES   PHASE
update-service-operator.v4.6.0   OpenShift Update Service   4.6.0             Succeeded
...
```

如果列出了 OpenShift Update Service Operator，则会成功安装。版本号可能与所示不同。

其他资源

- 在命名空间中安装 Operator。

10.3.6. 创建 OpenShift Update Service 图形数据容器镜像

OpenShift Update Service 需要图形数据容器镜像，OpenShift Update Service 从中检索有关频道成员资格和阻止更新边缘的信息。图形数据通常直接从升级图形数据仓库中获取。在互联网连接不可用的环境中，从 init 容器加载此信息是使图形数据可供 OpenShift Update Service 使用的另一种方式。init 容器的角色是提供图形数据的本地副本，在 pod 初始化期间，init 容器会将数据复制到该服务可访问的卷中。

流程

1. 创建一个 Dockerfile，如 **./Dockerfile**，包含以下内容：

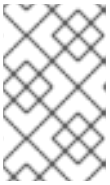
```
FROM registry.access.redhat.com/ubi8/ubi:8.1

RUN curl -L -o cincinnati-graph-data.tar.gz
https://api.openshift.com/api/upgrades_info/graph-data

RUN mkdir -p /var/lib/cincinnati-graph-data && tar xvzf cincinnati-graph-data.tar.gz -C
/var/lib/cincinnati-graph-data/ --no-overwrite-dir --no-same-owner

CMD ["/bin/bash", "-c" ,"exec cp -rp /var/lib/cincinnati-graph-data/* /var/lib/cincinnati/graph-
data"]
```

2. 使用上一步中创建的 docker 文件来构建图形数据容器镜像，如 **registry.example.com/openshift/graph-data:latest**：

```
$ podman build -f ./Dockerfile -t registry.example.com/openshift/graph-data:latest
```

3. 将上一步中创建的 graph-data 容器镜像推送到 OpenShift Update Service 可以访问的存储库，如 **registry.example.com/openshift/graph-data:latest**：

```
$ podman push registry.example.com/openshift/graph-data:latest
```



注意

要将图形数据镜像推送到受限网络中的本地 registry，请将上一步中创建的 graph-data 容器镜像复制到可供 OpenShift Update Service 访问的存储库。运行 **oc image mirror --help** 查看可用选项。

10.3.7. 创建 OpenShift Update Service 应用程序

您可以使用 OpenShift Container Platform Web 控制台或 CLI 创建 OpenShift Update Service 应用程序。

10.3.7.1. 使用 Web 控制台创建 OpenShift Update Service 应用程序

您可以使用 OpenShift Container Platform Web 控制台使用 OpenShift Update Service Operator 创建 OpenShift Update Service 应用程序。

先决条件

- 已安装 OpenShift Update Service Operator。
- OpenShift Update Service graph-data 容器镜像已创建并推送到 OpenShift Update Service 访问的存储库。
- 当前发行版本和更新目标版本已被 mirror 到本地可访问的 registry 中。

流程

1. 在 Web 控制台中，点 **Operators → Installed Operators**。
2. 从安装的 Operator 列表中选择 **OpenShift Update Service**。
3. 点 **Update Service** 选项卡。
4. 点 **Create UpdateService**。
5. 在 **Name** 字段中输入名称，如 **service**。
6. 在 **Graph Data Image** 字段中输入本地 pullspec，指向在"创建 OpenShift Update Service 图形数据容器镜像"中创建的图形数据容器镜像，如 **registry.example.com/openshift/graph-data:latest**。
7. 在 **Releases** 字段中，输入创建的本地 registry 和存储库，以在"镜像 OpenShift Container Platform 镜像存储库"中包括发行镜像，例如 **registry.example.com/ocp4/openshift4-release-images**。
8. 在 **Replicas** 字段中输入 **2**。
9. 单击 **Create** 以创建 OpenShift Update Service 应用。
10. 验证 OpenShift Update Service 应用程序：
    - 从 **Update Service** 选项卡中的 **UpdateServices** 列表中，点刚才创建的 Update Service 应用程序。
    - 单击 **Resources** 选项卡。
    - 验证每个应用资源的状态是否为 **Created**。

10.3.7.2. 使用 CLI 创建 OpenShift Update Service 应用程序

您可以使用 OpenShift CLI（**oc**）来创建 OpenShift Update Service 应用。

先决条件

- 已安装 OpenShift Update Service Operator。
- OpenShift Update Service graph-data 容器镜像已创建并推送到 OpenShift Update Service 访问的存储库。
- 当前发行版本和更新目标版本已被 mirror 到本地可访问的 registry 中。

流程

1. 配置 OpenShift Update Service 目标命名空间，如 **openshift-update-service**：

