Red Hat Quay 3

部署 Red Hat Quay - 高可用性

部署 Red Hat Quay HA

Last Updated: 2024-07-31

Red Hat Quay 3 部署 Red Hat Quay - 高可用性

部署 Red Hat Quay HA

法律通告

摘要

在 HA 环境中部署 Red Hat Quay

目录

前言

Red Hat Quay 是一个企业级容器 registry。使用 Quay 构建和存储容器，然后将其部署到整个企业中的服务器。

此流程描述了如何部署高可用性的 Red Hat Quay 设置。

第 1 章 RED HAT QUAY 功能

Red Hat Quay 会定期发布新的功能和软件更新。以下功能适用于 Red Hat Quay 部署，但列表并不完整：

- 高可用性
- geo-replication
- 仓库镜像
- Docker v2、模式 2 （多架构）支持
- 持续集成
- 使用 Clair 进行安全扫描
- 自定义日志轮转
- 零停机时间垃圾回收
- 24/7 支持

用户应检查 Red Hat Quay 发行注记 以获取最新的功能信息。

第 2 章 RED HAT QUAY 支持

Red Hat Quay 提供以下支持：

- 多个身份验证和访问方法
- 多个存储后端
- **Quay**、**Clair** 和存储后端容器的自定义证书
- 应用程序 registry
- 不同的容器镜像类型

2.1. 架构

Red Hat Quay 包括几个核心组件，包括内部和外部。

有关完整的架构分类，请参阅 Red Hat Quay 架构指南。

2.1.1. 内部组件

Red Hat Quay 包括以下内部组件：

- **Quay （容器注册表）**将 **Quay** 容器作为服务运行，由 pod 中的多个组件组成。
- **Clair**.扫描容器镜像中的漏洞并推荐修复。

2.1.2. 外部组件

Red Hat Quay 包括以下外部组件：

- **数据库**.Red Hat Quay 用作其主要元数据存储。请注意，这不是镜像存储。
- **redis （键-值存储）**存储实时构建器日志和 Red Hat Quay 指南。还包括垃圾回收所需的锁定机制。
- **云存储**。对于支持的部署，必须使用以下存储类型之一：
  - **公有云存储**.在公有云环境中，您应该使用云供应商的对象存储，如 Amazon Web Services 的 Amazon S3 或 Google Cloud 的 Google Cloud Storage。
  - **私有云存储**.在私有云中，需要 S3 或 Swift 兼容对象存储，如 Ceph RADOS 或 OpenStack Swift。

警告

不要将"本地挂载的目录"存储引擎用于任何生产配置。不支持挂载的 NFS 卷。本地存储用于 Red Hat Quay 测试安装。

第 3 章 准备 RED HAT QUAY （高可用性）

注意

此流程详细介绍了如何设置 Red Hat Quay 的高可用性、生产级部署的指导。

3.1. 先决条件

在开始 Red Hat Quay 高可用性部署前，您需要了解以下内容：

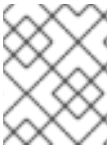
- Postgres 或 MySQL 可用于提供数据库服务。Postgres 被在此处选择为数据库，因为它包含了支持 Clair 安全扫描所需的功能。其他选项包括：
  - Crunchy Data PostgreSQL Operator：虽然不受红帽直接支持，但 CrunchDB Operator 可从 Crunchy Data for Red Hat Quay 提供。如果您有此路由，您应该具有 Crunchy Data 的支持合同，并直接与它们合作，以便与 Operator 及其数据库相关的使用指导或问题。
  - 如果您的组织已具有高可用性(HA)数据库，您可以将该数据库用于 Red Hat Quay。如需有关支持第三方数据库和其他组件的详细信息，请参阅 Red Hat Quay 支持政策。
- Ceph 对象网关（也称为 RADOS 网关）是产品的一个示例，可提供 Red Hat Quay 所需的对象存储。如果您希望 Red Hat Quay 设置进行 geo-replication，则需要 Ceph 对象网关或其他支持的对象存储。对于云安装，您可以使用以下任何云对象存储：
  - Amazon S3（请参阅 S3 IAM Bucket Policy ）以了解有关为 Quay 配置 S3 存储桶策略的详情。）
  - Azure Blob Storage
  - Google Cloud Storage
  - Ceph 对象网关
  - OpenStack Swift
  - CloudFront + S3
  - NooBaa S3 Storage
- 本例中使用 haproxy 服务器，但您可以使用适用于您的环境的任何代理服务。
- 系统数量：此步骤使用在以下任务分配的七种系统（物理或虚拟）：
  - **答：db01: 负载均衡器和数据库** ：运行 haproxy 负载均衡器和 Postgres 数据库。请注意，这些组件本身并不具有高可用性，但用于指明如何设置自己的负载均衡器或生产数据库。
  - **B: quay01, quay02, quay03: Quay 和 Redis**：分配三个（或更多）系统来运行 Quay 和 Redis 服务。
  - **C: ceph01, ceph02, ceph03, ceph04, ceph05: Ceph** Three （或更多）系统为存储提供 Ceph 服务。如果要部署到云，您可以使用前面描述的云存储功能。此流程将额外的系统用于 Ansible(ceph05)，一个用于 Ceph 对象网关(ceph04)。

每个系统都应该有以下属性：

- **Red Hat Enterprise Linux (RHEL)**8：获取 下载页面中 的最新 Red Hat Enterprise Linux 8 服务器介质，并按照 Red Hat Enterprise Linux 9 产品文档中的安装说明进行操作。
  - **有效的红帽订阅：配置** 有效的 Red Hat Enterprise Linux 8 服务器订阅。
  - **CPU** ：两个或更多虚拟 CPU
  - **RAM** ：每个 A 和 B 系统的 4GB；每个 C 系统有 8GB
  - **磁盘空间** ：为每个 A 和 B 系统大约 20GB 磁盘空间（10GB 用于操作系统，10GB 用于 docker 存储）。至少 30GB 磁盘空间用于 C 系统（或取决于所需的容器存储）。

3.2. 使用 PODMAN

本文档使用 podman 来创建和部署容器。如果您的系统中没有 podman，则应该可以使用等同的 docker 命令。有关 podman 及相关技术的更多信息，请参阅在 Red Hat Enterprise Linux 8 上构建、运行和管理 Linux 容器。



