Red Hat Quay 3.11

Red Hat Quay 架构

Red Hat Quay 架构

Last Updated: 2024-06-06

Red Hat Quay 3.11 Red Hat Quay 架构

Red Hat Quay 架构

法律通告

摘要

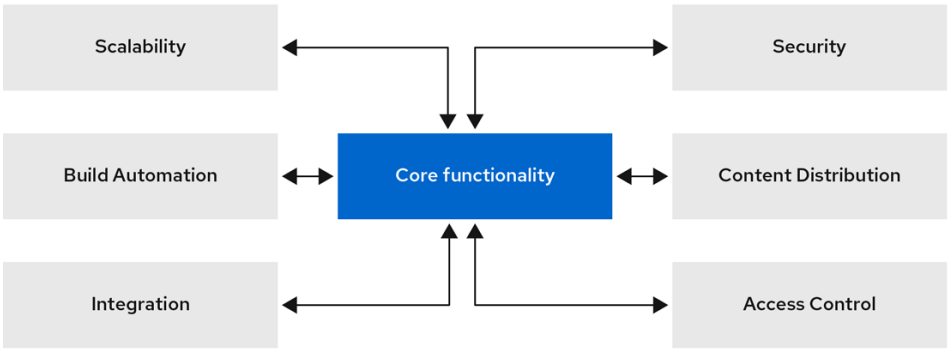
Red Hat Quay 架构

目录

第 1 章 RED HAT QUAY 概述

Red Hat Quay 是适用于您的企业的分布式和高可用性容器镜像 registry。

Red Hat Quay 容器 registry 平台在任意基础架构上提供安全存储、分发、访问控制、地理复制、存储库镜像和管理容器及云原生工件。它作为独立组件或 OpenShift Container Platform 的 Operator 提供，可部署于预备或公共云上。



本指南为部署 Red Hat Quay 时使用的架构模式提供了深入了解。本指南还提供大小调整指导和部署先决条件，以及确保 Red Hat Quay registry 高可用性的最佳实践。

1.1. 可扩展性和高可用性(HA)

用于 Red Hat Quay 的代码库与用于 Quay.io 的代码库相同，后者是由红帽托管的高可用性容器镜像 registry。Quay.io 和 Red Hat Quay 提供了一个多租户 SaaS 解决方案。因此，用户可以确信其部署可以在具有高可用性的大规模交付，无论其部署是预置型还是公共云上。

1.2. 内容发布

Red Hat Quay 中的内容发布功能包括：

存储库镜像

Red Hat Quay 存储库镜像可让您将 Red Hat Quay 和其他容器 registry 中的镜像（如 JFrog Artifactory、Harbor 或 Sonatype Nexus 仓库）镜像到 Red Hat Quay 集群中。使用存储库镜像，您可以根据存储库名称和标签将镜像同步到 Red Hat Quay。

geo-replication

Red Hat Quay geo-replication 允许多个地理位置分散的 Red Hat Quay 部署，从客户端或用户的角度来看，作为单个 registry 工作。它显著提高了在全局分布式 Red Hat Quay 设置中的推送和拉取性能。镜像数据在后台使用透明故障转移和客户端重定向进行异步复制。

在断开连接的或 air-gapped 环境中部署

Red Hat Quay 以两种方式之一在断开连接的环境中部署：

- Red Hat Quay 和 Clair 连接到互联网，它有一个 air-gapped OpenShift Container Platform 集群，通过防火墙中的显式允许列表漏洞访问 Red Hat Quay registry。
- 使用两个独立的 Red Hat Quay 和 Clair 安装。一个安装连接到互联网，另一个安装在断开连接的或防火墙的环境中。使用离线介质手动将镜像和漏洞数据从连接的环境传送到断开连接的环境中。

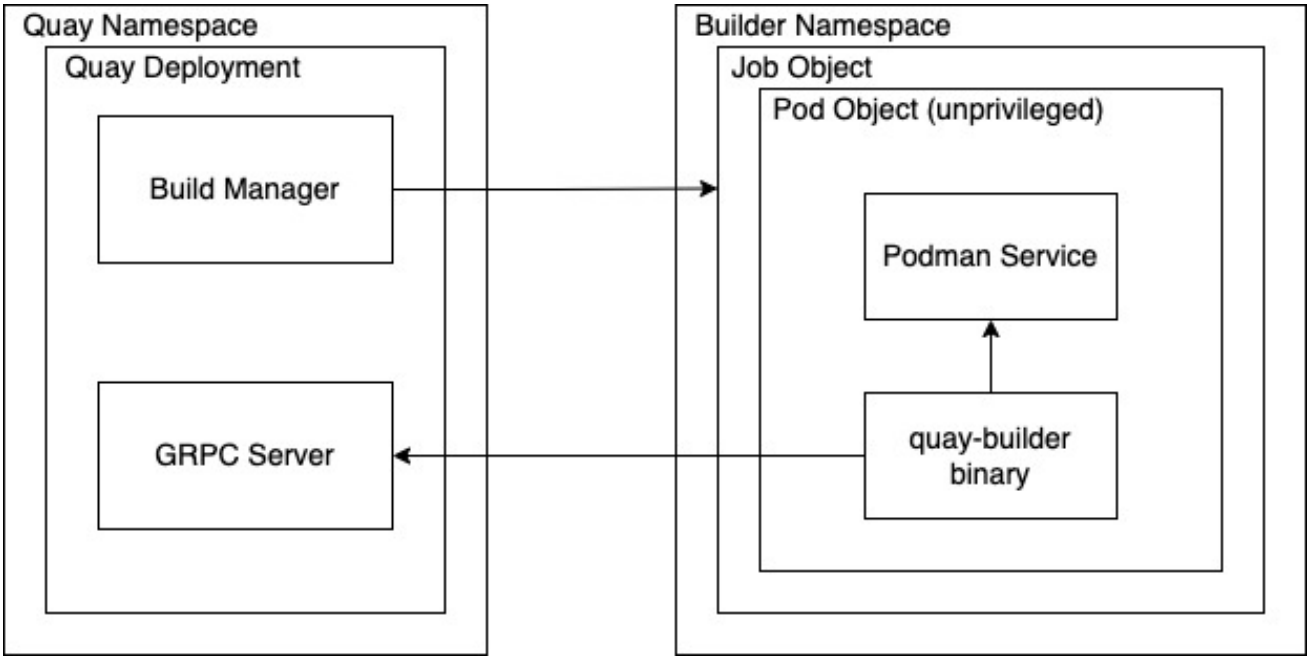
1.3. 构建自动化

Red Hat Quay 支持使用 OpenShift Container Platform 或 Kubernetes 平台上的一组 worker 节点来构建 Dockerfile。构建触发器（如 GitHub Webhook）可以配置为在提交新代码时自动构建新版本的存储库。

在 Red Hat Quay 3.7 之前，Red Hat Quay 在 Pod 启动的虚拟机中运行 Podman 命令。在虚拟平台上运行构建需要启用嵌套虚拟化，这不适用于 Red Hat Enterprise Linux (RHEL)或 OpenShift Container Platform。因此，构建必须在裸机集群上运行，这效率较低。使用 Red Hat Quay 3.7 时，这个要求已被删除，构建可以在虚拟化或裸机平台上运行的 OpenShift Container Platform 集群。

1.4. RED HAT QUAY 增强的构建架构

下图显示了增强功能的预期设计流和架构：



在这个版本中，构建管理器首先会创建**作业对象**。然后，**作业对象** 使用 **quay-builder-image** 创建 pod。**quay-builder-image** 将包含 **quay-builder 二进制文件** 和 **Podman** 服务。创建的 pod **作为非特权**运行。然后，**quay-builder 二进制文件** 会在通信状态和从 Build Manager 检索构建信息时构建镜像。

1.5. 集成

Red Hat Quay 可以和几乎所有与 Git 兼容的系统集成。Red Hat Quay 为 GitHub、GitLab 或 BitBucket 提供自动配置，允许用户持续构建并提供其容器化软件。

1.5.1. REST API

Red Hat Quay 提供了完整的 OAuth 2 RESTful API。RESTful API 提供以下优点：

- 从 URL 的每个 Red Hat Quay 实例的端点（例如 **https://quay-server.example.com/api/v1**）提供
- 允许用户通过浏览器连接到端点，连接到由 Swagger 使用的发现端点提供的 **GET**、**DELETE**、**POST** 和 **PUT** Red Hat Quay 设置。
- API 可以被 URL 调用，例如 **https://quay-server.example.com/api/v1**，并使用 JSON 对象作为有效负载。

1.6. 安全性

Red Hat Quay 专为真实企业用例而构建，其中内容监管和安全性是两个主要的领域。

Red Hat Quay 内容监管和安全包括通过 Clair 进行内置漏洞扫描。

1.6.1. TLS/SSL 配置

您可以在配置工具 UI 或配置捆绑包中为 Red Hat Quay registry 配置 SSL/TLS。SSL/TSL 连接到数据库、到镜像存储，也可以通过配置工具指定到 Redis。

数据库中的敏感字段以及运行时会自动加密。您还可以需要 HTTPS，并在镜像操作过程中验证 Red Hat Quay registry 的证书。

1.6.2. Clair

Clair 是一个开源应用，它利用静态代码分析来解析镜像内容并报告影响内容的漏洞。Clair 打包了 Red Hat Quay，并可用于独立和 Operator 部署。它可以在高度可扩展的配置中运行，其中组件可以根据企业环境单独进行扩展。

1.6.3. Red Hat Quay Operator 安全性

当使用 Red Hat Quay Operator 部署 Red Hat Quay Operator 时，**tls** 组件默认设置为 **managed**，OpenShift Container Platform 的证书颁发机构用于创建 HTTPS 端点并轮转 TLS 证书。

