



Red Hat Quay: Technical Overview and Roadmap

Laine Vyvyan
Channel Solutions Architect

Joshua Smith
Cloud Solutions Architect

Dirk Hermann
Senior Principal Product Manager, Quay

If you're doing containers and Kubernetes,
you need a container image registry

**Just like
money,
containers
need safe and
reliable storage
and
transmission**



The background image is an aerial photograph of a dense urban area, likely a financial district, featuring numerous skyscrapers of varying heights and architectural styles. A major highway or bridge structure is visible in the upper right corner. In the foreground, the words "H THIRD BANK" are partially visible on the side of a building, and a "usbank" logo is prominently displayed on the roof of a building in the center-right.

Core Services of a Container Registry

**Pushes (Deposits),
Pulls (Withdrawals),
and *Security***



What is Quay?

Massive Scale Testing Quay.io
Real Time Garbage Collection
Automated Squashing

SCALABILITY

Seamless Git Integration
Build Workers
Webhooks

BUILD AUTOMATION

Extensible API
Webhooks, OAuth
Robot Accounts

INTEGRATION

ENTERPRISE REGISTRY

High Availability
Full Standards / Spec Support
Long-Term Protocol Support
Application Registry
Enterprise Grade Support
Regular Updates

SECURITY

Vulnerability Scanning
Logging & Auditing
Notifications & Alerting

CONTENT DISTRIBUTION

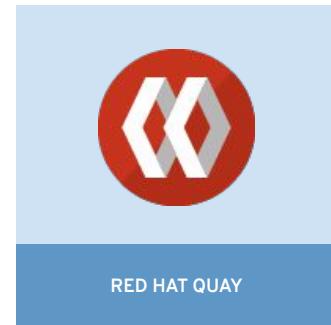
Geo-Replication
Repository Mirroring
Air-Gapped Environments

ACCESS CONTROL

Authentication Providers
Fine-Grained RBAC
Organizations & Teams

What is Quay?

- Enterprise container registry
- Available on-premise, on public cloud, and as a hosted service (SaaS)
- Key strengths:
 - Security
 - Robustness & speed
 - Automation
- Quay works with any container environment or orchestration platform