注意

对于高可用性、生产环境质量部署 Red Hat Quay，我们强烈建议使用 Podman。Docker 尚未使用 Red Hat Quay 3 测试，并将在以后的发行版本中删除。

3.3. 设置 HAPROXY 负载均衡器和 POSTGRESQL 数据库

使用以下步骤设置 HAProxy 负载均衡器和 PostgreSQL 数据库。

先决条件

- 已安装 Podman 或 Docker CLI。

流程

1. 在前两个系统(**q01** 和 **q02**)上，安装 HAProxy 负载均衡器和 PostgreSQL 数据库。这会将 HAProxy 配置为在不同系统上运行的以下服务的接入点和负载均衡器：
   - Red Hat Quay (B 系统中的端口 80 和 443)
   - Redis（B 个系统中的端口 6379）
   - RADOS（C 系统上端口 7480）
1. 在 SELinux 中打开所有 HAProxy 端口，并在防火墙中打开所选 HAProxy 端口：

```
# setsebool -P haproxy_connect_any=on
# firewall-cmd --permanent --zone=public --add-port=6379/tcp --add-port=7480/tcp
success
# firewall-cmd --reload
success
```

1. 将 **/etc/haproxy/haproxy.cfg** 配置为指向提供 Red Hat Quay、Red Hat Quay、Redis 和 Ceph RADOS 服务的系统和端口。以下是默认值和添加了 frontend 和 backend 设置的示例：

```
#---------------------------------------------------------------------
# common defaults that all the 'listen' and 'backend' sections will
```

```
# use if not designated in their block
#---------------------------------------------------------------------
defaults
    mode                    tcp
    log                     global
    option                  httplog
    option                  dontlognull
    option http-server-close
    option forwardfor       except 127.0.0.0/8
    option                  redispatch
    retries                 3
    timeout http-request    10s
    timeout queue           1m
    timeout connect         10s
    timeout client          1m
    timeout server          1m
    timeout http-keep-alive 10s
    timeout check           10s
    maxconn                 3000

#---------------------------------------------------------------------
# main frontend which proxys to the backends
#---------------------------------------------------------------------

frontend  fe_http *:80
    default_backend             be_http
frontend  fe_https *:443
    default_backend             be_https
frontend fe_redis *:6379
   default_backend              be_redis
frontend  fe_rdgw *:7480
    default_backend             be_rdgw
backend be_http
    balance     roundrobin
    server quay01 quay01:80 check
    server quay02 quay02:80 check
    server quay03 quay03:80 check
backend be_https
    balance     roundrobin
    server quay01 quay01:443 check
    server quay02 quay02:443 check
    server quay03 quay03:443 check
backend be_rdgw
    balance     roundrobin
    server ceph01 ceph01:7480 check
    server ceph02 ceph02:7480 check
    server ceph03 ceph03:7480 check
backend be_redis
server quay01 quay01:6379 check inter 1s
server quay02 quay02:6379 check inter 1s
server quay03 quay03:6379 check inter 1s
```

在新的 **haproxy.cfg** 文件就位后，输入以下命令重启 HAProxy 服务：

```
# systemctl restart haproxy
```

2. 输入以下命令为 PostgreSQL 数据库创建一个文件夹：

```
$ mkdir -p /var/lib/pgsql/data
```

3. 为 **/var/lib/pgsql/data** 文件夹设置以下权限：

```
$ chmod 777 /var/lib/pgsql/data
```

4. 输入以下命令启动 PostgreSQL 数据库：

```
$ sudo podman run -d --name postgresql_database \
  -v /var/lib/pgsql/data:/var/lib/pgsql/data:Z  \
  -e POSTGRESQL_USER=quayuser -e POSTGRESQL_PASSWORD=quaypass \
  -e POSTGRESQL_DATABASE=quaydb -p 5432:5432 \
  registry.redhat.io/rhel8/postgresql-13:1-109
```

注意

容器的数据将保存在 **/var/lib/pgsql/data** 目录中的主机系统中。

5. 输入以下命令列出可用的扩展：

```
$ sudo podman exec -it postgresql_database /bin/bash -c 'echo "SELECT * FROM
pg_available_extensions" | /opt/rh/rh-postgresql96/root/usr/bin/psql'
```

输出示例

```
   name    | default_version | installed_version |          comment
-----------+-----------------+-------------------+---------------------------------------
 adminpack | 1.0             |                   | administrative functions for PostgreSQL
...
```

6. 输入以下命令来创建 **pg_trgm** 扩展：

```
$ sudo podman exec -it postgresql_database /bin/bash -c 'echo "CREATE EXTENSION IF
NOT EXISTS pg_trgm;" | /opt/rh/rh-postgresql96/root/usr/bin/psql -d quaydb'
```

7. 输入以下命令确认 **pg_trgm** 已创建：

```
$ sudo podman exec -it postgresql_database /bin/bash -c 'echo "SELECT * FROM
pg_extension" | /opt/rh/rh-postgresql96/root/usr/bin/psql'
```

输出示例

```
 extname | extowner | extnamespace | extrelocatable | extversion | extconfig | extcondition
---------+----------+--------------+----------------+------------+-----------+--------------
 plpgsql |       10 |           11 | f              | 1.0        |           |
 pg_trgm |       10 |         2200 | t              | 1.3        |           |
(2 rows)
```

8. 更改 Postgres 用户 **quayuser** 的权限，并 **授予其超级用户** 角色，授予用户对数据库的不受限制的访问权限：

```
$ sudo podman exec -it postgresql_database /bin/bash -c 'echo "ALTER USER quayuser
WITH SUPERUSER;" | /opt/rh/rh-postgresql96/root/usr/bin/psql'
```

输出示例

```
ALTER ROLE
```

9. 如果您的系统中活跃了 firewalld 服务，请运行以下命令使 PostgreSQL 端口通过防火墙可用：

```
# firewall-cmd --permanent --zone=trusted --add-port=5432/tcp
```

```
# firewall-cmd --reload
```

10. 可选。如果您没有安装 **postgres** CLI 软件包，请输入以下命令安装它：

```
# yum install postgresql -y
```

11. 使用 **psql** 命令测试与 PostgreSQL 数据库的连接。

注意

要验证您可以远程访问该服务，请在远程系统中运行以下命令：

```
# psql -h localhost quaydb quayuser
```

输出示例

```
Password for user test:
psql (9.2.23, server 9.6.5)
WARNING: psql version 9.2, server version 9.6.
         Some psql features might not work.
Type "help" for help.

test=> \q
```