```
$ NAMESPACE=openshift-update-service
```

   命名空间必须与 operator 组中的 **targetNamespaces** 值匹配。

2. 配置 OpenShift Update Service 应用程序的名称，如 **service**：

```
$ NAME=service
```

3. 按照"镜像 OpenShift Container Platform 镜像存储库"中配置，为发行镜像配置本地 registry 和存储库，如 **registry.example.com/ocp4/openshift4-release-images**：

```
$ RELEASE_IMAGES=registry.example.com/ocp4/openshift4-release-images
```

4. 将 graph-data 镜像的本地 pullspec 设置为在"创建 OpenShift Update Service 图形数据容器镜像"中创建的图形数据容器镜像，如 **registry.example.com/openshift/graph-data:latest**:

```
$ GRAPH_DATA_IMAGE=registry.example.com/openshift/graph-data:latest
```

5. 创建 OpenShift Update Service 应用程序对象：

```
$ oc -n "${NAMESPACE}" create -f - <<EOF
apiVersion: updateservice.operator.openshift.io/v1
kind: UpdateService
metadata:
  name: ${NAME}
spec:
  replicas: 2
  releases: ${RELEASE_IMAGES}
  graphDataImage: ${GRAPH_DATA_IMAGE}
EOF
```

6. 验证 OpenShift Update Service 应用程序：

   a. 使用以下命令获取策略引擎路由：

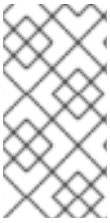
```
$ while sleep 1; do POLICY_ENGINE_GRAPH_URI="$(oc -n "${NAMESPACE}" get -o
jsonpath='{.status.policyEngineURI}/api/upgrades_info/v1/graph{"\n"}' updateservice
"${NAME}")"; SCHEME="${POLICY_ENGINE_GRAPH_URI%%:*}"; if test "${SCHEME}"
= http -o "${SCHEME}" = https; then break; fi; done
```

   您可能需要轮询，直到命令成功为止。

b. 从策略引擎检索图形。确保为 **channel** 指定一个有效版本。例如，如果在 OpenShift Container Platform 4.9 中运行，请使用 **stable-4.9**：

```
$ while sleep 10; do HTTP_CODE="$(curl --header Accept:application/json --output
/dev/stderr --write-out "%{http_code}" "${POLICY_ENGINE_GRAPH_URI}?
channel=stable-4.6")"; if test "${HTTP_CODE}" -eq 200; then break; fi; echo
"${HTTP_CODE}"; done
```

这会轮询到图形请求成功为止，但生成的图形可能为空，具体取决于您已镜像的发行镜像。



注意

基于 RFC-1123 的策略引擎路由名称不能超过 63 个字符。如果您看到 **ReconcileCompleted** 状态为 **false**，原因为 **CreateRouteFailed** caused by **host must conform to DNS 1123 naming convention and must be no more than 63 characters**，请尝试使用较短的名称创建 Update Service。

10.3.7.2.1. 配置 Cluster Version Operator（CVO）

安装 OpenShift Update Service Operator 并创建 OpenShift Update Service 应用程序后，可以更新 Cluster Version Operator（CVO）从本地安装的 OpenShift Update Service 中拉取图形数据。

先决条件

- 已安装 OpenShift Update Service Operator。
- OpenShift Update Service graph-data 容器镜像已创建并推送到 OpenShift Update Service 访问的存储库。
- 当前发行版本和更新目标版本已被 mirror 到本地可访问的 registry 中。
- OpenShift Update Service 应用已创建。

流程

1. 设置 OpenShift Update Service 目标命名空间，如 **openshift-update-service**：

```
$ NAMESPACE=openshift-update-service
```

2. 设置 OpenShift Update Service 应用程序的名称，如 **service**：

```
$ NAME=service
```

3. 获取策略引擎路由：

```
$ POLICY_ENGINE_GRAPH_URI="$(oc -n "${NAMESPACE}" get -o
jsonpath='{.status.policyEngineURI}/api/upgrades_info/v1/graph{"\n"}' updateservice
"${NAME}")"
```

4. 为拉取图形数据设置补丁：

```
$ PATCH="{\"spec\":{\"upstream\":\"${POLICY_ENGINE_GRAPH_URI}\"}}"
```

5. 对 CVO 进行补丁以使用本地 OpenShift 更新服务：

