



Red Hat Quay 3

Upgrade Red Hat Quay

Upgrade Red Hat Quay

Red Hat Quay 3 Upgrade Red Hat Quay

Upgrade Red Hat Quay

Legal Notice

Copyright © 2023 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at

<http://creativecommons.org/licenses/by-sa/3.0/>

. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

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

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

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

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

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

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

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

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

All other trademarks are the property of their respective owners.

Abstract

Upgrade Red Hat Quay

Table of Contents

CHAPTER 1. UPGRADE OVERVIEW	4
CHAPTER 2. UPGRADING THE RED HAT QUAY OPERATOR OVERVIEW	5
2.1. OPERATOR LIFECYCLE MANAGER	5
2.2. UPGRADING THE QUAY OPERATOR	5
2.2.1. Upgrading Quay	6
2.2.2. Updating Red Hat Quay from 3.8 → 3.9	6
2.2.3. Upgrading directly from 3.3.z or 3.4.z to 3.6	8
2.2.3.1. Upgrading with edge routing enabled	8
2.2.3.2. Upgrading with custom SSL/TLS certificate/key pairs without Subject Alternative Names	8
2.2.3.3. Configuring Clair v4 when upgrading from 3.3.z or 3.4.z to 3.6 using the Red Hat Quay Operator	9
2.2.4. Swift configuration when upgrading from 3.3.z to 3.6	9
2.2.5. Changing the update channel for the Red Hat Quay Operator	9
2.2.6. Manually approving a pending Operator upgrade	9
2.3. UPGRADING A QUAYREGISTRY	10
2.4. UPGRADING A QUAYECOSYSTEM	10
2.4.1. Reverting QuayEcosystem Upgrade	11
2.4.2. Supported QuayEcosystem Configurations for Upgrades	11
CHAPTER 3. STANDALONE UPGRADE	13
3.1. ACCESSING IMAGES	14
3.2. UPGRADE TO 3.9.Z FROM 3.8.Z	14
3.2.1. Target images	16
3.3. UPGRADE TO 3.9.Z FROM 3.7.Z	16
3.3.1. Target images	16
3.4. UPGRADE TO 3.8.Z FROM 3.7.Z	16
3.4.1. Target images	16
3.5. UPGRADE TO 3.7.Z FROM 3.6.Z	17
3.5.1. Target images	17
3.6. UPGRADE TO 3.7.Z FROM 3.5.Z	17
3.6.1. Target images	17
3.7. UPGRADE TO 3.7.Z FROM 3.4.Z	17
3.7.1. Target images	17
3.8. UPGRADE TO 3.7.Z FROM 3.3.Z	17
3.9. UPGRADE TO 3.6.Z FROM 3.5.Z	17
3.9.1. Target images	17
3.10. UPGRADE TO 3.6.Z FROM 3.4.Z	18
3.10.1. Target images	18
3.11. UPGRADE TO 3.6.Z FROM 3.3.Z	18
3.11.1. Target images	18
3.11.2. Swift configuration when upgrading from 3.3.z to 3.6	18
3.12. UPGRADE TO 3.5.7 FROM 3.4.Z	19
3.12.1. Target images	19
3.13. UPGRADE TO 3.4.6 FROM 3.3.Z	19
3.13.1. Target images	19
3.14. UPGRADE TO 3.3.4 FROM 3.2.Z	19
3.14.1. Target images	19
3.15. UPGRADE TO 3.2.2 FROM 3.1.Z	20
3.15.1. Target images	20
3.16. UPGRADE TO 3.1.3 FROM 3.0.Z	21
3.16.1. Target images	21

3.17. UPGRADE TO 3.0.5 FROM 2.9.5	21
3.17.1. Overview of upgrade	21
3.17.2. Prerequisites	22
3.17.3. Choosing upgrade type	22
3.17.4. Running a synchronous upgrade	22
3.17.5. Running a background upgrade	23
3.17.6. Target images	24
3.18. UPGRADING A GEO-REPLICATION DEPLOYMENT OF RED HAT QUAY	24
CHAPTER 4. UPGRADE QUAY BRIDGE OPERATOR	28
4.1. UPGRADING A GEO-REPLICATION DEPLOYMENT OF THE RED HAT QUAY OPERATOR	28
CHAPTER 5. DOWNGRADING RED HAT QUAY	31

CHAPTER 1. UPGRADE OVERVIEW

The upgrade procedure for Red Hat Quay depends on the type of installation you are using.

The Red Hat Quay Operator provides a simple method to deploy and manage a Red Hat Quay cluster. This is the preferred procedure for deploying Red Hat Quay on OpenShift.

The Red Hat Quay Operator should be upgraded using the [Operator Lifecycle Manager \(OLM\)](#) as described in the section "Upgrading Quay using the Quay Operator".

The procedure for upgrading a proof-of-concept or highly available installation of Red Hat Quay and Clair is documented in the section "Standalone upgrade".

CHAPTER 2. UPGRADING THE RED HAT QUAY OPERATOR OVERVIEW

The Red Hat Quay Operator follows a *synchronized versioning* scheme, which means that each version of the Operator is tied to the version of Red Hat Quay and the components that it manages. There is no field on the **QuayRegistry** custom resource which sets the version of Red Hat Quay to **deploy**; the Operator can only deploy a single version of all components. This scheme was chosen to ensure that all components work well together and to reduce the complexity of the Operator needing to know how to manage the lifecycles of many different versions of Red Hat Quay on Kubernetes.

2.1. OPERATOR LIFECYCLE MANAGER

The Red Hat Quay Operator should be installed and upgraded using the [Operator Lifecycle Manager \(OLM\)](#). When creating a **Subscription** with the default **approvalStrategy: Automatic**, OLM will automatically upgrade the Red Hat Quay Operator whenever a new version becomes available.



WARNING

When the Red Hat Quay Operator is installed by Operator Lifecycle Manager, it might be configured to support automatic or manual upgrades. This option is shown on the **Operator Hub** page for the Red Hat Quay Operator during installation. It can also be found in the Red Hat Quay Operator **Subscription** object by the **approvalStrategy** field. Choosing **Automatic** means that your Red Hat Quay Operator will automatically be upgraded whenever a new Operator version is released. If this is not desirable, then the **Manual** approval strategy should be selected.

2.2. UPGRADING THE QUAY OPERATOR

The standard approach for upgrading installed Operators on OpenShift Container Platform is documented at [Upgrading installed Operators](#).

In general, Red Hat Quay supports upgrades from a prior (N-1) minor version only. For example, upgrading directly from Red Hat Quay 3.0.5 to the latest version of 3.5 is not supported. Instead, users would have to upgrade as follows:

1. 3.0.5 → 3.1.3
2. 3.1.3 → 3.2.2
3. 3.2.2 → 3.3.4
4. 3.3.4 → 3.4.z
5. 3.4.z → 3.5.z

This is required to ensure that any necessary database migrations are done correctly and in the right order during the upgrade.