3.4. 设置 CEPH

对于此 Red Hat Quay 配置，我们创建一个三节点 Ceph 集群，它们带有几个其他支持节点，如下所示：

- ceph01、ceph02 和 ceph03 - Ceph Monitor、Ceph Manager 和 Ceph OSD 节点
- ceph04 - Ceph RGW 节点
- ceph05 - Ceph Ansible 管理节点

有关安装 Ceph 节点的详情，请参考在 Red Hat Enterprise Linux 上安装 Red Hat Ceph Storage 。

设置 Ceph 存储集群后，创建 Ceph 对象网关（也称为 RADOS 网关）。详情请参阅 安装 Ceph 对象网关。

3.4.1. 安装每个 Ceph 节点

在 ceph01、ceph02、ceph03、ceph04 和 ceph05 上，执行以下操作：

1. 查看根据 安装 Red Hat Ceph Storage 的要求设置 Ceph 节点的先决条件。特别是：
   - 决定是否要在 OSD 节点上使用 RAID 控制器。
   - 确定是否要将单独的集群网络用于您的 Ceph 网络配置。
2. 准备 OSD 存储（仅限ceph01、ceph02 和 ceph03）。在三个 OSD 节点上（ceph01、ceph02 和 ceph03）设置 OSD 存储。有关以后您将进入 Ansible 配置的受支持存储类型的详细信息，请参阅 table 3.2 中的 OSD Ansible 设置。在本例中，每个 OSD 节点上配置了一个未格式化的块设备（**/dev/sdb**），与操作系统分开。如果您要在裸机上安装，您可能需要向机器添加额外的硬盘以实现这一目的。
3. 如 RHEL 7 安装指南 中所述，安装 Red Hat Enterprise Linux Server 版本。
4. 注册并订阅每个 Ceph 节点，如 注册 Red Hat Ceph Storage 节点 中所述。以下是如何订阅所需仓库：

```
# subscription-manager repos --disable=*
# subscription-manager repos --enable=rhel-7-server-rpms
# subscription-manager repos --enable=rhel-7-server-extras-rpms
# subscription-manager repos --enable=rhel-7-server-rhceph-3-mon-rpms
# subscription-manager repos --enable=rhel-7-server-rhceph-3-osd-rpms
# subscription-manager repos --enable=rhel-7-server-rhceph-3-tools-rpms
```

5. 在每个节点上创建一个具有 root 特权的 ansible 用户。选择您要访问的任何名称。例如：

```
# USER_NAME=ansibleadmin
# useradd $USER_NAME -c "Ansible administrator"
# passwd $USER_NAME
New password: *********
Retype new password: *********
# cat << EOF >/etc/sudoers.d/admin
admin ALL = (root) NOPASSWD:ALL
EOF
# chmod 0440 /etc/sudoers.d/$USER_NAME
```

3.4.2. 配置 Ceph Ansible 节点(ceph05)

登录 Ceph Ansible 节点(ceph05)，再按如下所示配置它：您将需要 ceph01、ceph02 和 ceph03 节点才能运行，以完成这些步骤。

1. 在 Ansible 用户的主目录中，创建一个目录来存储从 ceph-ansible playbook 创建的临时值

```
# USER_NAME=ansibleadmin
# sudo su - $USER_NAME
[ansibleadmin@ceph05 ~]$ mkdir ~/ceph-ansible-keys
```

2. 为 ansible 用户启用无密码 ssh。在 ceph05 上运行 ssh-keygen(leave passphrase empty)，然后重复 ssh-copy-id 将公钥复制到 ceph01、ceph02 和 ceph03 系统上的 Ansible 用户：

```
# USER_NAME=ansibleadmin
# sudo su - $USER_NAME
[ansibleadmin@ceph05 ~]$ ssh-keygen
[ansibleadmin@ceph05 ~]$ ssh-copy-id $USER_NAME@ceph01
[ansibleadmin@ceph05 ~]$ ssh-copy-id $USER_NAME@ceph02
[ansibleadmin@ceph05 ~]$ ssh-copy-id $USER_NAME@ceph03
[ansibleadmin@ceph05 ~]$ exit
#
```

3. 安装 ceph-ansible 软件包：

```
# yum install ceph-ansible
```

4. 在这两个目录间创建一个符号链接：

```
# ln -s /usr/share/ceph-ansible/group_vars \
    /etc/ansible/group_vars
```

5. 创建要修改的 Ceph 示例 yml 文件的副本：

```
# cd /usr/share/ceph-ansible
# cp group_vars/all.yml.sample group_vars/all.yml
# cp group_vars/osds.yml.sample group_vars/osds.yml
# cp site.yml.sample site.yml
```

6. 编辑复制的 group_vars/all.yml 文件。详情请参阅 General Ansible 设置（ 表 3.1 ）。例如：

```
ceph_origin: repository
ceph_repository: rhcs
ceph_repository_type: cdn
ceph_rhcs_version: 3
monitor_interface: eth0
public_network: 192.168.122.0/24
```

请注意，您的网络设备和地址范围可能有所不同。

7. 编辑复制的 **group_vars/osds.yml** 文件。详情请参阅 OSD Ansible 设置（ 表 3.2 ）。在本例中，每个 OSD 节点上的第二个磁盘设备(**/dev/sdb**)用于数据和日志存储：

```
osd_scenario: collocated
devices:
  - /dev/sdb
dmcrypt: true
osd_auto_discovery: false
```

8. 编辑 **/etc/ansible/hosts** 清单文件，以识别 Ceph 监控、OSD 和管理器节点中的 Ceph 节点。在本例中，每个节点上也会识别存储设备：

```
[mons]
ceph01
ceph02
```

```
ceph03

[osds]
ceph01 devices="[ '/dev/sdb' ]"
ceph02 devices="[ '/dev/sdb' ]"
ceph03 devices="[ '/dev/sdb' ]"

[mgrs]
ceph01 devices="[ '/dev/sdb' ]"
ceph02 devices="[ '/dev/sdb' ]"
ceph03 devices="[ '/dev/sdb' ]"
```