```
$ oc patch clusterversion version -p $PATCH --type merge
```

注意

请参阅启用集群范围代理以将 CA 配置为信任更新服务器。

10.3.8. 后续步骤

在更新集群前，请确认满足以下条件：

- Cluster Version Operator (CVO) 被配置为使用本地安装的 OpenShift Update Service 应用程序。
- 新发行版本的发行镜像签名配置映射应用到集群。
- 当前发行版本和更新目标发行镜像被镜像到本地可访问的 registry 中。
- 最近图数据容器镜像已镜像到本地 registry。

将集群配置为使用本地安装的 OpenShift Update Service 和本地镜像 registry 后，您可以使用以下任何更新方法：

- 使用 Web 控制台更新集群
- 使用 CLI 更新集群
- 准备执行 EUS 更新
- 执行 canary rollout 更新
- 更新包含使用 RHEL 的计算（compute）系统的集群

10.4. 在没有 OPENSHIFT UPDATE SERVICE 的断开连接的环境中更新集群

使用以下步骤在断开连接的环境中更新集群，而无需访问 OpenShift Update Service。

10.4.1. 先决条件

- 您必须安装了 **oc** 命令行界面（CLI）工具。
- 您必须使用容器镜像置备本地容器镜像 registry，如镜像 OpenShift Container Platform 镜像存储库 中所述。
- 您必须可以使用具有 **admin** 权限的用户访问集群。请参阅使用 RBAC 定义和应用权限。
- 您需要具有最新的 etcd 备份，以防因为升级失败需要将集群恢复到以前的状态。
- 确保所有机器配置池 (MCP) 都正在运行且未暂停。在更新过程中跳过与暂停 MCP 关联的节点。如果要执行 canary rollout 更新策略，可以暂停 MCP。
- 如果您的集群使用手动维护的凭证，请确保 Cloud Credential Operator（CCO）处于可升级状态。如需更多信息，请参阅为 AWS、Azure 或 GCP *手动维护凭证升级集群*。
- 如果您运行 Operator 或您已配置了 pod 中断预算，您可能会在升级过程中遇到中断。如果在 **PodDisruptionBudget** 中将 **minAvailable** 设置为 1，则节点会排空以应用可能会阻止驱除过程

的待处理机器配置。如果重启了几个节点，则所有 pod 只能有一个节点上运行，**PodDisruptionBudget** 字段可能会阻止节点排空。

10.4.2. 暂停 MachineHealthCheck 资源

在升级过程中，集群中的节点可能会临时不可用。对于 worker 节点，机器健康检查可能会认为这样的节点不健康，并重新引导它们。为避免重新引导这样的节点，请在更新集群前暂停所有 **MachineHealthCheck** 资源。

先决条件

- 安装 OpenShift CLI (**oc**) 。

流程

1. 要列出您要暂停的所有可用 **MachineHealthCheck** 资源，请运行以下命令：

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

2. 要暂停机器健康检查，请将 **cluster.x-k8s.io/paused=""** 注解添加到 **MachineHealthCheck** 资源。运行以下命令:

```
$ oc -n openshift-machine-api annotate mhc <mhc-name> cluster.x-k8s.io/paused=""
```

注解的 **MachineHealthCheck** 资源类似以下 YAML 文件：

```
apiVersion: machine.openshift.io/v1beta1
kind: MachineHealthCheck
metadata:
  name: example
  namespace: openshift-machine-api
  annotations:
    cluster.x-k8s.io/paused: ""
spec:
  selector:
    matchLabels:
      role: worker
  unhealthyConditions:
  - type:    "Ready"
    status:  "Unknown"
    timeout: "300s"
  - type:    "Ready"
    status:  "False"
    timeout: "300s"
  maxUnhealthy: "40%"
status:
  currentHealthy: 5
  expectedMachines: 5
```

重要

更新集群后恢复机器健康检查。要恢复检查，请运行以下命令从 **MachineHealthCheck** 资源中删除暂停注解：

```
$ oc -n openshift-machine-api annotate mhc <mhc-name> cluster.x-k8s.io/paused-
```

10.4.3. 升级断开连接的集群

将受限网络集群更新至您下载的发行镜像的 OpenShift Container Platform 版本。

注意

如果您有一个本地 OpenShift Update Service，您可以使用连接的 Web 控制台或 CLI 指令来更新，而不是使用此流程。

先决条件

- 您已将新发行版本的镜像镜像（mirror）到 registry。
- 您已将发行镜像签名 ConfigMap 在新发行版本中应用到集群。
- 从镜像签名 ConfigMap 中获取了发行版本的 sha256 sum 值。
- 安装 OpenShift CLI (**oc**) 。
- 暂停所有 **MachineHealthCheck** 资源。

流程

- 更新集群：

