OpenShift Container Platform 4.4

容器镜像仓库（Registry）

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

Last Updated: 2021-03-11

OpenShift Container Platform 4.4 容器镜像仓库（Registry）

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

法律通告

摘要

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。

目录

第 1 章 镜像 REGISTRY **3**
1.1. 集成的OPENSHIFT CONTAINER PLATFORM REGISTRY 3

第 2 章 OPENSHIFT CONTAINER PLATFORM中的IMAGE REGISTRY OPERATOR **4**
2.1. 云平台和 OPENSTACK 上的镜像 REGISTRY 4
2.2. 裸机和 VSPHERE 上的镜像 REGISTRY 4
2.3. IMAGE REGISTRY OPERATOR 配置参数 5
2.4. 使用 CRD 启用 IMAGE REGISTRY 默认路由 5
2.5. 为镜像 REGISTRY 访问配置额外的信任存储 6
2.6. 为 IMAGE REGISTRY OPERATOR 配置一个存储凭证 6
2.7. 其他资源 7

第 3 章 设置和配置 REGISTRY **8**
3.1. 为 AWS 用户置备的基础架构配置 REGISTRY 8
3.2. 为 GCP 用户置备的基础架构配置 REGISTRY 9
3.3. 为 AZURE 用户置备的基础架构配置 REGISTRY 10
3.4. 为裸机配置 REGISTRY 11
3.5. 为 VSPHERE 配置 REGISTRY 14

第 4 章 REGISTRY 选项 **19**
4.1. 集成的OPENSHIFT CONTAINER PLATFORM REGISTRY 19
4.2. 第三方 REGISTRY 19
4.3. RED HAT QUAY REGISTRIES 19
4.4. 启用了身份验证的红帽 REGISTRY 19

第 5 章 访问REGISTRY **21**
5.1. 先决条件 21
5.2. 直接从集群访问REGISTRY 21
5.3. 检查 REGISTRY POD 的状态 23
5.4. 查看REGISTRY日志 23
5.5. 访问REGISTRY的指标数据（METRICS） 23

第 6 章 开放REGISTRY **26**
6.1. 手动公开受保护的REGISTRY 26

第 1 章 镜像 REGISTRY

1.1. 集成的OPENSHIFT CONTAINER PLATFORM 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。它为用户提供了一种现成的解决方案，供用户管理在已有集群基础架构上运行的，用于处理实际工作负载的镜像。这个registry可以象集群中的其他负载一样进行扩展，且不需要置备特殊的基础架构。此外，它已被集成到集群用户身份验证和授权系统中。这意味着，通过定义镜像资源上的用户权限就可以控制对镜像的创建和访问权限。

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。当一个新镜像被推送到registry时，集群会收到新镜像的通知，其他组件就可以对更新的镜像做出反应。

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。

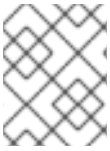
其他资源

- OpenShift Container Platform中的Image Registry Operator

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 Container Platform 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、GCP、Azure 或 OpenStack 中安装一个由安装程序置备的基础架构集群时，存储才会被自动配置。

当部署了control plane后，Operator 将会根据集群中的配置创建一个默认的 **configs.imageregistry.operator.openshift.io** 资源实例。

如果没有足够的信息来定义完整的**configs.imageregistry.operator.openshift.io**资源，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。

Image Registry Operator在**openshift-image-registry**命名空间中运行，并管理该位置中的 registry 实例。registry的所有配置和工作负载资源都位于该命名空间中。

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

2.2. 裸机和 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 本身的状态是 **Removed**。这允许 **openshift-installer** 在这些平台类型上完成安装。

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

注意

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

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

2.3. IMAGE REGISTRY OPERATOR 配置参数

configs.imageregistry.operator.openshift.io资源提供以下配置参数。

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。
logging	设置 registry 实例的 **loglevel**。
httpSecret	安全上载时 registry 所需的值，默认生成。
proxy	定义在调用 master API 和上游 registry 时要使用的代理。
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。
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。
replicas	registry 的副本数量。

2.4. 使用 CRD 启用 IMAGE REGISTRY 默认路由

在 OpenShift Container Platform 中 **Registry** Operator 控制 registry 的功能。这个 Operator 由 **configs.imageregistry.operator.openshift.io** CRD（Custom Resource Definition）定义。

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}}'
```

2.5. 为镜像 REGISTRY 访问配置额外的信任存储

Image.config.openshift.io/cluster 自定资源可包含对 ConfigMap 的引用，该 ConfigMap 包含要在镜像 registry 访问期间被信任的额外证书颁发机构。

先决条件

- CA必须经过PEM编码。

流程

您可以在**openshift-config**命名空间中创建ConfigMap，并在 **image.config.openshift.io** 子定义资源中的 **AdditionalTrustedCA** 中使用其名称，以提供与外部 registry 联系时可以被信任的额外CA。

对于每个要信任的额外 registry CA，ConfigMap 键是带有要信任此 CA 的端口的 registry 主机的名称，值是经 base64 编码的证书。

Image registry CA ConfigMap 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: | 1
    -----BEGIN CERTIFICATE-----
    ...
    -----END CERTIFICATE-----
```