In some cases, Red Hat Quay supports direct, single-step upgrades from prior (N-2, N-3) minor versions. This simplifies the upgrade procedure for customers on older releases. The following upgrade paths are supported:

1. 3.3.z → 3.6.z
2. 3.4.z → 3.6.z
3. 3.4.z → 3.7.z
4. 3.5.z → 3.7.z
5. 3.7.z → 3.8.z
6. 3.6.z → 3.9.z
7. 3.7.z → 3.9.z
8. 3.8.z → 3.9.z

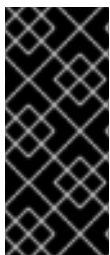
For users on standalone deployments of Red Hat Quay wanting to upgrade to 3.9, see the [Standalone upgrade](#) guide.

2.2.1. Upgrading Quay

To update Red Hat Quay from one minor version to the next, for example, 3.4 → 3.5, you must change the update channel for the Red Hat Quay Operator.

For **z** stream upgrades, for example, 3.4.2 → 3.4.3, updates are released in the major-minor channel that the user initially selected during install. The procedure to perform a **z** stream upgrade depends on the **approvalStrategy** as outlined above. If the approval strategy is set to **Automatic**, the Quay Operator will upgrade automatically to the newest **z** stream. This results in automatic, rolling Quay updates to newer **z** streams with little to no downtime. Otherwise, the update must be manually approved before installation can begin.

2.2.2. Updating Red Hat Quay from 3.8 → 3.9



IMPORTANT

If your Red Hat Quay deployment is upgrading from one y-stream to the next, for example, from 3.8.10 → 3.8.11, you must not switch the upgrade channel from **stable-3.8** to **stable-3.9**. Changing the upgrade channel in the middle of a y-stream upgrade will disallow Red Hat Quay from upgrading to 3.9. This is a known issue and will be fixed in a future version of Red Hat Quay.

When updating Red Hat Quay 3.8 → 3.9, the Operator automatically upgrades the existing PostgreSQL databases for Clair and Red Hat Quay from version 10 to version 13.



IMPORTANT

- This upgrade is irreversible. It is highly recommended that you upgrade to PostgreSQL 13. PostgreSQL 10 had its final release on November 10, 2022 and is no longer supported. For more information, see the [PostgreSQL Versioning Policy](#).
- By default, Red Hat Quay is configured to remove old persistent volume claims (PVCs) from PostgreSQL 10. To disable this setting and backup old PVCs, you must set **POSTGRES_UPGRADE_RETAIN_BACKUP** to **True** in your **quay-operator Subscription** object.

Prerequisites

- You have installed Red Hat Quay 3.8 on OpenShift Container Platform.
- 100 GB of free, additional storage.
During the upgrade process, additional persistent volume claims (PVCs) are provisioned to store the migrated data. This helps prevent a destructive operation on user data. The upgrade process rolls out PVCs for 50 GB for both the Red Hat Quay database upgrade, and the Clair database upgrade.

Procedure

1. Optional. Back up your old PVCs from PostgreSQL 10 by setting **POSTGRES_UPGRADE_RETAIN_BACKUP** to **True** your **quay-operator Subscription** object. For example:

```
apiVersion: operators.coreos.com/v1alpha1
kind: Subscription
metadata:
  name: quay-operator
  namespace: quay-enterprise
spec:
  channel: stable-3.8
  name: quay-operator
  source: redhat-operators
  sourceNamespace: openshift-marketplace
config:
  env:
    - name: POSTGRES_UPGRADE_RETAIN_BACKUP
      value: "true"
```

2. In the OpenShift Container Platform Web Console, navigate to **Operators → Installed Operators**.
3. Click on the Red Hat Quay Operator.
4. Navigate to the **Subscription** tab.
5. Under **Subscription details** click **Update channel**.
6. Select **stable-3.9** and save the changes.
7. Check the progress of the new installation under **Upgrade status**. Wait until the upgrade status changes to **1 installed** before proceeding.

8. In your OpenShift Container Platform cluster, navigate to **Workloads → Pods**. Existing pods should be terminated, or in the process of being terminated.
9. Wait for the following pods, which are responsible for upgrading the database and alembic migration of existing data, to spin up: **clair-postgres-upgrade**, **quay-postgres-upgrade**, and **quay-app-upgrade**.
10. After the **clair-postgres-upgrade**, **quay-postgres-upgrade**, and **quay-app-upgrade** pods are marked as **Completed**, the remaining pods for your Red Hat Quay deployment spin up. This takes approximately ten minutes.
11. Verify that the **quay-database** and **clair-postgres** pods now use the **postgresql-13** image.
12. After the **quay-app** pod is marked as **Running**, you can reach your Red Hat Quay registry.

2.2.3. Upgrading directly from 3.3.z or 3.4.z to 3.6

The following section provides important information when upgrading from Red Hat Quay 3.3.z or 3.4.z to 3.6.

2.2.3.1. Upgrading with edge routing enabled

- Previously, when running a 3.3.z version of Red Hat Quay with edge routing enabled, users were unable to upgrade to 3.4.z versions of Red Hat Quay. This has been resolved with the release of Red Hat Quay 3.6.
- When upgrading from 3.3.z to 3.6, if **tls.termination** is set to **none** in your Red Hat Quay 3.3.z deployment, it will change to HTTPS with TLS edge termination and use the default cluster wildcard certificate. For example:

```
apiVersion: redhatcop.redhat.io/v1alpha1
kind: QuayEcosystem
metadata:
  name: quay33
spec:
  quay:
    imagePullSecretName: redhat-pull-secret
    enableRepoMirroring: true
    image: quay.io/quay/quay:v3.3.4-2
    ...
  externalAccess:
    hostname: quayv33.apps.devcluster.openshift.com
    tls:
      termination: none
  database:
    ...
```

2.2.3.2. Upgrading with custom SSL/TLS certificate/key pairs without Subject Alternative Names

There is an issue for customers using their own SSL/TLS certificate/key pairs without Subject Alternative Names (SANs) when upgrading from Red Hat Quay 3.3.4 to Red Hat Quay 3.6 directly. During the upgrade to Red Hat Quay 3.6, the deployment is blocked, with the error message from the Red Hat Quay Operator pod logs indicating that the Red Hat Quay SSL/TLS certificate must have SANs.

If possible, you should regenerate your SSL/TLS certificates with the correct hostname in the SANs. A possible workaround involves defining an environment variable in the **quay-app**, **quay-upgrade** and **quay-config-editor** pods after upgrade to enable CommonName matching:

```
GODEBUG=x509ignoreCN=0
```

The **GODEBUG=x509ignoreCN=0** flag enables the legacy behavior of treating the CommonName field on X.509 certificates as a hostname when no SANs are present. However, this workaround is not recommended, as it will not persist across a redeployment.

2.2.3.3. Configuring Clair v4 when upgrading from 3.3.z or 3.4.z to 3.6 using the Red Hat Quay Operator

To set up Clair v4 on a new Red Hat Quay deployment on OpenShift Container Platform, it is highly recommended to use the Red Hat Quay Operator. By default, the Red Hat Quay Operator will install or upgrade a Clair deployment along with your Red Hat Quay deployment and configure Clair automatically.