如果将 **tls** 组件设置为 **非受管** 组件，您可以将自定义证书提供给直通路由，但您需要负责证书轮转。

1.6.4. 完全隔离的构建

Red Hat Quay 现在支持构建使用裸机和虚拟构建器的 Dockerfile。

通过使用裸机 worker 节点，每个构建都是在一个临时虚拟机中执行的，以确保构建运行时的隔离和安全性。这提供了对 rogue 有效负载的最佳保护。

直接在容器中运行构建与使用虚拟机时没有相同的隔离，但它仍然提供良好的保护。

1.6.5. 基于角色的访问控制

Red Hat Quay 提供机构和团队的完整隔离 registry 内容，以及对用户和自动化工具进行读、写入和管理访问权限的精细权利。

1.7. 最近添加的功能

有关最新功能、增强功能、弃用和已知问题的信息，请参阅 Red Hat Quay 发行注记。

第 2 章 RED HAT QUAY 的先决条件

在部署 Red Hat Quay 前，您必须置备镜像存储、数据库和 Redis。

2.1. 镜像存储后端

Red Hat Quay 将所有二进制 Blob 存储在其存储后端中。

本地存储

Red Hat Quay 可以使用本地存储，但这只用于概念验证或测试设置，因为无法保证二进制 Blob 的持久性。

HA 存储设置

对于 Red Hat Quay HA 部署，您必须提供 HA 镜像存储，例如：

- **Red Hat OpenShift Data Foundation**（以前称为 Red Hat OpenShift Container Storage）是软件定义的容器存储。Red Hat OpenShift Data Foundation 作为 OpenShift Container Platform 的数据和存储服务平台设计，可帮助团队在云中快速高效地开发和部署应用程序。如需更多信息，请参阅 https://www.redhat.com/en/technologies/cloud-computing/openshift-data-foundation。

- **Ceph 对象网关**（也称为 RADOS 网关）是一个存储解决方案示例，可以提供 Red Hat Quay 所需的对象存储。有关如何将 Ceph 存储用作高可用性存储后端的详细信息，请参阅 Quay 高可用性指南。有关 Red Hat Ceph Storage 和 HA 设置的更多信息，请参阅 Red Hat Ceph Storage 架构指南

geo-replication

本地存储无法用于异地复制，因此必须部署受支持的内部或基于云的对象存储解决方案。本地化镜像存储在每个地区提供，镜像拉取则从最接近的可用存储引擎提供。容器镜像推送将写入 Red Hat Quay 实例的首选存储引擎，然后在后台将复制到其他存储引擎。这需要镜像存储能够从所有区域访问。

2.1.1. 支持的镜像存储引擎

Red Hat Quay 支持以下内部存储类型：

- Ceph/Rados RGW
- OpenStack Swift
- Red Hat OpenShift Data Foundation 4 （通过 NooBaa）

Red Hat Quay 支持以下公有云存储引擎：

- Amazon Web Services (AWS) S3
- Google Cloud Storage
- Azure Blob Storage

2.1.2. 不支持的镜像存储引擎

目前，不支持 Hitachi HCP。因为 S3 的每个实现都不同，因此过去 Hitachi HCP 存在问题。如果使用 Ceph/RADOS 驱动程序，Hitachi HCP 可能会正常工作，但 Red Hat Quay 无法保证它在所有场景中都可以正常工作，且不受支持。

2.2. 数据库后端

Red Hat Quay 将其所有配置信息存储在 **config.yaml** 文件中。注册表元数据，如用户信息、机器人帐户、团队、权限、机构、镜像、标签、清单等存储在数据库后端内。如果需要，日志可以推送到 ElasticSearch。PostgreSQL 是首选的数据库后端，因为它可用于 Red Hat Quay 和 Clair。

以后的 Red Hat Quay 版本将删除对使用 MySQL 和 MariaDB 作为数据库后端的支持，该后端自 Red Hat Quay 3.6 版本起已被弃用。在此之前，MySQL 仍根据 支持列表进行支持，但不会收到其他功能或明确的测试覆盖。Red Hat Quay Operator 仅在管理数据库时支持 PostgreSQL 部署。如果要使用 MySQL，您必须手动部署，并将数据库组件设置为 **managed: false**。

在高可用性(HA)配置中部署 Red Hat Quay 需要置备您的数据库服务以实现高可用性。如果 Red Hat Quay 在公共云基础架构上运行，建议您使用云供应商提供的 PostgreSQL 服务，但也支持 MySQL。

地理复制需要一个可从所有区域访问的、共享数据库。

2.3. REDIS

Red Hat Quay 将构建程序日志存储在 Redis 缓存中。由于存储的数据是临时的，因此 Redis 不需要高度可用，即使它处于有状态状态。

如果 Redis 失败，您将丢失对构建日志、构建器和垃圾收集器服务的访问。此外，用户事件将不可用。

您可以使用 Red Hat Software Collections 中的 Redis 镜像，或来自您喜欢的任何其他源。

第 3 章 RED HAT QUAY 基础架构

Red Hat Quay 在内部或公有云的任何物理或虚拟基础架构上运行。部署范围从简单到大规模扩展，如下所示：

- 开发人员笔记本上的 all-in-one 设置
- 虚拟机或 OpenShift Container Platform 上的高可用性
- 地理位置分散在多个可用区和区域

3.1. 在独立主机上运行 RED HAT QUAY

您可以使用 Ansible 或其他自动化套件自动化独立部署过程。所有独立主机都需要有效的 Red Hat Enterprise Linux (RHEL)订阅。

概念验证部署

Red Hat Quay 在带有镜像存储、容器化数据库、Redis 以及可选的 Clair 安全扫描的机器上运行。

高可用性设置

Red Hat Quay 和 Clair 在跨多个主机的容器中运行。您可以使用 **systemd** 单元来确保在失败时重启或重启时重启。
独立主机上的高可用性设置需要客户提供的负载均衡器，可以是低级 TCP 负载均衡器或应用程序负载均衡器，能够终止 TLS。

3.2. 在 OPENSHIFT 上运行 RED HAT QUAY

Red Hat Quay Operator for OpenShift Container Platform 提供以下功能：

- 使用自定义选项自动部署和管理 Red Hat Quay
- 管理 Red Hat Quay 及其所有依赖项
- 自动扩展和更新
- 与现有的 OpenShift Container Platform 进程集成，如 GitOps、监控、警报、日志记录
- 作为 Red Hat OpenShift Data Foundation (ODF) Operator 的一部分，置备具有有限可用性的对象存储，由多云对象网关(NooBaa)支持。该服务不需要额外订阅。
- ODF Operator 提供的横向扩展、高可用性对象存储。该服务需要额外的订阅。

Red Hat Quay 可以在 OpenShift Container Platform 基础架构节点上运行。因此，不需要进一步的订阅。在 OpenShift Container Platform 上运行 Red Hat Quay 有以下优点：

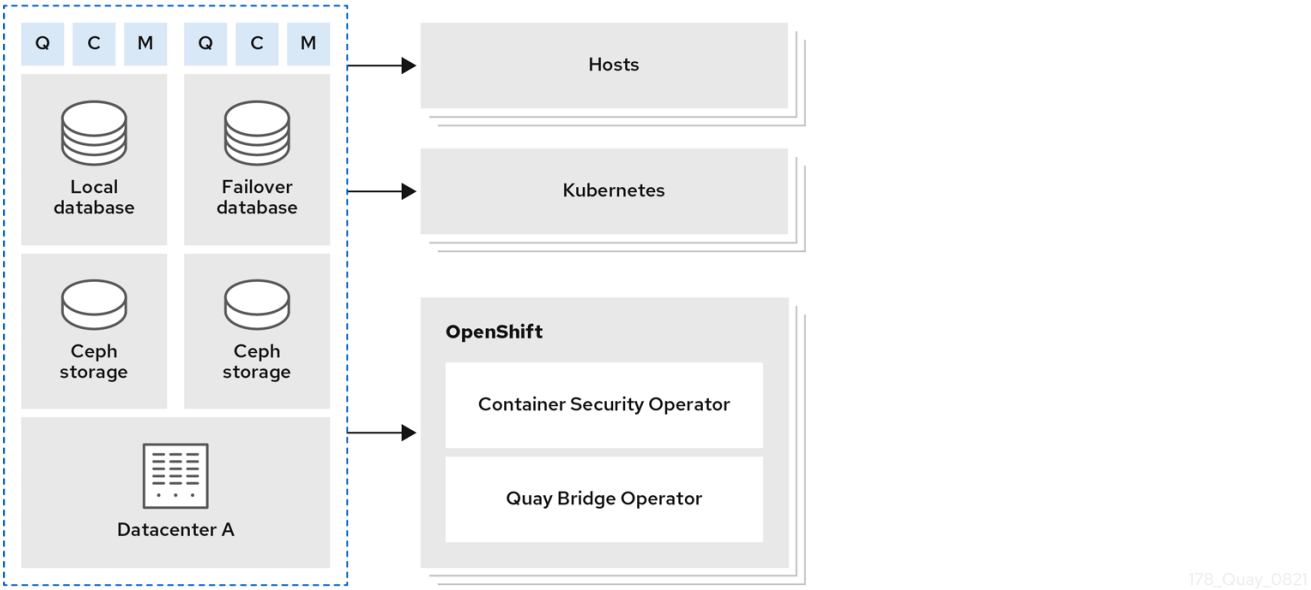
- **零到这里：** 简化 Red Hat Quay 和相关组件的部署意味着您可以立即开始使用产品
- **可扩展性：** 根据实际负载，使用集群计算容量通过自动化扩展来管理需求
- **简化网络 ：** 使用 OpenShift Container Platform TLS 证书和路由通过 HTTPS 自动配置负载均衡器和流量入口安全
- **声明性配置管理：存储在 CustomResource 对象中用于 GitOps 友好生命周期管理的配置**
- **可重复性 ：** 无论 Red Hat Quay 和 Clair 的副本数如何