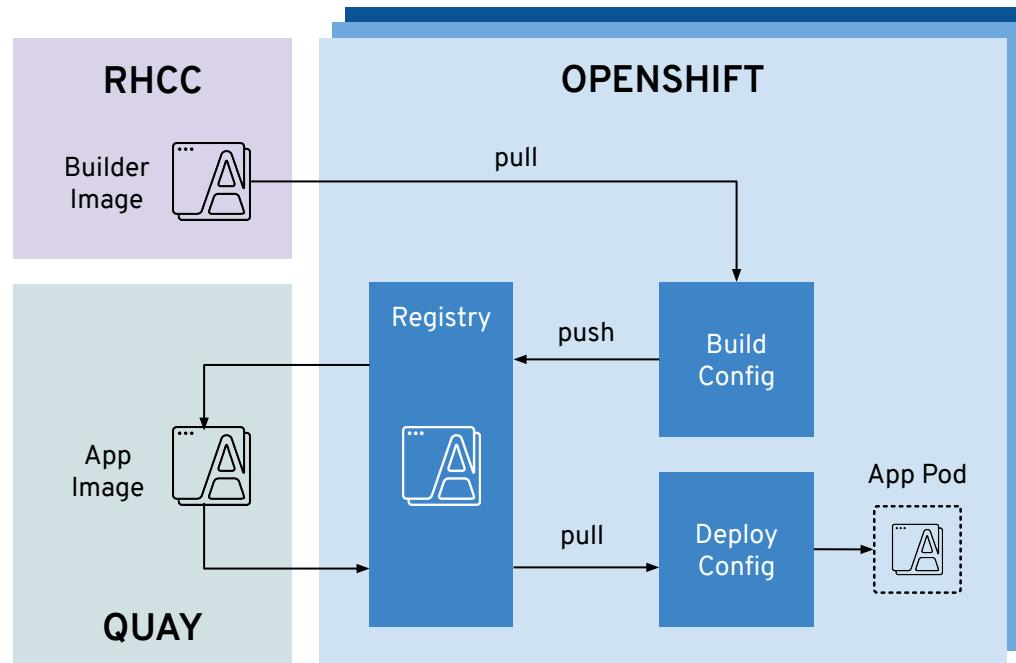


First hosted registry in the market with private repos

One of the top 5 biggest hosted registries overall

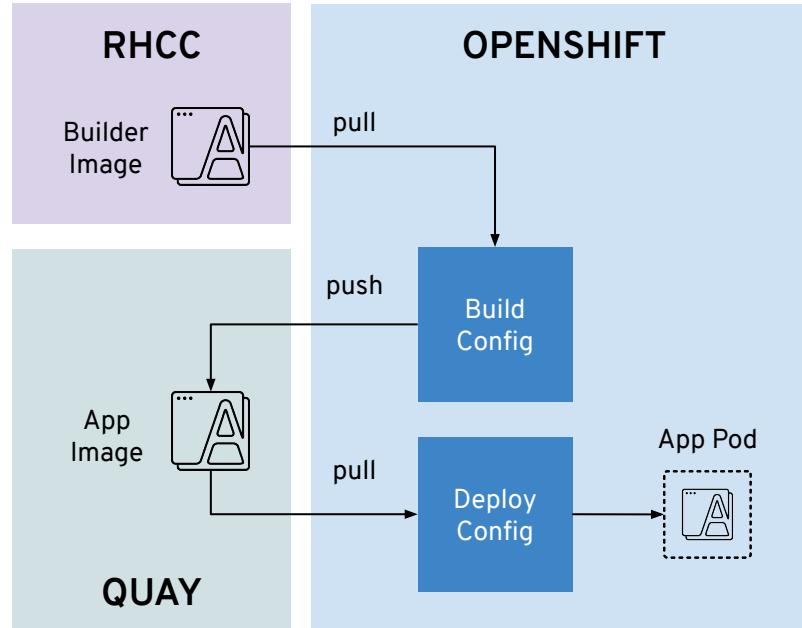
Quay as an Upstream Registry with OpenShift

- Images **pulled** from Quay into the integrated OpenShift registry
- Images are **pushed** to the integrated OpenShift registry, and synced externally with Quay



Quay as the OpenShift Registry

- Images are pushed directly by builds to Quay
- Images are pulled directly from Quay