For instructions about setting up Clair v4 in a disconnected OpenShift Container Platform cluster, see [Setting Up Clair on a Red Hat Quay OpenShift deployment](#).

2.2.4. Swift configuration when upgrading from 3.3.z to 3.6

When upgrading from Red Hat Quay 3.3.z to 3.6.z, some users might receive the following error: **Switch auth v3 requires tenant_id (string) in os_options**. As a workaround, you can manually update your **DISTRIBUTED_STORAGE_CONFIG** to add the **os_options** and **tenant_id** parameters:

```
DISTRIBUTED_STORAGE_CONFIG:
  brscale:
    - SwiftStorage
    - auth_url: http://****/v3
      auth_version: "3"
      os_options:
        tenant_id: ****
        project_name: ocp-base
        user_domain_name: Default
      storage_path: /datastorage/registry
      swift_container: ocp-svc-quay-ha
      swift_password: *****
      swift_user: *****
```

2.2.5. Changing the update channel for the Red Hat Quay Operator

The subscription of an installed Operator specifies an update channel, which is used to track and receive updates for the Operator. To upgrade the Red Hat Quay Operator to start tracking and receiving updates from a newer channel, change the update channel in the **Subscription** tab for the installed Red Hat Quay Operator. For subscriptions with an **Automatic** approval strategy, the upgrade begins automatically and can be monitored on the page that lists the Installed Operators.

2.2.6. Manually approving a pending Operator upgrade

If an installed Operator has the approval strategy in its subscription set to **Manual**, when new updates are released in its current update channel, the update must be manually approved before installation can begin. If the Red Hat Quay Operator has a pending upgrade, this status will be displayed in the list of

Installed Operators. In the **Subscription** tab for the Red Hat Quay Operator, you can preview the install plan and review the resources that are listed as available for upgrade. If satisfied, click **Approve** and return to the page that lists Installed Operators to monitor the progress of the upgrade.

The following image shows the **Subscription** tab in the UI, including the update **Channel**, the **Approval** strategy, the **Upgrade status** and the **InstallPlan**:

The screenshot shows the Red Hat Quay UI with the 'Subscription' tab selected for the 'quay-enterprise' project. The left sidebar contains navigation links: Administrator, Home, Operators, OperatorHub, Installed Operators, Workloads, Networking, Storage, Builds, Monitoring, Compute, User Management, and Administration. The main content area displays the 'Subscription details' for the 'Red Hat Quay' operator (version 3.4.3 provided by Red Hat). The details include:

- Channel:** quay-v3.4
- Approval:** Automatic
- Upgrade status:** 1 installed, 0 installing (Up to date)
- Name:** quay-operator
- Namespace:** quay-enterprise
- Labels:** operators.coreos.com/quay-operator.quay-enterprise
- Created at:** Mar 25, 12:17 pm
- Owner:** No owner
- Installed version:** quay-operatorv3.4.3
- Starting version:** quay-operatorv3.4.3
- CatalogSource:** redhat-operators (Healthy)
- InstallPlan:** install-wf26n

The list of Installed Operators provides a high-level summary of the current Quay installation:

The screenshot shows the 'Installed Operators' page in the Red Hat Quay UI. The left sidebar is the same as the previous screenshot. The main content area displays a table of installed operators. The table has columns: Name, Managed Namespaces, Status, Last updated, and Provided APIs. The table contains one entry:

Name	Managed Namespaces	Status	Last updated	Provided APIs
Red Hat Quay 3.4.3 provided by Red Hat	quay-enterprise	Succeeded Up to date	Mar 25, 12:18 pm	Quay Registry

2.3. UPGRADING A QUAYREGISTRY

When the Red Hat Quay Operator starts, it immediately looks for any **QuayRegistries** it can find in the namespace(s) it is configured to watch. When it finds one, the following logic is used:

- If **status.currentVersion** is unset, reconcile as normal.
- If **status.currentVersion** equals the Operator version, reconcile as normal.
- If **status.currentVersion** does not equal the Operator version, check if it can be upgraded. If it can, perform upgrade tasks and set the **status.currentVersion** to the Operator's version once complete. If it cannot be upgraded, return an error and leave the **QuayRegistry** and its deployed Kubernetes objects alone.

2.4. UPGRADING A QUAYECOSYSTEM

Upgrades are supported from previous versions of the Operator which used the **QuayEcosystem** API

for a limited set of configurations. To ensure that migrations do not happen unexpectedly, a special label needs to be applied to the **QuayEcosystem** for it to be migrated. A new **QuayRegistry** will be created for the Operator to manage, but the old **QuayEcosystem** will remain until manually deleted to ensure that you can roll back and still access Quay in case anything goes wrong. To migrate an existing **QuayEcosystem** to a new **QuayRegistry**, use the following procedure.

Procedure

1. Add **"quay-operator/migrate": "true"** to the **metadata.labels** of the **QuayEcosystem**.

```
$ oc edit quayecosystem <quayecosystemname>
```

```
metadata:
  labels:
    quay-operator/migrate: "true"
```

2. Wait for a **QuayRegistry** to be created with the same **metadata.name** as your **QuayEcosystem**. The **QuayEcosystem** will be marked with the label **"quay-operator/migration-complete": "true"**.
3. After the **status.registryEndpoint** of the new **QuayRegistry** is set, access Red Hat Quay and confirm that all data and settings were migrated successfully.
4. If everything works correctly, you can delete the **QuayEcosystem** and Kubernetes garbage collection will clean up all old resources.

2.4.1. Reverting QuayEcosystem Upgrade

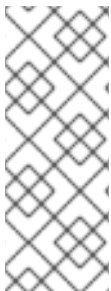
If something goes wrong during the automatic upgrade from **QuayEcosystem** to **QuayRegistry**, follow these steps to revert back to using the **QuayEcosystem**:

Procedure

1. Delete the **QuayRegistry** using either the UI or **kubectl**:

```
$ kubectl delete -n <namespace> quayregistry <quayecosystem-name>
```

2. If external access was provided using a **Route**, change the **Route** to point back to the original **Service** using the UI or **kubectl**.



NOTE

If your **QuayEcosystem** was managing the PostgreSQL database, the upgrade process will migrate your data to a new PostgreSQL database managed by the upgraded Operator. Your old database will not be changed or removed but Red Hat Quay will no longer use it once the migration is complete. If there are issues during the data migration, the upgrade process will exit and it is recommended that you continue with your database as an unmanaged component.

2.4.2. Supported QuayEcosystem Configurations for Upgrades

The Red Hat Quay Operator reports errors in its logs and in **status.conditions** if migrating a **QuayEcosystem** component fails or is unsupported. All unmanaged components should migrate

successfully because no Kubernetes resources need to be adopted and all the necessary values are already provided in Red Hat Quay's **config.yaml** file.