```
$ oc adm upgrade --allow-explicit-upgrade --to-image ${LOCAL_REGISTRY}/${LOCAL_REPOSITORY}<sha256_sum_value> 1
```

1 **<sha256_sum_value>** 值是镜像签名 ConfigMap 的 sha256 sum 值，例如 **@sha256:81154f5c03294534e1eaf0319BEF7a601134f891689ccede5d705ef659aa8c92**

如果镜像 registry 使用 **ImageContentSourcePolicy**，可以使用 Canonical registry 名称而非 **LOCAL_REGISTRY**。

注意

您只能为具有 **ImageContentSourcePolicy** 对象的集群配置全局 pull secret。您不能在项目中添加 pull secret。

10.4.4. 配置镜像 registry 存储库镜像

通过设置容器 registry 存储库镜像，您可以进行以下操作：

- 配置 OpenShift Container Platform 集群，以便重定向从源镜像 registry 上的存储库拉取（pull）镜像的请求，并通过已镜像 (mirror) 的镜像 registry 上的存储库来解决该请求。

- 为每个目标存储库识别多个已镜像 (mirror)的存储库，以确保如果一个镜像停止运作，仍可使用其他镜像。

OpenShift Container Platform 中存储库镜像的属性包括：

- 镜像拉取（pull）可应对 registry 停机的问题。
- 在断开连接的环境中的集群可以从关键位置（如 quay.io）拉取镜像，并让公司防火墙后面的 registry 提供请求的镜像。
- 发出镜像拉取（pull）请求时尝试特定 registry 顺序，通常最后才会尝试持久性 registry。
- 您所输入的镜像信息会添加到 OpenShift Container Platform 集群中每个节点上的 **/etc/containers/registries.conf** 文件中。
- 当节点从源存储库中请求镜像时，它会依次尝试每个已镜像的存储库，直到找到所请求的内容。如果所有镜像均失败，集群则会尝试源存储库。如果成功，则镜像拉取至节点中。

可通过以下方式设置存储库镜像：

- 在 OpenShift Container Platform 安装中：
  通过拉取（pull） OpenShift Container Platform 所需的容器镜像，然后将这些镜像放至公司防火墙后，即可将 OpenShift Container Platform 安装到受限网络中的数据中心。
- 安装 OpenShift Container Platform 后：
  即使没有在 OpenShift Container Platform 安装期间配置镜像 (mirror)，之后您仍可使用 **ImageContentSourcePolicy** 对象进行配置。

以下流程提供安装后镜像配置，您可在此处创建 **ImageContentSourcePolicy** 对象来识别：

- 您希望镜像 (mirror) 的容器镜像存储库的源。
- 您希望为其提供从源存储库请求的内容的每个镜像存储库的单独条目。

注意

您只能为具有 **ImageContentSourcePolicy** 对象的集群配置全局 pull secret。您不能在项目中添加 pull secret。

先决条件

- 使用具有 **cluster-admin** 角色的用户访问集群。

流程

1. 通过以下方法配置已镜像的存储库：
   - 按照 Red Hat Quay 存储库镜像 中所述，使用 Red Hat Quay 来设置已镜像的存储库。使用 Red Hat Quay 有助于您将镜像从一个存储库复制到另一存储库，并可随着时间的推移重复自动同步这些存储库。
   - 使用 **skopeo** 等工具手动将镜像从源目录复制到已镜像的存储库。
     例如：在 Red Hat Enterprise Linux（RHEL 7 或 RHEL 8）系统上安装 skopeo RPM 软件包后，使用 **skopeo** 命令，如下例所示：

     ```
     $ skopeo copy \
     ```

```
docker://registry.access.redhat.com/ubi8/ubi-minimal@sha256:5cfbaf45ca96806917830c183e9f37df2e913b187adb32e89fd83fa455ebaa6 \
docker://example.io/example/ubi-minimal
```

在本例中，您有一个名为 **example.io** 的容器镜像 registry，其中包含一个名为 **example** 的镜像存储库，您希望将 **ubi8/ubi-minimal** 镜像从 **registry.access.redhat.com** 复制到此镜像存储库。创建该 registry 后，您可将 OpenShift Container Platform 集群配置为将源存储库的请求重定向到已镜像的存储库。

2. 登录您的 OpenShift Container Platform 集群。

3. 创建 **ImageContentSourcePolicy** 文件（如：**registryrepomirror.yaml**），将源和镜像 (mirror) 替换为您自己的 registry、存储库对和镜像中的源和镜像：

