OpenShift Container Platform 4.19

容器镜像仓库（Registry）

为 OpenShift Container Platform 配置容器镜像仓库（Registry）

Last Updated: 2025-12-11

OpenShift Container Platform 4.19 容器镜像仓库（Registry）

为 OpenShift Container Platform 配置容器镜像仓库（Registry）

Abstract

本文档提供有关为 OpenShift Container Platform 配置和管理内部 registry 的说明。它还介绍了与 OpenShift Container Platform 相关的 registry 的一般信息。

Table of Contents

第 1 章 {PRODUCT-REGISTRY} 概述

OpenShift Container Platform可以使用您的源代码构建镜像，并进行部署及管理其生命周期。OpenShift Container Platform 提供了一个内部集成的容器镜像 registry，可在 OpenShift Container Platform 环境中部署以在本地管理镜像。概述部分包括 OpenShift 镜像 registry 参考信息和 OpenShift Container Platform 常用的 registry 链接。

1.1. OPENSHIFT 镜像 REGISTRY 常用术语表

该术语表定义了 registry 内容中使用的常用术语。

container

包括软件及其所有依赖项的轻量级和可执行镜像。由于容器虚拟化操作系统，因此您可以在数据中心、公有云或私有云或本地主机中运行容器。

镜像仓库

镜像仓库是相关容器镜像和标识它们的标签（tag）的集合。

镜像 registry

mirror registry 是一个 registry，其中包含 OpenShift Container Platform 镜像的 mirror。

namespace

命名空间隔离单个集群中的一组资源。

pod

pod 是 Kubernetes 中的最小逻辑单元。pod 由一个或多个容器组成，可在一个计算节点中运行。

私有 registry

registry 是实现容器镜像 registry API 的服务器。私有 registry 是需要身份验证的 registry，以便用户可以访问 registry 的内容。

公共 registry

registry 是实现容器镜像 registry API 的服务器。公共 registry 是一个公开提供其内容的 registry。

Quay.io

由红帽提供和维护的公共 Red Hat Quay Container Registry 实例，为 OpenShift Container Platform 集群提供大多数容器镜像和 Operator。

OpenShift 镜像 registry

OpenShift 镜像 registry 是 OpenShift Container Platform 提供的 registry，用于管理镜像。

registry 身份验证

要将镜像推送 (push) 到私有镜像仓库，registry 需要根据凭据验证其用户。

route

公开服务，以允许从 OpenShift Container Platform 实例外的用户和应用程序对 pod 进行网络访问。

缩减（scale down）

减少副本数。

扩展（scale up）

增加副本数量。

service

服务在一组 pod 上公开正在运行的应用程序。

1.2. 集成的 OPENSHIFT 镜像 REGISTRY

OpenShift Container Platform 提供了一个内建的镜像 registry，它作为一个标准的工作负载在集群中运行。这个 registry 由一个 infrastructure Operator 配置并管理。registry 提供了在现有集群基础架构上运行的开箱即用的解决方案，以便您可以管理运行工作负载的镜像。

您可以像任何其他集群工作负载一样扩展或缩减 registry，且 registry 不需要特定的基础架构置备。registry 也集成到集群用户身份验证和授权系统中，这意味着定义镜像资源上的用户权限控制对创建和检索镜像的访问。

该 registry 通常作为集群中构建的镜像的发布目标，以及在集群中运行的工作负载的镜像源。当您将新镜像推送到 registry 时，会将通知发送到新镜像的集群，以便其他组件可以响应和使用更新的镜像。

镜像数据会存储在两个位置。实际镜像数据存储在可配置的存储位置，例如云存储或一个文件系统卷中。镜像的元数据被保存为标准的API资源（镜像(image)及镜像流(imagestream)），它们可以通过标准的集群 API 进行访问。

其他资源

- OpenShift Container Platform中的Image Registry Operator

1.3. 自动修剪镜像

因为年龄、状态或超过限制，已不再被系统需要的来自 OpenShift 镜像 registry 的镜像会被自动修剪。作为集群管理员，可以配置或暂停修剪自定义资源(CR)。

先决条件

- 您可以使用具有集群管理员权限的账户访问 OpenShift Container Platform 集群。
- 安装 **oc** CLI。

重要

用于管理修剪器的 Image Registry Operator 的行为与在 Image Registry Operator 的 **ClusterOperator** 对象上指定的 **managementState** 无关。如果 Image Registry Operator 没有处于 **Managed** 状态，则镜像修剪器仍然可以被 Pruning Custom Resource 配置和管理。

但是，Image Registry Operator 的 **managementState** 会更改部署的镜像修剪器任务的行为：

- **Managed**: 镜像修剪器的 **--prune-registry** 标志被设置为 **true**。
- **Removed**: 镜像修剪器的 **--prune-registry** 标志被设置为 **false**，这意味着它只在 etcd 中修剪镜像元数据。

流程

- 验证名为 **imagepruners.imageregistry.operator.openshift.io/cluster** 的项包括以下 **spec** 和 **status** 字段：

```
spec:
  schedule: 0 0 * * *
  suspend: false
  keepTagRevisions: 3
  keepYoungerThanDuration: 60m
```

```
    keepYoungerThan: 3600000000000
    resources: {}
    affinity: {}
    nodeSelector: {}
    tolerations: []
    successfulJobsHistoryLimit: 3
    failedJobsHistoryLimit: 3
  status:
    observedGeneration: 2
    conditions:
    - type: Available
      status: "True"
      lastTransitionTime: 2019-10-09T03:13:45
      reason: Ready
      message: "Periodic image pruner has been created."
    - type: Scheduled
      status: "True"
      lastTransitionTime: 2019-10-09T03:13:45
      reason: Scheduled
      message: "Image pruner job has been scheduled."
    - type: Failed
      staus: "False"
      lastTransitionTime: 2019-10-09T03:13:45
      reason: Succeeded
      message: "Most recent image pruning job succeeded."
```

- **schedule**: **CronJob** 格式的调度。这是可选字段，默认为每日的午夜。
- **suspend**: 如果设置为 **true**，**CronJob** 运行的修建操作会被挂起。这是可选字段，默认为 **false**。新集群上的初始值为 **false**。
- **keepTagRevisions**：要保留的每个标签的修订版本数量。这是可选字段，默认为 **3**。初始值为 **3**。
- **keepYoungerDuration**：保留比此时间段更早的镜像。这是可选字段。如果没有指定值，则使用 **keepYoungerThan**，或默认值 **60m**（60分钟）。
- **keepYoungerThan**：已弃用。与 **keepYoungerThanDuration** 相同，但持续时间被指定为纳秒的整数。这是可选字段。当设置 **keepYoungerThanDuration** 时，会忽略此字段。
- **资源**：标准 pod 资源请求和限值。这是可选字段。
- **affinity**：标准 pod 关联性。这是可选字段。
- **nodeSelector**：标准 pod 节点选择器。这是可选字段。
- **tolerations**：标准 pod 容限。这是可选字段。
- **successfulJobsHistoryLimit**：要保留的作业的最大值。必须大于或等于 **1** 以确保报告指标。这是可选字段，默认为 **3**。初始值为 **3**。
- **failedJobsHistoryLimit**：要保留的最大失败作业数。必须大于或等于 **1** 以确保报告指标。这是可选字段，默认为 **3**。初始值为 **3**。
- **observedGeneration**: Operator 观察到的生成。

- **条件**：带有以下类型的标准条件对象：
    - **可用**：指示修剪任务是否已创建。原因可以是 **Ready** 或 **Error**。
    - **调度**：指示是否调度了下一个修剪任务。原因可以是 **Scheduled**, **Suspended**, 或 **Error**。
    - **失败**：指示最新修剪任务是否失败。

1.4. 使用来自第三方 REGISTRY 的镜像创建容器

有些容器镜像 registry 需要访问授权。Podman 是一个开源工具，用于管理容器和容器镜像，并与镜像 registry 交互。您可以使用 Podman 来验证凭证、拉取 registry 镜像，并将本地镜像存储在本地文件系统中。此流程提供了使用 Podman 验证 registry 的通用示例。

OpenShift Container Platform 使用用户提供的凭证与 registry 进行联系来访问私有镜像仓库。这样，OpenShift Container Platform就可以对私有仓库进行镜像的 push 和 pull 操作。

OpenShift Container Platform 可以使用由第三方 registry 提供的镜像创建容器，但这些 registry 可能不会象集成的 OpenShift 镜像 registry 一样提供相同的镜像通知支持。在这种情况下，OpenShift Container Platform 会在创建镜像流时从远程 registry 获取标签。要刷新获取的标签，请运行 **oc import-image <stream>**。当检测到新的镜像时，以前的构建和部署将会被重新创建。

流程

1. 使用红帽生态系统目录从红帽仓库搜索特定容器镜像并选择所需的镜像。
2. 点 **Get this image** 来查找您的容器镜像的命令。
3. 运行以下命令，并输入您的用户名和密码进行验证。显示的输出示例用于演示目的。

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

    ```
    Username:<your_registry_account_username>
    Password:<your_registry_account_password>
    ```

4. 运行以下命令下载镜像并将其保存在本地：

    ```
    $ podman pull registry.redhat.io/<repository_name>
    ```

1.5. RED HAT QUAY REGISTRIES

Red Hat Quay 为您提供了一个企业级的容器镜像 registry。它可以作为一个托管的服务，也可以在您自己的数据中心或环境中安装它。Red Hat Quay 中的高级功能包括跨区域复制、镜像扫描及镜像回滚（roll back）功能。

请通过 Quay.io 网站设置您自己的托管 Quay registry 帐户。之后，请按照Quay教程中的内容登录到Quay registry并开始管理镜像。

您可以象访问其他远程镜像 registry 一样，通过 OpenShift Container Platform 访问您的 Red Hat Quay registry。

其他资源

- Red Hat Quay 产品文档

1.6. 启用了身份验证的红帽 REGISTRY

红帽生态系统目录的容器镜像部分提供的所有容器镜像都托管在镜像 registry 上（**registry.redhat.io**）。registry（**Registry.redhat.io**）需要进行身份验证才能访问OpenShift Container Platform上的镜像及内容。

注意

OpenShift Container Platform从**Registry.redhat.io**中提取（pull）镜像，因此需要配置集群以使用它。

新registry使用标准的OAuth机制进行身份验证：

- **身份验证令牌。**令牌（token）是服务帐户，由管理员生成。系统可以使用它们与容器镜像 registry进行身份验证。服务帐户不受用户帐户更改的影响，因此使用令牌进行身份验证是一个可靠且具有弹性的方法。这是生产环境集群中唯一受支持的身份验证选项。
- **Web用户名和密码。**这是用于登录到诸如**access.redhat.com**之类的资源的标准凭据集。虽然您可以在 OpenShift Container Platform 中使用此验证方法，但红帽并不支持在生产环境部署中使用这个方法。此身份验证方法应该只限于在OpenShift Container Platform之外的独立项目中使用。

您可以在 **podman login** 中使用您的凭证（用户名和密码，或身份验证令牌）来访问新 registry 中的内容。

所有镜像流都指向使用安装 pull secret 进行身份验证的新 registry。

您必须将凭证放在以下任一位置：

- **OpenShift 命名空间**。您的凭证必须存在于 **openshift** 命名空间中，以便 **openshift** 命名空间中的镜像流可以被导入。
- **您的主机**。您的凭据必须存在于主机上，因为在抓取（pull）镜像时，Kubernetes会使用主机中的凭据。

其他资源

- registry 服务帐户

第 2 章 OPENSHIFT CONTAINER PLATFORM中的IMAGE REGISTRY OPERATOR

2.1. 云平台和 OPENSTACK 上的镜像 REGISTRY

Image Registry Operator 安装一个单独的 OpenShift 镜像 registry 实例，并对 registry 的所有配置进行管理（包括设置 registry 存储）。

注意

只有在 AWS、Azure、Google Cloud、IBM® 或 RHOSP 上安装安装程序置备的基础架构集群时，存储才会被自动配置。

当您在 AWS、Azure、Google Cloud、IBM® 或 RHOSP 上安装或升级安装程序置备的基础架构集群时，Image Registry Operator 会将 **spec.storage.managementState** 参数设置为 **Managed**。如果 **spec.storage.managementState** 参数设置为 **Unmanaged**，则 Image Registry Operator 不会执行与存储相关的操作。

在管理集群中部署 control plane 后，Operator 会根据集群中的配置创建一个默认的 **configs.imageregistry.operator.openshift.io** 自定义资源（CR）实例。

如果没有足够的信息来定义完整的**configs.imageregistry.operator.openshift.io**CR，则将定义不完整的资源，Operator 将更新资源状态以提供缺失的内容。