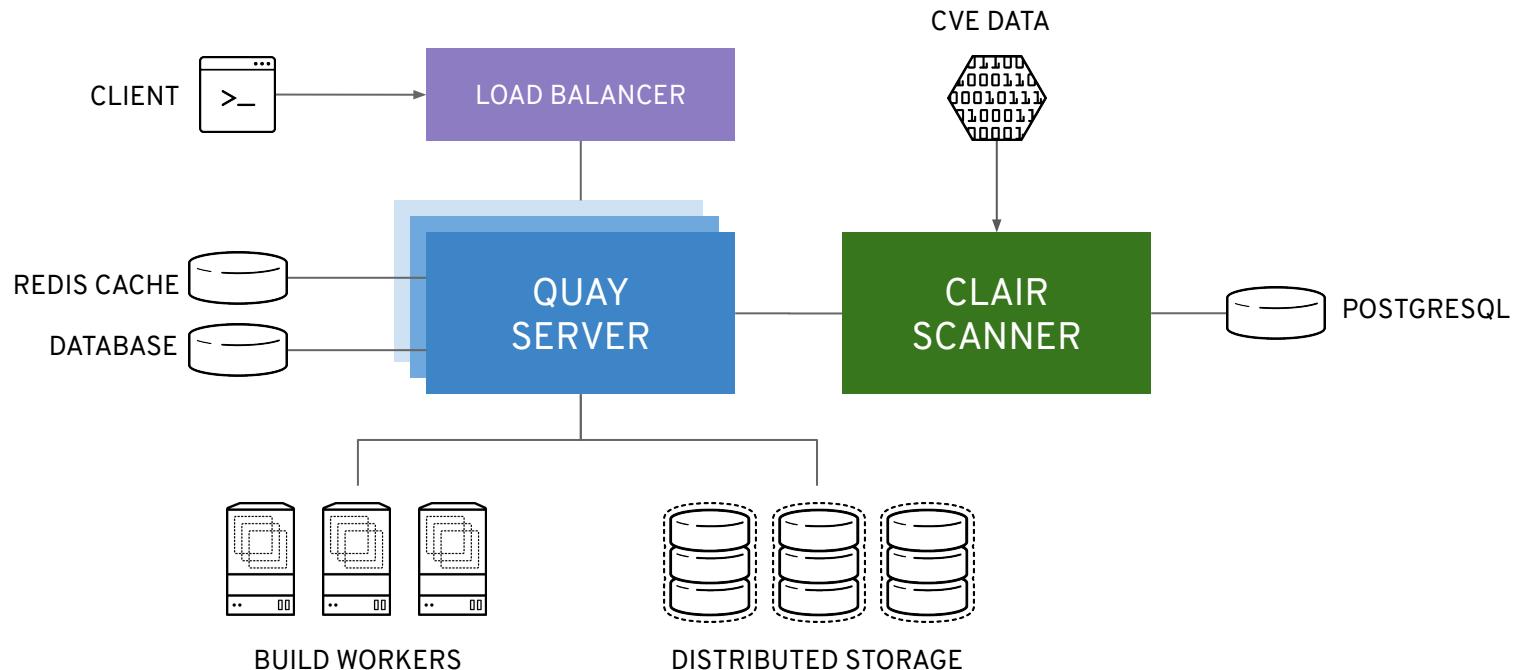
Common Quay Use Cases

- Large-scale and distributed environments (thousands of users and images)
- Images shared across multiple OpenShift clusters
- OpenShift clusters in multiple geographical regions
- Governance/security of container images
- High image maintenance and automation requirements
- Large number of build and high requirements on image delivery throughput

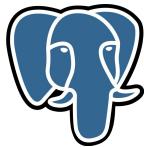


Red Hat Quay Architecture

Red Hat Quay Architecture



Prerequisite #1: The Database



- **PostgreSQL** is the preferred database backend since it can be used for both Quay and Clair



- If Clair isn't used: Quay works fine with MySQL too (5.7+)



Google Cloud

- **If Quay runs on public cloud infrastructure** we recommend to use the PostgreSQL services provided by your cloud provider



- If the database runs on **OpenShift** we recommend to use an operator, such as Crunchy Data PostgreSQL Operator

Prerequisite #2: Storage Backend

Public Cloud Storage

- AWS S3
- Google Cloud Storage
- Azure Blob Storage



On-Prem Storage

- Ceph Rados RGW
- OpenStack Swift
- RHOCs3 (via NooBaa) (TP)
- RHOCs4 (via NooBaa)



Note: Local Storage for PoC's only, NFS not recommended

Prerequisite #3: Redis Cache

- Provided via Red Hat Software Collections but any other redis works, too
- Mostly used by builds, workers and tutorial
- Data stored is ephemeral in nature, Redis does **not** need to be HA.
- If Redis goes down you will lose access to:
 - Live build logs
 - Tutorial



Quay Setup Operator

The diagram illustrates the integration of the Quay Setup Operator with the Red Hat Container Platform and the Container Community of Practice.

Red Hat Container Platform: A screenshot of the OpenShift Container Platform interface showing the OperatorHub. It lists various operators, including the Quay operator, which is highlighted. The Quay operator page shows details such as its version (0.0.7), provider (Red Hat), and repository (<https://quay.io/operatorhub/quay-operator>). It also includes prerequisites, operator features (Automated installation of Red Hat Quay, Provision Database(s) to support relational database for Quay and optional component Clair, Installation of Clair for container scanning and integration with Quay), and a note about protected registries.

Container Community of Practice: A screenshot of the Container Community of Practice page on the Red Hat website. The page title is "Container Community of Practice". Below the title, there is a red banner with the text "All Places > Communities at Red Hat > Applications". The banner also features the Red Hat logo and the text "Container Community of Practice".

Quay Setup Operator: A central circular icon containing a server rack with a box labeled 'A' on top, representing the operator.

Key Features and Benefits:

- Automates the initial deployment of Quay and Clair
- Simplifies installation of Quay
- Configures all relevant OpenShift objects (routes, secrets, etc.)
- Aids in certificate management