1 如果 registry 带有端口，如 **registry-with-port.example.com:5000**，**:** 需要被 **..** 替换。

您可以按照以下过程配置其他CA。

1. 配置其他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
```

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

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 提供存储凭证配置。

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。

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=KEY1=value1
--from-literal=KEY2=value2 --namespace openshift-image-registry
```

2.7. 其他资源

- 为 AWS 用户置备的基础架构配置 registry
- 为 GCP 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

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

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。

先决条件

- 使用用户置备的 AWS 中有一个集群。
- 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

如果 Registry Operator 无法创建 S3 存储桶并自动配置存储，请进行以下操作。

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
```

```
storage:
  s3:
    bucket: <bucket-name>
    region: <region-name>
```

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。

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。

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 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。
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。
ImageRegistryConfigStorageS3CloudFront	CloudFront 将 Amazon Cloudfront 配置为 registry 中的存储中间件。它是可选的。

3.2. 为 GCP 用户置备的基础架构配置 REGISTRY

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 GCP 存储上的 GCS，secret 应该包含以下这个键，其值是 GCP 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. 带有用户置备基础架构的 GCP 的 registry 存储

您必须手动设置存储介质，并在 registry CRD 中配置设置。

先决条件

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

3.2.3. GCP GCS 的 Image Registry Operator 配置参数

流程

以下配置参数可用于 GCP 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 GCP 项目的项目 ID。它是可选的。
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 GCP 上是加密的。这将允许使用一个自定义加密密钥。

3.3. 为 AZURE 用户置备的基础架构配置 REGISTRY

3.3.1. 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
```

3.3.2. 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**中中输入存储配置：

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

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

3.4. 为裸机配置 REGISTRY

3.4.1. 安装过程中删除的镜像 registry

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

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

注意

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

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

3.4.2. 更改镜像 registry 的管理状态

要启动镜像 registry，需要把 Image Registry Operator 配置的 **managementState** 从 **Removed** 改为 **Managed**。

流程

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

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

3.4.3. 镜像 registry 存储配置

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

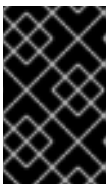
提供了配置持久性卷的说明，这是生产集群所需要的；也提供了将空目录配置为存储位置的说明，这仅适用于非生产集群。

3.4.4. 为裸机配置 registry 存储

作为集群管理员，在安装后需要配置 registry 来使用存储。

先决条件

- 具有 Cluster Administrator 权限
- 在裸机上有一个集群。
- 为集群置备的持久性存储，如 Red Hat OpenShift Container Storage。



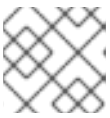
重要

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

- 必须具有 100Gi 容量。

流程

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



注意

使用共享存储时，请查看您的安全设置以防止被外部访问。

2. 验证您没有 registry pod:

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

注意

如果存储类型为 **emptyDIR**，则副本数不能超过 **1**。

3. 检查 registry 配置：

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

输出示例

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

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

4. 检查 **clusteroperator** 的状态：

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

3.4.5. 在非生产集群中配置镜像 registry 存储

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

流程

- 将镜像 registry 存储设置为空目录：

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

警告

仅可为非生产集群配置这个选项。

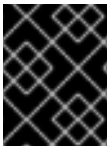
如果在 Image Registry Operator 初始化其组件前运行此命令，**oc patch** 命令会失败并显示以下错误：

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

等待几分钟，然后再次运行该命令。

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

流程

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

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

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

3. 编辑 registry 配置，使其引用正确的 PVC。

3.4.7. 其他资源

有关为裸机配置 registry 存储的详情，请参考推荐的可配置存储技术。

3.5. 为 VSPHERE 配置 REGISTRY

3.5.1. 安装过程中删除的镜像 registry

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

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

注意

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

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

3.5.2. 更改镜像 registry 的管理状态

要启动镜像 registry，需要把 Image Registry Operator 配置的 **managementState** 从 **Removed** 改为 **Managed**。

流程

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

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

3.5.2.1. 镜像 registry 存储配置

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

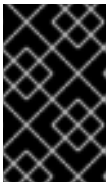
提供了配置持久性卷的说明，这是生产集群所需要的；也提供了将空目录配置为存储位置的说明，这仅适用于非生产集群。

3.5.3. 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

- 具有 Cluster Administrator 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 Container Storage。