重要

Image Registry Operator 管理修剪器的行为与在 Image Registry Operator 的 **ClusterOperator** 对象上指定的 **managementState** 关联。如果 Image Registry Operator 没有处于 **Managed** 状态，则镜像修剪器仍然可以被 **Pruning** 自定义资源配置和管理。

但是，Image Registry Operator 的 **managementState** 会更改部署的镜像修剪器任务的行为：

- **Managed**: 镜像修剪器的 **--prune-registry** 标志被设置为 **true**。
- **Removed**：镜像修剪器的 **--prune-registry** 标志被设置为 **false**，这意味着镜像修剪器作业仅在 etcd 中修剪镜像元数据。

2.2. 裸机、NUTANIX 和 VSPHERE 上的镜像 REGISTRY

2.2.1. 安装过程中删除的镜像 registry

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

安装后，您必须编辑 Image Registry Operator 配置，将 **managementState** 从 **Removed** 切换到 **Managed**。完成此操作后，您必须配置存储。

2.3. IMAGE REGISTRY OPERATOR 在可用区间分布

Image Registry Operator 的默认配置现在将镜像 registry pod 分散到拓扑区中，以防止在完全区失败时造成延迟恢复时间，因为所有 pod 都会受到影响。引用以下 YAML，以了解当 Operator 使用与区相关的拓扑约束部署时，Image Registry Operator 使用的默认参数值：

```yaml
topologySpreadConstraints:
- labelSelector:
    matchLabels:
      docker-registry: default
  maxSkew: 1
  topologyKey: kubernetes.io/hostname
  whenUnsatisfiable: DoNotSchedule
- labelSelector:
    matchLabels:
      docker-registry: default
  maxSkew: 1
  topologyKey: node-role.kubernetes.io/worker
  whenUnsatisfiable: DoNotSchedule
- labelSelector:
    matchLabels:
      docker-registry: default
  maxSkew: 1
  topologyKey: topology.kubernetes.io/zone
  whenUnsatisfiable: DoNotSchedule
```

引用以下 YAML 以了解当 Operator 使用与区相关的拓扑约束部署时，Image Registry Operator 使用的默认参数值，它适用于裸机和 vSphere 实例：

```yaml
topologySpreadConstraints:
- labelSelector:
    matchLabels:
      docker-registry: default
  maxSkew: 1
  topologyKey: kubernetes.io/hostname
  whenUnsatisfiable: DoNotSchedule
- labelSelector:
    matchLabels:
      docker-registry: default
  maxSkew: 1
  topologyKey: node-role.kubernetes.io/worker
  whenUnsatisfiable: DoNotSchedule
```

作为集群管理员。您可以通过配置 **configs.imageregistry.operator.openshift.io/cluster** spec 文件来覆盖默认的 **topologySpreadConstraints** 部分值。

2.4. 其他资源

- 配置 pod 拓扑分布限制

2.5. IMAGE REGISTRY OPERATOR 配置参数

您可以使用 **configs.imageregistry.operator.openshift.io** 资源配置 Image Registry Operator。资源提供用于管理 registry 状态、存储、日志记录、路由和部署设置的参数。

参数	描述
managementState	**Managed**: Operator 在资源被更新时对 registry 进行相应更新。 **Unmanaged**: Operator 忽略配置资源的改变。 **Removed**: Operator 删除 registry 实例，并清除所有由 Operator 置备的存储。
logLevel	设置 registry 实例的 **logLevel**。默认为 **Normal**。 支持 **logLevel** 的以下值： • **Normal** • **Debug** • **Trace** • **TraceAll**
httpSecret	安全上载时 registry 所需的值，默认生成。
operatorLogLevel	**operatorLogLevel** 配置参数为 Operator 本身提供基于意图的日志记录，以及管理 Operator 本身必须解释的粗粒度日志记录选项的简单方法。此配置参数默认为 **Normal**。它没有提供精细的控制。 支持 **operatorLogLevel** 的以下值： • **Normal** • **Debug** • **Trace** • **TraceAll**
proxy	定义在调用 master API 和上游 registry 时要使用的代理。
affinity	您可以使用 **affinity** 参数为 Image Registry Operator pod 配置 pod 调度首选项和限制。 关联性设置可以使用 **podAffinity** 或 **podAntiAffinity** spec。两个选项都可以使用 **preferredDuringSchedulingIgnoredDuringExecution** 规则或 **requiredDuringSchedulingIgnoredDuringExecution** 规则。
storage	**Storagetype** ：用于配置 registry 存储的详细信息，例如S3 bucket坐标。通常会被默认配置。
readOnly	registry 实例是否应该拒绝推送新镜像或删除现有镜像的尝试。

参数	描述
requests	API Request Limit 详情。控制在把请求放入队列前，registry 实例可以并行处理的请求数量。
defaultRoute	确定是否使用默认主机名定义外部路由。如果启用，该路由将会对加密进行重新加密。默认值为 **false**。
Routes	要创建的其他路由。您需要提供路由的主机名和证书。
rolloutStrategy	为镜像 registry 部署定义 rollout 策略。默认为 **RollingUpdate**。
replicas	registry 的副本数量。
disableRedirect	控制是否通过 registry 路由所有数据，而不是重定向到后端。默认值为 **false**。
spec.storage.managementState	在 AWS 或 Azure 的安装程序置备的基础架构中的新安装或升级的集群中，Image Registry Operator 会把 **spec.storage.managementState** 参数设置为 **Managed**。 ● **Managed**: 确定 Image Registry Operator 管理底层存储。如果 Image Registry Operator 的 **managementState** 被设置为 **Removed**，则存储将被删除。 ○ 如果 **managementState** 设为 **Managed**，Image Registry Operator 会尝试对底层存储单元应用一些默认配置。例如，如果设置为 **Managed**，Operator 会尝试在 S3 存储桶上启用加密，然后提供给 registry。如果您不希望默认设置应用到您提供的存储中，请确保将 **managementState** 设置为 **Unmanaged**。 ● **Unmanaged**：确定 Image Registry Operator 忽略存储设置。如果 Image Registry Operator 的 **managementState** 设置为 **Removed**，则存储不会被删除。如果您提供了底层存储单元配置，如存储桶或容器名称，并且 **spec.storage.managementState** 尚未设置为任何值，则 Image Registry Operator 会将其配置为 **Unmanaged**。

2.6. 使用 CRD 启用 IMAGE REGISTRY 默认路由

在 OpenShift Container Platform 中，**Registry** Operator 控制 OpenShift 镜像 registry 功能，并在 **configs.imageregistry.operator.openshift.io** 自定义资源定义 (CRD) 中定义此 Operator。如果您需要自动启用Image Registry默认路由，请对 Image Registry Operator CRD 进行 patch 处理。

流程

- 对 Image Registry Operator CRD 进行 patch 处理:

```
$ oc patch configs.imageregistry.operator.openshift.io/cluster --type merge -p '{"spec":{"defaultRoute":true}}'
```

2.7. 为镜像 REGISTRY 访问配置额外的信任存储

您可以在镜像 registry 访问 **image.config.openshift.io/cluster** 自定义资源(CR)时添加对具有信任额外证书颁发机构(CA)的配置映射的引用。

先决条件

- 证书颁发机构(CA)必须经过 PEM 编码。

流程

1. 在 **openshift-config** 命名空间中创建配置映射，然后在 **image.config.openshift.io** CR 的 **AdditionalTrustedCA** 参数中使用配置映射名称。这会添加集群联系外部镜像 registry 时应信任的 CA。

   镜像 registry CA 配置映射示例

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

   其中：

   data:registry.example.com:
   要信任此 CA 的 registry 主机名示例。

   data:registry-with-port.example.com..5000:
   信任此 CA 的端口的 registry 主机名示例。如果 registry 有一个端口，如 **registry-with-port.example.com:5000**，**:** 必须替换为 **..**。
   PEM 证书内容是要信任的每个额外 registry CA 的值。

2. 可选。运行以下命令来配置额外的 CA：

   ```
   $ oc create configmap registry-config --from-file=<external_registry_address>=ca.crt -n openshift-config
   ```

   ```
   $ oc edit image.config.openshift.io cluster
   ```

   ```
   spec:
     additionalTrustedCA:
       name: registry-config
   ```

2.8. 为 IMAGE REGISTRY OPERATOR 配置一个存储凭证

除了 **configs.imageregistry.operator.openshift.io** 自定义资源 (CR) 和 ConfigMap 资源外，还通过单独的 secret 资源为 Operator 提供存储凭证配置。此资源位于 **openshift-image-registry** 命名空间中。

您可以创建一个 **image-registry-private-configuration-user** secret，它会创建存储访问和管理所需的自定义凭证。如果存在默认凭证，自定义凭证会覆盖 Operator 使用的默认凭证。

流程

- 创建一个包括了所需键的 OpenShift Container Platform secret。

```
$ oc create secret generic image-registry-private-configuration-user --from-literal=KEY1=value1 --from-literal=KEY2=value2 --namespace openshift-image-registry
```

2.9. 其他资源

- 为 AWS 用户置备的基础架构配置 registry
- 为 Google Cloud 用户置备的基础架构配置 registry
- 为 Azure 用户置备的基础架构配置 registry
- 为裸机配置 registry
- 为 vSphere 配置 registry
- 为 RHOSP 配置 registry
- 为 Red Hat OpenShift Data Foundation 配置 registry
- 为 Nutanix 配置 registry

第 3 章 设置和配置 REGISTRY

3.1. 为 AWS 用户置备的基础架构配置 REGISTRY

通过将内置镜像 registry 配置为使用专用 AWS 存储，将容器镜像保存到持久存储位置。此设置为您的 registry 提供持久的可扩展存储，与临时集群存储分开。

3.1.1. 为 Image Registry Operator 配置一个 secret

除了**configs.imageregistry.operator.openshift.io**及 ConfigMap 资源外，还可以通过**openshift-image-registry**命名空间中的独立的 secret 资源为 Operator 提供其他配置。

image-registry-private-configuration-user secret 提供了存储访问和管理所需的凭证。如果找到默认凭据，它将覆盖 Operator 使用的默认凭据。

对于 AWS 上的 S3 存储，secret 应该包含以下两个键：

- **REGISTRY_STORAGE_S3_ACCESSKEY**
- **REGISTRY_STORAGE_S3_SECRETKEY**

流程

- 创建一个包括了所需键的 OpenShift Container Platform secret。

```
$ oc create secret generic image-registry-private-configuration-user --from-literal=REGISTRY_STORAGE_S3_ACCESSKEY=myaccesskey --from-literal=REGISTRY_STORAGE_S3_SECRETKEY=mysecretkey --namespace openshift-image-registry
```

3.1.2. 为使用用户置备的基础架构的AWS配置registry存储

在安装过程中，使用您的云凭据就可以创建一个 Amazon S3 存储桶，Registry Operator 将会自动配置存储。

如果 Registry Operator 无法创建 S3 存储桶或自动配置存储，您可以按照以下流程创建 S3 存储桶并配置存储。

> **警告**
>
> 为了保护 AWS 中 registry 镜像的安全，阻止对 S3 存储桶的公共访问。

先决条件

- 在带有用户置备的基础架构的 AWS 上有一个集群。
- 对于 Amazon S3 存储，secret 应该包含以下两个键：
    - **REGISTRY_STORAGE_S3_ACCESSKEY**

- **REGISTRY_STORAGE_S3_SECRETKEY**

流程

1. 设置一个 Bucket Lifecycle Policy 用来终止已有一天之久的未完成的分段上传操作。
2. 在**configs.imageregistry.operator.openshift.io/cluster**中中输入存储配置：

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

配置示例

```
apiVersion: imageregistry.operator.openshift.io/v1
kind: Config
metadata:
  name: cluster
spec:
  storage:
    s3:
      bucket: <bucket_name>
      region: <region_name>
```

3.1.3. AWS S3 的 Image Registry Operator 配置参数

以下配置参数可用于AWS S3 registry 存储。

镜像 registry **spec.storage.s3** 配置参数包含用于将 registry 配置为使用 AWS S3 服务进行后端存储的信息。如需更多信息，请参阅 S3 存储驱动程序文档。

参数	描述
bucket	要在其中存储 registry 数据的存储桶名称。它是可选的，如果未提供会自动生成。
chunkSizeMiB	ChunkSizeMiB 是 S3 API 的多部分上传块的大小。默认值为 **10** MiB，最少为 **5** MiB。
region	存储桶所在的 AWS 区。它是可选的，并根据已安装的 AWS 区域进行设置。
regionEndpoint	RegionEndpoint 是 S3 兼容存储服务的端点。它是可选的，默认基于提供的 Region。
virtualHostedStyle	VirtualHostedStyle 启用 S3 虚拟主机风格的存储桶路径，使用自定义的 RegionEndpoint。它是可选的，默认为 false。 设置此参数以将 OpenShift Container Platform 部署到隐藏的区域。
encrypt	用来指定 registry 是否以加密的形式存储镜像。它是可选的，默认为 false。
keyID	KeyID 是用于加密的 KMS 密钥 ID。它是可选的。encrypt 必须为 true，否则此参数将被忽略。