Database

Ephemeral database not supported (**volumeSize** field must be set).

Redis

Nothing special needed.

External Access

Only passthrough **Route** access is supported for automatic migration. Manual migration required for other methods.

- **LoadBalancer** without custom hostname: After the **QuayEcosystem** is marked with label "**quay-operator/migration-complete**": **"true"**, delete the **metadata.ownerReferences** field from existing **Service** *before* deleting the **QuayEcosystem** to prevent Kubernetes from garbage collecting the **Service** and removing the load balancer. A new **Service** will be created with **metadata.name** format **<QuayEcosystem-name>-quay-app**. Edit the **spec.selector** of the existing **Service** to match the **spec.selector** of the new **Service** so traffic to the old load balancer endpoint will now be directed to the new pods. You are now responsible for the old **Service**; the Quay Operator will not manage it.
- **LoadBalancer/NodePort/Ingress** with custom hostname: A new **Service** of type **LoadBalancer** will be created with **metadata.name** format **<QuayEcosystem-name>-quay-app**. Change your DNS settings to point to the **status.loadBalancer** endpoint provided by the new **Service**.

Clair

Nothing special needed.

Object Storage

QuayEcosystem did not have a managed object storage component, so object storage will always be marked as unmanaged. Local storage is not supported.

Repository Mirroring

Nothing special needed.

CHAPTER 3. STANDALONE UPGRADE

In general, Red Hat Quay supports upgrades from a prior (N-1) minor version only. For example, upgrading directly from Red Hat Quay 3.0.5 to the latest version of 3.5 is not supported. Instead, users would have to upgrade as follows:

1. 3.0.5 → 3.1.3
2. 3.1.3 → 3.2.2
3. 3.2.2 → 3.3.4
4. 3.3.4 → 3.4.z
5. 3.4.z → 3.5.z

This is required to ensure that any necessary database migrations are done correctly and in the right order during the upgrade.

In some cases, Red Hat Quay supports direct, single-step upgrades from prior (N-2, N-3) minor versions. This exception to the normal, prior minor version-only, upgrade simplifies the upgrade procedure for customers on older releases. The following upgrade paths are supported:

1. 3.3.z → 3.6.z
2. 3.4.z → 3.6.z
3. 3.4.z → 3.7.z
4. 3.5.z → 3.7.z
5. 3.7.z → 3.9.z

For users wanting to upgrade the Red Hat Quay Operator, see [Upgrading the Red Hat Quay Operator Overview](#).

This document describes the steps needed to perform each individual upgrade. Determine your current version and then follow the steps in sequential order, starting with your current version and working up to your desired target version.

- [Upgrade to 3.9.z from 3.8.z](#)
- [Upgrade to 3.9.z from 3.7.z](#)
- [Upgrade to 3.8.z from 3.7.z](#)
- [Upgrade to 3.7.z from 3.6.z](#)
- [Upgrade to 3.7.z from 3.5.z](#)
- [Upgrade to 3.7.z from 3.4.z](#)
- [Upgrade to 3.7.z from 3.3.z](#)
- [Upgrade to 3.6.z from 3.5.z](#)
- [Upgrade to 3.6.z from 3.4.z](#)

- [Upgrade to 3.6.z from 3.3.z](#)
- [Upgrade to 3.5.z from 3.4.z](#)
- [Upgrade to 3.4.z from 3.3.4](#)
- [Upgrade to 3.3.4 from 3.2.2](#)
- [Upgrade to 3.2.2 from 3.1.3](#)
- [Upgrade to 3.1.3 from 3.0.5](#)
- [Upgrade to 3.0.5 from 2.9.5](#)

See the [Red Hat Quay Release Notes](#) for information on features for individual releases.

The general procedure for a manual upgrade consists of the following steps:

1. Stop the Quay and Clair containers.
2. Backup the database and image storage (optional but recommended).
3. Start Clair using the new version of the image.
4. Wait until Clair is ready to accept connections before starting the new version of Quay.

3.1. ACCESSING IMAGES

Images for Quay 3.4.0 and later are available from [registry.redhat.io](#) and [registry.access.redhat.com](#), with authentication set up as described in [Red Hat Container Registry Authentication](#) .

Images for Quay 3.3.4 and earlier are available from [quay.io](#), with authentication set up as described in [Accessing Red Hat Quay without a CoreOS login](#) .

3.2. UPGRADE TO 3.9.Z FROM 3.8.Z

If you are upgrading your standalone Red Hat Quay deployment from 3.8.z → 3.9, it is highly recommended that you upgrade PostgreSQL from version 10 → 13. To upgrade PostgreSQL from 10 → 13, you must bring down your PostgreSQL 10 database and run a migration script to initiate the process.

Use the following procedure to upgrade PostgreSQL from 10 → 13 on a standalone Red Hat Quay deployment.

Procedure

1. Enter the following command to scale down the Red Hat Quay container:

```
$ sudo podman stop <quay_container_name>
```

2. Optional. If you are using Clair, enter the following command to stop the Clair container:

```
$ sudo podman stop <clair_container_id>
```

3. Run the Podman process from SCLOrg's [Data Migration](#) procedure, which allows for data migration from a remote PostgreSQL server:

```
$ sudo podman run -d --name <migration_postgresql_database> ❶
-e POSTGRESQL_MIGRATION_REMOTE_HOST=172.17.0.2 \ ❷
-e POSTGRESQL_MIGRATION_ADMIN_PASSWORD=remoteAdminP@ssword \
-v </host/data/directory:/var/lib/pgsql/data:Z> ❸
[ OPTIONAL_CONFIGURATION_VARIABLES ]
rhel8/postgresql-13
```

- ❶ The name of your PostgreSQL 13 migration database.
- ❷ Your current Red Hat Quay PostgreSQL 10 database container IP address. Can be obtained by running the following command: **sudo podman inspect -f "{{.NetworkSettings.IPAddress}}" postgresql-quay.**
- ❸ You must specify a different volume mount point than the one from your initial PostgreSQL 10 deployment, and modify the access control lists for said directory. For example:

```
$ mkdir -p /host/data/directory
```

```
$ setfacl -m u:26:-wx /host/data/directory
```

This prevents data from being overwritten by the new container.

4. Optional. If you are using Clair, repeat the previous step for the Clair PostgreSQL database container.
5. Stop the PostgreSQL 10 container:

```
$ sudo podman stop <postgresql_container_name>
```

6. After completing the PostgreSQL migration, run the PostgreSQL 13 container, using the new data volume mount from Step 3, for example, **</host/data/directory:/var/lib/postgresql/data>**:

```
$ sudo podman run -d --rm --name postgresql-quay \
-e POSTGRESQL_USER=<username> \
-e POSTGRESQL_PASSWORD=<password> \
-e POSTGRESQL_DATABASE=<quay_database_name> \
-e POSTGRESQL_ADMIN_PASSWORD=<admin_password> \
-p 5432:5432 \
-v </host/data/directory:/var/lib/pgsql/data:Z> \
registry.redhat.io/rhel8/postgresql-13:1-109
```

7. Optional. If you are using Clair, repeat the previous step for the Clair PostgreSQL database container.
8. Start the Red Hat Quay container:

```
$ sudo podman run -d --rm -p 80:8080 -p 443:8443 --name=quay \
-v /home/<quay_user>/quay-poc/config:/conf/stack:Z \
-v /home/<quay_user>/quay-poc/storage:/datastorage:Z \
{productrepo}/{quayimage}:{productminv}
```

9. Optional. Restart the Clair container, for example:

```
$ sudo podman run -d --name clairv4 \
-p 8081:8081 -p 8088:8088 \
-e CLAIR_CONF=/clair/config.yaml \
-e CLAIR_MODE=combo \
registry.redhat.io/quay/clair-rhel8:v3.9.0
```

For more information, see [Data Migration](#).