- **OpenShift 集成：** 额外的服务使用 OpenShift Container Platform 监控和 Alerting 功能来管理单一集群上的多个 Red Hat Quay 部署

3.3. 将独立 RED HAT QUAY 与 OPENSHIFT CONTAINER PLATFORM 集成

虽然 Red Hat Quay Operator 确保无缝部署和管理在 OpenShift Container Platform 上运行的 Red Hat Quay，但也可以以独立模式运行 Red Hat Quay，然后向一个或多个 OpenShift Container Platform 集群提供内容。

将独立 Red Hat Quay 与 OpenShift Container Platform 集成



有几个 Operator 可用于帮助将基于 Red Hat Quay 的 Red Hat Quay 部署与 OpenShift Container Platform 集成，如下所示：

Red Hat Quay Cluster Security Operator

将 Red Hat Quay 漏洞扫描结果转发到 OpenShift Container Platform 控制台

Red Hat Quay Bridge Operator

通过将 Red Hat Quay 与 OpenShift Container Platform 与 OpenShift Container Platform 搭配使用来确保无缝集成和用户体验

3.4. MIRROR REGISTRY FOR RED HAT OPENSHIFT

mirror registry for Red Hat Quay 是 Red Hat Quay 的小型版本，您可以将其用作为断开连接的安装镜像 OpenShift Container Platform 所需的容器镜像的目标。

对于断开连接的 OpenShift Container Platform 部署，需要一个容器 registry 来安装集群。要在这样的集群中运行 production-grade registry 服务，您必须创建一个单独的 registry 部署来安装第一个集群。*mirror registry for Red Hat OpenShift* 可解决这个问题，并包含在每个 OpenShift Container Platform 订阅中。它可用于从 OpenShift 控制台 **Downloads**页面下载。

mirror registry for Red Hat OpenShift 允许用户使用 **mirror-registry** 命令行界面(CLI)工具安装一个较小的 Red Hat Quay 版本及其所需的组件。*mirror registry for Red Hat OpenShift* 会自动部署，它带有预先配置的本地存储和一个本地的数据库。它还包括自动生成的用户凭证和访问权限，其中只有一个输入集，且不需要额外配置选项。

mirror registry for Red Hat OpenShift 提供了一个预先确定的网络配置，并在成功时报告部署的组件凭证并访问 URL。另外还提供了一组有限的可选配置输入，如完全限定域名(FQDN)服务、超级用户名称和密码，以及自定义 TLS 证书。这为用户提供了一个容器 registry，以便在受限网络环境中运行 OpenShift Container Platform 时，轻松创建所有 OpenShift Container Platform 发行版本内容的离线镜像。

mirror registry for Red Hat OpenShift 仅限于托管安装断开连接的 OpenShift Container Platform 集群（如发行镜像或 Operator 镜像）所需的镜像。它使用本地存储。由客户创建的内容不应由 *mirror registry for Red Hat OpenShift* 托管。

与 Red Hat Quay 不同，*mirror registry for Red Hat OpenShift* 不是高可用性 registry。仅支持本地文件系统存储。对于多个集群的环境，不推荐使用 *mirror registry for Red Hat OpenShift*，因为有多个集群可以在更新集群时会存在单点故障。建议为 *Red Hat OpenShift 使用 mirror registry* 来安装一个集群，该集群可以托管生产环境级别的、高度可用的 registry，如 Red Hat Quay，它可以为其他集群提供 OpenShift Container Platform 内容。

如需更多信息，请参阅 *为 Red Hat OpenShift 创建带有镜像 registry 的镜像 registry*。

3.5. 与多个 REGISTRY 相比，单个

许多用户都考虑运行多个不同的 registry。Red Hat Quay 的首选方法是使用单个共享 registry：

- 如果您希望在开发和生产镜像之间明确分离，或者内容来源明确分离，例如保留与内部源不同的第三方镜像，您可以使用机构和存储库以及基于角色的访问控制(RBAC)，以实现所需的分离。
- 鉴于镜像 registry 是企业环境中的一个关键组件，您可能需要使用不同的部署来测试 registry 软件的升级。Red Hat Quay Operator 为补丁版本以及次要或主要更新更新 registry。这意味着，任何复杂的流程都是自动化的，因此您无需置备多个 registry 实例来测试升级。
- 在 Red Hat Quay 中，您部署的每个集群都需要有一个单独的 registry。Red Hat Quay 经验证可在 Quay.io 进行扩展，并可向数千个集群提供内容。
- 即使在多个数据中心中进行部署，您仍然可以使用单个 Red Hat Quay 实例向多个物理披露数据中心提供内容，或使用具有负载均衡器的 HA 功能在数据中心间扩展。或者，您可以使用 Red Hat Quay geo-replication 功能在物理距离数据中心之间扩展。这需要调配全局负载均衡器或基于 DNS 的地理感知负载平衡。
- 一个场景，可能需要运行多个不同的 registry，即您要为每个 registry 指定不同的配置。

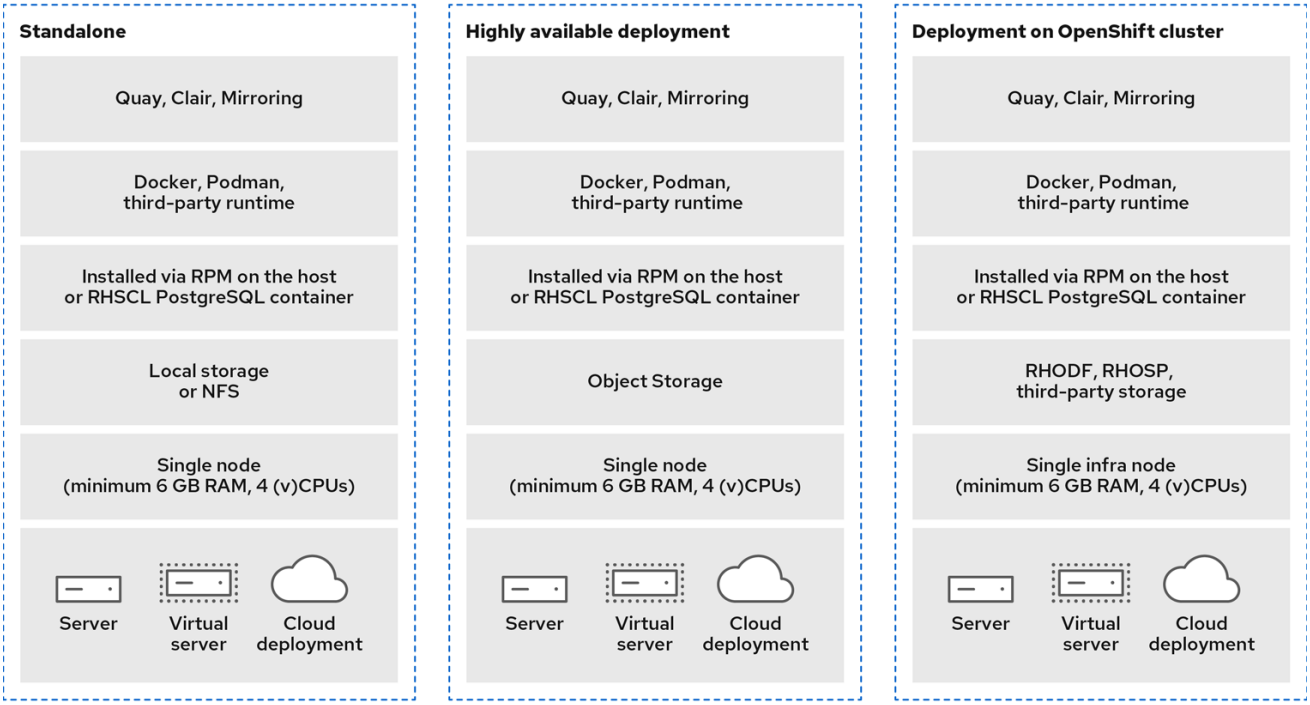
总之，运行共享 registry 可帮助您节省存储、基础架构和操作成本，但在某些情况下可能需要专用的 registry。

第 4 章 在内部部署 RED HAT QUAY

下图显示了以下部署类型的内部配置示例：

- 独立概念
- 在多个主机上部署高可用性
- 使用 Red Hat Quay Operator 在 OpenShift Container Platform 集群上部署