重要

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

- 必须有“ 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可能会出现问题。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，不推荐使用 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. 为了配置 registry 使用存储，需要修改 **configs.imageregistry/cluster** 资源中的 **spec.storage.pvc** 。

注意

使用共享存储时，请查看您的安全设置以防止被外部访问。

2. 验证您没有 registry pod:

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

注意

如果存储类型为 **emptyDIR**，则副本数不能超过 **1**。

3. 检查 registry 配置：

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

输出示例

```
storage:
  pvc:
    claim: 1
```

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

4. 检查 **clusteroperator** 的状态：

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

3.5.4. 在非生产集群中配置镜像 registry 存储

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

流程

- 将镜像 registry 存储设置为空目录：

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

警告

仅可为非生产集群配置这个选项。

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

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

等待几分钟，然后再次运行该命令。

3.5.5. 为 VMware vSphere 配置块 registry 存储

在作为集群管理员升级时，要允许镜像 registry 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，但不建议将其用于生产环境中的镜像 registry。在块存储上配置 registry 的安装不具有高可用性，因为 registry 无法拥有多个副本。

流程

1. 要将镜像 registry 存储设置为块存储类型，对 registry 进行补丁，使其使用 **Recreate** rollout 策略，且仅使用 **1** 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 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**：

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: image-registry-storage 1
spec:
  accessModes:
  - ReadWriteOnce 2
  resources:
    requests:
      storage: 100Gi 3
```

   1. 代表 **PersistentVolumeClaim** 对象的唯一名称。
   2. PersistentVolumeClaim 的访问模式。使用 **ReadWriteOnce** 时，单个节点可以通过读写权限挂载这个卷。
   3. PersistentVolumeClaim 的大小。

   b. 从文件创建 **PersistentVolumeClaim** 对象：

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

3. 编辑 registry 配置，使其可以正确引用 PVC:

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

输出示例

```
storage:
  pvc:
    claim: 1
```

1. 通过创建自定义 PVC，您可以将 **claim** 字段留空以用于默认自动创建 **image-registry-storage** PVC。

有关配置 registry 存储以便引用正确的 PVC 的说明，请参阅为 vSphere 配置 registry。

3.5.6. 其他资源

有关为 vSphere 配置 registry 存储的详情，请参考推荐的可配置存储技术。

第 4 章 REGISTRY 选项

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。为此，OpenShift Container Platform提供了一个内部集成的容器镜像 registry。您可以在OpenShift Container Platform环境中部署它以在本地管理镜像。

4.1. 集成的OPENSHIFT CONTAINER PLATFORM 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。它为用户提供了一种现成的解决方案，供用户管理在已有集群基础架构上运行的，用于处理实际工作负载的镜像。这个registry可以象集群中的其他负载一样进行扩展，且不需要置备特殊的基础架构。此外，它已被集成到集群用户身份验证和授权系统中。这意味着，通过定义镜像资源上的用户权限就可以控制对镜像的创建和访问权限。

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。当一个新镜像被推送到registry时，集群会收到新镜像的通知，其他组件就可以对更新的镜像做出反应。

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。

4.2. 第三方 REGISTRY

OpenShift Container Platform可以使用由第三方 registry 提供的镜像创建容器，但是这些registry 可能不会象OpenShift Container Platform中集成的registry一样提供相同的镜像通知支持。在这种情况下，OpenShift Container Platform将在创建镜像流时从远程 registry 中获取 tag。

oc import-image <stream>就可以更新获取的 tag。当检测到新的镜像时，以前的构建和部署将会被重新创建。

4.2.1. 身份验证

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。

4.3. 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 registry 功能包括跨区域复制、镜像扫描及镜像回滚（roll back）功能。

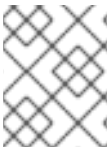
请通过 Quay.io 网站设置您自己的托管 Quay registry 帐户。之后，请按照Quay教程中的内容登录到Quay registry并开始管理镜像。

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。

4.4. 启用了身份验证的红帽 REGISTRY

Red Hat Container Catalog （**Registry.redhat.io**)是一个托管的镜像 registry，通过它可以获得所需的容器镜像。

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。当迁移到新registry后，现有的registry仍将在一段时间内可用。



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。虽然可以在OpenShift Container Platform上使用此身份验证方法，但在生产环境部署中不支持此方法。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。

所有镜像流均指向新的registry。因为 registry 需要进行身份验证才可以访问，所以 Samples Operator 会创建 **samples-registry-credentials** secret。

您需要将凭据放在两个位置：

- **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。

第 5 章 访问REGISTRY

使用以下部分介绍的内容来访问registry，包括查看日志和指标，以及保护和公开registry。

您可以使用**podman**命令直接访问registry。您可以使用**podman push**或**podman pull**等操作直接从集成的registry中进行镜像的 pull 和 push 操作。您需要首先使用**oc login**命令登录到registry。您可以执行的操作取决于您的用户权限，如以下各节所述。

5.1. 先决条件

- 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。添加此角色：

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

- 为了写或推送镜像，例如使用 **podman push**命令，用户必须具有 **registry-editor** 角色。添加此角色：

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

5.2. 直接从集群访问REGISTRY

您可以从集群内部访问registry。

流程

通过使用内部路由从集群访问registry：

1. 使用节点地址来访问节点：