参数	描述
cloudFront	CloudFront 将 Amazon Cloudfront 配置为 registry 中的存储中间件。它是可选的。
trustedCA	**trustedCA** 引用的配置映射的命名空间是 **openshift-config**。配置映射中的捆绑包键是 **ca-bundle.crt**。它是可选的。

> **注意**
>
> 当 **regionEndpoint** 参数的值配置为 Rados 网关的 URL 时，必须指定显式端口。例如：
>
> ```
> regionEndpoint: http://rook-ceph-rgw-ocs-storagecluster-cephobjectstore.openshift-storage.svc.cluster.local
> ```

3.2. 为 GOOGLE CLOUD 用户置备的基础架构配置 REGISTRY

通过将内置镜像 registry 配置为使用专用 Google Cloud 存储，将容器镜像保存到持久存储位置。此设置为您的 registry 提供持久的可扩展存储，与临时集群存储分开。

3.2.1. 为 Image Registry Operator 配置一个 secret

除了**configs.imageregistry.operator.openshift.io**及 ConfigMap 资源外，还可以通过**openshift-image-registry**命名空间中的独立的 secret 资源为 Operator 提供其他配置。

image-registry-private-configuration-user secret 提供了存储访问和管理所需的凭证。如果找到默认凭据，它将覆盖 Operator 使用的默认凭据。

对于 Google Cloud 存储上的 GCS，secret 应该包含以下这个键，其值是 Google Cloud 提供的凭据文件的内容：

- **REGISTRY_STORAGE_GCS_KEYFILE**

流程

- 创建一个包括了所需键的 OpenShift Container Platform secret。

```
$ oc create secret generic image-registry-private-configuration-user --from-file=REGISTRY_STORAGE_GCS_KEYFILE=<path_to_keyfile> --namespace openshift-image-registry
```

3.2.2. 使用用户置备的基础架构为 Google Cloud 配置 registry 存储

如果 Registry Operator 无法创建 Google Cloud 存储桶，您必须手动设置存储介质，并在 registry 自定义资源(CR)中配置设置。

先决条件

- 使用用户置备的 Google Cloud 中的一个集群。
- 要为 Google Cloud 配置 registry 存储，您需要提供 Registry Operator 云凭据。
- 对于 Google Cloud 存储上的 GCS，secret 应该包含以下这个键，其值是 Google Cloud 提供的凭据文件的内容：
    - **REGISTRY_STORAGE_GCS_KEYFILE**

> **警告**
>
> 您可以通过设置 公共访问防止来保护使用 Google Cloud Storage 存储桶的 registry 镜像。

流程

1. 设置 对象生命周期管理策略，以中止过去一天的未完成的多部分上传。
2. 在**configs.imageregistry.operator.openshift.io/cluster**中中输入存储配置：

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

配置示例

```
apiVersion: imageregistry.operator.openshift.io/v1
kind: Config
metadata:
  name: cluster
spec:
  storage:
    gcs:
      bucket: <bucket_name>
      projectID: <project_id>
      region: <region_name>
```

3.2.3. Google Cloud GCS 的 Image Registry Operator 配置参数

以下配置参数可用于 Google Cloud GCS registry 存储。

参数	描述
bucket	要在其中存储 registry 数据的存储桶名称。它是可选的，如果未提供会自动生成。
region	存储桶所在的 GCS 的区。它是可选的，并根据已安装的 GCS 区域进行设置。
projectID	ProjectID 是此存储桶应与之关联的 Google Cloud 项目的项目 ID。它是可选的。

参数	描述
keyID	KeyID 是用于加密的 KMS 密钥 ID。它是可选的，因为默认情况下，存储桶在 Google Cloud 上是加密的。这将允许使用一个自定义加密密钥。

3.3. 为 OPENSTACK 用户置备的基础架构配置 REGISTRY

您可以配置在您自己的 Red Hat OpenStack Platform(RHOSP)基础架构中运行的集群的 registry。

3.3.1. 配置 Image Registry Operator 重定向

通过禁用重定向，您可以配置 Image Registry Operator，以控制 OpenShift Container Platform 集群构建或开发人员机器等外部系统是否直接从 Red Hat OpenStack Platform (RHOSP) Swift 存储拉取镜像。此配置是可选的，它取决于客户端是否信任存储的 SSL/TLS 证书。

注意

当客户端不信任存储证书时，可以通过镜像 registry 将 **disableRedirect** 选项设置为 **true** 代理流量。但是，镜像 registry 可能需要更多资源（特别是网络带宽）来处理增加的负载。

或者，如果客户端信任存储证书，registry 可以允许重定向。这可减少对 registry 本身的资源需求。

有些用户可能希望将其客户端配置为信任其自签名证书颁发机构 (CA)，而不是禁用重定向。如果使用自签名 CA，您必须决定信任自定义 CA 或禁用重定向。

流程

- 要确保镜像 registry 代理流量而不是依赖 Swift 存储，请运行以下命令将 **config.imageregistry** 对象中的 **spec.disableRedirect** 字段的值更改为 **true** ：

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

3.3.2. 为 Image Registry Operator 配置一个 secret

除了**configs.imageregistry.operator.openshift.io**及 ConfigMap 资源外，还可以通过**openshift-image-registry**命名空间中的独立的 secret 资源为 Operator 提供其他配置。

image-registry-private-configuration-user secret 提供了存储访问和管理所需的凭证。如果找到默认凭据，它将覆盖 Operator 使用的默认凭据。

对于 Red Hat OpenStack Platform(RHOSP)存储上的 Swift，secret 应该包含以下两个键：

- **REGISTRY_STORAGE_SWIFT_USERNAME**
- **REGISTRY_STORAGE_SWIFT_PASSWORD**

流程

- 创建一个包括了所需键的 OpenShift Container Platform secret。

```
$ oc create secret generic image-registry-private-configuration-user --from-literal=REGISTRY_STORAGE_SWIFT_USERNAME=<username> --from-literal=REGISTRY_STORAGE_SWIFT_PASSWORD=<password> -n openshift-image-registry
```

3.3.3. 使用用户置备的基础架构 RHOSP 的 registry 存储

如果 Registry Operator 无法创建 Swift 存储桶，您必须手动设置存储介质，并在 registry 自定义资源 (CR)中配置设置。

先决条件

- 在具有用户置备的 Red Hat OpenStack Platform(RHOSP)上的集群。
- 要为 RHOSP 配置 registry 存储，您需要提供 Registry Operator 云凭证。
- 对于 RHOSP 存储上的 Swift，secret 应该包含以下两个键：
  - **REGISTRY_STORAGE_SWIFT_USERNAME**
  - **REGISTRY_STORAGE_SWIFT_PASSWORD**

流程

- 在**configs.imageregistry.operator.openshift.io/cluster**中中输入存储配置：

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

配置示例

```
apiVersion: imageregistry.operator.openshift.io/v1
kind: Config
metadata:
  name: cluster
spec:
  storage:
    swift:
      container: <container_id>
```

3.3.4. RHOSP Swift 的 Image Registry Operator 配置参数

以下参数可用于配置 Red Hat OpenStack Platform (RHOSP) Swift registry 存储。

参数	描述
authURL	定义获取身份验证令牌的 URL。这个值是可选的。
authVersion	指定 RHOSP 的 Auth 版本，例如 **authVersion: "3"**。这个值是可选的。

参数	描述
container	定义用于存储 registry 数据的 Swift 容器的名称。这个值是可选的。
domain	指定 Identity v3 API 的 RHOSP 域名。这个值是可选的。
domainID	指定 Identity v3 API 的 RHOSP 域 ID。这个值是可选的。
tenant	定义 registry 使用的 RHOSP 租户名称。这个值是可选的。
tenantID	定义 registry 使用的 RHOSP 租户 ID。这个值是可选的。
regionName	定义容器所在的 RHOSP 区域。这个值是可选的。

3.4. 为 AZURE 用户置备的基础架构配置 REGISTRY

通过将内置镜像 registry 配置为使用专用 Azure 存储，将容器镜像保存到持久存储位置。此设置为您的 registry 提供持久的可扩展存储，与临时集群存储分开。

3.4.1. 为 Image Registry Operator 配置一个 secret

除了**configs.imageregistry.operator.openshift.io**及 ConfigMap 资源外，还可以通过**openshift-image-registry**命名空间中的独立的 secret 资源为 Operator 提供其他配置。

image-registry-private-configuration-user secret 提供了存储访问和管理所需的凭证。如果找到默认凭据，它将覆盖 Operator 使用的默认凭据。

对于 Azure registry 存储，secret 应该包含这个键，其值是 Azure 提供的凭据文件的内容：

- **REGISTRY_STORAGE_AZURE_ACCOUNTKEY**

流程

- 创建一个包括了所需键的 OpenShift Container Platform secret。

```
$ oc create secret generic image-registry-private-configuration-user --from-literal=REGISTRY_STORAGE_AZURE_ACCOUNTKEY=<accountkey> --namespace openshift-image-registry
```

3.4.2. 为 Azure 配置 registry 存储

在安装过程中，使用您的云凭据就可以创建 Azure Blob Storage，Registry Operator 将会自动配置存储。

先决条件

- 用户置备的 Azure 中的集群。
- 要为 Azure 配置 registry 存储，您需要提供 Registry Operator 云凭据。
- 对于Azure 存储，secret 应该包含以下键：

- **REGISTRY_STORAGE_AZURE_ACCOUNTKEY**

流程

1. 创建一个 Azure 存储容器。

2. 在**configs.imageregistry.operator.openshift.io/cluster**中输入存储配置：

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

配置示例

```
apiVersion: imageregistry.operator.openshift.io/v1
kind: Config
metadata:
  name: cluster
spec:
  storage:
    azure:
      accountName: <storage_account_name>
      container: <container_name>
```

3.4.3. 为 Azure 政府配置 registry 存储

在安装过程中，使用您的云凭据就可以创建 Azure Blob Storage， Registry Operator 将会自动配置存储。

先决条件

- 在一个政府区域中使用用户置备基础架构的 Azure 中的一个集群。
- 要为 Azure 配置 registry 存储，您需要提供 Registry Operator 云凭据。
- 对于 Azure 存储，secret 应该包含以下一个键：
  - **REGISTRY_STORAGE_AZURE_ACCOUNTKEY**

流程

1. 创建一个 Azure 存储容器。

2. 在**configs.imageregistry.operator.openshift.io/cluster**中输入存储配置：

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

配置示例

```
storage:
  azure:
    accountName: <storage-account-name>
    container: <container-name>
    cloudName: AzureUSGovernmentCloud
```

CloudName 是 Azure 云环境的名称，可用于使用适当的 Azure API 端点配置 Azure SDK。默认为 **AzurePublicCloud**。在有足够凭证时，可以将 **cloudName** 设置为 **AzureUSGovernmentCloud**、**AzureChinaCloud** 或 **AzureGermanCloud**。

3.5. 为 RHOSP 配置 REGISTRY

您可以将镜像 registry 配置为使用在 RHOSP 上运行的集群中的自定义存储。您必须将自定义存储配置为在特定可用区中使用 Cinder 卷。

3.5.1. 在 RHOSP 上运行的集群中使用自定义存储配置镜像 registry

在 Red Hat OpenStack Platform（RHOSP）上安装集群后，您可以使用位于 registry 存储的特定可用区的 Cinder 卷。

流程

1. 创建一个 YAML 文件，用于指定要使用的存储类和可用性区域。例如：

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: custom-csi-storageclass
provisioner: cinder.csi.openstack.org
volumeBindingMode: WaitForFirstConsumer
allowVolumeExpansion: true
parameters:
  availability: <availability_zone_name>
```

注意

OpenShift Container Platform 不验证您选择的可用区是否存在。应用配置前，请验证可用性区域的名称。

2. 在命令行中应用配置：

```
$ oc apply -f <storage_class_file_name>
```

输出示例

```
storageclass.storage.k8s.io/custom-csi-storageclass created
```

3. 创建一个 YAML 文件，用于指定使用存储类和 **openshift-image-registry** 命名空间的持久性卷声明（PVC）。例如：

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: csi-pvc-imageregistry
  namespace: openshift-image-registry
  annotations:
    imageregistry.openshift.io: "true"
spec:
  accessModes:
```

```
- ReadWriteOnce
volumeMode: Filesystem
resources:
  requests:
    storage: 100Gi
storageClassName: <your_custom_storage_class>
```