```
apiVersion: operator.openshift.io/v1alpha1
kind: ImageContentSourcePolicy
metadata:
  name: ubi8repo
spec:
  repositoryDigestMirrors:
  - mirrors:
    - example.io/example/ubi-minimal (1)
    - example.com/example/ubi-minimal (2)
    source: registry.access.redhat.com/ubi8/ubi-minimal (3)
  - mirrors:
    - mirror.example.com/redhat
    source: registry.redhat.io/openshift4 (4)
  - mirrors:
    - mirror.example.com
    source: registry.redhat.io (5)
  - mirrors:
    - mirror.example.net/image
    source: registry.example.com/example/myimage (6)
  - mirrors:
    - mirror.example.net
    source: registry.example.com/example (7)
  - mirrors:
    - mirror.example.net/registry-example-com
    source: registry.example.com (8)
```

(1) 指明镜像 registry 和存储库的名称。

(2) 表示每个目标仓库的多个镜像仓库。如果一个镜像停机，则目标仓库可以使用另一个镜像。

(3) 指明包含所镜像内容的 registry 和存储库。

(4) 您可以在 registry 中配置命名空间以使用该命名空间中的任何镜像。如果您使用 registry 域作为源，**ImageContentSourcePolicy** 资源将应用到 registry 中的所有存储库。

(5) 如果配置 registry 名称，则 **ImageContentSourcePolicy** 资源将应用到源 registry 中的所有软件仓库。

(6) 拉取镜像 **mirror.example.net/image@sha256:...**。

7 从 mirror **mirror.example.net/myimage@sha256:...** 的源 registry 命名空间中拉取镜像 **myimage**。

8 从 mirror registry **mirror.example.net/registry-example-com/example/myimage@sha256:...** 拉取镜像 **registry.example.com/example/myimage**。**ImageContentSourcePolicy** 资源会应用到源 registry 中的所有仓库到到 mirror registry **mirror.example.net/registry-example-com**。

4. 创建新的 **ImageContentSourcePolicy** 对象：

```
$ oc create -f registryrepomirror.yaml
```

创建 **ImageContentSourcePolicy** 对象后，新的设置将部署到每个节点，集群开始使用已镜像的存储库来响应源存储库请求。

5. 要检查是否应用了已镜像的配置设置，在其中一个节点上执行以下内容。

   a. 列出您的节点：

```
$ oc get node
```

输出示例

```
NAME                         STATUS                ROLES    AGE  VERSION
ip-10-0-137-44.ec2.internal    Ready                    worker   7m   v1.24.0
ip-10-0-138-148.ec2.internal   Ready                     master   11m  v1.24.0
ip-10-0-139-122.ec2.internal   Ready                     master   11m  v1.24.0
ip-10-0-147-35.ec2.internal    Ready                    worker   7m   v1.24.0
ip-10-0-153-12.ec2.internal    Ready                    worker   7m   v1.24.0
ip-10-0-154-10.ec2.internal    Ready                    master   11m  v1.24.0
```

Imagecontentsourcepolicy 资源不会重启节点。

   b. 启动调试过程以访问节点：

```
$ oc debug node/ip-10-0-147-35.ec2.internal
```

输出示例

```
Starting pod/ip-10-0-147-35ec2internal-debug ...
To use host binaries, run `chroot /host`
```

   c. 将您的根目录改为 **/host** ：

```
sh-4.2# chroot /host
```

   d. 检查 **/etc/containers/registries.conf** 文件，确保已完成更改：

```
sh-4.2# cat /etc/containers/registries.conf
```

输出示例

```
unqualified-search-registries = ["registry.access.redhat.com", "docker.io"]
short-name-mode = ""

[[registry]]
  prefix = ""
  location = "registry.access.redhat.com/ubi8/ubi-minimal"
  mirror-by-digest-only = true

  [[registry.mirror]]
    location = "example.io/example/ubi-minimal"

  [[registry.mirror]]
    location = "example.com/example/ubi-minimal"

[[registry]]
  prefix = ""
  location = "registry.example.com"
  mirror-by-digest-only = true

  [[registry.mirror]]
    location = "mirror.example.net/registry-example-com"

[[registry]]
  prefix = ""
  location = "registry.example.com/example"
  mirror-by-digest-only = true

  [[registry.mirror]]
    location = "mirror.example.net"

[[registry]]
  prefix = ""
  location = "registry.example.com/example/myimage"
  mirror-by-digest-only = true

  [[registry.mirror]]
    location = "mirror.example.net/image"

[[registry]]
  prefix = ""
  location = "registry.redhat.io"
  mirror-by-digest-only = true

  [[registry.mirror]]
    location = "mirror.example.com"

[[registry]]
  prefix = ""
  location = "registry.redhat.io/openshift4"
  mirror-by-digest-only = true

  [[registry.mirror]]
    location = "mirror.example.com/redhat"
```

e. 将镜像摘要从源拉取到节点，并检查是否通过镜像解析。**ImageContentSourcePolicy** 对象仅支持镜像摘要，不支持镜像标签。