3.2.1. Target images

- **Quay:** registry.redhat.io/quay/quay-rhel8:v3.9.0
- **Clair:** registry.redhat.io/quay/clair-rhel8:4.9.0
- **PostgreSQL:** registry.redhat.io/rhel8/postgresql-13:1-109
- **Redis:** registry.redhat.io/rhel8/redis-6:1-110)

3.3. UPGRADE TO 3.9.Z FROM 3.7.Z

If you are upgrading your standalone Red Hat Quay deployment from 3.8.z → 3.9, it is highly recommended that you upgrade PostgreSQL from version 10 → 13. To upgrade PostgreSQL from 10 → 13, you must bring down your PostgreSQL 10 database and run a migration script to initiate the process:



NOTE

- When upgrading from Red Hat Quay 3.7 to 3.9, you might receive the following error: **pg_dumpall: error: query failed: ERROR: xlog flush request 1/B446CCD8 is not satisfied --- flushed only to 1/B0013858**. As a workaround to this issue, you can delete the **quayregistry-clair-postgres-upgrade** job on your OpenShift Container Platform deployment, which should resolve the issue.

3.3.1. Target images

- **Quay:** registry.redhat.io/quay/quay-rhel8:v3.9.0
- **Clair:** registry.redhat.io/quay/clair-rhel8:4.9.0
- **PostgreSQL:** registry.redhat.io/rhel8/postgresql-13:1-109
- **Redis:** registry.redhat.io/rhel8/redis-6:1-110)

3.4. UPGRADE TO 3.8.Z FROM 3.7.Z

3.4.1. Target images

- **Quay:** registry.redhat.io/quay/quay-rhel8:v3.8.0
- **Clair:** registry.redhat.io/quay/clair-rhel8:4.9.0
- **PostgreSQL:** registry.redhat.io/rhel8/postgresql-13:1-109
- **Redis:** registry.redhat.io/rhel8/redis-6:1-110)

3.5. UPGRADE TO 3.7.Z FROM 3.6.Z

3.5.1. Target images

- **Quay:** registry.redhat.io/quay/quay-rhel8:v3.7.0
- **Clair:** registry.redhat.io/quay/clair-rhel8:4.9.0
- **PostgreSQL:** registry.redhat.io/rhel8/postgresql-13:1-109
- **Redis:** registry.redhat.io/rhel8/redis-6:1-110)

3.6. UPGRADE TO 3.7.Z FROM 3.5.Z

3.6.1. Target images

- **Quay:** registry.redhat.io/quay/quay-rhel8:v3.7.0
- **Clair:** registry.redhat.io/quay/clair-rhel8:4.9.0
- **PostgreSQL:** registry.redhat.io/rhel8/postgresql-13:1-109
- **Redis:** registry.redhat.io/rhel8/redis-6:1-110)

3.7. UPGRADE TO 3.7.Z FROM 3.4.Z

3.7.1. Target images

- **Quay:** registry.redhat.io/quay/quay-rhel8:v3.7.0
- **Clair:** registry.redhat.io/quay/clair-rhel8:4.9.0
- **PostgreSQL:** registry.redhat.io/rhel8/postgresql-13:1-109
- **Redis:** registry.redhat.io/rhel8/redis-6:1-110)

3.8. UPGRADE TO 3.7.Z FROM 3.3.Z

Upgrading to Red Hat Quay 3.7 from 3.3. is unsupported. Users must first upgrade to 3.6 from 3.3, and then upgrade to 3.7. For more information, see [Upgrade to 3.6.z from 3.3.z](#) .

3.9. UPGRADE TO 3.6.Z FROM 3.5.Z

3.9.1. Target images

- **Quay:** registry.redhat.io/quay/quay-rhel8:v3.6.0
- **Clair:** registry.redhat.io/quay/clair-rhel8:4.9.0
- **PostgreSQL:** registry.redhat.io/rhel8/postgresql-13:1-109
- **Redis:** registry.redhat.io/rhel8/redis-6:1-110)

3.10. UPGRADE TO 3.6.Z FROM 3.4.Z



NOTE

Red Hat Quay 3.6 supports direct, single-step upgrade from 3.4.z. This exception to the normal, prior minor version-only, upgrade simplifies the upgrade procedure for customers on older releases.

Upgrading to Red Hat Quay 3.6 from 3.4.z requires a database migration which does not support downgrading back to a prior version of Red Hat Quay. Please back up your database before performing this migration.

Users will also need to configure a completely new Clair v4 instance to replace the old Clair v2 when upgrading from 3.4.z. For instructions on configuring Clair v4, see [Setting up Clair on a non-OpenShift Red Hat Quay deployment](#).

3.10.1. Target images

- **Quay:** registry.redhat.io/quay/quay-rhel8:v3.6.0
- **Clair:** registry.redhat.io/quay/clair-rhel8:v3.6.0
- **PostgreSQL:** registry.redhat.io/rhel8/postgresql-13:1-109
- **Redis:** registry.redhat.io/rhel8/redis-6:1-110)

3.11. UPGRADE TO 3.6.Z FROM 3.3.Z



NOTE

Red Hat Quay 3.6 supports direct, single-step upgrade from 3.3.z. This exception to the normal, prior minor version-only, upgrade simplifies the upgrade procedure for customers on older releases.

Upgrading to Red Hat Quay 3.6.z from 3.3.z requires a database migration which does not support downgrading back to a prior version of Red Hat Quay. Please back up your database before performing this migration.

Users will also need to configure a completely new Clair v4 instance to replace the old Clair v2 when upgrading from 3.3.z. For instructions on configuring Clair v4, see [Setting up Clair on a non-OpenShift Red Hat Quay deployment](#).