内部配置示例

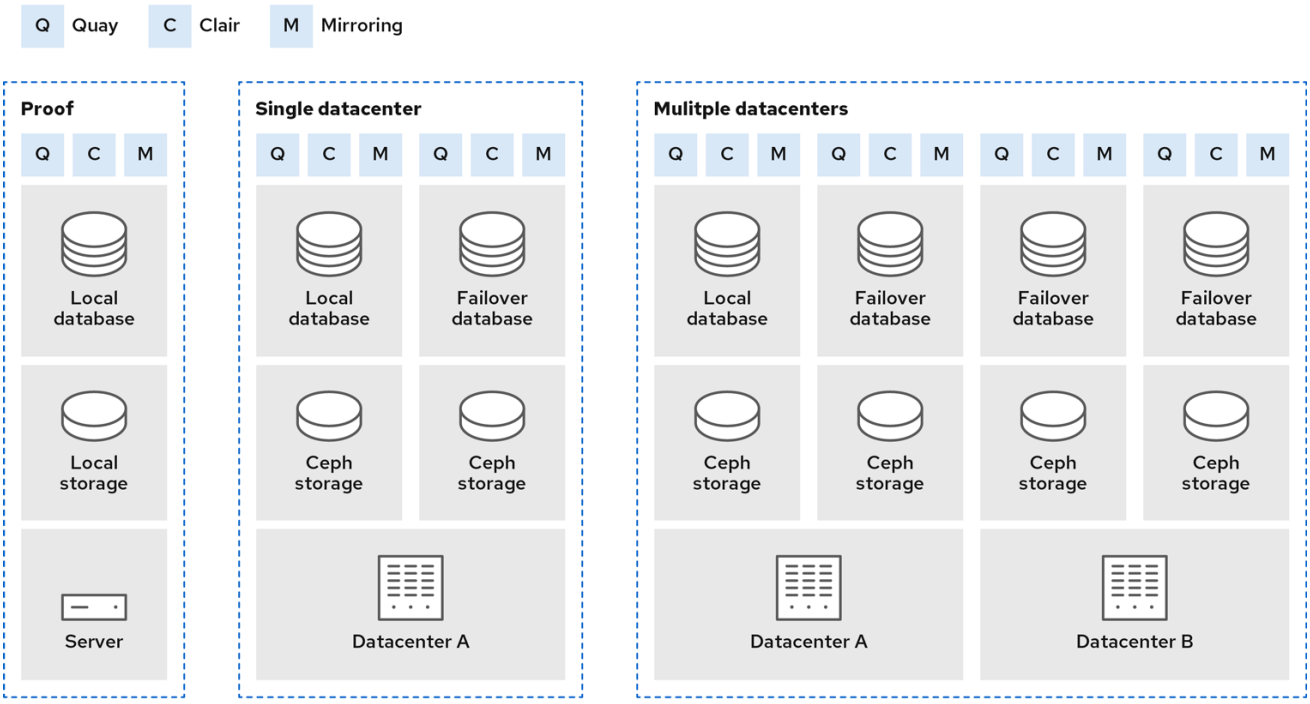


178_Quay_0821

4.1. RED HAT QUAY 示例部署

下图显示了 Red Hat Quay 的 3 个可能部署：

部署示例



Concept 的证明

在单一节点上运行 Red Hat Quay、Clair 和 mirror，使用本地镜像存储和本地数据库

单个数据中心

在多个节点上运行高度可用的 Red Hat Quay、Clair 和 mirroring，带有 HA 数据库和镜像存储

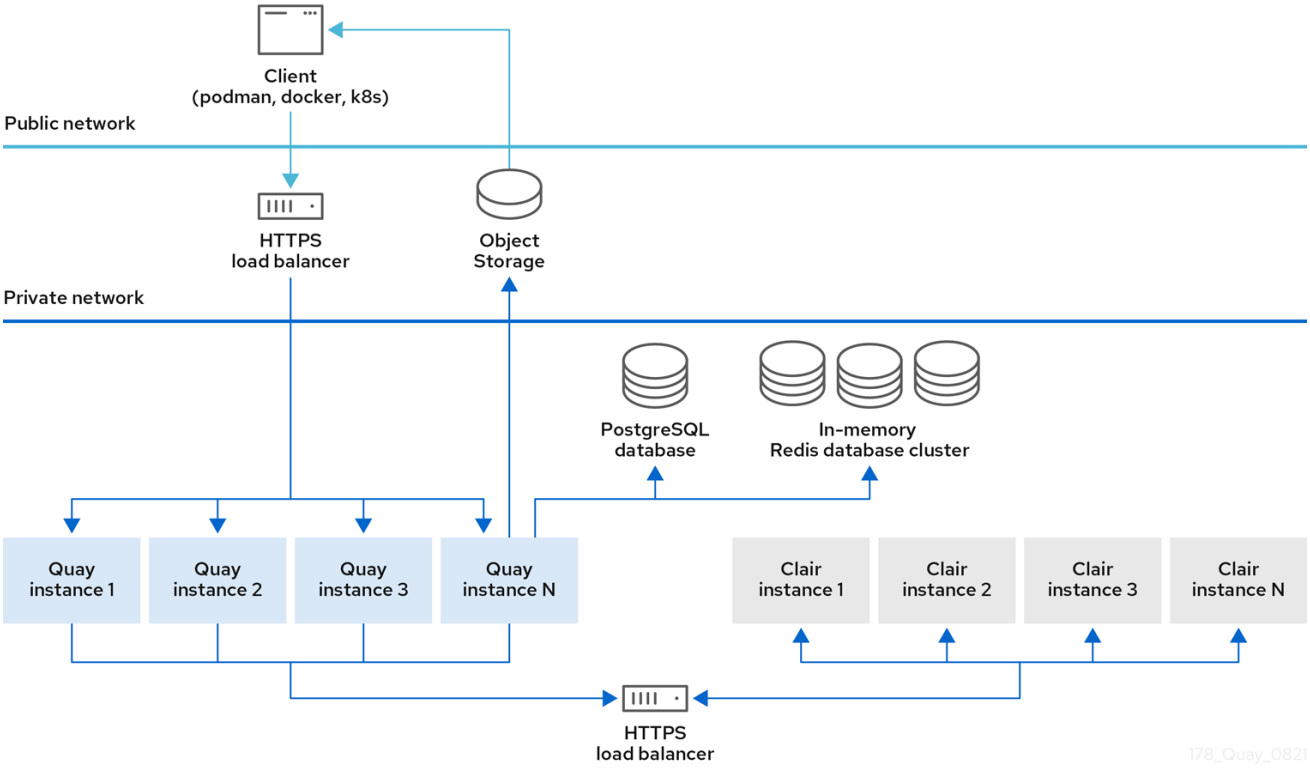
多个数据中心

在多个数据中心的多个节点上运行高度可用的 Red Hat Quay、Clair 和 mirroring，并使用 HA 数据库和镜像存储

4.2. RED HAT QUAY 部署拓扑

下图提供了 Red Hat Quay 部署拓扑的高级概述：

Red Hat Quay 部署拓扑

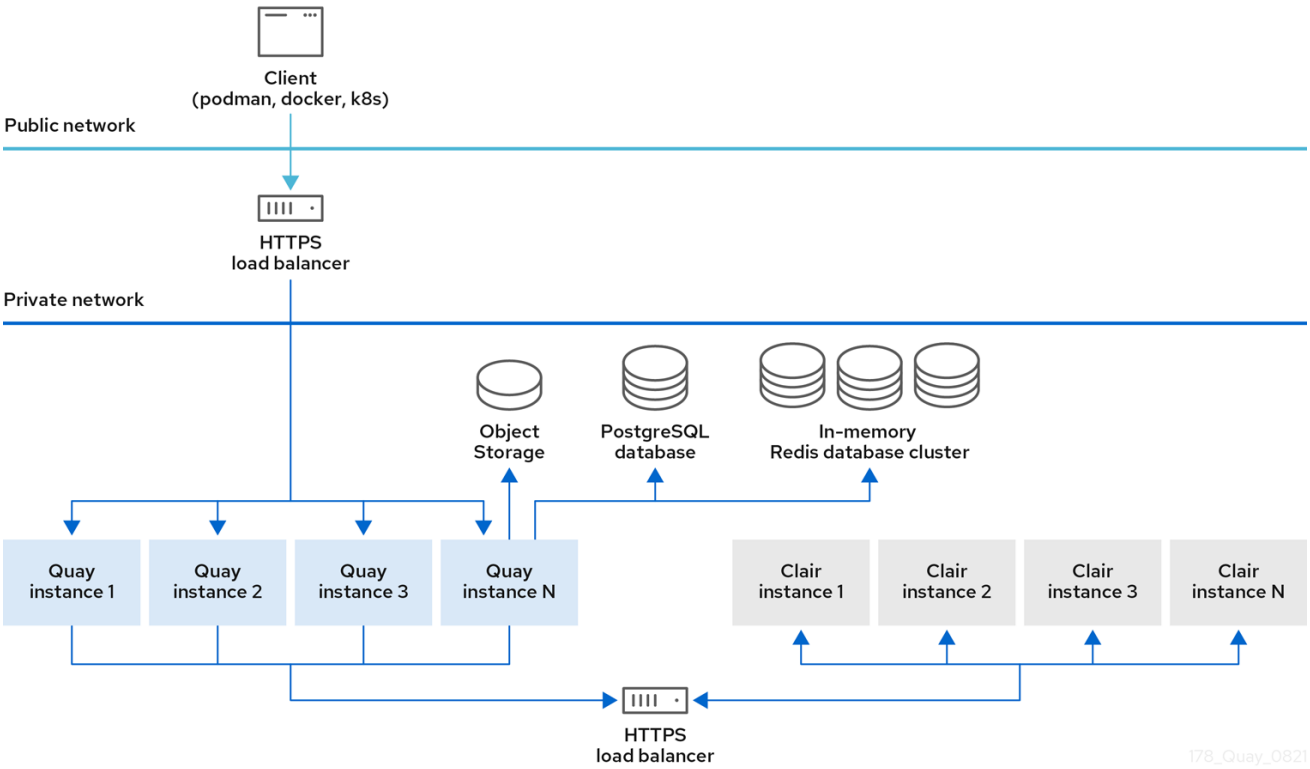


在本部署中，所有推送、用户界面和 API 请求都由公共 Red Hat Quay 端点接收。拉取直接从 **对象存储提供。**

4.3. 使用存储代理的 RED HAT QUAY 部署拓扑

下图提供了配置了存储代理的 Red Hat Quay 部署拓扑的高级概述：

使用存储代理的 Red Hat Quay 部署拓扑



配置存储代理后，所有流量都通过公共 Red Hat Quay 端点。

第 5 章 在公有云上部署 RED HAT QUAY

Red Hat Quay 可以在公有云上运行，可以是独立模式，或 OpenShift Container Platform 本身部署在公有云中。可在 Red Hat Quay **Tested Integrations Matrix** 中找到经过测试和支持的配置的完整列表，网址为 https://access.redhat.com/articles/4067991。