其中：

openshift-image-registry
: 指定此命名空间可以允许 Cluster Image Registry Operator 使用 PVC。

storage
: 此可选字段调整卷大小。

storageClassName
: 指定您创建的存储类的名称。

4. 在命令行中应用配置：

```
$ oc apply -f <pvc_file_name>
```

输出示例

```
persistentvolumeclaim/csi-pvc-imageregistry created
```

5. 将镜像 registry 配置中的原始持久性卷声明替换为新声明：

```
$ oc patch configs.imageregistry.operator.openshift.io/cluster --type 'json' -p='[{"op":
"replace", "path": "/spec/storage/pvc/claim", "value": "csi-pvc-imageregistry"}]'
```

输出示例

```
config.imageregistry.operator.openshift.io/cluster patched
```

接下来的几分钟内，配置将更新。

验证

确认 registry 正在使用您定义的资源：

1. 验证 PVC 声明值是否与您在 PVC 定义中提供的名称相同：

```
$ oc get configs.imageregistry.operator.openshift.io/cluster -o yaml
```

输出示例

```
...
status:
  ...
  managementState: Managed
  pvc:
    claim: csi-pvc-imageregistry
...
```

2. 验证 PVC 的状态是否为 **Bound**：

```
$ oc get pvc -n openshift-image-registry csi-pvc-imageregistry
```

输出示例

```
NAME                  STATUS   VOLUME                                     CAPACITY   ACCESS MODES
STORAGECLASS          AGE
csi-pvc-imageregistry  Bound    pvc-72a8f9c9-f462-11e8-b6b6-fa163e18b7b5   100Gi
RWO          custom-csi-storageclass  11m
```

3.6. 为裸机配置 REGISTRY

安装后为裸机集群配置镜像 registry 存储。因为裸机安装不会自动置备存储，所以您必须将 registry 管理状态从 **Removed** 改为 **Managed**，并配置持久性存储，或使用 Red Hat OpenShift Data Foundation，然后才能存储容器镜像。

3.6.1. 安装过程中删除的镜像 registry

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

安装后，您必须编辑 Image Registry Operator 配置，将 **managementState** 从 **Removed** 切换到 **Managed**。完成此操作后，您必须配置存储。

3.6.2. 更改镜像 registry 的管理状态

要启动镜像 registry，您必须将 Image Registry Operator 配置的 **managementState** 从 **Removed** 改为 **Managed**。

流程

- 将 **managementState** Image Registry Operator 配置从 **Removed** 改为 **Managed**。例如：

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

3.6.3. 镜像 registry 存储配置

对于不提供默认存储的平台，Image Registry Operator 最初不可用。安装后，您必须将 registry 配置为使用存储，以便 Registry Operator 可用。

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

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

3.6.3.1. 为裸机和其他手动安装配置 registry 存储

作为集群管理员，在安装后需要配置 registry 来使用存储。

先决条件

- 您可以使用具有 **cluster-admin** 角色的用户访问集群。
- 您有一个使用手动置备的 Red Hat Enterprise Linux CoreOS(RHCOS)节点（如裸机）的集群。
- 您已为集群置备持久性存储，如 Red Hat OpenShift Data Foundation。

  重要

  当您只有一个副本时，OpenShift Container Platform 支持对镜像 registry 存储的 **ReadWriteOnce** 访问。**ReadWriteOnce** 访问还要求 registry 使用 **Recreate** rollout 策略。要部署支持高可用性的镜像 registry，需要两个或多个副本，**ReadWriteMany** 访问。

- 必须具有 100Gi 容量。

流程

1. 要将 registry 配置为使用存储，修改 **configs.imageregistry/cluster** 资源中的 **spec.storage.pvc**。

   注意

   使用共享存储时，请查看您的安全设置以防止外部访问。

2. 验证您没有 registry pod:

   ```
   $ oc get pod -n openshift-image-registry -l docker-registry=default
   ```

   输出示例

   ```
   No resources found in openshift-image-registry namespace
   ```

   注意

   如果您的输出中有一个 registry pod，则不需要继续这个过程。

3. 检查 registry 配置：

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

   输出示例

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

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

4. 检查 **clusteroperator** 状态：

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

输出示例

```
NAME           VERSION              AVAILABLE   PROGRESSING   DEGRADED   SINCE
MESSAGE
image-registry   4.19               True        False         False      6h50m
```

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

   - 运行：

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

     然后，更改行

     ```
     managementState: Removed
     ```

     至

     ```
     managementState: Managed
     ```

3.6.3.2. 在非生产集群中为镜像 registry 配置存储

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

流程

- 将镜像 registry 存储设置为空目录：

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

  > **警告**
  >
  > 仅为非生产集群配置这个选项。

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

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

  等待几分钟，然后再次运行 命令。

3.6.3.3. 为裸机配置块 registry 存储

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

> **重要**
>
> 支持块存储卷或块持久性卷，但不建议在生产环境中使用镜像 registry。在块存储上配置 registry 的安装不具有高可用性，因为 registry 无法具有多个副本。
>
> 如果您选择将块存储卷与镜像 registry 搭配使用，则必须使用文件系统持久性卷声明 (PVC)。

流程

1. 输入以下命令将镜像 registry 存储设置为块存储类型，对 registry 进行补丁，使其使用 **Recreate** rollout 策略，并只使用一个副本运行：**1**

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

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

   a. 创建包含以下内容的 **pvc.yaml** 文件以定义 VMware vSphere **PersistentVolumeClaim** 对象：

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: image-registry-storage
  namespace: openshift-image-registry
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 100Gi
```

   其中：

   name
   指定代表 **PersistentVolumeClaim** 对象的唯一名称。

   namespace
   指定 **PersistentVolumeClaim** 对象的 **namespace**，即 **openshift-image-registry**。

   accessModes
   指定持久性卷声明的访问模式。使用 **ReadWriteOnce** 时，单个节点可以通过读写权限挂载该卷。

   storage
   持久性卷声明的大小。

   b. 输入以下命令从文件创建 **PersistentVolumeClaim** 对象：

```
$ oc create -f pvc.yaml -n openshift-image-registry
```

3. 输入以下命令编辑 registry 配置，使其引用正确的 PVC：

```
$ oc edit config.imageregistry.operator.openshift.io -o yaml
```

输出示例

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

通过创建自定义 PVC，您可以将 **claim** 字段留空，以便默认自动创建 **image-registry-storage** PVC。

3.6.3.4. 将 Image Registry Operator 配置为使用带有 Red Hat OpenShift Data Foundation 的 Ceph RGW 存储

Red Hat OpenShift Data Foundation 集成了多个可与 OpenShift 镜像 registry 搭配使用的存储类型：

- Ceph、共享和分布式文件系统以及内部对象存储
- NooBaa 提供多云对象网关

使用以下步骤将镜像 registry 配置为使用 Ceph RGW 存储。

先决条件

- 您可以使用具有 **cluster-admin** 角色的用户访问集群。
- 访问 OpenShift Container Platform web 控制台。
- 已安装 **oc** CLI。
- 已安装 OpenShift Data Foundation Operator，提供对象存储和 Ceph RGW 对象存储。

流程

1. 使用 **ocs-storagecluster-ceph-rgw** 存储类创建对象存储桶声明。例如：

```
cat <<EOF | oc apply -f -
apiVersion: objectbucket.io/v1alpha1
kind: ObjectBucketClaim
metadata:
  name: rgwbucket
  namespace: openshift-storage
spec:
  storageClassName: ocs-storagecluster-ceph-rgw
  generateBucketName: rgwbucket
EOF
```

或者，您可以使用 **openshift-image-registry** 作为 **namespace** 值。

2. 输入以下命令来获取存储桶名称：

```
$ bucket_name=$(oc get obc -n openshift-storage rgwbucket -o
jsonpath='{.spec.bucketName}')
```

3. 输入以下命令来获取 AWS 凭证：

```
$ AWS_ACCESS_KEY_ID=$(oc get secret -n openshift-storage rgwbucket -o
jsonpath='{.data.AWS_ACCESS_KEY_ID}' | base64 --decode)
```

```
$ AWS_SECRET_ACCESS_KEY=$(oc get secret -n openshift-storage rgwbucket -o
jsonpath='{.data.AWS_SECRET_ACCESS_KEY}' | base64 --decode)
```

4. 输入以下命令，在 **openshift-image-registry** 项目下，使用新存储桶的 AWS 凭证创建 secret **image-registry-private-configuration-user** ：

```
$ oc create secret generic image-registry-private-configuration-user --from-
literal=REGISTRY_STORAGE_S3_ACCESSKEY=${AWS_ACCESS_KEY_ID} --from-
literal=REGISTRY_STORAGE_S3_SECRETKEY=${AWS_SECRET_ACCESS_KEY} --
namespace openshift-image-registry
```

5. 输入以下命令来获取 **route** 主机：

```
$ route_host=$(oc get route ocs-storagecluster-cephobjectstore -n openshift-storage --
template='{{ .spec.host }}')
```

6. 输入以下命令来创建使用入口证书的配置映射：

```
$ oc extract secret/$(oc get ingresscontroller -n openshift-ingress-operator default -o json | jq
'.spec.defaultCertificate.name // "router-certs-default"' -r) -n openshift-ingress --confirm
```

```
$ oc create configmap image-registry-s3-bundle --from-file=ca-bundle.crt=./tls.crt  -n
openshift-config
```

7. 输入以下命令配置镜像 registry，以使用 Ceph RGW 对象存储：

```
$ oc patch config.image/cluster -p '{"spec":
{"managementState":"Managed","replicas":2,"storage":
{"managementState":"Unmanaged","s3":{"bucket":'\"${bucket_name}\"',"region":"us-east-
1","regionEndpoint":'\"https://${route_host}\"',"virtualHostedStyle":false,"encrypt":false,"trustedC
A":{"name":"image-registry-s3-bundle"}}}}}' --type=merge
```

3.6.3.5. 配置 Image Registry Operator，将 Noobaa 存储与 Red Hat OpenShift Data Foundation 一起使用

Red Hat OpenShift Data Foundation 集成了多个可与 OpenShift 镜像 registry 搭配使用的存储类型：

- Ceph、共享和分布式文件系统以及内部对象存储
- NooBaa 提供多云对象网关

使用以下步骤将镜像 registry 配置为使用 Noobaa 存储。

先决条件

- 您可以使用具有 **cluster-admin** 角色的用户访问集群。
- 访问 OpenShift Container Platform web 控制台。
- 已安装 **oc** CLI。
- 已安装 OpenShift Data Foundation Operator，以提供对象存储和 Noobaa 对象存储。

流程

1. 使用 **openshift-storage.nooba.io** 存储类创建对象存储桶声明。例如：

```
cat <<EOF | oc apply -f -
apiVersion: objectbucket.io/v1alpha1
kind: ObjectBucketClaim
metadata:
  name: noobaatest
  namespace: openshift-storage
spec:
  storageClassName: openshift-storage.noobaa.io
  generateBucketName: noobaatest
EOF
```

   或者，您可以使用 **openshift-image-registry** 作为 **namespace** 值。

2. 输入以下命令来获取存储桶名称：

```
$ bucket_name=$(oc get obc -n openshift-storage noobaatest -o
jsonpath='{.spec.bucketName}')
```

3. 输入以下命令来获取 AWS 凭证：

```
$ AWS_ACCESS_KEY_ID=$(oc get secret -n openshift-storage noobaatest -o yaml | grep -w
"AWS_ACCESS_KEY_ID:" | head -n1 | awk '{print $2}' | base64 --decode)
```

```
$ AWS_SECRET_ACCESS_KEY=$(oc get secret -n openshift-storage noobaatest -o yaml |
grep -w "AWS_SECRET_ACCESS_KEY:" | head -n1 | awk '{print $2}' | base64 --decode)
```

4. 输入以下命令，在 **openshift-image-registry** 项目下，使用新存储桶的 AWS 凭证创建 secret **image-registry-private-configuration-user** ：

```
$ oc create secret generic image-registry-private-configuration-user --from-
literal=REGISTRY_STORAGE_S3_ACCESSKEY=${AWS_ACCESS_KEY_ID} --from-
literal=REGISTRY_STORAGE_S3_SECRETKEY=${AWS_SECRET_ACCESS_KEY} --
namespace openshift-image-registry
```

5. 输入以下命令来获取路由主机：

```
$ route_host=$(oc get route s3 -n openshift-storage -o=jsonpath='{.spec.host}')
```

6. 输入以下命令来创建使用入口证书的配置映射：

```
$ oc extract secret/$(oc get ingresscontroller -n openshift-ingress-operator default -o json | jq
'.spec.defaultCertificate.name // "router-certs-default"' -r) -n openshift-ingress --confirm
```