9. 将此行添加到 **/etc/ansible/ansible.cfg** 文件中，将每个 Ansible playbook 中的输出保存到 Ansible 用户的主目录中：

```
retry_files_save_path = ~/
```

10. 检查 Ansible 是否可以访问您配置为 Ansible 用户的所有 Ceph 节点：

```
# USER_NAME=ansibleadmin
# sudo su - $USER_NAME
[ansibleadmin@ceph05 ~]$ ansible all -m ping
ceph01 | SUCCESS => {
    "changed": false,
    "ping": "pong"
}
ceph02 | SUCCESS => {
    "changed": false,
    "ping": "pong"
}
ceph03 | SUCCESS => {
    "changed": false,
    "ping": "pong"
}
[ansibleadmin@ceph05 ~]$
```

11. 运行 ceph-ansible playbook（作为 Ansible 用户）：

```
[ansibleadmin@ceph05 ~]$ cd /usr/share/ceph-ansible/
[ansibleadmin@ceph05 ~]$ ansible-playbook site.yml
```

此时，Ansible playbook 将检查您的 Ceph 节点，并为您请求的服务配置它们。如果有任何失败，请进行更正并重新运行该命令。

12. 登录到三个 Ceph 节点中的一个（ceph01、ceph02 或 ceph03），并检查 Ceph 集群的健康状态：

```
# ceph health
HEALTH_OK
```

13. 在同一节点上，使用 rados 验证监控是否正常工作：

```
# ceph osd pool create test 8
# echo 'Hello World!' > hello-world.txt
```

```
# rados --pool test put hello-world hello-world.txt
# rados --pool test get hello-world fetch.txt
# cat fetch.txt
Hello World!
```

3.4.3. 安装 Ceph 对象网关

在 Ansible 系统上，将 Ceph 对象网关配置为 Ceph Storage 集群（最终会在 ceph04 上运行）。详情请参阅 安装 Ceph 对象网关。

3.5. 设置 REDIS

在三个 Red Hat Quay 系统(quay01、quay02 和 quay03)上安装 Red Hat Enterprise Linux 8 服务器后，安装并启动 Redis 服务，如下所示：

1. **安装 / DeployRedis** ：将 Redis 作为容器在三个 quay0 the 系统上安装：

```
# mkdir -p /var/lib/redis
# chmod 777 /var/lib/redis
# sudo podman run -d -p 6379:6379 \
    -v /var/lib/redis:/var/lib/redis/data:Z \
    registry.redhat.io/rhel8/redis-5
```

2. 检查 **redis connectivity** ：您可以使用 **telnet** 命令测试到 redis 服务的连接。键入 MONITOR（以开始监控服务）和 QUIT 退出：

```
# yum install telnet -y
# telnet 192.168.122.99 6379
Trying 192.168.122.99...
Connected to 192.168.122.99.
Escape character is '^]'.
MONITOR
+OK
+1525703165.754099 [0 172.17.0.1:43848] "PING"
QUIT
+OK
Connection closed by foreign host.
```

注意

有关使用 **podman** 和重启容器的更多信息，请参阅本文档前面"使用 podman"部分。

第 4 章 配置 RED HAT QUAY

在将 Red Hat Quay 服务作为容器运行前，您需要使用相同的 **Quay** 容器来创建部署 Red Hat Quay 所需的配置文件(**config.yaml**)。为此，您可以将配置参数和密码（这里替换 my-secret-password）传递给 **Quay** 容器。**之后，您可以使用该密码以用户 quayconfig 身份登录配置工具。**

以下是如何进行此操作的示例：

1. **以设置 模式启动 quay ：在第一个 quay 节点上，运行以下命令：**

```
# sudo podman run --rm -it --name quay_config -p 8080:8080
registry.redhat.io/quay/quay-rhel8:v3.12.0 config my-secret-password
```

2. **打开浏览器 ：当 quay 配置工具启动时，打开浏览器到您要运行配置工具的系统的 URL 和端口 8080 （例如 http://myquay.example.com:8080）。会提示您输入用户名和密码。**

3. **以 quayconfig 身份登录 ：在提示时，输入 quayconfig 用户名和密码(podman run 命令行中的一个)。**

4. **填写必填字段 ：当您在不挂载现有配置捆绑包的情况下启动配置工具时，您将引导至初始设置会话。在设置会话中，默认值将自动填写。以下步骤将了解如何填写剩余的必填字段。**

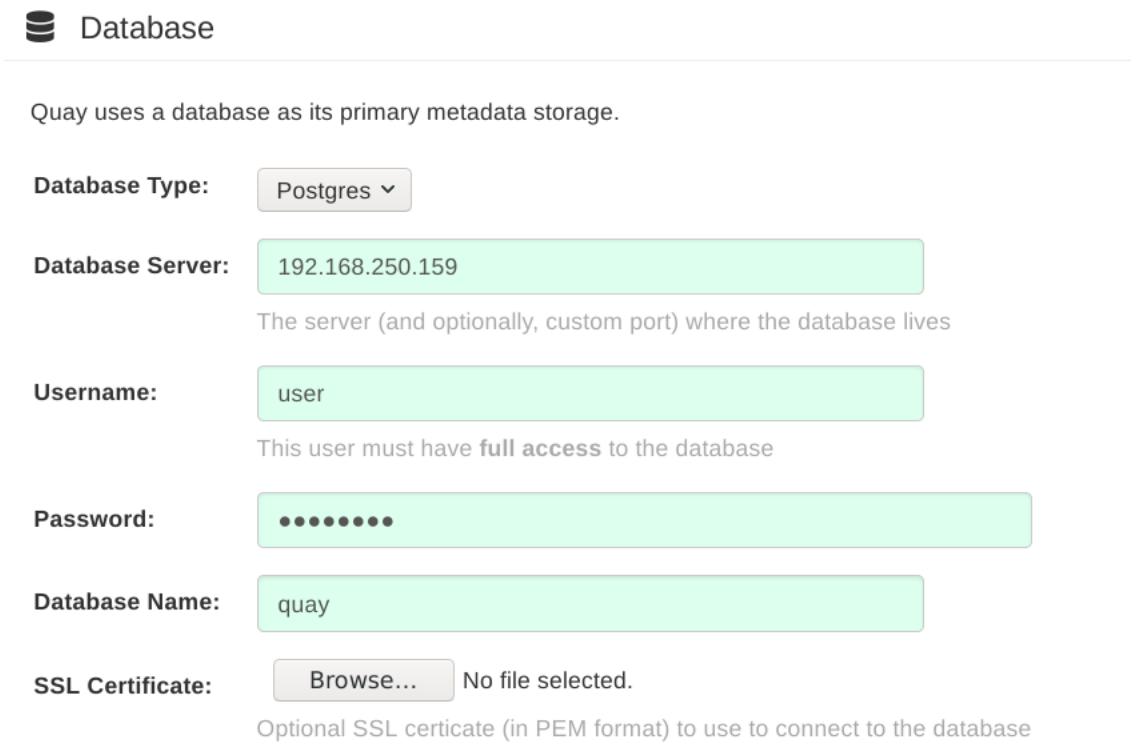
5. 识别**数据库 ： 对于初始设置，您必须包含有关 Red Hat Quay 要使用的数据库类型和位置的以下信息：**