建议： 如果 Red Hat Quay 在公有云上运行，则您应该为 Red Hat Quay 后端服务使用公有云服务来确保正确的高可用性和可扩展性。

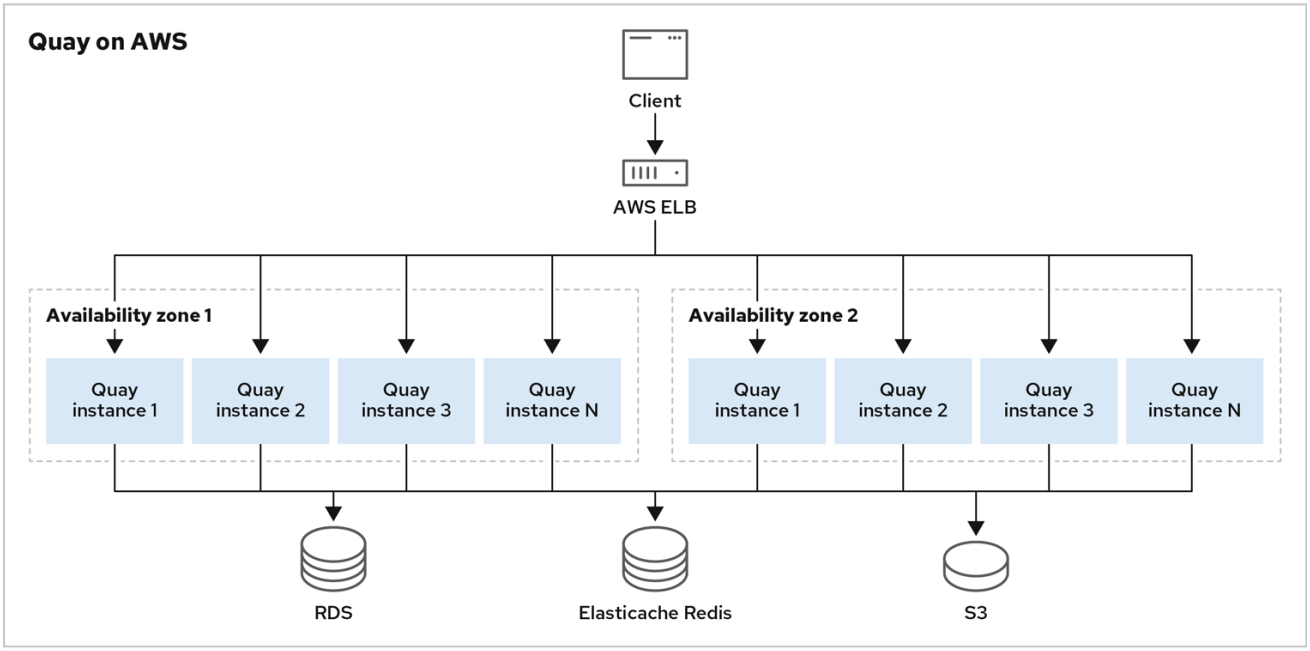
5.1. 在 AMAZON WEB SERVICES 上运行 RED HAT QUAY

如果 Red Hat Quay 在 Amazon Web Services (AWS)上运行，您可以使用以下功能：

- AWS Elastic Load Balancer
- AWS S3 (hot) blob 存储
- AWS RDS 数据库
- AWS ElastiCache Redis
- EC2 虚拟机建议：M3.Large 或 M4.XLarge

下图提供了在 AWS 上运行的 Red Hat Quay 的高级别概述：

Red Hat Quay on AWS



178_Quay_0821

5.2. 在 MICROSOFT AZURE 上运行 RED HAT QUAY

如果 Red Hat Quay 在 Microsoft Azure 上运行，您可以使用以下功能：

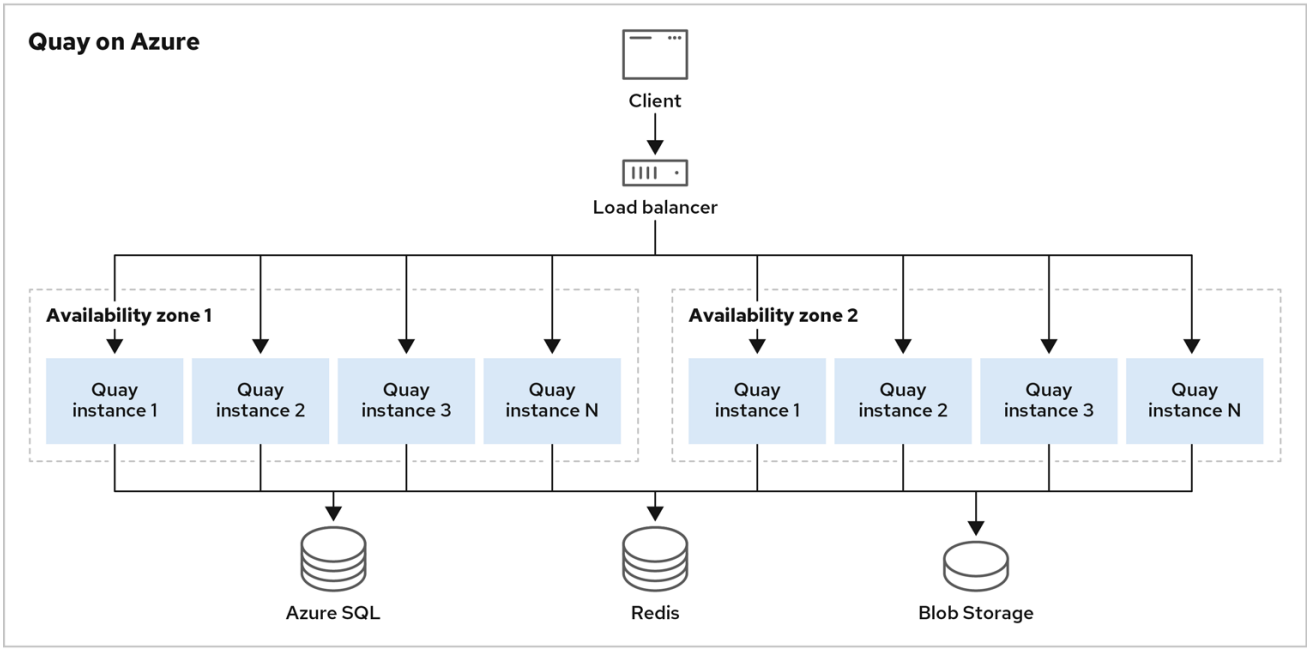
- Azure 管理的服务，如高可用性 PostgreSQL
- Azure Blob Storage 必须是热存储

- Azure cool 存储不适用于 Red Hat Quay

- Azure Cache for Redis

下图提供了在 Microsoft Azure 上运行的 Red Hat Quay 的高级别概述：

Microsoft Azure 上的 Red Hat Quay



178_Quay_0821

第 6 章 RED HAT QUAY 的内容发布

Red Hat Quay 中的内容发布功能包括：

- 存储库镜像
- geo-replication
- 在 air-gapped 环境中部署

6.1. 存储库镜像

Red Hat Quay 存储库镜像可让您将外部容器 registry 或另一个本地 registry 的镜像 mirror 到 Red Hat Quay 集群。使用存储库镜像，您可以根据存储库名称和标签将镜像同步到 Red Hat Quay。

在启用了存储库镜像的 Red Hat Quay 集群中，您可以执行以下操作：

- 从外部 registry 中选择一个仓库(mirror)
- 添加用于访问外部 registry 的凭证
- 识别要同步的特定容器镜像存储库名称和标签
- 设置同步存储库的间隔
- 检查同步的当前状态

要使用镜像功能，您需要执行以下操作：

- 在 Red Hat Quay 配置文件中启用存储库镜像
- 运行存储库镜像 worker
- 创建镜像的存储库

所有存储库镜像配置都可以使用配置工具 UI 或 Red Hat Quay API 执行。

6.1.1. 使用存储库镜像

以下列表显示了 Red Hat Quay 存储库镜像的功能和限制：

- 使用存储库镜像，您可以镜像整个存储库或有选择限制哪些镜像同步。过滤器可以基于以逗号分隔的标签列表、一系列标签或其他通过 Unix shell 样式通配符识别标签的方法。如需更多信息，请参阅 通配符 的文档。
- 当将存储库设置为镜像时，您无法手动将其他镜像添加到该存储库中。
- 由于已镜像的存储库和标签基于您设置的存储库和标签，它只会保存由存储库和标签对表示的内容。例如，如果您更改了标签，使存储库中的一些镜像不再匹配，则这些镜像将被删除。
- 只有指定的机器人可以将镜像推送到已镜像的存储库，替换存储库上设置的任何基于角色的访问控制权限。
- 镜像可以配置为失败时回滚，*或者以* 最佳方式运行。

- 使用已镜像的存储库时，*具有读取权限* 的用户可以从存储库中拉取镜像，但不能将镜像推送到存储库。
- 在镜像的存储库上更改设置，可以在 Red Hat Quay 用户界面中使用您创建的已镜像存储库的 **Repositories → Mirrors** 选项卡执行。
- 镜像以设定间隔同步，但也可以按需同步。

6.1.2. 存储库镜像建议

存储库镜像的最佳实践包括：

- 存储库镜像 pod 可以在任何节点上运行。这意味着您可以在已运行 Red Hat Quay 的节点上运行镜像功能。
- 存储库镜像在数据库中调度，并批量运行。因此，存储库 worker 会检查每个存储库镜像配置文件，并在需要下一次同步时读取。更多镜像 worker 意味着可以同时镜像更多存储库。例如，运行 10 镜像 worker 意味着用户可以并行运行 10 个镜像 Operator。如果用户只有 2 个带有 10 个镜像配置的 worker，则只能执行 2 个 operator。
- 镜像 pod 的最佳数量取决于以下条件：
  - 要镜像的存储库总数
  - 存储库中的镜像和标签数量以及更改的频率
  - 并行批处理
    例如，如果用户镜像有 100 标签的存储库，则镜像将由一个 worker 完成。用户必须考虑要并行镜像多少个仓库，并基于 worker 的数量。