3.11.1. Target images

- **Quay:** registry.redhat.io/quay/quay-rhel8:v3.6.0
- **Clair:** registry.redhat.io/quay/clair-rhel8:v3.6.0
- **PostgreSQL:** registry.redhat.io/rhel8/postgresql-13:1-109
- **Redis:** registry.redhat.io/rhel8/redis-6:1-110)

3.11.2. Swift configuration when upgrading from 3.3.z to 3.6

When upgrading from Red Hat Quay 3.3.z to 3.6.z, some users might receive the following error: **Switch auth v3 requires tenant_id (string) in os_options**. As a workaround, you can manually update your **DISTRIBUTED_STORAGE_CONFIG** to add the **os_options** and **tenant_id** parameters:

```
DISTRIBUTED_STORAGE_CONFIG:
  brscale:
  - SwiftStorage
  - auth_url: http://****/v3
    auth_version: "3"
    os_options:
      tenant_id: ****
      project_name: ocp-base
      user_domain_name: Default
    storage_path: /datastorage/registry
    swift_container: ocp-svc-quay-ha
    swift_password: *****
    swift_user: *****
```

3.12. UPGRADE TO 3.5.7 FROM 3.4.Z

3.12.1. Target images

- **Quay:** registry.redhat.io/quay/quay-rhel8:v3.5.7
- **Clair:** registry.redhat.io/quay/clair-rhel8:v3.9.0
- **PostgreSQL:** registry.redhat.io/rhel8/postgresql-13:1-109
- **Redis:** registry.redhat.io/rhel8/redis-6:1-110)

3.13. UPGRADE TO 3.4.6 FROM 3.3.Z

Upgrading to Quay 3.4 requires a database migration which does not support downgrading back to a prior version of Quay. Please back up your database before performing this migration.

3.13.1. Target images

- **Quay:** registry.redhat.io/quay/quay-rhel8:v3.4.6
- **Clair:** registry.redhat.io/quay/clair-rhel8:v3.9.0
- **PostgreSQL:** registry.redhat.io/rhel8/postgresql-13:1-109
- **Redis:** registry.redhat.io/rhel8/redis-6:1-110)

3.14. UPGRADE TO 3.3.4 FROM 3.2.Z

3.14.1. Target images

- **Quay:** quay.io/redhat/quay:v3.3.4
- **Clair:** registry.redhat.io/quay/clair-rhel8:v3.9.0

- **PostgreSQL:** rhsc1/postgresql-96-rhel7
- **Redis:** registry.access.redhat.com/rhsc1/redis-32-rhel7

3.15. UPGRADE TO 3.2.2 FROM 3.1.Z

Once your cluster is running any Red Hat Quay 3.1.z version, to upgrade your cluster to 3.2.2 you must bring down your entire cluster and make a small change to the configuration before bringing it back up with the 3.2.2 version.



WARNING

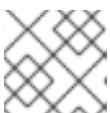
Once you set the value of `DATABASE_SECRET_KEY` in this procedure, do not ever change it. If you do so, then existing robot accounts, API tokens, etc. cannot be used anymore. You would have to create a new robot account and API tokens to use with Quay.

1. Take all hosts in the Red Hat Quay cluster out of service.
2. Generate some random data to use as a database secret key. For example:

```
$ openssl rand -hex 48
2d023adb9c477305348490aa0fd9c
```

3. Add a new `DATABASE_SECRET_KEY` field to your **config.yaml** file. For example:

```
DATABASE_SECRET_KEY: "2d023adb9c477305348490aa0fd9c"
```



NOTE

For an OpenShift installation, the **config.yaml** file is stored as a secret.

4. Bring up one **Quay** container to complete the migration to 3.2.2.
5. Once the migration is done, make sure the same **config.yaml** is available on all nodes and bring up the new quay 3.2.2 service on those nodes.
6. Start 3.0.z versions of quay-builder and Clair to replace any instances of those containers you want to return to your cluster.

3.15.1. Target images

- **Quay:** quay.io/redhat/quay:v3.2.2
- **Clair:** registry.redhat.io/quay/clair-rhel8:v3.9.0
- **PostgreSQL:** rhsc1/postgresql-96-rhel7
- **Redis:** registry.access.redhat.com/rhsc1/redis-32-rhel7

3.16. UPGRADE TO 3.1.3 FROM 3.0.Z

3.16.1. Target images

- **Quay:** quay.io/redhat/quay:v3.1.3
- **Clair:** registry.redhat.io/quay/clair-rhel8:v3.9.0
- **PostgreSQL:** rhsc/postgresql-96-rhel7
- **Redis:** registry.access.redhat.com/rhsc/redis-32-rhel7

3.17. UPGRADE TO 3.0.5 FROM 2.9.5

For the 2.9.5 to 3.0.5 upgrade, you can either do the whole upgrade with Red Hat Quay down (synchronous upgrade) or only bring down Red Hat Quay for a few minutes and have the bulk of the upgrade continue with Red Hat Quay running (background upgrade).

A background upgrade could take longer to run the upgrade depending on how many tags need to be processed. However, there is less total downtime. The downside of a background upgrade is that you will not have access to the latest features until the upgrade completes. The cluster runs from the Quay v3 container in v2 compatibility mode until the upgrade is complete.

3.17.1. Overview of upgrade

Follow the procedure below if you are starting with a Red Hat Quay 2.y.z cluster. Before upgrading to the latest Red Hat Quay 3.x version, you must first migrate that cluster to 3.0.5, as described [here](#). Once your cluster is running 3.0.5, you can then upgrade to the latest 3.x version by sequentially upgrading to each minor version in turn. For example:

1. 3.0.5 → 3.1.3
2. 3.1.3 → 3.2.2
3. 3.2.2 → 3.3.4
4. 3.3.4 → 3.4.z

Before beginning your Red Hat Quay 2.y.z to 3.0 upgrade, please note the following:

- **Synchronous upgrade:** For a synchronous upgrade, expect less than one hour of total downtime for small installations. Consider a small installation to contain a few thousand container image tags or fewer. For that size installation, you could probably get by with just a couple hours of scheduled downtime. The entire Red Hat Quay service is down for the duration, so if you were to try a synchronous upgrade on a registry with millions of tags, you could potentially be down for several days.
- **Background upgrade:** For a background upgrade (also called a compatibility mode upgrade), after a short shutdown your Red Hat Quay cluster upgrade runs in the background. For large Red Hat Quay registries, this could take weeks to complete, but the cluster continues to operate in v2 mode for the duration of the upgrade. As a point of reference, one Red Hat Quay v3 upgrade took four days to process approximately 30 million tags across six machines.
- **Full features on completion:** Before you have access to features associated with Docker version 2, schema 2 changes (such as support for containers of different architectures), the

entire migration must complete. Other v3 features are immediately available when you switch over.

- **Upgrade complete:** When the upgrade is complete, you need to set **V3_UPGRADE_MODE: complete** in the Red Hat Quay **config.yaml** file for the new features to be available. All new Red Hat Quay v3 installations automatically have that set.

3.17.2. Prerequisites

To assure the best results, we recommend the following prerequisites:

- Back up your Red Hat Quay database before starting the upgrade (doing regular backups is a general best practice). A good time to do this is right after you have taken down the Red Hat Quay cluster to do the upgrade.
- Back up your storage (also a general best practice).
- Upgrade your current Red Hat Quay 2.y.z setup to the latest 2.9.z version (currently 2.9.5) before starting the v3 upgrade. To do that:
 - While the Red Hat Quay cluster is still running, take one node and change the **Quay** container on that system to a **Quay** container that is running the latest 2.9.z version.
 - Wait for all the database migrations to run, bringing the database up to the latest 2.9.z version. This should only take a few minutes to a half an hour.
 - Once that is done, replace the **Quay** container on all the existing nodes with the same latest 2.9.z version. With the entire Red Hat Quay cluster on the new version, you can proceed to the v3 upgrade.