   - **数据库类型 ：选择 MySQL 或 PostgreSQL。MySQL 将在基本示例中使用；PostgreSQL 与 OpenShift 示例的高可用性 Red Hat Quay 一起使用。**
   - **数据库服务器 ：识别数据库的 IP 地址或主机名，如果端口号不同，则标识其端口号。**
   - **用户名 ：识别具有数据库的完整访问权限的用户。**
   - **Password ：输入分配给所选用户的密码。**

- **数据库名称 ：输入您启动数据库服务器时分配的数据库名称。**

- **SSL 证书 ：对于生产环境，您应该提供 SSL 证书来连接数据库。**

  下图显示了用于标识 Red Hat Quay 使用的数据库的屏幕示例：



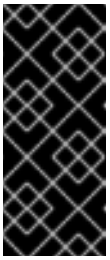
6. **识别 Redis 主机名、服务器配置并添加其他所需的设置 ：您可以添加的其他设置来完成设置，如下所示。更多用于基本部署的 Red Hat Quay 部署的设置：**

   - **对于基本的测试配置，标识 Redis Hostname 需要的所有配置都是您需要执行的。但是，您可以添加其他功能，如 Clair 扫描和存储库镜像，如此过程末尾所述。**

   - **对于高可用性和 OpenShift 配置，需要更多设置（如下所示）以允许共享存储、系统之间的安全通信和其他功能。**

     以下是您需要考虑的设置：

- 自定义 SSL 证书：上传供 Red Hat Quay 使用的自定义或自签名 SSL 证书。详情请参阅 使用 SSL 保护到 Red Hat Quay 的连接。建议提供高可用性。



重要

对于基本和高可用性部署，建议使用 SSL 证书。如果您决定不使用 SSL，您必须将容器客户端配置为使用新的 Red Hat Quay 设置作为不安全的 registry，如 测试 Insecure Registry 所述。

- 基本配置：上传公司徽标以重新命名 Red Hat Quay registry。
- 服务器配置：用于访问 Red Hat Quay 服务的主机名或 IP 地址，以及 TLS 表示（针对生产环境安装推荐）。所有 Red Hat Quay 部署都需要 Server Hostname。TLS 终止可以通过两种不同的方式完成：
  - 在实例本身上，具有由 Quay 容器中 nginx 服务器管理的所有 TLS 流量（推荐）。
  - 在负载均衡器中。不建议这样做。如果没有在负载均衡器上正确完成 TLS 设置，则对 Red Hat Quay 的访问可能会丢失。
- 数据一致性设置：选择 relax 日志记录一致性保证以提高性能和可用性。
- 时间机器：允许旧镜像标签保留在存储库中，以设定的时间，并允许用户选择自己的标签过期时间。
- Redis：标识主机名或 IP 地址（以及可选密码）以连接到 Red Hat Quay 使用的 redis 服务。
- 存储库镜像：选择启用存储库镜像的复选框。启用此后，您可以在 Red Hat Quay 集群中创建软件仓库，从远程 registry 中镜像所选的存储库。在启用存储库镜像前，请启动存储库镜像 worker，如此流程所述。
- registry 存储：识别存储位置。各种云和本地存储选项均可用。高可用性需要远程存储。如果您遵循 Red Hat Quay 高可用性存储的示例，请识别 Ceph 存储位置。在

OpenShift 上，示例使用 Amazon S3 存储。

- 操作日志存储配置 ： Action 日志默认存储在 Red Hat Quay 数据库中。如果您有大量操作日志，您可以将这些日志定向到 Elasticsearch，以便稍后搜索和分析。为此，请将 Action Logs Storage 的值改为 Elasticsearch 并配置相关的设置，如 Configure action log storage 所述。

- 操作日志轮转和 Archiving：选择该选项启用日志轮转，该轮转 时间超过 30 天，然后指示存储区域。

- 安全扫描 器：通过选择安全扫描程序端点和身份验证密钥来启用安全扫描。要设置 Clair 进行镜像扫描，请参阅 Clair Setup 和 配置 Clair。建议提供高可用性。

- 应用程序 Registry ：启用包含 Kubernetes 清单或 Helm chart 等内容的额外应用程序 registry（请参阅 App Registry 规格）。

- rkt Conversion ：允许使用 rkt fetch 从 Red Hat Quay registry 获取镜像。需要公共和私有 GPG2 密钥。此字段已弃用。

- 电子邮件 ：允许电子邮件用于通知和用户密码重置.

- 内部身份验证 ：将注册表的默认身份验证从 Local Database 更改为 LDAP、Keystone(OpenStack)、JWT 自定义身份验证或外部应用令牌。