    同一存储库中的多个标签无法并行镜像。

6.1.3. 镜像的事件通知

存储库镜像有三个通知事件：

- 仓库镜像已启动
- 仓库镜像成功
- 仓库 Mirror Unsuccessful

事件可以在每个存储库的 **Settings** 选项卡内配置，并且支持电子邮件、Slack、Quay UI 和 Webhook 等所有现有通知方法。

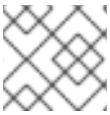
6.1.4. 镜像 API

您可以使用 Red Hat Quay API 配置存储库镜像：

镜像 API

如需更多信息，请参阅 Red Hat Quay API 指南

6.2. GEO-REPLICATION



注意

目前，IBM Power 不支持 geo-replication 功能。

地理复制允许多个地理分布的 Red Hat Quay 部署从客户端或用户的角度来看作为单个 registry 工作。它显著提高了在全局分布式 Red Hat Quay 设置中的推送和拉取性能。镜像数据在后台异步复制，并带有透明故障转移，并为客户端重定向。

在独立和 Operator 部署中支持部署带有 geo-replication 的 Red Hat Quay。

6.2.1. 地域复制功能

- 配置 geo-replication 后，容器镜像推送将写入该 Red Hat Quay 实例的首选存储引擎。这通常是区域内最接近的存储后端。
- 在初始推送后，镜像数据将在后台复制到其他存储引擎。
- 复制位置列表可以配置，它们可以是不同的存储后端。
- 镜像拉取(pull)将始终使用最接近的可用存储引擎来最大化拉取性能。
- 如果复制还没有完成，则拉取将使用源存储后端。

6.2.2. 地域复制要求和限制

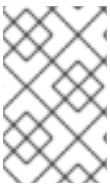
- 在地理复制设置中，Red Hat Quay 要求所有区域都可以读取和写入所有其他区域的对象存储。对象存储必须可以被所有其他区域访问。
- 如果一个地理复制站点的对象存储系统失败，该站点的 Red Hat Quay 部署必须被关闭，以便客户端由全局负载均衡器重定向到具有完整存储系统的剩余站点。否则，客户端将遇到拉取和推送失败。
- Red Hat Quay 没有内部感知连接的对象存储系统的健康状态或可用性。用户必须配置一个全局负载均衡器(LB)，以监控分布式系统的健康状况，并根据存储状态将流量路由到不同的站点。

- 要检查 geo-replication 部署的状态，您必须使用 **/health/endtoend** checkpoint，该检查点用于全局健康监控。您必须使用 **/health/endtoend** 端点手动配置重定向。**/health/instance** 端点仅检查本地实例健康状况。
- 如果一个站点的对象存储系统不可用，则其余站点或站点没有自动重定向到剩余的存储系统或系统。
- 地理复制（geo-replication）是异步的。如果一个站点永久丢失，则已存储在该站点的对象存储系统中，但在失败时还没有复制到剩余的站点的数据会丢失。
- 单个数据库（因此所有元数据和 Red Hat Quay 配置）在所有区域之间共享。
  地理复制不会复制数据库。如果出现停机，启用了地理复制功能的 Red Hat Quay 不会切换到另一个数据库。
- 单个 Redis 缓存在整个 Red Hat Quay 设置间共享，需要可以被所有 Red Hat Quay pod 访问。
- 完全相同的配置应该在所有区域间使用，但存储后端除外，该配置也可以使用 **QUAY_DISTRIBUTED_STORAGE_PREFERENCE** 环境变量明确配置。
- 地理复制需要每个区域中的对象存储。它不适用于本地存储。
- 每个区域必须能够访问每个区域中的每个存储引擎，这需要一个网络路径。
- 或者，可以使用存储代理选项。
- 整个存储后端（如所有 blob）都会被复制。相反，存储库镜像可以限制为存储库或镜像。
- 所有 Red Hat Quay 实例都必须共享相同的入口点，通常通过负载均衡器。
- 所有 Red Hat Quay 实例都必须具有相同的超级用户集合，因为它们在通用配置文件中定义。
- 地理复制要求您的 Clair 配置设置为 **unmanaged**。非受管 Clair 数据库允许 Red Hat Quay Operator 在地理复制环境中工作，其中多个 Red Hat Quay Operator 实例必须与同一数据库通信。如需更多信息，请参阅高级 Clair 配置。
- geo-Replication 需要 SSL/TLS 证书和密钥。如需更多信息，请参阅使用 SSL/TLS 保护到 Red Hat Quay 的连接。

如果无法满足上述要求，您应该使用两个不同的 Red Hat Quay 部署，并利用存储库镜像功能。

6.2.3. 使用独立 Red Hat Quay 的 geo-replication

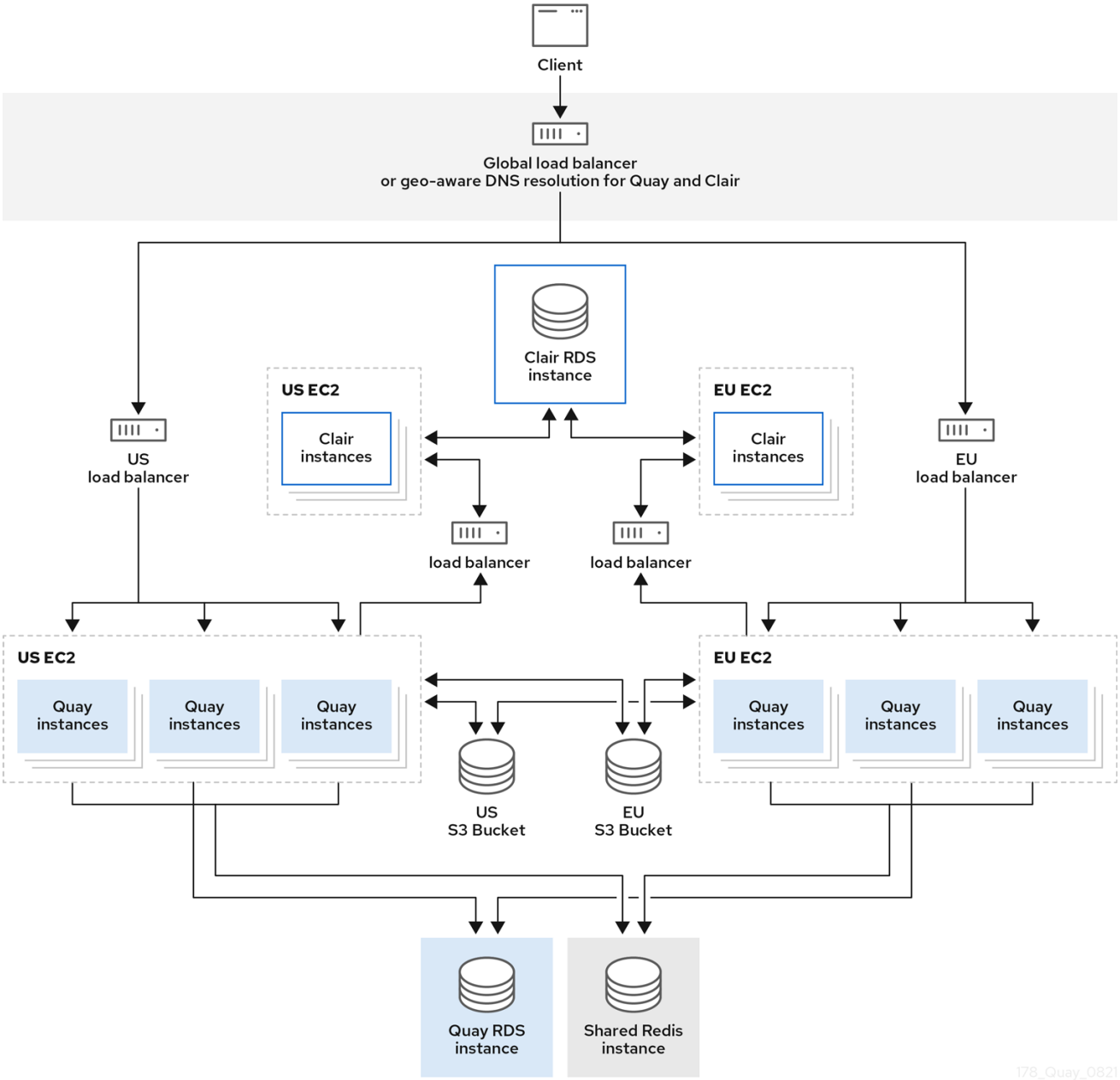
在以下镜像中，Red Hat Quay 在两个不同的区域（带有通用数据库和一个通用的 Redis 实例）中运行独立运行。本地化镜像存储在每个地区提供，镜像拉取则从最接近的可用存储引擎提供。容器镜像推送将写入 Red Hat Quay 实例的首选存储引擎，然后在后台将复制到其他存储引擎。