```
$ oc create configmap image-registry-s3-bundle --from-file=ca-bundle.crt=./tls.crt  -n
openshift-config
```

7. 输入以下命令将镜像 registry 配置为使用 Nooba 对象存储：

```
$ oc patch config.image/cluster -p '{"spec":
{"managementState":"Managed","replicas":2,"storage":
{"managementState":"Unmanaged","s3":{"bucket":'\"${bucket_name}\"',"region":"us-east-
1","regionEndpoint":'\"https://${route_host}\"',"virtualHostedStyle":false,"encrypt":false,"trustedC
A":{"name":"image-registry-s3-bundle"}}}}}' --type=merge
```

3.6.4. 配置 Image Registry Operator，以使用 Red Hat OpenShift Data Foundation 的 CephFS 存储

Red Hat OpenShift Data Foundation 集成了多个可与 OpenShift 镜像 registry 搭配使用的存储类型：

- Ceph、共享和分布式文件系统以及内部对象存储
- NooBaa 提供多云对象网关

使用以下步骤将镜像 registry 配置为使用 CephFS 存储。

注意

CephFS 使用持久性卷声明 (PVC) 存储。如果存在其他选项，如 Ceph RGW 或 Noobaa，则不建议将 PVC 用于镜像 registry 存储。

先决条件

- 您可以使用具有 **cluster-admin** 角色的用户访问集群。
- 访问 OpenShift Container Platform web 控制台。
- 已安装 **oc** CLI。
- 已安装 OpenShift Data Foundation Operator，以提供对象存储和 CephFS 文件存储。

流程

1. 创建一个 PVC 来使用 **cephfs** 存储类。例如：

```
cat <<EOF | oc apply -f -
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
 name: registry-storage-pvc
 namespace: openshift-image-registry
spec:
 accessModes:
 - ReadWriteMany
```

```
resources:
  requests:
    storage: 100Gi
storageClassName: ocs-storagecluster-cephfs
EOF
```

2. 输入以下命令配置镜像 registry，以使用 CephFS 文件系统存储：

```
$ oc patch config.image/cluster -p '{"spec":
{"managementState":"Managed","replicas":2,"storage":
{"managementState":"Unmanaged","pvc":{"claim":"registry-storage-pvc"}}}}' --type=merge
```

3.6.5. 其他资源

- 推荐的可配置存储技术
- 将镜像 registry 配置为使用 OpenShift Data Foundation

3.7. 为 VSPHERE 配置 REGISTRY

安装后为 vSphere 集群配置镜像 registry 存储。因为 vSphere 安装不会自动置备存储，所以您必须将 registry 管理状态从 **Removed** 改为 **Managed**，并配置持久性存储，或使用 Red Hat OpenShift Data Foundation，然后才能存储容器镜像。

3.7.1. 安装过程中删除的镜像 registry

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

安装后，您必须编辑 Image Registry Operator 配置，将 **managementState** 从 **Removed** 切换到 **Managed**。完成此操作后，您必须配置存储。

3.7.2. 更改镜像 registry 的管理状态

要启动镜像 registry，您必须将 Image Registry Operator 配置的 **managementState** 从 **Removed** 改为 **Managed**。

流程

- 将 **managementState** Image Registry Operator 配置从 **Removed** 改为 **Managed**。例如：

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

3.7.3. 镜像 registry 存储配置

对于不提供默认存储的平台，Image Registry Operator 最初不可用。安装后，您必须将 registry 配置为使用存储，以便 Registry Operator 可用。

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

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

3.7.3.1. 为 VMware vSphere 配置 registry 存储

作为集群管理员，在安装后需要配置 registry 来使用存储。

先决条件

- 集群管理员权限。
- VMware vSphere 上有一个集群。
- 为集群置备的持久性存储，如 Red Hat OpenShift Data Foundation。

  重要

  当您只有一个副本时，OpenShift Container Platform 支持对镜像 registry 存储的 **ReadWriteOnce** 访问。**ReadWriteOnce** 访问还要求 registry 使用 **Recreate** rollout 策略。要部署支持高可用性的镜像 registry，需要两个或多个副本，**ReadWriteMany** 访问。

- 必须具有"100Gi"容量.

重要

测试显示在 RHEL 中使用 NFS 服务器作为核心服务的存储后端的问题。这包括 OpenShift Container Registry 和 Quay，Prometheus 用于监控存储，以及 Elasticsearch 用于日志存储。因此，不建议使用 RHEL NFS 作为 PV 后端用于核心服务。

市场上的其他 NFS 实现可能没有这些问题。如需了解更多与此问题相关的信息，请联络相关的 NFS 厂商。

流程

1. 更改 **configs.imageregistry/cluster** 资源中的 **spec.storage.pvc** 字段。

   注意

   使用共享存储时，请查看您的安全设置以防止外部访问。

2. 运行以下命令验证您没有 registry pod：

   ```
   $ oc get pod -n openshift-image-registry -l docker-registry=default
   ```

   输出示例

   ```
   No resourses found in openshift-image-registry namespace
   ```

   注意

   如果您的输出中有一个 registry pod，则不需要继续这个过程。

3. 运行以下命令检查 registry 配置：

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

输出示例

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

将 **claim** 字段留空以允许自动创建 **image-registry-storage** 持久性卷声明(PVC)。PVC 基于默认存储类生成。但请注意，默认存储类可能会提供 ReadWriteOnce (RWO)卷，如 RADOS 块设备 (RBD)，这可能会在复制到多个副本时导致问题。

4. 运行以下命令检查 **clusteroperator** 状态：

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

输出示例

```
NAME             VERSION   AVAILABLE   PROGRESSING   DEGRADED   SINCE
MESSAGE
image-registry   4.7       True        False         False      6h50m
```

3.7.3.2. 在非生产集群中为镜像 registry 配置存储

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

流程

- 将镜像 registry 存储设置为空目录：

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

> **警告**
>
> 仅为非生产集群配置这个选项。

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

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

等待几分钟，然后再次运行 命令。

3.7.3.3. 为 VMware vSphere 配置块 registry 存储

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

重要

支持块存储卷，但不建议在生产环境中用于镜像 registry。在块存储上配置 registry 的安装不具有高可用性，因为 registry 无法具有多个副本。

流程

1. 输入以下命令将镜像 registry 存储设置为块存储类型，对 registry 进行补丁，使其使用 **Recreate** rollout 策略，并只使用**一个**副本运行：

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

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

   a. 创建包含以下内容的 **pvc.yaml** 文件以定义 VMware vSphere **PersistentVolumeClaim** 对象：

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: image-registry-storage
  namespace: openshift-image-registry
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 100Gi
```

   其中：

   name

   指定代表 **PersistentVolumeClaim** 对象的唯一名称。

   namespace

   指定 **PersistentVolumeClaim** 对象的 **namespace**，即 **openshift-image-registry**。

   accessModes

   指定持久性卷声明的访问模式。使用 **ReadWriteOnce** 时，单个节点可以通过读写权限挂载该卷。

   storage

   持久性卷声明的大小。

   b. 输入以下命令从文件创建 **PersistentVolumeClaim** 对象：

```
$ oc create -f pvc.yaml -n openshift-image-registry
```

3. 输入以下命令编辑 registry 配置，使其引用正确的 PVC：

```
$ oc edit config.imageregistry.operator.openshift.io -o yaml
```

输出示例

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

通过创建自定义 PVC，您可以将 **claim** 字段留空，以便默认自动创建 **image-registry-storage** PVC。

3.7.3.4. 将 Image Registry Operator 配置为使用带有 Red Hat OpenShift Data Foundation 的 Ceph RGW 存储

Red Hat OpenShift Data Foundation 集成了多个可与 OpenShift 镜像 registry 搭配使用的存储类型：

- Ceph、共享和分布式文件系统以及内部对象存储
- NooBaa 提供多云对象网关

使用以下步骤将镜像 registry 配置为使用 Ceph RGW 存储。

先决条件

- 您可以使用具有 **cluster-admin** 角色的用户访问集群。
- 访问 OpenShift Container Platform web 控制台。
- 已安装 **oc** CLI。
- 已安装 OpenShift Data Foundation Operator，提供对象存储和 Ceph RGW 对象存储。

流程

1. 使用 **ocs-storagecluster-ceph-rgw** 存储类创建对象存储桶声明。例如：

```
cat <<EOF | oc apply -f -
apiVersion: objectbucket.io/v1alpha1
kind: ObjectBucketClaim
metadata:
  name: rgwbucket
  namespace: openshift-storage
spec:
  storageClassName: ocs-storagecluster-ceph-rgw
  generateBucketName: rgwbucket
EOF
```

或者，您可以使用 **openshift-image-registry** 作为 **namespace** 值。

2. 输入以下命令来获取存储桶名称：

```
$ bucket_name=$(oc get obc -n openshift-storage rgwbucket -o
jsonpath='{.spec.bucketName}')
```

3. 输入以下命令来获取 AWS 凭证：

```
$ AWS_ACCESS_KEY_ID=$(oc get secret -n openshift-storage rgwbucket -o
jsonpath='{.data.AWS_ACCESS_KEY_ID}' | base64 --decode)
```

```
$ AWS_SECRET_ACCESS_KEY=$(oc get secret -n openshift-storage rgwbucket -o
jsonpath='{.data.AWS_SECRET_ACCESS_KEY}' | base64 --decode)
```

4. 输入以下命令，在 **openshift-image-registry** 项目下，使用新存储桶的 AWS 凭证创建 secret **image-registry-private-configuration-user** ：

```
$ oc create secret generic image-registry-private-configuration-user --from-
literal=REGISTRY_STORAGE_S3_ACCESSKEY=${AWS_ACCESS_KEY_ID} --from-
literal=REGISTRY_STORAGE_S3_SECRETKEY=${AWS_SECRET_ACCESS_KEY} --
namespace openshift-image-registry
```

5. 输入以下命令来获取 **route** 主机：

```
$ route_host=$(oc get route ocs-storagecluster-cephobjectstore -n openshift-storage --
template='{{ .spec.host }}')
```

6. 输入以下命令来创建使用入口证书的配置映射：

```
$ oc extract secret/$(oc get ingresscontroller -n openshift-ingress-operator default -o json | jq
'.spec.defaultCertificate.name // "router-certs-default"' -r) -n openshift-ingress --confirm
```

```
$ oc create configmap image-registry-s3-bundle --from-file=ca-bundle.crt=./tls.crt  -n
openshift-config
```

7. 输入以下命令配置镜像 registry，以使用 Ceph RGW 对象存储：

```
$ oc patch config.image/cluster -p '{"spec":
{"managementState":"Managed","replicas":2,"storage":
{"managementState":"Unmanaged","s3":{"bucket":'\"${bucket_name}\"',"region":"us-east-
1","regionEndpoint":'\"https://${route_host}\"',"virtualHostedStyle":false,"encrypt":false,"trustedC
A":{"name":"image-registry-s3-bundle"}}}}}' --type=merge
```

3.7.3.5. 配置 Image Registry Operator，将 Noobaa 存储与 Red Hat OpenShift Data Foundation 一起使用

Red Hat OpenShift Data Foundation 集成了多个可与 OpenShift 镜像 registry 搭配使用的存储类型：

- Ceph、共享和分布式文件系统以及内部对象存储
- NooBaa 提供多云对象网关

使用以下步骤将镜像 registry 配置为使用 Noobaa 存储。

先决条件

- 您可以使用具有 **cluster-admin** 角色的用户访问集群。
- 访问 OpenShift Container Platform web 控制台。
- 已安装 **oc** CLI。
- 已安装 OpenShift Data Foundation Operator，以提供对象存储和 Noobaa 对象存储。

流程

1. 使用 **openshift-storage.nooba.io** 存储类创建对象存储桶声明。例如：

```
cat <<EOF | oc apply -f -
apiVersion: objectbucket.io/v1alpha1
kind: ObjectBucketClaim
metadata:
  name: noobaatest
  namespace: openshift-storage
spec:
  storageClassName: openshift-storage.noobaa.io
  generateBucketName: noobaatest
EOF
```

   或者，您可以使用 **openshift-image-registry** 作为 **namespace** 值。

2. 输入以下命令来获取存储桶名称：

```
$ bucket_name=$(oc get obc -n openshift-storage noobaatest -o
jsonpath='{.spec.bucketName}')
```

3. 输入以下命令来获取 AWS 凭证：

```
$ AWS_ACCESS_KEY_ID=$(oc get secret -n openshift-storage noobaatest -o yaml | grep -w
"AWS_ACCESS_KEY_ID:" | head -n1 | awk '{print $2}' | base64 --decode)
```

```
$ AWS_SECRET_ACCESS_KEY=$(oc get secret -n openshift-storage noobaatest -o yaml |
grep -w "AWS_SECRET_ACCESS_KEY:" | head -n1 | awk '{print $2}' | base64 --decode)
```