- 外部授权(OAuth) ：允许 GitHub 或 GitHub Enterprise 向 registry 进行身份验证。

- Google Authentication ：启用 Google 以允许 Google 向 registry 进行身份验证。

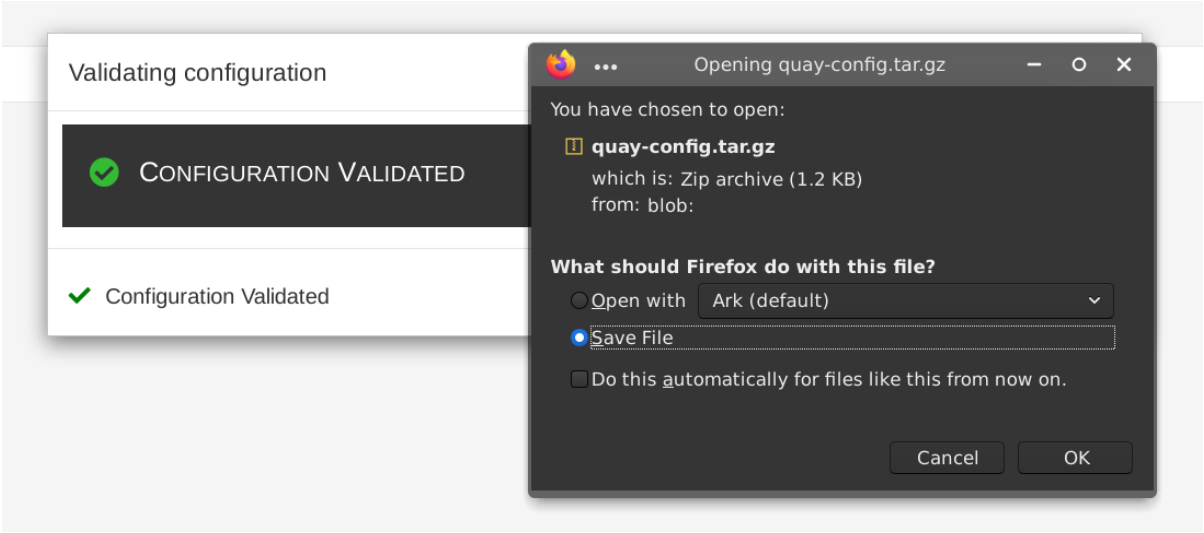
- Access Settings ：默认启用基本用户名/密码身份验证。可启用的其他身份验证类型包括：外部应用令牌（通过 docker 或 rkt 命令使用的用户生成的令牌）、匿名访问（对于可以获取 registry 的公共访问权限）、用户创建（允许用户创建自己的帐户）、加密客户端密码（查询命令行用户访问包含加密的密码），以及前缀用户名自动完成（禁用需要自动完成的用户名自动完成）。

- registry Protocol Settings ：启用 Restrict V1 Push Support 复选框，以限制对

Docker V1 协议推送的访问。虽然红帽建议启用 Docker V1 推送协议（如果允许），但您必须明确将启用的命名空间列入白名单中。

- Dockerfile 构建支持 ：启用以允许用户构建并推送到 Red Hat Quay。不建议在多租户环境中这样做。

7. 验证更改 ：选择 验证配置更改。如果验证成功，您会看到以下 Download Configuration 模式：



8. 下载配置 ：选择 Download Configuration 按钮，并将 tarball (quay-config.tar.gz)保存到本地目录，以便稍后用于启动 Red Hat Quay。

此时，您可以关闭 Red Hat Quay 配置工具并关闭浏览器。接下来，将 tarball 文件复制到您要在其上安装第一个 Red Hat Quay 节点的系统中。对于基本安装，您可能只是在同一系统上运行 Red Hat Quay。

第 5 章 部署 RED HAT QUAY

要在集群中的节点上部署 Red Hat Quay 服务，您可以使用您用来创建配置文件的同一 Quay 容器。这里的区别是您：

- 识别存储配置文件和数据的目录
- 使用 --sysctl net.core.somaxconn=4096运行命令
- 不要使用 配置选项 或密码

对于基本设置，您可以在单个节点上部署；对于高可用性，您可能需要三个或更多节点（例如 quay01、quay02 和 quay03）。

注意

生成的 Red Hat Quay 服务将侦听常规端口 8080 和 SSL 端口 8443。这与之前的 Red Hat Quay 版本不同，分别侦听标准端口 80 和 443。在本文档中，我们将 8080 和 8443 分别映射到主机上的标准端口 80 和 443。在本文档的其余部分中，我们假设您以这种方式映射了端口。

以下是您做什么：

1. 创建目录 ：创建两个目录来存储主机上的配置信息和数据。例如：

```
# mkdir -p /mnt/quay/config
# #optional: if you don't choose to install an Object Store
# mkdir -p /mnt/quay/storage
```

2. 将配置文件 复制：将 tarball (quay-config.tar.gz)复制到配置目录并解包它。例如：

```
# cp quay-config.tar.gz /mnt/quay/config/
# tar xvf quay-config.tar.gz
config.yaml ssl.cert ssl.key
```

3.

部署 Red Hat Quay ：已经过 Quay.io 身份验证（请参阅 访问 Red Hat Quay）以容器的形式运行，如下所示：

注意

在 Quay 容器的 podman run 命令行中添加 -e DEBUGLOG=true 来启用调试级别日志记录。添加 -e IGNORE_VALIDATION=true 以绕过启动过程中的验证。

```
# sudo podman run --restart=always -p 443:8443 -p 80:8080 \
  --sysctl net.core.somaxconn=4096 \
  --privileged=true \
  -v /mnt/quay/config:/conf/stack:Z \
  -v /mnt/quay/storage:/datastorage:Z \
  -d registry.redhat.io/quay/quay-rhel8:v3.12.0
```

4. 打开 UI 的浏览器 ：在 Quay 容器启动后，进入 Web 浏览器并打开运行 Quay 容器的节点的 URL。

5. 登录 Red Hat Quay ：使用您在配置过程中创建的超级用户帐户，登录并确保 Red Hat Quay 正常工作。

6. 添加更多 Red Hat Quay 节点 ：此时，您可以选择向这个 Red Hat Quay 集群添加更多节点，方法是进入每个节点，然后添加 tarball 并启动 Quay 容器，如上所示。

7. 添加可选功能 ：要在 Red Hat Quay 集群中添加更多功能，如 Clair 镜像扫描和存储库镜像，请继续下一部分。

5.1. 在 RED HAT QUAY 中添加 CLAIR 镜像扫描

为您的 Red Hat Quay 部署设置和部署 Clair 镜像扫描，请参阅 Clair安全扫描

5.2. 添加软件仓库镜像 RED HAT QUAY

启用存储库镜像允许您在 Red Hat Quay 集群上创建容器镜像仓库，它们与所选外部 registry 的内容完全匹配，然后定期根据需要同步这些存储库的内容。