Check out the Quay Setup Operator Demo: <https://youtu.be/TCDmylIt1Fns>

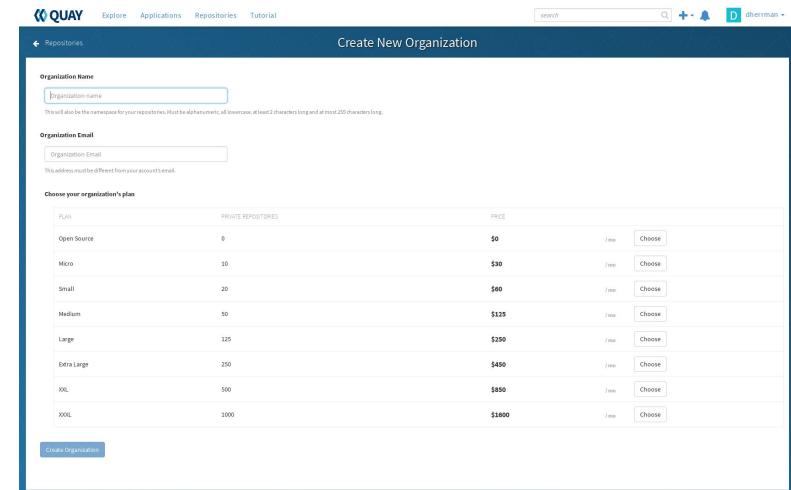




Authorization and Authentication

Organizations, Teams, Users, Robot Accounts

- Organizations
 - sharing repositories under a common namespace that belongs to many users
 - divided into **teams** which provide access to repositories under that namespace
- Teams
 - Provide a way for an organization to delegate permissions (both global and on specific repositories) to groups of users
 - Permissions: Member, Creator, Admin



Organizations, Teams, Users, Robot Accounts

- Users
 - Key element of setting repository permissions / RBAC
- Robot accounts
 - Allow for automatic software deployments
 - Can be shared by multiple repositories owned by a user or organization
 - Managed inside the organization view -> *Robot Accounts* tab

The screenshot shows the Quay interface. At the top, there's a navigation bar with 'Explore', 'Repositories', 'Tutorial', and a user dropdown. A modal window titled 'Create robot account' is open, prompting for a name ('jenkins_ci') and an optional description ('Jenkins CI Account'). Below the modal, the main organization view is visible, featuring a sidebar with icons for 'Explore', 'Repositories', 'Teams', 'Robot Accounts', 'Builds', 'Metrics', and 'Environments'. The main area is titled 'Robot Accounts' and contains a table with one entry:

ROBOT ACCOUNT NAME	DESCRIPTION	TEAMS	REPOSITORIES	CREATED	LAST ACCESSED
redhat+jenkins_ci	Jenkins CI Account	No teams	1 repository	a minute ago	Never

On the right side of the table, there are three buttons: 'View Credentials', 'Set Repository Permissions', and 'Delete Robot redhat+jenkins_ci'.

Enterprise Authorization and Authentication

Red Hat Quay allows you to integrate your existing identity infrastructure and use a fine-grained permissions system to map to your organizational structure and grant access to whole teams to manage specific repositories.

Supported auth providers:

- Built-in Database Auth
- LDAP auth and sync
- External OIDC provider
- OpenStack Keystone

The screenshot shows the 'Internal Authentication' section of the Red Hat Quay configuration. It includes a note about requiring encrypted client passwords and a warning about storing passwords in plaintext. Under 'Authentication', 'Local Database' is selected, while 'Keystone (OpenStack Identity)', 'JWT Custom Authentication', and 'External Application Token' are listed as options. Below this are fields for 'LDAP URI', 'Base DN', and 'User Relative DN'. The 'External Authorization (OAuth)' section contains a 'GitHub (Enterprise) Authentication' subsection with a note about GitHub OAuth applications and a 'Enable GitHub Authentication' checkbox. The 'Google Authentication' section has a similar structure. At the bottom are buttons for 'Add OIDC Provider' and 'What is OIDC?'



Note: Auth integration for OCP coming soon (next slide...).

Quay and OpenShift Mapping (coming soon)

- Each OpenShift **namespace** results in an **organization** within Quay
- Each **ImageStream** within an namespace results in a new **repository** within Quay
- The three primary service accounts (Builder, Deployer and Default) result in **Robot accounts** in each organization with the following permissions
 - Builder - Write
 - Default - Read
 - Deployer - Read
- **Secrets** from each Robot account within an org are created in each OpenShift namespace
- **Service accounts** are configured to leverage these secrets as mountable secrets in order to push images as well as pull secrets in order to push images

Early prototype already exists developed in collaboration with our Community of Practice

Repository Notifications

Quay triggers different notifications for various repository events.

E-mail, Quay UI notification, webhook POST (can be used to trigger other integrations - for example an OpenShift Pipeline or a Jenkins job), Flowdock, Hipchat, Slack notifications based on various events inside of Quay Enterprise.