```
sh-4.2# podman pull --log-level=debug registry.access.redhat.com/ubi8/ubi-minimal@sha256:5cfbaf45ca96806917830c183e9f37df2e913b187adb32e89fd83fa455ebaa6
```

存储库镜像故障排除

如果存储库镜像流程未按规定工作，请使用以下有关存储库镜像如何工作的信息协助排查问题。

- 首个工作镜像用于提供拉取（pull）的镜像。
- 只有在无其他镜像工作时，才会使用主 registry。
- 从系统上下文，**Insecure** 标志用作回退。
- 最近更改了 **/etc/containers/registries.conf** 文件的格式。现在它是第 2 版，采用 TOML 格式。

10.4.5. 镜像镜像目录的范围，以减少集群节点重启的频率

您可以在存储库级别或更广泛的 registry 级别限定镜像目录。一个范围广泛的 **ImageContentSourcePolicy** 资源可减少节点在响应资源更改时需要重启的次数。

要强化 **ImageContentSourcePolicy** 资源中镜像目录的范围，请执行以下步骤。

先决条件

- 安装 OpenShift Container Platform CLI **oc**。
- 以具有 **cluster-admin** 权限的用户身份登录。
- 配置镜像镜像目录，以便在断开连接的集群中使用。

流程

1. 运行以下命令，为 **<local_registry>**、**<pull_spec>** 和 **<pull_secret_file>** 指定值：

```
$ oc adm catalog mirror <local_registry>/<pull_spec> <local_registry> -a <pull_secret_file> --icsp-scope=registry
```

其中：

<local_registry>
: 您为断开连接的集群配置的本地 registry，如 **local.registry:5000**。

<pull_spec>
: 断开连接的 registry 中配置的拉取规格，如 **redhat/redhat-operator-index:v4.9**

<pull_secret_file>
: 是 **.json** 文件格式的 **registry.redhat.io** pull secret。您可以从 Red Hat OpenShift Cluster Manager 下载 pull secret。

oc adm catalog mirror 命令创建 **/redhat-operator-index-manifests** 目录，并生成 **imageContentSourcePolicy.yaml**、**catalogSource.yaml** 和 **mapping.txt** 文件。

2. 将新的 **ImageContentSourcePolicy** 资源应用到集群：

```
$ oc apply -f imageContentSourcePolicy.yaml
```

-

验证

- 验证 **oc apply** 是否成功将更改应用到 **ImageContentSourcePolicy**:

```
$ oc get ImageContentSourcePolicy -o yaml
```

输出示例

```
apiVersion: v1
items:
- apiVersion: operator.openshift.io/v1alpha1
  kind: ImageContentSourcePolicy
  metadata:
    annotations:
      kubectl.kubernetes.io/last-applied-configuration: |

{"apiVersion":"operator.openshift.io/v1alpha1","kind":"ImageContentSourcePolicy","metadata":
{"annotations":{},"name":"redhat-operator-index"},"spec":{"repositoryDigestMirrors":
[{"mirrors":["local.registry:5000"],"source":"registry.redhat.io"}]}}
...
```

更新 **ImageContentSourcePolicy** 资源后，OpenShift Container Platform 会将新设置部署到每个节点，集群开始使用已镜像的存储库向源存储库发出请求。

10.4.6. 其他资源

- 在受限网络中使用 Operator Lifecycle Manager
- 机器配置概述

10.5. 从集群中删除 OPENSHIFT UPDATE SERVICE

要从集群中删除 OpenShift Update Service (OSUS) 的本地副本，您必须首先删除 OSUS 应用程序，然后卸载 OSUS Operator。

10.5.1. 删除 OpenShift Update Service 应用程序

您可以使用 OpenShift Container Platform Web 控制台或 CLI 删除 OpenShift Update Service 应用程序。

10.5.1.1. 使用 Web 控制台删除 OpenShift Update Service 应用程序

您可以使用 OpenShift Container Platform Web 控制台使用 OpenShift Update Service Operator 删除 OpenShift Update Service 应用程序。

先决条件

- 已安装 OpenShift Update Service Operator。

流程

1. 在 Web 控制台中，点 **Operators → Installed Operators**。

2. 从安装的 Operator 列表中选择 **OpenShift Update Service**。
3. 点 **Update Service** 选项卡。
4. 从安装的 OpenShift Update Service 应用列表中，选择要删除的应用，然后单击 **Delete UpdateService**。
5. 从 **Delete UpdateService?** 确认对话框中，单击 **Delete** 以确认删除。

10.5.1.2. 使用 CLI 删除 OpenShift Update Service 应用程序

您可以使用 OpenShift CLI（**oc**）删除 OpenShift Update Service 应用。

流程

1. 使用 OpenShift Update Service 应用程序在其中创建的命名空间获取 OpenShift Update Service 应用程序的名称，如 **openshift-update-service**：