将存储库镜像功能添加到 Red Hat Quay 集群中：

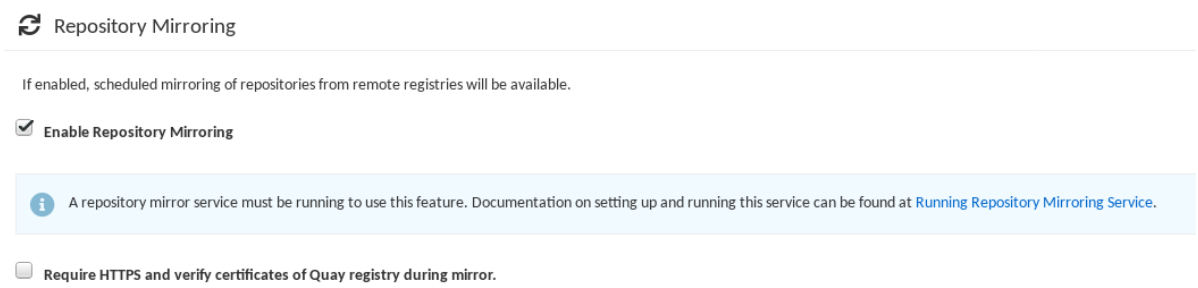
- 运行存储库镜像 worker。要做到这一点，您可以使用 repomirror 选项启动 quay pod。
- 在 Red Hat Quay Setup 工具中选择 "Enable Repository Mirroring。
- 登录您的 Red Hat Quay Web UI，并开始创建镜像的存储库，如 Red Hat Quay 中的 存储库镜像 中所述。

以下流程假设您已在 OpenShift 平台上运行 Red Hat Quay 集群，并在浏览器中运行 Red Hat Quay Setup 容器：

1. 启动存储库镜像 worker: 以 repomirror 模式启动 Quay 容器。本例假设您已使用当前存储在 /root/ca.crt 中的证书配置了 TLS 通信。如果没有，请删除向容器添加 /root/ca.crt 的行：

```
$ sudo podman run -d --name mirroring-worker \
  -v /mnt/quay/config:/conf/stack:Z \
  -v /root/ca.crt:/etc/pki/ca-trust/source/anchors/ca.crt \
  registry.redhat.io/quay/quay-rhel8:v3.12.0 repomirror
```

2. 登录到配置工具 ：登录到 Red Hat Quay Setup Web UI (config 工具)。
3. Enable repository mirroring: Scroll the Repository Mirroring 部分，然后选择 Enable Repository Mirroring 复选框，如下所示：
4. 选择 HTTPS 和证书验证 ：如果要在镜像过程中需要 HTTPS 通信并验证证书，请选择此复选框。



5. **保存配置 ：**选择 **Save Configuration Changes** 按钮。现在，应在 **Red Hat Quay** 集群上启用存储库镜像。有关设置您自己的 镜像容器镜像存储库的详细信息，请参阅 **Red Hat Quay** 中的存储库。

第 6 章 开始使用 RED HAT QUAY

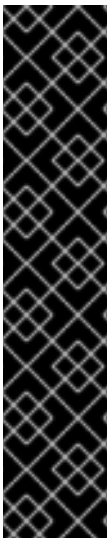
使用 Red Hat Quay 现在，您可以：

- 从 Quay 主页选择教程，以尝试 15 分钟教程。在教程中，您可以了解如何登录到 Quay，启动容器，创建镜像，推送镜像，推送存储库，查看存储库，以及使用 Quay 更改存储库权限。

- 有关使用 Red Hat Quay 存储库的信息，请参阅使用 Red Hat Quay。

第 7 章 升级独立 RED HAT QUAY 的地理复制部署

使用以下步骤升级您的 geo-replication Red Hat Quay 部署。



重要

- 当将 geo-replication Red Hat Quay 部署升级到下一个 y-stream 版本（例如，Red Hat Quay 3.7 → Red Hat Quay 3.8）或 geo-replication 部署时，您必须在升级前停止操作。
- 故障时间从一个 y-stream 版本升级到下一个版本会间歇性。
- 在升级前，强烈建议您备份 Red Hat Quay 部署。

先决条件

- 已登录到 registry.redhat.io

流程

此流程假设您在三个（或更多）系统上运行 Red Hat Quay 服务。如需更多信息，请参阅准备 Red Hat Quay 高可用性。

1. 获取运行 Red Hat Quay 实例的每个系统上所有 Red Hat Quay 实例的列表。

   a. 在 System A 中输入以下命令来显示 Red Hat Quay 实例：

```
$ sudo podman ps
```

输出示例

```
CONTAINER ID  IMAGE                                COMMAND       CREATED
STATUS          PORTS                               NAMES
ec16ece208c0  registry.redhat.io/quay/quay-rhel8:v3.7.0  registry       6 minutes
ago  Up 6 minutes ago  0.0.0.0:80->8080/tcp, 0.0.0.0:443->8443/tcp  quay01
```

b. 在 System B 中输入以下命令来显示 Red Hat Quay 实例：

```
$ sudo podman ps
```

输出示例

```
CONTAINER ID  IMAGE                                COMMAND         CREATED
STATUS            PORTS                                NAMES
7ae0c9a8b37d  registry.redhat.io/quay/quay-rhel8:v3.7.0  registry        5 minutes
ago   Up 2 seconds ago   0.0.0.0:82->8080/tcp, 0.0.0.0:445->8443/tcp  quay02
```

c. 在 System C 中输入以下命令来显示 Red Hat Quay 实例：

```
$ sudo podman ps
```

输出示例

```
CONTAINER ID  IMAGE                                COMMAND         CREATED
STATUS            PORTS                                NAMES
e75c4aebfee9  registry.redhat.io/quay/quay-rhel8:v3.7.0  registry        4 seconds
ago   Up 4 seconds ago   0.0.0.0:84->8080/tcp, 0.0.0.0:447->8443/tcp  quay03
```

2. 在每个系统中临时关闭所有 Red Hat Quay 实例。

a. 在 System A 中输入以下命令来关闭 Red Hat Quay 实例：

```
$ sudo podman stop ec16ece208c0
```

b. 在 System B 中输入以下命令来关闭 Red Hat Quay 实例：

```
$ sudo podman stop 7ae0c9a8b37d
```

c. 在 System C 中输入以下命令来关闭 Red Hat Quay 实例：

```
$ sudo podman stop e75c4aebfee9
```