```
$ oc get nodes
$ oc debug nodes/<node_address>
```

2. 要使用节点上的 **oc** 和 **podman** 等工具程序，请运行以下命令：

```
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
```

您应该看到一条确认登录的消息，例如：

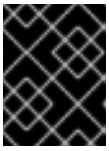
```
Login Succeeded!
```

注意

用户名可以是任何值，令牌包含了所有必要的信息。如果用户名包含冒号，则会导致登录失败。

因为 Image Registry Operator 创建了路由，所以它将与 **default-route-openshift-image-registry.<cluster_name>** 类似。

4. 针对您的registry执行**podman pull**和**podman push**操作：



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。

在以下示例中，使用：

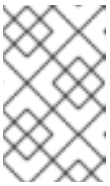
组件	值
<registry_ip>	**172.30.124.220**
<port>	**5000**
<project>	**openshift**
<image>	**image**
<tag>	忽略 (默认为 **latest**)

a. 抓取任意镜像：

```
$ podman pull name.io/image
```

b. 使用 **<registry_ip>:<port>/<project>/<image>** 格式标记（tag）新镜像。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：

```
$ 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：

```
$ podman push image-registry.openshift-image-registry.svc:5000/openshift/image
```

5.3. 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。
- 已安装 OpenShift CLI（**oc**）。

流程

1. 列出 **openshift-image-registry** 项目中的 pod 并查看其状态：

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

输出示例

```
NAME READY STATUS RESTARTS AGE
cluster-image-registry-operator-764bd7f846-qqtpb 1/1 Running 0 78m
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
```

5.4. 查看REGISTRY日志

使用**oc logs**命令可以查看registry中的日志信息。

流程

1. 使用带有 deployments 的**oc logs**命令查看容器镜像 registry 的日志：

```
$ oc logs deployments/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
```

5.5. 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。

metrics 可以通过registry端点的*/extensions/v2/metrics*路径获得。

流程

您可以通过两种方式访问指标数据：运行指标数据查询或使用集群角色。

指标数据查询

1. 运行指标查询，例如：

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

1 **<user>**可以是任意的，但**<secret>**必须与registry配置中指定的值匹配。

集群角色

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 |
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: prometheus-scraper
rules:
- apiGroups:
  - image.openshift.io
  resources:
  - registry/metrics
  verbs:
```

```
  - get
EOF
$ oc create -f -
```

2. 将此角色添加到用户，运行以下命令：

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

3. 使用集群角色访问指标数据。配置文件中与指标数据相关的部分应如下所示：

```
openshift:
  version: 1.0
  metrics:
    enabled: true
...
```

其他资源

- **kubeadmin**可以访问 registry，直到其被删除。有关更多信息，请参阅删除 kubeadmin 用户 。
- 有关配置身份供应商的更多信息，请参阅了解身份供应商配置 。

第 6 章 开放REGISTRY

默认情况下，OpenShift Container Platform registry在集群安装期间是加密的，它需要使用TLS进行访问。与早期版本的OpenShift Container Platform不同，安装时registry不会向集群外部公开。

6.1. 手动公开受保护的REGISTRY

通过使用路由可以开放从外部访问OpenShift Container Platform registry的通道，用户不再需要从集群内部登录到OpenShift Container Platform registry。您可以使用路由地址从集群以外登陆到 registry，并使用路由主机进行镜像的 tag 和 push 操作。

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。

流程

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。

使用**DefaultRoute**公开registry：

1. 将**DefaultRoute**设置为**True** ：

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

2. 使用 **podman** 登录：

```
$ HOST=$(oc get route default-route -n openshift-image-registry --template='{{ .spec.host }}')
$ podman login -u $(oc whoami) -p $(oc whoami -t) --tls-verify=false $HOST (1)
```

(1) 如果集群的默认路由证书不受信任，则需要**--tls-verify=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。

使用自定义路由公开registry：

1. 使用路由的 TLS 密钥创建一个 secret：

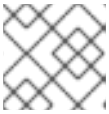
```
$ oc create secret tls public-route-tls \
    -n openshift-image-registry \
    --cert=</path/to/tls.crt> \
    --key=</path/to/tls.key>
```

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。

2. 在 Registry Operator 中：

```
spec:
  routes:
```

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



注意

如果为registry的路由提供了一个自定义的 TLS 配置，则仅需设置**secretName**。