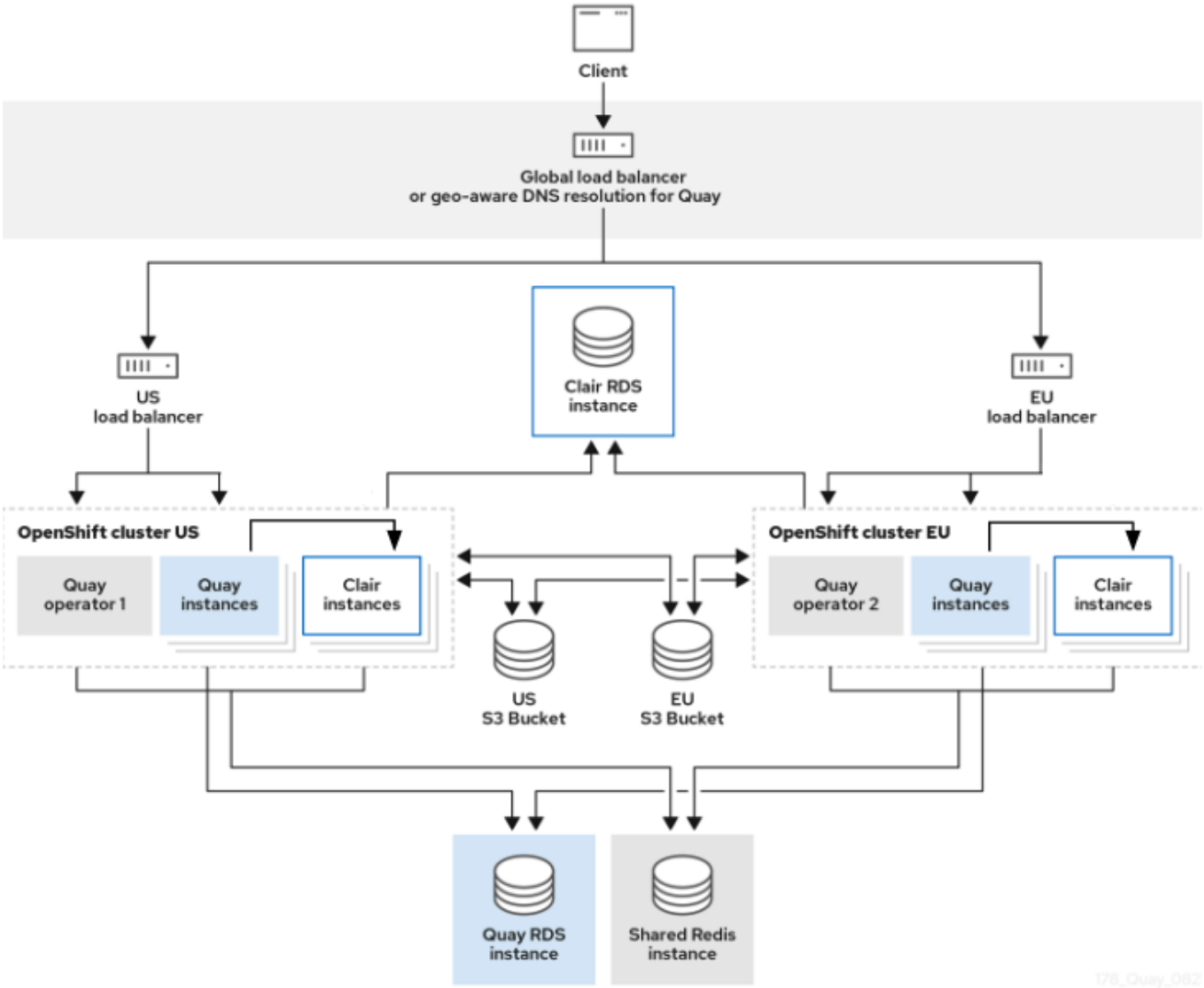
注意

如果 Clair 在一个集群中失败，例如 US 集群，则 US 用户不会在 Red Hat Quay 中为第二个集群(EU)看到漏洞报告。这是因为所有 Clair 实例都有相同的状态。当 Clair 失败时，这通常是由于集群中的问题。

地域复制架构



6.2.4. 使用 Red Hat Quay Operator 的 geo-replication



在上例中，Red Hat Quay Operator 部署到两个单独的区域，它们有一个通用数据库和一个通用的 Redis 实例。本地化镜像存储在每个地区提供，镜像拉取则从最接近的可用存储引擎提供。容器镜像推送将写入 Quay 实例的首选存储引擎，然后在后台将复制到其他存储引擎。

由于 Operator 现在单独管理 Clair 安全扫描程序及其数据库，因此可以利用异地复制设置，使其不管理 Clair 数据库。取而代之，需要使用外部共享数据库。Red Hat Quay 和 Clair 支持多个 PostgreSQL 供应商和供应商，它们可在 Red Hat Quay 3.x 测试列表中找到。另外，Operator 还支持可注入部署的自定义 Clair 配置，允许用户使用外部数据库的连接凭证配置 Clair。

6.2.5. 地域复制的混合存储

Red Hat Quay geo-replication 支持使用不同和多个复制目标，例如，在公共云中使用 AWS S3 存储并在内部使用 Ceph 存储。这与授予对所有 Red Hat Quay Pod 和集群节点中所有存储后端的访问权限的关键要求。因此，建议您使用以下内容：

- VPN 以防止内部存储可见性，*或者*
- 允许访问 Red Hat Quay 使用的指定存储桶的令牌对

这会导致 Red Hat Quay 的公共云实例可以访问内部存储，但网络会被加密、保护并将使用 ACL，从而满足安全要求。

如果您无法实现这些安全措施，则最好部署两个不同的 Red Hat Quay registry，并使用存储库镜像作为地理复制的替代选择。

6.3. 与 GEO-REPLICATION 相比的存储库镜像

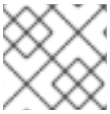
Red Hat Quay geo-replication 镜像在 2 个或更多不同存储后端间的整个镜像存储后端数据，而数据库是共享的一个 Red Hat Quay registry，它有两个不同的 blob 存储端点。地理复制的主要用例包括：

- 为地理分布的设置加快对二进制 blob 的访问
- 确保镜像内容在区域之间相同

存储库镜像将所选存储库或存储库的子集与另一个 registry 同步。registry 有所不同，每个 registry 都有单独的数据库和单独的镜像存储。

镜像的主要用例包括：

- 不同数据中心或区域中的独立 registry 部署，其中整个内容的某些子集应该在数据中心和区域间共享
- 从外部 registry 自动同步或镜像（允许列表）上游存储库到本地 Red Hat Quay 部署



注意

存储库镜像和异地复制可以同时使用。

表 6.1. Red Hat Quay 存储库镜像和异地复制比较

功能/能力	geo-replication	存储库镜像
设计的功能是什么？	共享、全局 registry	不同的 registry
如果复制或镜像还没有完成，会发生什么？	使用远程副本(slower)	未提供镜像
是否可以访问两个区域所需的所有存储后端？	是（所有 Red Hat Quay 节点）	no (distinct storage)
用户是否可以将镜像从两个站点推送到同一存储库？	是	否
所有 registry 内容和配置是否适用于所有区域（共享数据库）？	是	否
用户是否可以选择要镜像的独立命名空间或存储库？	否	是
用户能否将过滤器应用到同步规则？	否	是
每个区域是否允许独立/不同的角色访问控制配置	否	是

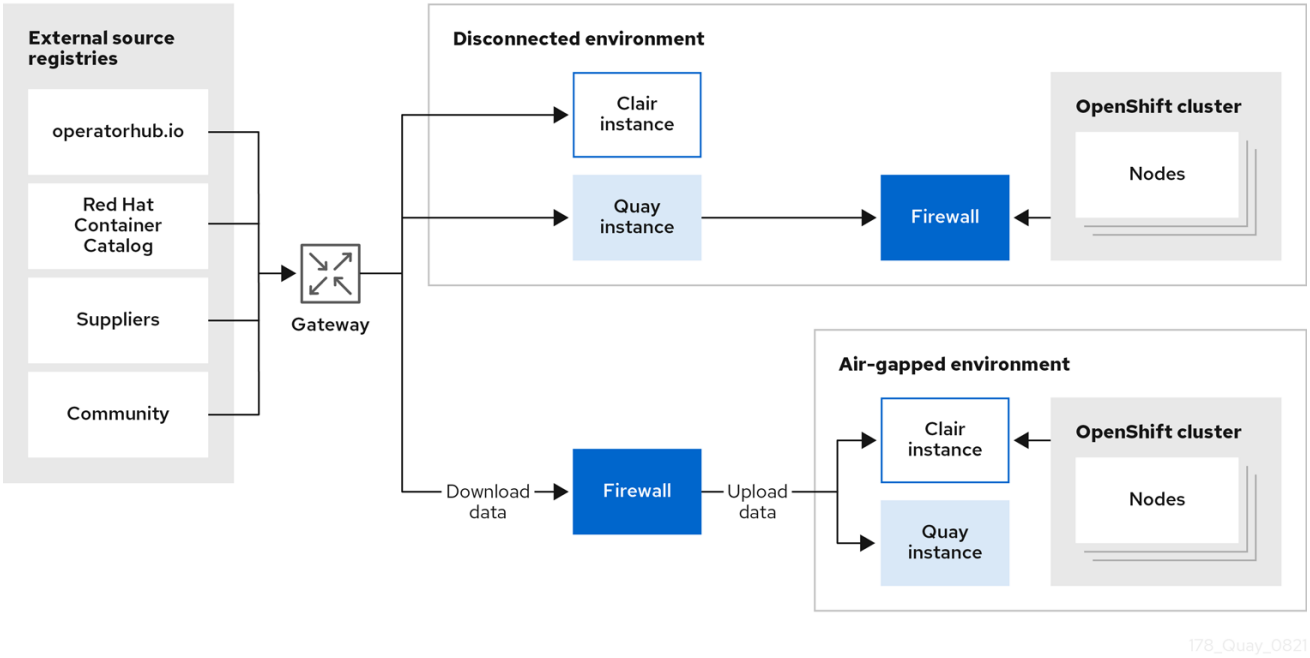
6.4. AIR-GAPPED 或断开连接的部署

在下图中，图表中的上部部署显示了连接到互联网的 Red Hat Quay 和 Clair，air-gapped OpenShift Container Platform 集群通过防火墙中的显式允许列表访问 Red Hat Quay registry。

图中的较低部署显示了在防火墙内运行的 Red Hat Quay 和 Clair，使用离线介质传输到目标系统的镜像和 CVE 数据。数据从连接到互联网的独立 Red Hat Quay 和 Clair 部署中导出。