```
$ oc get updateservice -n openshift-update-service
```

输出示例

```
NAME      AGE
service   6s
```

2. 使用上一步中的 **NAME** 值以及 OpenShift Update Service 应用程序创建命名空间删除 OpenShift Update Service 应用程序，如 **openshift-update-service**：

```
$ oc delete updateservice service -n openshift-update-service
```

输出示例

```
updateservice.updateservice.operator.openshift.io "service" deleted
```

10.5.2. 卸载 OpenShift Update Service Operator

您可以使用 OpenShift Container Platform Web 控制台或 CLI 卸载 OpenShift Update Service Operator。

10.5.2.1. 使用 Web 控制台卸载 OpenShift Update Service Operator

您可以使用 OpenShift Container Platform Web 控制台卸载 OpenShift Update Service Operator。

先决条件

- 所有 OpenShift Update Service 应用都已删除。

流程

1. 在 Web 控制台中，点 **Operators → Installed Operators**。
2. 从安装的 Operator 列表中选择 **OpenShift Update Service** 并点 **Uninstall Operator**。

3. 在 **Uninstall Operator?** 确认对话框中点 **Uninstall** 确认卸载。

10.5.2.2. 使用 CLI 卸载 OpenShift Update Service Operator

您可以使用 OpenShift CLI（**oc**）卸载 OpenShift Update Service Operator。

先决条件

- 所有 OpenShift Update Service 应用都已删除。

流程

1. 更改到包含 OpenShift Update Service Operator 的项目，如 **openshift-update-service**：

```
$ oc project openshift-update-service
```

输出示例

```
Now using project "openshift-update-service" on server "https://example.com:6443".
```

2. 获取 OpenShift Update Service Operator operator 组的名称：

```
$ oc get operatorgroup
```

输出示例

```
NAME                              AGE
openshift-update-service-fprx2   4m41s
```

3. 删除 operator 组，如 **openshift-update-service-fprx2**：

```
$ oc delete operatorgroup openshift-update-service-fprx2
```

输出示例

```
operatorgroup.operators.coreos.com "openshift-update-service-fprx2" deleted
```

4. 获取 OpenShift Update Service Operator 订阅的名称：

```
$ oc get subscription
```

输出示例

```
NAME                        PACKAGE                     SOURCE                       CHANNEL
update-service-operator   update-service-operator   updateservice-index-catalog   v1
```

5. 使用上一步中的 **Name** 值，在 **currentCSV** 字段中检查订阅的 OpenShift Update Service Operator 的当前版本：

```
$ oc get subscription update-service-operator -o yaml | grep " currentCSV"
```

输出示例

```
  currentCSV: update-service-operator.v0.0.1
```

6. 删除订阅，如 **update-service-operator**：

```
$ oc delete subscription update-service-operator
```

输出示例

```
subscription.operators.coreos.com "update-service-operator" deleted
```

7. 使用上一步中的 **currentCSV** 值删除 OpenShift Update Service Operator 的 CSV：

```
$ oc delete clusterserviceversion update-service-operator.v0.0.1
```

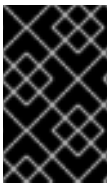
输出示例

```
clusterserviceversion.operators.coreos.com "update-service-operator.v0.0.1" deleted
```

第 11 章 更新在 VSPHERE 上运行的节点上运行的硬件

您必须确保您在 vSphere 中运行的节点在 OpenShift Container Platform 支持的硬件版本上运行。目前，硬件版本 13 或更高版本都支持集群中的 vSphere 虚拟机。

您可以立即更新虚拟硬件，或在 vCenter 中计划更新。



重要

在 vSphere 上运行的集群节点使用硬件版本 13 现已弃用。这个版本仍然被完全支持，但在以后的 OpenShift Container Platform 版本中将会删除支持。现在，硬件版本 15 是 OpenShift Container Platform 中 vSphere 虚拟机的默认版本。

11.1. 更新 VSPHERE 上的虚拟硬件

要在 VMware vSphere 上更新虚拟机 (VM) 的硬件，请单独更新您的虚拟机，以减少集群停机风险。

11.1.1. 为 vSphere 上的 control plane 节点更新虚拟硬件

要减少停机的风险，建议按顺序更新 control plane 节点。这样可确保 Kubernetes API 保持可用，etcd 保留仲裁。

先决条件

- 在托管 OpenShift Container Platform 集群的 vCenter 实例中具有执行所需权限的权限。
- 您的 vSphere ESXi 主机是 6.7U3 或更高版本。

流程

1. 列出集群中的 control plane 节点。