3. 在每个系统上获取最新的 Red Hat Quay 版本，如 Red Hat Quay 3。

a. 在 System A 中输入以下命令来获取最新的 Red Hat Quay 版本：

```
$ sudo podman pull registry.redhat.io/quay/quay-rhel8:v3.8.0
```

b. 在 System B 中输入以下命令来获取最新的 Red Hat Quay 版本：

```
$ sudo podman pull registry.redhat.io/quay/quay-rhel8:v3.8.0
```

c. 在 System C 中输入以下命令来获取最新的 Red Hat Quay 版本：

```
$ sudo podman pull registry.redhat.io/quay/quay-rhel8:v3.8.0
```

4. 在高可用性 Red Hat Quay 部署的系统 A 中，运行新镜像版本，例如 Red Hat Quay 3：

```
# sudo podman run --restart=always -p 443:8443 -p 80:8080 \
  --sysctl net.core.somaxconn=4096 \
  --name=quay01 \
  -v /mnt/quay/config:/conf/stack:Z \
  -v /mnt/quay/storage:/datastorage:Z \
  -d registry.redhat.io/quay/quay-rhel8:v3.8.0
```

5. 等待新的 Red Hat Quay 容器在 System A 上完全正常工作。您可以输入以下命令检查容器的状态：

```
$ sudo podman ps
```

输出示例

```
CONTAINER ID  IMAGE                                  COMMAND       CREATED       STATUS
PORTS                                NAMES
70b9f38c3fb4  registry.redhat.io/quay/quay-rhel8:v3.8.0  registry      2 seconds ago
Up 2 seconds ago   0.0.0.0:82->8080/tcp, 0.0.0.0:445->8443/tcp  quay01
```

6. 可选：通过进入到 Red Hat Quay UI 来确保 Red Hat Quay 已完全操作。

7. 确保 System A 上的 Red Hat Quay 完全正常工作后，在 System B 和 System C 上运行新镜像版本。

   a. 在高可用性 Red Hat Quay 部署的 System B 中，运行新镜像版本，例如 Red Hat Quay 3：

```
# sudo podman run --restart=always -p 443:8443 -p 80:8080 \
  --sysctl net.core.somaxconn=4096 \
  --name=quay02 \
  -v /mnt/quay/config:/conf/stack:Z \
  -v /mnt/quay/storage:/datastorage:Z \
  -d registry.redhat.io/quay/quay-rhel8:v3.8.0
```

   b. 在高可用性 Red Hat Quay 部署的系统 C 中，运行新镜像版本，例如 Red Hat Quay 3：

```
# sudo podman run --restart=always -p 443:8443 -p 80:8080 \
  --sysctl net.core.somaxconn=4096 \
  --name=quay03 \
  -v /mnt/quay/config:/conf/stack:Z \
  -v /mnt/quay/storage:/datastorage:Z \
  -d registry.redhat.io/quay/quay-rhel8:v3.8.0
```

8. 您可以输入以下命令来检查 System B 和 System C 中的容器状态：

```
$ sudo podman ps
```

第 8 章 在 RED HAT QUAY 部署上执行健康检查

健康检查机制旨在评估系统、服务或组件的健康和功能。健康检查有助于确保一切正常工作，并可用于在潜在问题成为严重问题之前识别潜在问题。通过监控系统的健康状况，Red Hat Quay 管理员可以针对地域复制部署、Operator 部署、独立 Red Hat Quay 部署、对象存储问题等方面解决异常或潜在的故障。执行健康检查有助于降低遇到故障排除场景的可能性。

通过提供有关系统当前状态的宝贵信息，健康检查机制可以在诊断问题方面扮演角色。通过将健康检查结果与预期的基准测试或预定义的阈值进行比较，可以更快地识别 deviations 或 anomalies。

8.1. RED HAT QUAY 健康检查端点

重要

此处包含的任何外部网站的链接仅为方便用户而提供。红帽没有审阅链接的内容，并不对其内容负责。包含到外部网站的任何链接并不意味着红帽认可该网站或其实体、产品或服务。您同意红帽对因您使用（或依赖）外部网站或内容而导致的任何损失或费用不承担任何责任。

Red Hat Quay 有几个健康检查端点。下表显示了健康检查、描述、端点和示例输出。

表 8.1. 健康检查端点

健康检查	描述	端点	输出示例
实例	实例 端点获取特定 Red Hat Quay 实例的完整状态。返回带有以下键值对 的字典 ： auth,database,disk_space,registry_gunicorn,service_key, 和 web_gunicorn。 返回指示 200 的健康检查响应的数字，这表示实例处于健康状态，或者 503，这表示您的部署有问题。	https://{quay-ip-endpoint}/health/instance 或 https://{quay-ip-endpoint}/health	{"data":{"services":{"auth":true,"database":true,"disk_space":true,"registry_gunicorn":true,"service_key":true,"web_gunicorn":true}},"status_code":200}
endtoend	端到端端点对 Red Hat Quay 实例的所有服务进行检查。使用以下的键值对返回 字典 ： auth、数据库、redis、存储。返回指示 200 的健康检查响应的数字，这表示实例处于健康状态，或者 503，这表示您的部署有问题。	https://{quay-ip-endpoint}/health/endtoend	{"data":{"services":{"auth":true,"database":true,"redis":true,"storage":true}},"status_code":200}

健康检查	描述	端点	输出示例
warning	警告 端点对警告进行检查。为以下内容返回一个带有键值对的字典：disk_space_warning。返回指示 200 的健康检查响应的数字，这表示实例处于健康状态，或者 503，这表示您的部署有问题。	https://{quay-ip-endpoint}/health/warning	{"data":{"services":{"disk_space_warning":true}},"status_code":503}

8.2. 导航到 RED HAT QUAY 健康检查端点

使用以下步骤导航到 实例 端点。对于 端到端 和警告 端点，这个过程可以重复。

流程

1. 在 Web 浏览器中，导航到 https://{quay-ip-endpoint}/health/instance。
2. 您使用健康实例页面，它会返回类似如下的信息：

```
{"data":{"services":
{"auth":true,"database":true,"disk_space":true,"registry_gunicorn":true,"service_key"
:true,"web_gunicorn":true}},"status_code":200}
```

对于 Red Hat Quay，"status_code": 200 表示实例是健康的。相反，如果您收到 "status_code": 503，则部署有问题。

其他资源