下图显示了如何在 air-gapped 或断开连接的环境中部署 Red Hat Quay 和 Clair：

断开连接的或 air-gapped 环境中的 Red Hat Quay 和 Clair



第 7 章 RED HAT QUAY 大小和订阅

Red Hat Quay 的可扩展性是其关键优势之一，单一代码库支持广泛的部署大小，包括：

- 在单一开发机器上部署概念证明
- 大约 2,000 个用户，可以向数十个 Kubernetes 集群提供内容的中型部署
- 高端部署，如 Quay.io，它可以为全球数以千计的 Kubernetes 集群提供服务

因为大小主要取决于多个因素，如用户、镜像、并发拉取和推送的数量，因此没有标准大小建议。

以下是运行 Red Hat Quay （每个容器/pod 实例）系统的最低要求：

- **Quay:** 最小 6 GB；建议 8 GB，2 个更多 vCPU
- **Clair：** 建议 2 GB RAM 和 2 个或更多 vCPU
- **存储：** 推荐的 30 GB
- **NooBaa：** 最少 2 GB、1 个 vCPU （当 **objectstorage** 组件被 Operator 选择时）
- **Clair 数据库：** 安全元数据至少需要 5 GB

Red Hat Quay 的无状态组件可以扩展，但这会导致有状态后端服务的负载更高。

7.1. RED HAT QUAY 示例大小

下表显示了概念证明、中型和高端部署的大约大小。同一指标是否适当地运行部署取决于以下多种因素。

指标	概念验证	mid-size	High End (Quay.io)
否。默认情况下 Quay 容器	1	4	15
No. Quay 容器在横向扩展时最大数	N/A	8	30
否。默认情况下 Clair 容器	1	3	10
no. Clair 容器在横向扩展时最大	N/A	6	15
否。镜像 pod （镜像 100 个软件仓库）	1	5-10	N/A
数据库大小	2 -4 个内核 6-8 GB RAM 10-20 GB 磁盘	4-8 个内核 6-32 GB RAM 100 GB - 1 TB 磁盘	32 个内核 244 GB 1+ TB 磁盘
对象存储后端大小	10-100 GB	1 - 20 TB	50+ TB，最多 PB
Redis 缓存大小		2 个内核 2-4 GB RAM	4 个内核 28 GB RAM

指标	概念验证	mid-size	High End (Quay.io)
底层节点大小（物理或虚拟）	4 个内核 8 GB RAM	4-6 个内核 12-16 GB RAM	Quay: 13 个内核 56GB RAM Clair: 2 个内核 4 GB RAM

有关调整镜像大小和相关建议的详情，请参阅有关 存储库镜像的部分。

只有当您使用 Quay 构建器时，Red Hat Redis 缓存的大小才相关，否则这并不重要。

7.2. RED HAT QUAY 订阅信息

Red Hat Quay 具有标准（Standard）和高级（Premium）支持，订阅则基于部署。

注意

Deployment 意味着使用共享数据后端安装单个 Red Hat Quay registry。

使用 Red Hat Quay 订阅时，可用选项：

- 对于可以部署的 pod 数量，如 Quay、Clair、Builder 等 pod 没有限制。
- Red Hat Quay pod 可以在多个数据中心或可用区中运行。
- 存储和数据库后端可以部署到多个数据中心或可用区，但只能作为单一的共享存储后端和单一的共享数据库后端。
- Red Hat Quay 可以为无限数量的集群或单机服务器管理内容。
- 客户端无论其物理位置如何，都可以访问 Red Hat Quay 部署。
- 您可以在 OpenShift Container Platform 基础架构节点上部署 Red Hat Quay，以最大程度降低订阅要求。
- 您可以在 OpenShift Container Platform 集群上运行 Container Security Operator (CSO)和 Quay Bridge Operator (QBO)，而无需额外成本。

注意

Red Hat Quay geo-replication 需要为每个存储复制订阅。但是，数据库是共享的。

有关购买 Red Hat Quay 订阅的更多信息，请参阅 Red Hat Quay。

7.3. 使用带有内部 REGISTRY 或没有内部 REGISTRY 的 RED HAT QUAY

Red Hat Quay 可以在多个 OpenShift Container Platform 集群前使用其内部 registry 作为外部 registry。

在进行构建和部署推出时，Red Hat Quay 也可以用于代替内部 registry。**Secret** 和 **ImageStreams** 所需的协调由 Quay Bridge Operator 自动进行，该 Operator 可以从 OperatorHub 为 OpenShift Container Platform 启动。

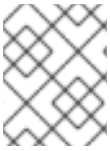
第 8 章 配额管理架构

启用配额管理功能后，单独的 blob 大小在存储库和命名空间级别总和。例如，如果同一存储库中的两个标签引用同一 blob，则该 Blob 的大小仅计算为仓库总计一次。另外，清单列表总数计算为存储库总计。

重要

因为清单列表总数被计算为仓库总计，所以从以前的 Red Hat Quay 版本升级时消耗的总配额可能会在 Red Hat Quay 3.9 中被报告不同。在某些情况下，新总数可能会超过存储库的之前设置的限制。Red Hat Quay 管理员可能需要调整存储库所分配的配额，以考虑这些更改。

配额管理功能的工作原理是，使用回填 worker 计算现有存储库和命名空间的大小，然后从总镜像添加或减去，这些镜像被推送或垃圾收集到每个镜像。另外，在收集清单时，从总的减法中减去。



注意

因为减法发生在清单垃圾回收时的总数，所以大小计算会有一个延迟，直到它能够收集垃圾回收为止。有关垃圾回收的更多信息，请参阅 Red Hat Quay 垃圾回收。

以下数据库表包含机构中 Red Hat Quay 仓库的配额存储库大小、配额命名空间大小和配额 registry 大小（以字节为单位）：

- **QuotaRepositorySize**
- **QuotaNameSpaceSize**
- **QuotaRegistrySize**

组织大小由回填 worker 计算，以确保不会重复。初始化镜像推送时，会验证用户的机构存储来检查它是否超出配置的配额限制。如果镜像推送超过定义的配额限制，则会出现软或硬检查：

- 对于软检查，用户会收到通知。
- 对于硬检查，推送将停止。

如果存储消耗在配置的配额限制内，则允许推送进行。

镜像清单删除遵循类似的流程，其中关联的镜像标签和清单之间的链接会被删除。另外，在镜像清单被删除后，会在 **QuotaRepositorySize**、**QuotaNameSpaceSize** 和 **QuotaRegistrySize** 表中重新计算和更新存储库大小。

第 9 章 命名空间自动运行架构

对于命名空间 auto-pruning 功能，创建数据库模式中的两个不同的数据库表：一个用于 **namespaceautoprunepolicy**，另一个用于 **autoprunetaskstatus**。自动修剪 worker 会执行配置策略。

命名空间自动修剪策略数据库表

namespaceautoprunepolicy 数据库表包含单个命名空间的策略配置。每个命名空间只有一个条目，但每个 **namespace_id** 支持多个行。**policy** 字段包含策略详情，如 **{method: "creation_date", olderThan: "2w"}** 或 **{method: "number_of_tags", numTags: 100}**。

表 9.1. namespaceautoprunepolicy 数据库表

字段	类型	属性	描述
uuid	character varying (225)	unique, indexed	此策略的唯一标识符
namespace_id	整数	外键	策略属于的命名空间
policy	text	JSON	策略配置

自动修剪任务状态数据库表

autoprunetaskstatus 表注册由自动修剪 worker 执行的任务。任务在单个命名空间上下文中执行。每个命名空间只有一个任务。

表 9.2. autoprunetaskstatus 数据库表

字段	类型	属性	描述
namespace_id	整数	外键	此任务所属的命名空间
last_ran_ms	大整数(bigint)	nullable, indexed	worker 为此命名空间执行策略最后一次的时间
status	text	nullable	最后一次执行任务的详情

9.1. 自动修剪 WORKER

以下小节详细介绍了自动修剪 worker 的信息。

9.1.1. Auto-prune-task-creation

在 **namespaceautoprunepolicy** 数据库表中创建新策略时，也会在 **autoprunetask** 表中创建一个行。这在同一事务中完成。auto-prune worker 使用 **autoprunetask** 表中的条目来识别它应该执行策略的命名空间。

9.1.2. 自动修剪 worker 执行

auto-pruning worker 是一个执行配置策略的异步作业。其工作流基于 **autoprunetask** 表中的值。当任务启动时，会出现以下情况：

- 自动修剪 worker 以集合间隔启动，默认值为 30 秒。
- auto-prune worker 从 **autoprunetask** 中选择一个行，其中至少为 null，**last_ran_ms** 和 **FOR UPDATE SKIP LOCKED**。
  - null **last_ran_ms** 表示任务从未运行。
  - 尚未运行任务的时间最长，或者尚未完全运行，则优先选择。
- auto-prune worker 从 **namespaceautoprunepolicy** 表中获取策略配置。
  - 如果没有策略配置，则会为这个命名空间删除 **autoprunetask** 的条目，这个过程会立即停止。
- 自动修剪 worker 开始机构下所有存储库的分页循环。
  - 自动修剪工作程序根据 **policy.method** 确定要使用的大量修剪方法。
- auto-prune worker 使用前面检索到的策略配置来执行修剪方法。
  - 使用标签数量进行修剪： auto-pruner worker 获取当前按创建日期排序的活动标签数量，并将旧的标签删除到配置的数量。
  - 要按日期进行修剪： auto-pruner worker 获取超过指定时间范围的活跃标签，返回的任何标签都会被删除。
- auto-prune worker 添加已删除标签的审计日志。
- 选择 **autoprunetask** 行后，**last_ran_ms** 会被更新。
- 自动修剪 worker 结束。