3.17.3. Choosing upgrade type

Choose between a synchronous upgrade (complete the upgrade in downtime) and a background upgrade (complete the upgrade while Red Hat Quay is still running). Both of these major-release upgrades require that the Red Hat Quay cluster be down for at least a short period of time.

Regardless of which upgrade type you choose, during the time that the Red Hat Quay cluster is down, if you are using builder and Clair images, you need to also upgrade to those new images:

- **Builder:** quay.io/redhat/quay-builder:v3.0.5
- **Clair:** quay.io/redhat/clair-jwt:v3.0.5

Both of those images are available from the registry.redhat.io/quay repository.

3.17.4. Running a synchronous upgrade

To run a synchronous upgrade, where your whole cluster is down for the entire upgrade, do the following:

1. Take down your entire Red Hat Quay cluster, including any quay-builder and Clair containers.
2. Add the following setting to the **config.yaml** file on all nodes:
V3_UPGRADE_MODE: complete
3. Pull and start up the v3 container on a single node and wait for however long it takes to do the upgrade (it will take a few minutes). Use the following container or later:

- **Quay:** quay.io/redhat/quay:v3.0.5

Note that the **Quay** container comes up on ports 8080 and 8443 for Red Hat Quay 3, instead of 80 and 443, as they did for Red Hat Quay 2. Therefore, we recommend remapping 8080 and 8443 into 80 and 443, respectively, as shown in this example:

```
# docker run --restart=always -p 80:8080 -p 443:8443 \
--sysctl net.core.somaxconn=4096 \
--privileged=true \
-v /mnt/quay/config:/conf/stack:Z \
-v /mnt/quay/storage:/datastorage:Z \
-d quay.io/redhat/quay:v3.0.5
```

4. After the upgrade completes, bring the Red Hat Quay 3 container up on all other nodes.
5. Start 3.0.z versions of quay-builder and Clair to replace any instances of those containers you want to return to your cluster.
6. Verify that Red Hat Quay is working, including pushes and pulls of containers compatible with Docker version 2, schema 2. This can include windows container images and images of different computer architectures (arm, ppc, etc.).

3.17.5. Running a background upgrade

To run a background upgrade, you need only bring down your cluster for a short period of time on two occasions. When you bring the cluster back up after the first downtime, the quay v3 container runs in v2 compatibility mode as it backfills the database. This background process can take hours or even days to complete. Background upgrades are recommended for large installations where downtime of more than a few hours would be a problem.

For this type of upgrade, you put Red Hat Quay into a compatibility mode, where you have a **Quay 3** container running, but it is running on the old data model while the upgrade completes. Here's what you do:

1. Pull the Red Hat Quay 3 container to all the nodes. Use the following container or later: quay.io/redhat/quay:v3.0.5
2. Take down your entire Red Hat Quay cluster, including any quay-builder and Clair containers.
3. Edit the **config.yaml** file on each node and set the upgrade mode to background as follows: V3_UPGRADE_MODE: background
4. Bring the Red Hat Quay 3 container up on a single node and wait for the migrations to complete (should take a few minutes maximum). Here is an example of that command:
Note that the **Quay** container comes up on ports 8080 and 8443 for Red Hat Quay 3, instead of 80 and 443, as they did for Red Hat Quay 2. Therefore, we recommend remapping 8080 and 8443 into 80 and 443, respectively, as shown in this example:

```
# docker run --restart=always -p 80:8080 -p 443:8443 \
--sysctl net.core.somaxconn=4096 \
--privileged=true \
-v /mnt/quay/config:/conf/stack:Z \
-v /mnt/quay/storage:/datastorage:Z \
-d quay.io/redhat/quay:v3.0.5
```

5. Bring the Red Hat Quay 3 container up on all the other nodes.

6. Monitor the **/upgradeprogress** API endpoint until it reports done enough to move to the next step (the status reaches 99%). For example, view <https://myquay.example.com/upgradeprogress> or use some other tool to query the API.
7. Once the background process is far enough along you have to schedule another maintenance window.
8. During your scheduled maintenance, take the entire Red Hat Quay cluster down.
9. Edit the **config.yaml** file on each node and set the upgrade mode to **complete** as follows:

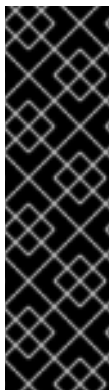

```
V3_UPGRADE_MODE: complete
```
10. Bring Red Hat Quay back up on one node to have it do a final check.
11. Once the final check is done, bring Red Hat Quay v3 back up on all the other nodes.
12. Start 3.0.z versions of quay-builder and Clair to replace any instances of those containers you want to return to your cluster.
13. Verify Quay is working, including pushes and pulls of containers compatible with Docker version 2, schema 2. This can include windows container images and images of different computer architectures (arm, ppc, etc.).

3.17.6. Target images

- **Quay:** quay.io/redhat/quay:v3.0.5
- **Clair:** quay.io/redhat/clair-jwt:v3.0.5
- **Redis:** registry.access.redhat.com/rhsc/redis-32-rhel7
- **PostgreSQL:** rhsc/postgresql-96-rhel7
- **Builder:** quay.io/redhat/quay-builder:v3.0.5

3.18. UPGRADING A GEO-REPLICATION DEPLOYMENT OF RED HAT QUAY

Use the following procedure to upgrade your geo-replication Red Hat Quay deployment.



IMPORTANT

- When upgrading geo-replication Red Hat Quay deployments to the next y-stream release (for example, Red Hat Quay 3.7 → Red Hat Quay 3.8), or geo-replication deployments, you must stop operations before upgrading.
- There is intermittent downtime down upgrading from one y-stream release to the next.
- It is highly recommended to back up your Red Hat Quay deployment before upgrading.

Prerequisites

- You have logged into **registry.redhat.io**



PROCEDURE

This procedure assumes that you are running Red Hat Quay services on three (or more) systems. For more information, see [Preparing for Red Hat Quay high availability](#).

1. Obtain a list of all Red Hat Quay instances on each system running a Red Hat Quay instance.

- a. Enter the following command on System A to reveal the Red Hat Quay instances:

```
$ sudo podman ps
```

Example output

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. Enter the following command on System B to reveal the Red Hat Quay instances:

```
$ sudo podman ps
```

Example output

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. Enter the following command on System C to reveal the Red Hat Quay instances:

```
$ sudo podman ps
```

Example output

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. Temporarily shut down all Red Hat Quay instances on each system.