4. 输入以下命令，在 **openshift-image-registry** 项目下，使用新存储桶的 AWS 凭证创建 secret **image-registry-private-configuration-user** ：

```
$ oc create secret generic image-registry-private-configuration-user --from-
literal=REGISTRY_STORAGE_S3_ACCESSKEY=${AWS_ACCESS_KEY_ID} --from-
literal=REGISTRY_STORAGE_S3_SECRETKEY=${AWS_SECRET_ACCESS_KEY} --
namespace openshift-image-registry
```

5. 输入以下命令来获取路由主机：

```
$ route_host=$(oc get route s3 -n openshift-storage -o=jsonpath='{.spec.host}')
```

6. 输入以下命令来创建使用入口证书的配置映射：

```
$ oc extract secret/$(oc get ingresscontroller -n openshift-ingress-operator default -o json | jq '.spec.defaultCertificate.name // "router-certs-default"' -r) -n openshift-ingress --confirm
```

```
$ oc create configmap image-registry-s3-bundle --from-file=ca-bundle.crt=./tls.crt  -n openshift-config
```

7. 输入以下命令将镜像 registry 配置为使用 Nooba 对象存储：

```
$ oc patch config.image/cluster -p '{"spec":{"managementState":"Managed","replicas":2,"storage":{"managementState":"Unmanaged","s3":{"bucket":'\"${bucket_name}\"',"region":"us-east-1","regionEndpoint":'\"https://${route_host}\"',"virtualHostedStyle":false,"encrypt":false,"trustedCA":{"name":"image-registry-s3-bundle"}}}}}' --type=merge
```

3.7.4. 配置 Image Registry Operator，以使用 Red Hat OpenShift Data Foundation 的 CephFS 存储

Red Hat OpenShift Data Foundation 集成了多个可与 OpenShift 镜像 registry 搭配使用的存储类型：

- Ceph、共享和分布式文件系统以及内部对象存储
- NooBaa 提供多云对象网关

使用以下步骤将镜像 registry 配置为使用 CephFS 存储。

注意

CephFS 使用持久性卷声明 (PVC) 存储。如果存在其他选项，如 Ceph RGW 或 Noobaa，则不建议将 PVC 用于镜像 registry 存储。

先决条件

- 您可以使用具有 **cluster-admin** 角色的用户访问集群。
- 访问 OpenShift Container Platform web 控制台。
- 已安装 **oc** CLI。
- 已安装 OpenShift Data Foundation Operator，以提供对象存储和 CephFS 文件存储。

流程

1. 创建一个 PVC 来使用 **cephfs** 存储类。例如：

```
cat <<EOF | oc apply -f -
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
 name: registry-storage-pvc
 namespace: openshift-image-registry
spec:
 accessModes:
 - ReadWriteMany
```

```
 resources:
   requests:
     storage: 100Gi
 storageClassName: ocs-storagecluster-cephfs
EOF
```

2. 输入以下命令配置镜像 registry，以使用 CephFS 文件系统存储：

```
$ oc patch config.image/cluster -p '{"spec":
{"managementState":"Managed","replicas":2,"storage":
{"managementState":"Unmanaged","pvc":{"claim":"registry-storage-pvc"}}}}' --type=merge
```

3.7.5. 其他资源

- 为 vSphere 配置 registry
- 推荐的可配置存储技术
- 将镜像 registry 配置为使用 OpenShift Data Foundation

3.8. 为 RED HAT OPENSHIFT DATA FOUNDATION 配置 REGISTRY

要在裸机和 vSphere 上配置 OpenShift 镜像 registry 以使用 Red Hat OpenShift Data Foundation 存储，您必须安装 OpenShift Data Foundation，然后使用 Ceph 或 Noobaa 配置镜像 registry。

3.8.1. 将 Image Registry Operator 配置为使用带有 Red Hat OpenShift Data Foundation 的 Ceph RGW 存储

Red Hat OpenShift Data Foundation 集成了多个可与 OpenShift 镜像 registry 搭配使用的存储类型：

- Ceph、共享和分布式文件系统以及内部对象存储
- NooBaa 提供多云对象网关

使用以下步骤将镜像 registry 配置为使用 Ceph RGW 存储。

先决条件

- 您可以使用具有 **cluster-admin** 角色的用户访问集群。
- 访问 OpenShift Container Platform web 控制台。
- 已安装 **oc** CLI。
- 已安装 OpenShift Data Foundation Operator，提供对象存储和 Ceph RGW 对象存储。

流程

1. 使用 **ocs-storagecluster-ceph-rgw** 存储类创建对象存储桶声明。例如：

```
cat <<EOF | oc apply -f -
apiVersion: objectbucket.io/v1alpha1
kind: ObjectBucketClaim
metadata:
```

```
  name: rgwbucket
  namespace: openshift-storage
spec:
  storageClassName: ocs-storagecluster-ceph-rgw
  generateBucketName: rgwbucket
EOF
```

或者，您可以使用 **openshift-image-registry** 作为 **namespace** 值。

2. 输入以下命令来获取存储桶名称：

```
$ bucket_name=$(oc get obc -n openshift-storage rgwbucket -o
jsonpath='{.spec.bucketName}')
```

3. 输入以下命令来获取 AWS 凭证：

```
$ AWS_ACCESS_KEY_ID=$(oc get secret -n openshift-storage rgwbucket -o
jsonpath='{.data.AWS_ACCESS_KEY_ID}' | base64 --decode)
```

```
$ AWS_SECRET_ACCESS_KEY=$(oc get secret -n openshift-storage rgwbucket -o
jsonpath='{.data.AWS_SECRET_ACCESS_KEY}' | base64 --decode)
```

4. 输入以下命令，在 **openshift-image-registry** 项目下，使用新存储桶的 AWS 凭证创建 secret **image-registry-private-configuration-user** ：

```
$ oc create secret generic image-registry-private-configuration-user --from-
literal=REGISTRY_STORAGE_S3_ACCESSKEY=${AWS_ACCESS_KEY_ID} --from-
literal=REGISTRY_STORAGE_S3_SECRETKEY=${AWS_SECRET_ACCESS_KEY} --
namespace openshift-image-registry
```

5. 输入以下命令来获取 **route** 主机：

```
$ route_host=$(oc get route ocs-storagecluster-cephobjectstore -n openshift-storage --
template='{{ .spec.host }}')
```

6. 输入以下命令来创建使用入口证书的配置映射：

```
$ oc extract secret/$(oc get ingresscontroller -n openshift-ingress-operator default -o json | jq
'.spec.defaultCertificate.name // "router-certs-default"' -r) -n openshift-ingress --confirm
```

```
$ oc create configmap image-registry-s3-bundle --from-file=ca-bundle.crt=./tls.crt  -n
openshift-config
```

7. 输入以下命令配置镜像 registry，以使用 Ceph RGW 对象存储：

```
$ oc patch config.image/cluster -p '{"spec":
{"managementState":"Managed","replicas":2,"storage":
{"managementState":"Unmanaged","s3":{"bucket":'\"${bucket_name}\"',"region":"us-east-
1","regionEndpoint":'\"https://${route_host}\"',"virtualHostedStyle":false,"encrypt":false,"trustedC
A":{"name":"image-registry-s3-bundle"}}}}}' --type=merge
```

3.8.2. 配置 Image Registry Operator，将 Noobaa 存储与 Red Hat OpenShift Data Foundation 一起使用

Red Hat OpenShift Data Foundation 集成了多个可与 OpenShift 镜像 registry 搭配使用的存储类型：

- Ceph、共享和分布式文件系统以及内部对象存储
- NooBaa 提供多云对象网关

使用以下步骤将镜像 registry 配置为使用 Noobaa 存储。

先决条件

- 您可以使用具有 **cluster-admin** 角色的用户访问集群。
- 访问 OpenShift Container Platform web 控制台。
- 已安装 **oc** CLI。
- 已安装 OpenShift Data Foundation Operator，以提供对象存储和 Noobaa 对象存储。

流程

1. 使用 **openshift-storage.nooba.io** 存储类创建对象存储桶声明。例如：

```
cat <<EOF | oc apply -f -
apiVersion: objectbucket.io/v1alpha1
kind: ObjectBucketClaim
metadata:
  name: noobaatest
  namespace: openshift-storage
spec:
  storageClassName: openshift-storage.noobaa.io
  generateBucketName: noobaatest
EOF
```

   或者，您可以使用 **openshift-image-registry** 作为 **namespace** 值。

2. 输入以下命令来获取存储桶名称：

```
$ bucket_name=$(oc get obc -n openshift-storage noobaatest -o
jsonpath='{.spec.bucketName}')
```

3. 输入以下命令来获取 AWS 凭证：

```
$ AWS_ACCESS_KEY_ID=$(oc get secret -n openshift-storage noobaatest -o yaml | grep -w
"AWS_ACCESS_KEY_ID:" | head -n1 | awk '{print $2}' | base64 --decode)
```

```
$ AWS_SECRET_ACCESS_KEY=$(oc get secret -n openshift-storage noobaatest -o yaml |
grep -w "AWS_SECRET_ACCESS_KEY:" | head -n1 | awk '{print $2}' | base64 --decode)
```

4. 输入以下命令，在 **openshift-image-registry** 项目下，使用新存储桶的 AWS 凭证创建 secret **image-registry-private-configuration-user** ：

```
$ oc create secret generic image-registry-private-configuration-user --from-literal=REGISTRY_STORAGE_S3_ACCESSKEY=${AWS_ACCESS_KEY_ID} --from-literal=REGISTRY_STORAGE_S3_SECRETKEY=${AWS_SECRET_ACCESS_KEY} --namespace openshift-image-registry
```

5. 输入以下命令来获取路由主机：

```
$ route_host=$(oc get route s3 -n openshift-storage -o=jsonpath='{.spec.host}')
```

6. 输入以下命令来创建使用入口证书的配置映射：

```
$ oc extract secret/$(oc get ingresscontroller -n openshift-ingress-operator default -o json | jq '.spec.defaultCertificate.name // "router-certs-default"' -r) -n openshift-ingress --confirm
```

```
$ oc create configmap image-registry-s3-bundle --from-file=ca-bundle.crt=./tls.crt  -n openshift-config
```

7. 输入以下命令将镜像 registry 配置为使用 Nooba 对象存储：

```
$ oc patch config.image/cluster -p '{"spec":{"managementState":"Managed","replicas":2,"storage":{"managementState":"Unmanaged","s3":{"bucket":'\"${bucket_name}\"',"region":"us-east-1","regionEndpoint":'\"https://${route_host}\"',"virtualHostedStyle":false,"encrypt":false,"trustedCA":{"name":"image-registry-s3-bundle"}}}}}' --type=merge
```

3.8.3. 配置 Image Registry Operator，以使用 Red Hat OpenShift Data Foundation 的 CephFS 存储

Red Hat OpenShift Data Foundation 集成了多个可与 OpenShift 镜像 registry 搭配使用的存储类型：

- Ceph、共享和分布式文件系统以及内部对象存储
- NooBaa 提供多云对象网关

使用以下步骤将镜像 registry 配置为使用 CephFS 存储。

注意

CephFS 使用持久性卷声明 (PVC) 存储。如果存在其他选项，如 Ceph RGW 或 Noobaa，则不建议将 PVC 用于镜像 registry 存储。

先决条件

- 您可以使用具有 **cluster-admin** 角色的用户访问集群。
- 访问 OpenShift Container Platform web 控制台。
- 已安装 **oc** CLI。
- 已安装 OpenShift Data Foundation Operator，以提供对象存储和 CephFS 文件存储。

流程

1. 创建一个 PVC 来使用 **cephfs** 存储类。例如：

```
cat <<EOF | oc apply -f -
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
 name: registry-storage-pvc
 namespace: openshift-image-registry
spec:
 accessModes:
 - ReadWriteMany
 resources:
   requests:
     storage: 100Gi
 storageClassName: ocs-storagecluster-cephfs
EOF
```

2. 输入以下命令配置镜像 registry，以使用 CephFS 文件系统存储：

```
$ oc patch config.image/cluster -p '{"spec":
{"managementState":"Managed","replicas":2,"storage":
{"managementState":"Unmanaged","pvc":{"claim":"registry-storage-pvc"}}}}' --type=merge
```

3.8.4. 其他资源

- 将镜像 registry 配置为使用 OpenShift Data Foundation
- 多云对象网关的性能调节指南(NooBaa)