```
$ oc get nodes -l node-role.kubernetes.io/master
```

输出示例

```
NAME                 STATUS   ROLES    AGE   VERSION
control-plane-node-0    Ready    master   75m   v1.22.1
control-plane-node-1    Ready    master   75m   v1.22.1
control-plane-node-2    Ready    master   75m   v1.22.1
```

请注意 control plane 节点的名称。

2. 将 control plane 节点标记为不可调度。

```
$ oc adm cordon <control_plane_node>
```

3. 关闭与 control plane 节点关联的虚拟机 (VM)。在 vSphere 客户端中通过右键单击虚拟机并选择 **Power → Shut Down Guest OS**进行此操作。不要使用 **Power Off** 来关闭虚拟机，因为它可能无法安全地关闭。

4. 更新 vSphere 客户端中的虚拟机。如需更多信息，请参阅 VMware 文档中的将虚拟机升级到最新硬件版本。

5. 打开与 control plane 节点关联的虚拟机。在 vSphere 客户端中通过右键单击虚拟机并选择 **Power On** 来进行此操作。

6. 等待节点报告为 **Ready** ：

```
$ oc wait --for=condition=Ready node/<control_plane_node>
```

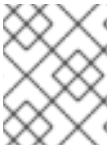
7. 再次将 control plane 节点标记为可以调度：

```
$ oc adm uncordon <control_plane_node>
```

8. 对集群中的每个 control plane 节点重复此步骤。

11.1.2. 更新 vSphere 上计算节点的虚拟硬件

要降低停机的风险，建议按顺序更新计算节点。



注意

可以在并行给定工作负载中更新多个计算节点，可以接受具有 **NotReady** 状态的多个节点。管理员负责确保所需的计算节点可用。

先决条件

- 在托管 OpenShift Container Platform 集群的 vCenter 实例中具有执行所需权限的权限。
- 您的 vSphere ESXi 主机是 6.7U3 或更高版本。

流程

1. 列出集群中的计算节点。

```
$ oc get nodes -l node-role.kubernetes.io/worker
```

输出示例

```
NAME            STATUS   ROLES    AGE   VERSION
compute-node-0   Ready    worker   30m   v1.22.1
compute-node-1   Ready    worker   30m   v1.22.1
compute-node-2   Ready    worker   30m   v1.22.1
```

注意计算节点的名称。

2. 将计算节点标记为不可调度：

```
$ oc adm cordon <compute_node>
```

3. 从计算节点撤离容器集。执行此操作有多种方法。例如，您可以撤离节点上的所有或选定 pod：

```
$ oc adm drain <compute_node> [--pod-selector=<pod_selector>]
```

如需从节点上撤离 pod 的其他选项，请参阅"如何撤离节点上的 pod"部分。

4. 关闭与计算节点关联的虚拟机 (VM)。在 vSphere 客户端中通过右键单击虚拟机并选择 **Power → Shut Down Guest OS**进行此操作。不要使用 **Power Off** 来关闭虚拟机，因为它可能无法安全地关闭。

5. 更新 vSphere 客户端中的虚拟机。如需更多信息，请参阅 VMware 文档中的将虚拟机升级到最新硬件版本。

6. 打开与计算节点关联的虚拟机。在 vSphere 客户端中通过右键单击虚拟机并选择 **Power On** 来进行此操作。

7. 等待节点报告为 **Ready** ：

```
$ oc wait --for=condition=Ready node/<compute_node>
```

8. 将计算节点再次标记为可调度：

```
$ oc adm uncordon <compute_node>
```

9. 对集群中的每个计算节点重复此步骤。

11.1.3. 为 vSphere 上的模板更新虚拟硬件

先决条件

- 在托管 OpenShift Container Platform 集群的 vCenter 实例中具有执行所需权限的权限。
- 您的 vSphere ESXi 主机是 6.7U3 或更高版本。

流程

1. 如果 RHCOS 模板被配置为 vSphere 模板，请在进行下一步前参阅将模板转换为一个虚拟机。

注意

从模板转换后，请不要打开虚拟机电源。

1. 更新 vSphere 客户端中的虚拟机。如需更多信息，请参阅 VMware 文档中的将虚拟机升级到最新硬件版本。

2. 将 vSphere 客户端中的虚拟机从虚拟机转换为模板。如需更多信息，请参阅 VMware 文档中的将虚拟机转换为 vSphere 客户端中的模板。

其他资源

- 了解如何撤离节点上的 pod

11.2. 在 VSPHERE 上调度虚拟硬件的更新

虚拟机开启或重新启动时，可以计划进行虚拟硬件更新。您可以按照 VMware 文档中的虚拟机兼容性升级调度专门在 vCenter 中调度虚拟硬件更新。

在 OpenShift Container Platform 升级前调度升级时，在 OpenShift Container Platform 升级过程中重启节点时会进行虚拟硬件更新。