- a. Enter the following command on System A to shut down the Red Hat Quay instance:

```
$ sudo podman stop ec16ece208c0
```

- b. Enter the following command on System B to shut down the Red Hat Quay instance:

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

- c. Enter the following command on System C to shut down the Red Hat Quay instance:

```
$ sudo podman stop e75c4aebfee9
```

3. Obtain the latest Red Hat Quay version, for example, Red Hat Quay 3, on each system.

- a. Enter the following command on System A to obtain the latest Red Hat Quay version:

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

- b. Enter the following command on System B to obtain the latest Red Hat Quay version:

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

- c. Enter the following command on System C to obtain the latest Red Hat Quay version:

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

4. On System A of your highly available Red Hat Quay deployment, run the new image version, for example, 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. Wait for the new Red Hat Quay container to become fully operational on System A. You can check the status of the container by entering the following command:

```
$ sudo podman ps
```

Example output

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS
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. Optional: Ensure that Red Hat Quay is fully operation by navigating to the Red Hat Quay UI.
7. After ensuring that Red Hat Quay on System A is fully operational, run the new image versions on System B and on System C.
- a. On System B of your highly available Red Hat Quay deployment, run the new image version, for example, 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. On System C of your highly available Red Hat Quay deployment, run the new image version, for example, 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. You can check the status of the containers on System B and on System C by entering the following command:

```
$ sudo podman ps
```

CHAPTER 4. UPGRADE QUAY BRIDGE OPERATOR

To upgrade the Quay Bridge Operator (QBO), change the Channel Subscription update channel in the Subscription tab to the desired channel.

When upgrading QBO from version 3.5 to 3.7, a number of extra steps are required:

1. You need to create a new **QuayIntegration** custom resource. This can be completed in the Web Console or from the command line.

upgrade-quay-integration.yaml

```
- apiVersion: quay.redhat.com/v1
  kind: QuayIntegration
  metadata:
    name: example-quayintegration-new
  spec:
    clusterID: openshift 1
    credentialsSecret:
      name: quay-integration
      namespace: openshift-operators
    insecureRegistry: false
    quayHostname: https://registry-quay-quay35.router-default.apps.cluster.openshift.com
```

- 1** Make sure that the **clusterID** matches the value for the existing **QuayIntegration** resource.

2. Create the new **QuayIntegration** custom resource:

```
$ oc create -f upgrade-quay-integration.yaml
```

3. Delete the old **QuayIntegration** custom resource.
4. Delete the old **mutatingwebhookconfigurations**:

```
$ oc delete mutatingwebhookconfigurations.admissionregistration.k8s.io quay-bridge-operator
```

4.1. UPGRADING A GEO-REPLICATION DEPLOYMENT OF THE RED HAT QUAY OPERATOR

Use the following procedure to upgrade your geo-replicated Red Hat Quay Operator.



IMPORTANT

- When upgrading geo-replicated Red Hat Quay Operator deployments to the next y-stream release (for example, Red Hat Quay 3.7 → Red Hat Quay 3.8), you must stop operations before upgrading.
- There is intermittent downtime down upgrading from one y-stream release to the next.
- It is highly recommended to back up your Red Hat Quay Operator deployment before upgrading.



PROCEDURE

This procedure assumes that you are running the Red Hat Quay Operator on three (or more) systems. For this procedure, we will assume three systems named **System A**, **System B**, and **System C**. **System A** will serve as the primary system in which the Red Hat Quay Operator is deployed.

1. On System B and System C, scale down your Red Hat Quay Operator deployment. This is done by disabling auto scaling and overriding the replica count for Red Hat Quay, mirror workers, and Clair (if it is managed). Use the following **quayregistry.yaml** file as a reference:

```
apiVersion: quay.redhat.com/v1
kind: QuayRegistry
metadata:
  name: registry
  namespace: ns
spec:
  components:
    ...
    - kind: horizontalpodautoscaler
      managed: false 1
    - kind: quay
      managed: true
      overrides: 2
        replicas: 0
    - kind: clair
      managed: true
      overrides:
        replicas: 0
    - kind: mirror
      managed: true
      overrides:
        replicas: 0
    ...
```

1 Disable auto scaling of Quay, Clair and Mirroring workers

2 Set the replica count to 0 for components accessing the database and objectstorage

**NOTE**

You must keep the Red Hat Quay Operator running on System A. Do not update the **quayregistry.yaml** file on System A.

- Wait for the **registry-quay-app**, **registry-quay-mirror**, and **registry-clair-app** pods to disappear. Enter the following command to check their status:

```
oc get pods -n <quay-namespace>
```

Example output

```
quay-operator.v3.7.1-6f9d859bd-p5ftc      1/1   Running   0      12m
quayregistry-clair-postgres-7487f5bd86-xnxpr 1/1   Running   1 (12m ago) 12m
quayregistry-quay-app-upgrade-xq2v6        0/1   Completed 0      12m
quayregistry-quay-config-editor-6dfdcfc44f-hlvwm 1/1   Running   0      73s
quayregistry-quay-redis-84f888776f-hhgms    1/1   Running   0      12m
```

- On System A, initiate a Red Hat Quay Operator upgrade to the latest y-stream version. This is a manual process. For more information about upgrading installed Operators, see [Upgrading installed Operators](#). For more information about Red Hat Quay upgrade paths, see [Upgrading the Red Hat Quay Operator](#).
- After the new Red Hat Quay Operator is installed, the necessary upgrades on the cluster are automatically completed. Afterwards, new Red Hat Quay pods are started with the latest y-stream version. Additionally, new **Quay** pods are scheduled and started.
- Confirm that the update has properly worked by navigating to the Red Hat Quay UI:
 - In the **OpenShift** console, navigate to **Operators → Installed Operators**, and click the **Registry Endpoint** link.

**IMPORTANT**

Do not execute the following step until the Red Hat Quay UI is available. Do not upgrade the Red Hat Quay Operator on System B and on System C until the UI is available on System A.

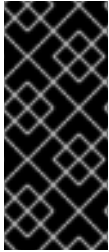
- After confirming that the update has properly worked on System A, initiate the Red Hat Quay Operator on System B and on System C. The Operator upgrade results in an upgraded Red Hat Quay installation, and the pods are restarted.

**NOTE**

Because the database schema is correct for the new y-stream installation, the new pods on System B and on System C should quickly start.

CHAPTER 5. DOWNGRADING RED HAT QUAY

Red Hat Quay only supports rolling back, or downgrading, to previous z-stream versions, for example, 3.7.2 → 3.7.1. Rolling back to previous y-stream versions (3.7.0 → 3.6.0) is not supported. This is because Red Hat Quay updates might contain database schema upgrades that are applied when upgrading to a new version of Red Hat Quay. Database schema upgrades are not considered backwards compatible.



IMPORTANT

Downgrading to previous z-streams is neither recommended nor supported by either Operator based deployments or virtual machine based deployments. Downgrading should only be done in extreme circumstances. The decision to rollback your Red Hat Quay deployment must be made in conjunction with the Red Hat Quay support and development teams. For more information, contact Red Hat Quay support.