3.9. 为 NUTANIX 配置 REGISTRY

用户可以优化容器镜像分发、安全性和访问控制，为 OpenShift Container Platform 上的 Nutanix 应用程序启用强大的基础

3.9.1. 安装过程中删除的镜像 registry

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

安装后，您必须编辑 Image Registry Operator 配置，将 **managementState** 从 **Removed** 切换到 **Managed**。完成此操作后，您必须配置存储。

3.9.2. 更改镜像 registry 的管理状态

要启动镜像 registry，您必须将 Image Registry Operator 配置的 **managementState** 从 **Removed** 改为 **Managed**。

流程

- 将 **managementState** Image Registry Operator 配置从 **Removed** 改为 **Managed**。例如：

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

3.9.3. 镜像 registry 存储配置

对于不提供默认存储的平台，Image Registry Operator 最初不可用。安装后，您必须将 registry 配置为使用存储，以便 Registry Operator 可用。

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

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

3.9.3.1. 为 Nutanix 配置 registry 存储

作为集群管理员，在安装后需要配置 registry 来使用存储。

先决条件

- 您可以使用具有 **cluster-admin** 角色的用户访问集群。
- 在 Nutanix 上有一个集群。
- 您已为集群置备持久性存储，如 Red Hat OpenShift Data Foundation。

  重要

  当您只有一个副本时，OpenShift Container Platform 支持对镜像 registry 存储的 **ReadWriteOnce** 访问。**ReadWriteOnce** 访问还要求 registry 使用 **Recreate** rollout 策略。要部署支持高可用性的镜像 registry，需要两个或多个副本，**ReadWriteMany** 访问。

- 您必须有 100 Gi 容量。

流程

1. 要将 registry 配置为使用存储，修改 **configs.imageregistry/cluster** 资源中的 **spec.storage.pvc**。

   注意

   使用共享存储时，请查看您的安全设置以防止外部访问。

2. 验证您没有 registry pod:

   ```
   $ oc get pod -n openshift-image-registry -l docker-registry=default
   ```

   输出示例

   ```
   No resourses found in openshift-image-registry namespace
   ```

   注意

   如果您的输出中有一个 registry pod，则不需要继续这个过程。

3. 检查 registry 配置：

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

输出示例

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

将 **claim** 字段留空以允许自动创建 **image-registry-storage** 持久性卷声明(PVC)。PVC 基于默认存储类生成。但请注意，默认存储类可能会提供 ReadWriteOnce (RWO)卷，如 RADOS 块设备(RBD)，这可能会在复制到多个副本时导致问题。

4. 检查 **clusteroperator** 状态：

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

输出示例

```
NAME           VERSION                              AVAILABLE   PROGRESSING   DEGRADED
SINCE   MESSAGE
image-registry   4.13                             True        False         False      6h50m
```

3.9.3.2. 在非生产集群中为镜像 registry 配置存储

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

流程

- 将镜像 registry 存储设置为空目录：

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

> **警告**
>
> 仅为非生产集群配置这个选项。

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

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

等待几分钟，然后再次运行 命令。

3.9.3.3. 为 Nutanix 卷配置块 registry 存储

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

重要

支持块存储卷或块持久性卷，但不建议在生产环境中使用镜像 registry。在块存储上配置 registry 的安装不具有高可用性，因为 registry 无法具有多个副本。

如果您选择将块存储卷与镜像 registry 搭配使用，则必须使用文件系统持久性卷声明 (PVC)。

流程

1. 输入以下命令将镜像 registry 存储设置为块存储类型，对 registry 进行补丁，使其使用 **Recreate** rollout 策略，并只使用一个副本运行：**1**

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

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

   a. 创建包含以下内容的 **pvc.yaml** 文件以定义 Nutanix **PersistentVolumeClaim** 对象：

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: image-registry-storage
  namespace: openshift-image-registry
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 100Gi
```

其中：

name

指定代表 **PersistentVolumeClaim** 对象的唯一名称。

namespace

指定 **PersistentVolumeClaim** 对象的命名空间，即 **openshift-image-registry**。

ReadWriteOnce

指定持久性卷声明的访问模式。使用 **ReadWriteOnce** 时，单个节点可以通过读写权限挂载该卷。

storage

指定持久性卷声明的大小。

   b. 输入以下命令从文件创建 **PersistentVolumeClaim** 对象：

```
$ oc create -f pvc.yaml -n openshift-image-registry
```

3. 输入以下命令编辑 registry 配置，使其引用正确的 PVC：

```
$ oc edit config.imageregistry.operator.openshift.io -o yaml
```

输出示例

```
storage:
  pvc:
    claim: (1)
```

通过创建自定义 PVC，您可以将 **claim** 字段留空，以便默认自动创建 **image-registry-storage** PVC。

3.9.3.4. 将 Image Registry Operator 配置为使用带有 Red Hat OpenShift Data Foundation 的 Ceph RGW 存储

Red Hat OpenShift Data Foundation 集成了多个可与 OpenShift 镜像 registry 搭配使用的存储类型：

- Ceph、共享和分布式文件系统以及内部对象存储
- NooBaa 提供多云对象网关

使用以下步骤将镜像 registry 配置为使用 Ceph RGW 存储。

先决条件

- 您可以使用具有 **cluster-admin** 角色的用户访问集群。
- 访问 OpenShift Container Platform web 控制台。
- 已安装 **oc** CLI。
- 已安装 OpenShift Data Foundation Operator，提供对象存储和 Ceph RGW 对象存储。

流程

1. 使用 **ocs-storagecluster-ceph-rgw** 存储类创建对象存储桶声明。例如：

```
cat <<EOF | oc apply -f -
apiVersion: objectbucket.io/v1alpha1
kind: ObjectBucketClaim
metadata:
  name: rgwbucket
  namespace: openshift-storage
spec:
  storageClassName: ocs-storagecluster-ceph-rgw
  generateBucketName: rgwbucket
EOF
```

或者，您可以使用 **openshift-image-registry** 作为 **namespace** 值。

2. 输入以下命令来获取存储桶名称：

```
$ bucket_name=$(oc get obc -n openshift-storage rgwbucket -o
jsonpath='{.spec.bucketName}')
```

3. 输入以下命令来获取 AWS 凭证：

```
$ AWS_ACCESS_KEY_ID=$(oc get secret -n openshift-storage rgwbucket -o
jsonpath='{.data.AWS_ACCESS_KEY_ID}' | base64 --decode)
```

```
$ AWS_SECRET_ACCESS_KEY=$(oc get secret -n openshift-storage rgwbucket -o
jsonpath='{.data.AWS_SECRET_ACCESS_KEY}' | base64 --decode)
```

4. 输入以下命令，在 **openshift-image-registry** 项目下，使用新存储桶的 AWS 凭证创建 secret **image-registry-private-configuration-user** ：

```
$ oc create secret generic image-registry-private-configuration-user --from-
literal=REGISTRY_STORAGE_S3_ACCESSKEY=${AWS_ACCESS_KEY_ID} --from-
literal=REGISTRY_STORAGE_S3_SECRETKEY=${AWS_SECRET_ACCESS_KEY} --
namespace openshift-image-registry
```

5. 输入以下命令来获取 **route** 主机：

```
$ route_host=$(oc get route ocs-storagecluster-cephobjectstore -n openshift-storage --
template='{{ .spec.host }}')
```

6. 输入以下命令来创建使用入口证书的配置映射：

```
$ oc extract secret/$(oc get ingresscontroller -n openshift-ingress-operator default -o json | jq
'.spec.defaultCertificate.name // "router-certs-default"' -r) -n openshift-ingress --confirm
```

```
$ oc create configmap image-registry-s3-bundle --from-file=ca-bundle.crt=./tls.crt  -n
openshift-config
```

7. 输入以下命令配置镜像 registry，以使用 Ceph RGW 对象存储：

```
$ oc patch config.image/cluster -p '{"spec":
{"managementState":"Managed","replicas":2,"storage":
{"managementState":"Unmanaged","s3":{"bucket":'\"${bucket_name}\"',"region":"us-east-
1","regionEndpoint":'\"https://${route_host}\"',"virtualHostedStyle":false,"encrypt":false,"trustedC
A":{"name":"image-registry-s3-bundle"}}}}}' --type=merge
```

3.9.3.5. 配置 Image Registry Operator，将 Noobaa 存储与 Red Hat OpenShift Data Foundation 一起使用

Red Hat OpenShift Data Foundation 集成了多个可与 OpenShift 镜像 registry 搭配使用的存储类型：

- Ceph、共享和分布式文件系统以及内部对象存储
- NooBaa 提供多云对象网关

使用以下步骤将镜像 registry 配置为使用 Noobaa 存储。

先决条件

- 您可以使用具有 **cluster-admin** 角色的用户访问集群。
- 访问 OpenShift Container Platform web 控制台。
- 已安装 **oc** CLI。
- 已安装 OpenShift Data Foundation Operator，以提供对象存储和 Noobaa 对象存储。

流程

1. 使用 **openshift-storage.nooba.io** 存储类创建对象存储桶声明。例如：

```
cat <<EOF | oc apply -f -
apiVersion: objectbucket.io/v1alpha1
kind: ObjectBucketClaim
metadata:
  name: noobaatest
  namespace: openshift-storage
spec:
  storageClassName: openshift-storage.noobaa.io
  generateBucketName: noobaatest
EOF
```

   或者，您可以使用 **openshift-image-registry** 作为 **namespace** 值。

2. 输入以下命令来获取存储桶名称：

```
$ bucket_name=$(oc get obc -n openshift-storage noobaatest -o
jsonpath='{.spec.bucketName}')
```

3. 输入以下命令来获取 AWS 凭证：

```
$ AWS_ACCESS_KEY_ID=$(oc get secret -n openshift-storage noobaatest -o yaml | grep -w
"AWS_ACCESS_KEY_ID:" | head -n1 | awk '{print $2}' | base64 --decode)
```

```
$ AWS_SECRET_ACCESS_KEY=$(oc get secret -n openshift-storage noobaatest -o yaml |
grep -w "AWS_SECRET_ACCESS_KEY:" | head -n1 | awk '{print $2}' | base64 --decode)
```

4. 输入以下命令，在 **openshift-image-registry** 项目下，使用新存储桶的 AWS 凭证创建 secret **image-registry-private-configuration-user** ：

```
$ oc create secret generic image-registry-private-configuration-user --from-
literal=REGISTRY_STORAGE_S3_ACCESSKEY=${AWS_ACCESS_KEY_ID} --from-
literal=REGISTRY_STORAGE_S3_SECRETKEY=${AWS_SECRET_ACCESS_KEY} --
namespace openshift-image-registry
```

5. 输入以下命令来获取路由主机：

```
$ route_host=$(oc get route s3 -n openshift-storage -o=jsonpath='{.spec.host}')
```

6. 输入以下命令来创建使用入口证书的配置映射：

```
$ oc extract secret/$(oc get ingresscontroller -n openshift-ingress-operator default -o json | jq
'.spec.defaultCertificate.name // "router-certs-default"' -r) -n openshift-ingress --confirm
```

```
$ oc create configmap image-registry-s3-bundle --from-file=ca-bundle.crt=./tls.crt  -n
openshift-config
```

7. 输入以下命令将镜像 registry 配置为使用 Nooba 对象存储：

```
$ oc patch config.image/cluster -p '{"spec":
{"managementState":"Managed","replicas":2,"storage":
{"managementState":"Unmanaged","s3":{"bucket":'\"${bucket_name}\"',"region":"us-east-
1","regionEndpoint":'\"https://${route_host}\"',"virtualHostedStyle":false,"encrypt":false,"trustedC
A":{"name":"image-registry-s3-bundle"}}}}}' --type=merge
```

3.9.4. 配置 Image Registry Operator，以使用 Red Hat OpenShift Data Foundation 的 CephFS 存储

Red Hat OpenShift Data Foundation 集成了多个可与 OpenShift 镜像 registry 搭配使用的存储类型：

- Ceph、共享和分布式文件系统以及内部对象存储
- NooBaa 提供多云对象网关

使用以下步骤将镜像 registry 配置为使用 CephFS 存储。

注意

CephFS 使用持久性卷声明 (PVC) 存储。如果存在其他选项，如 Ceph RGW 或 Noobaa，则不建议将 PVC 用于镜像 registry 存储。

先决条件

- 您可以使用具有 **cluster-admin** 角色的用户访问集群。
- 访问 OpenShift Container Platform web 控制台。
- 已安装 **oc** CLI。
- 已安装 OpenShift Data Foundation Operator，以提供对象存储和 CephFS 文件存储。

流程

1. 创建一个 PVC 来使用 **cephfs** 存储类。例如：

```
cat <<EOF | oc apply -f -
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
 name: registry-storage-pvc
 namespace: openshift-image-registry
spec:
 accessModes:
 - ReadWriteMany
```

```
  resources:
    requests:
      storage: 100Gi
  storageClassName: ocs-storagecluster-cephfs
EOF
```

2. 输入以下命令配置镜像 registry，以使用 CephFS 文件系统存储：

```
$ oc patch config.image/cluster -p '{"spec":
{"managementState":"Managed","replicas":2,"storage":
{"managementState":"Unmanaged","pvc":{"claim":"registry-storage-pvc"}}}}' --type=merge
```

3.9.5. 其他资源

- 推荐的可配置存储技术
- 将镜像 registry 配置为使用 OpenShift Data Foundation

第 4 章 访问REGISTRY

您可以访问 registry 来查看日志和指标。您还可以保护并公开 registry。

使用 **podman login** 命令登录到 registry 后，您可以使用 **podman push** 或 **podman pull** 命令直接从集成 registry 中推送或拉取镜像。您可以使用的命令取决于您的用户权限。

4.1. 先决条件

- 您可以使用具有 **cluster-admin** 角色的用户访问集群。
- 您必须已经配置了一个身份供应商（IDP）。
- 为了抓取镜像，例如使用**podman pull**命令，用户必须具有**registry-viewer**角色。要添加此角色，请运行以下命令：

```
$ oc policy add-role-to-user registry-viewer <user_name>
```

- 要编写或推送镜像，如使用 **podman push** 命令，请完成以下步骤：
  - 您的帐户具有 **registry-editor** 角色。要添加此角色，请运行以下命令：

    ```
    $ oc policy add-role-to-user registry-editor <user_name>
    ```

  - 集群必须有一个可以推送镜像的现有项目。

4.2. 直接从集群访问 REGISTRY

您可以使用内部路由从集群内部访问 registry。

流程

1. 通过获取节点的名称来访问节点：

```
$ oc get nodes
```

```
$ oc debug nodes/<node_name>
```

2. 要在节点上启用对 **oc** 和 **podman** 等工具的访问，请将您的根目录改为 **/host**。运行命令的成功输出为 **Login Succeeded!**。

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

3. 使用您的访问令牌登录到容器镜像registry：

```
sh-4.2# oc login -u kubeadmin -p <password_from_install_log> https://api-int.
<cluster_name>.<base_domain>:6443
```

```
sh-4.2# podman login -u kubeadmin -p $(oc whoami -t) image-registry.openshift-image-
registry.svc:5000
```

注意

您可以为用户名传递几乎任何值。令牌包含所有必要的信息。如果用户名包含冒号，则会导致登录失败。

Image Registry Operator 创建路由，如 **default-route-openshift-image-registry.<cluster_name>**。

4. 针对 registry 执行 **podman pull** 和 **podman push** 操作。以下示例命令演示了这些操作。

   a. 抓取任意镜像：

   ```
   sh-4.2# podman pull <name.io>/<image>
   ```

   重要

   您可以抓取任意镜像，但是如果已添加了**system:registry**角色，则只能将镜像推送到您自己的registry中。

   b. 使用 **<registry_ip>:<port>/<project>/<image>** 格式标记（tag）新镜像。例如，**172.30.124.220:5000/openshift/image**。项目名称必须出现在这个 pull 规范中，以供 OpenShift Container Platform 把这个镜像正确放置在 registry 中，并在以后正确访问 registry 中的这个镜像：

   ```
   sh-4.2# podman tag <name.io>/<image> image-registry.openshift-image-registry.svc:5000/openshift/<image>
   ```

   注意

   您必须具有指定项目的**system:image-builder**角色，该角色允许用户写或推送镜像。否则，下一步中的**podman push**将失败。为了进行测试，您可以创建一个新项目来推送镜像。

   c. 将新标记的镜像推送到 registry：

   ```
   sh-4.2# podman push image-registry.openshift-image-registry.svc:5000/openshift/<image>
   ```

   注意

   将镜像推送到内部 registry 时，存储库名称必须使用 **<project>/<name>** 格式。在存储库名称中使用多个项目级别会导致身份验证错误。

4.3. 检查 REGISTRY POD 的状态

作为集群管理员，您可以列出在 **openshift-image-registry** 项目中运行的镜像 registry pod，并检查其状态。

先决条件

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

流程

- 列出 **openshift-image-registry** 项目中的 pod 并查看其状态。显示的输出示例用于演示目的。

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

```
NAME READY STATUS RESTARTS AGE
image-registry-79fb4469f6-llrln 1/1 Running 0 77m
node-ca-hjksc 1/1 Running 0 73m
node-ca-tftj6 1/1 Running 0 77m
node-ca-wb6ht 1/1 Running 0 77m
node-ca-zvt9q 1/1 Running 0 74m
```

4.4. 查看REGISTRY日志

使用**oc logs**命令可以查看registry中的日志信息。

流程

- 使用带有 deployments 的**oc logs**命令查看容器镜像 registry 的日志。显示的输出示例用于演示目的。

```
$ oc logs deployments/image-registry -n openshift-image-registry
```

```
2015-05-01T19:48:36.300593110Z time="2015-05-01T19:48:36Z" level=info
msg="version=v2.0.0+unknown"
2015-05-01T19:48:36.303294724Z time="2015-05-01T19:48:36Z" level=info msg="redis not
configured" instance.id=9ed6c43d-23ee-453f-9a4b-031fea646002
2015-05-01T19:48:36.303422845Z time="2015-05-01T19:48:36Z" level=info msg="using
inmemory layerinfo cache" instance.id=9ed6c43d-23ee-453f-9a4b-031fea646002
2015-05-01T19:48:36.303433991Z time="2015-05-01T19:48:36Z" level=info msg="Using
OpenShift Auth handler"
2015-05-01T19:48:36.303439084Z time="2015-05-01T19:48:36Z" level=info msg="listening
on :5000" instance.id=9ed6c43d-23ee-453f-9a4b-031fea646002
```

4.5. 访问REGISTRY的指标数据（METRICS）

OpenShift Container Registry 为 Prometheus metrics 提供了一个端点。Prometheus是一个独立的开源系统监视和警报工具包。指标在 registry 端点的 ***/extensions/v2/metrics*** 路径上公开。您可以通过运行包含集群角色的指标查询来访问指标。

流程

1. 如果还没有一个访问指标的集群角色，创建一个集群角色：

```
$ cat <<EOF | oc create -f -
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: prometheus-scraper
rules:
- apiGroups:
  - image.openshift.io
```

```
    resources:
    - registry/metrics
    verbs:
    - get
EOF
```

2. 输入以下命令在用户帐户中添加集群角色：

```
$ oc adm policy add-cluster-role-to-user prometheus-scraper <username>
```

3. 对于指标查询，获取用户令牌。

```
openshift:
$ oc whoami -t
```

4. 在节点或 pod 中运行指标查询。以下示例命令和输出演示了此任务。

```
$ curl --insecure -s -u <user>:<secret> \ 1
    https://image-registry.openshift-image-registry.svc:5000/extensions/v2/metrics | grep
imageregistry | head -n 20
```

- **<user>:<secret>**：**<user>** 对象可以是任意对象，但 **<secret>** 标签必须使用用户令牌。

```
# HELP imageregistry_build_info A metric with a constant '1' value labeled by major, minor, git commit & git version from which the image registry was built.
# TYPE imageregistry_build_info gauge
imageregistry_build_info{gitCommit="9f72191",gitVersion="v3.11.0+9f72191-135-dirty",major="3",minor="11+"} 1
# HELP imageregistry_digest_cache_requests_total Total number of requests without scope to the digest cache.
# TYPE imageregistry_digest_cache_requests_total counter
imageregistry_digest_cache_requests_total{type="Hit"} 5
imageregistry_digest_cache_requests_total{type="Miss"} 24
# HELP imageregistry_digest_cache_scoped_requests_total Total number of scoped requests to the digest cache.
# TYPE imageregistry_digest_cache_scoped_requests_total counter
imageregistry_digest_cache_scoped_requests_total{type="Hit"} 33
imageregistry_digest_cache_scoped_requests_total{type="Miss"} 44
# HELP imageregistry_http_in_flight_requests A gauge of requests currently being served by the registry.
# TYPE imageregistry_http_in_flight_requests gauge
imageregistry_http_in_flight_requests 1
# HELP imageregistry_http_request_duration_seconds A histogram of latencies for requests to the registry.
# TYPE imageregistry_http_request_duration_seconds summary
imageregistry_http_request_duration_seconds{method="get",quantile="0.5"} 0.01296087
imageregistry_http_request_duration_seconds{method="get",quantile="0.9"}
0.014847248
imageregistry_http_request_duration_seconds{method="get",quantile="0.99"}
0.015981195
imageregistry_http_request_duration_seconds_sum{method="get"}
12.260727916000022
```

4.6. 其他资源

- 允许 pod 在项目间引用镜像
- 移除 kubeadmin 用户
- 了解身份提供程序配置

第 5 章 开放REGISTRY

默认情况下，OpenShift 镜像 registry 在集群安装过程中是安全的，以便通过传输层安全(TLS)协议提供流量。与早期版本的OpenShift Container Platform不同，安装时registry不会向集群外部公开。

5.1. 手动公开默认 REGISTRY

通过使用路由公开 registry，用户不再需要从集群内部登录到默认的 OpenShift 镜像 registry。通过此外部访问，您可以使用路由地址从集群外部登录到 registry。然后，您可以使用路由主机标记并将镜像推送到现有项目中。

先决条件

- 以下的先决条件会被自动执行：
  - 部署 Registry Operator。
  - 部署 Ingress Operator。
- 您可以使用具有 **cluster-admin** 角色的用户访问集群。

流程

1. 要使用 **configs.imageregistry.operator.openshift.io** 资源中存在的 **defaultRoute** 参数公开 registry，请运行以下命令将 defaultRoute' 设置为 **true** ：

```
$ oc patch configs.imageregistry.operator.openshift.io/cluster --patch '{"spec":
{"defaultRoute":true}}' --type=merge
```

2. 运行以下命令来获取默认 registry 路由：

```
$ HOST=$(oc get route default-route -n openshift-image-registry --template='{{ .spec.host }}')
```

3. 运行以下命令，获取 Ingress Operator 的证书：

```
$ oc extract secret/$(oc get ingresscontroller -n openshift-ingress-operator default -o json | jq
'.spec.defaultCertificate.name // "router-certs-default"' -r) -n openshift-ingress --confirm
```

4. 运行以下命令，将提取的证书移到系统的可信 CA 目录中：

```
$ sudo mv tls.crt /etc/pki/ca-trust/source/anchors/
```

5. 运行以下命令，启用集群的默认证书以信任路由：

```
$ sudo update-ca-trust enable
```

6. 运行以下命令，使用 podman 使用默认路由登录：

```
$ sudo podman login -u kubeadmin -p $(oc whoami -t) $HOST
```

5.2. 手动公开受保护的REGISTRY

通过使用路由公开 registry，用户不再需要从集群内部登录到 OpenShift 镜像 registry。通过此外部访问，您可以使用路由地址从集群外部登录到 registry。然后，您可以使用路由主机标记并将镜像推送到现有项目中。

您可以使用**configs.imageregistry.operator.openshift.io**资源中的**DefaultRoute**参数或使用自定义路由来公开路由。

先决条件

- 以下的先决条件会被自动执行：
  - 部署 Registry Operator。
  - 部署 Ingress Operator。
- 您可以使用具有 **cluster-admin** 角色的用户访问集群。

流程

1. 要使用 **DefaultRoute** 参数公开 registry，请将 **DefaultRoute** 设置为 **True** ：

```
$ oc patch configs.imageregistry.operator.openshift.io/cluster --patch '{"spec":
{"defaultRoute":true}}' --type=merge
```

2. 输入以下命令使用 **podman** 登录：

```
$ HOST=$(oc get route default-route -n openshift-image-registry --template='{{ .spec.host }}')
```

```
$ podman login -u kubeadmin -p $(oc whoami -t) --tls-verify=false $HOST
```

   - **--tls-verify=false** ：如果路由的默认证书不受信任，则将此参数设置为 **false**。您可以将一个自定义的可信证书设置为 Ingress Operator 的默认证书。

3. 要使用自定义路由公开 registry，请使用路由的 TLS 密钥创建一个 secret。此步骤是可选的。如果不创建一个secret，则路由将使用Ingress Operator的默认TLS配置。

```
$ oc create secret tls public-route-tls \
    -n openshift-image-registry \
    --cert=</path/to/tls.crt> \
    --key=</path/to/tls.key>
```

4. 在 Registry Operator 中输入以下命令：

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

```
spec:
  routes:
    - name: public-routes
      hostname: myregistry.mycorp.organization
      secretName: public-route-tls
...
```

注意

如果为 registry 的路由提供了自定义 TLS 配置，则仅设置 **secretName**。

故障排除

- 创建 TLS secret 时出错