How it Works:

- Events
- [Event threshold] (vulnerability: severity)
- Notification method

The screenshot shows the 'Create repository notification' page. At the top, there's a list of events: Push to Repository, Dockerfile Build Queued, Dockerfile Build Started, Dockerfile Build Successfully Completed, Dockerfile Build Failed, Docker Build Cancelled, and Package Vulnerability Found. Below this, there are two dropdown menus. The first dropdown, 'When this event occurs', is set to 'Push to Repository'. The second dropdown, 'Then issue a notification', has 'Please select a notification method' as its placeholder. A list of notification methods is visible: Red Hat Quay Notification, Webhook POST, Flowdock Team Notification, HipChat Room Notification, and Slack Room Notification. To the right of these dropdowns, there's explanatory text: 'Red Hat Quay supports a number of events around repositories (such as push completed), building (build queued, build completed, etc) and security (vulnerability detected). Some events also allow for filtering, for further granular control of when notifications fire.' At the bottom left is a 'Create Notification' button, and at the bottom right is another explanatory text block: 'Once an event has fired, Red Hat Quay supports a number of notification methods, including various chat systems (Slack, HipChat, etc), notification via e-mail or programmatic handling via the firing of a webhook.'



Image Management

Quay Build Triggers

Automatically build images on repository push actions with integration to GitHub, Bitbucket, and more.

How it Works:

- Must be enabled in Admin panel first
- Mounts in docker.socket
- Equivalent to running `docker build` cmd
- Robot accounts to lock down automated access and audit each deployment

Create Build Trigger ▾

- GitHub Repository Push
- Bitbucket Repository Push
- GitLab Repository Push
- Custom Git Repository Push

Configure Trigger

Configure trigger options for sir-rob/simplewebapp

- Trigger for all branches and tags (default)
Build a container image for each commit across all branches and tags
- Trigger only on branches and tags matching a regular expression
Only build container images for a subset of branches and/or tags.



Note: Docker version must be the same across all build nodes!

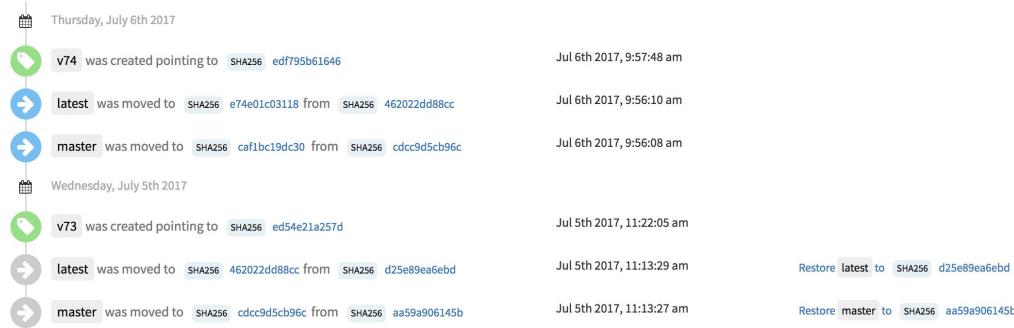
Note: If self-signed certs are used, the cert must be mounted in via the configuration directory!

Time Machine

Time Machine provides image rollback:
view history of tags for up to two
weeks and have the ability to revert
tags to a previous state.

How it Works:

- Select Repository → Click on Tags tab
- Add a new tag / move a tag / delete tag
- View tag history
- Click on Restore to roll back



Repository Tags

Compact Expanded

TAG	LAST MODIFIED	SECURITY SCAN	SIZE	IMAGE	
latest	16 hours ago	70 Medium + 10 fixable	711.0 MB	SHA256 9a347939468e	
master	16 hours ago	70 Medium + 10 fixable	711.0 MB	SHA256 014514e8ef9b	
dbb57f7	18 hours ago	70 Medium + 10 fixable	696.1 MB	SHA256 2592c71fe8f5	
3e28797	a day ago	75 Medium + 15 fixable	693.5 MB	SHA256 0d37d281173e	

Automatically Squash Images

Squash multiple docker layers into one in order to create an image with fewer and smaller layers - leading to reduced image size and faster fetch times.

How it Works:

- HTTP call triggers squashing and fetches squashed image
- Resulting squashed image is stored in Quay but you can't *docker pull* it

Fetch Tag:  v74

Image Format:  Squashed Docker Image

Pull Credentials:  sir_rob+saurabhsrobot

Command:
`> curl -L -f https://sir_rob+saurabhsrobot:5ASMRQ24SIRMR00140PESBYTVFLQNMZADK8PFNW80W456WEAI0:`



 Squashed Docker Image
 Rocket Fetch
 Basic Docker Pull



Note: Images provided by Red Hat are squashed as part of our internal image build factory.

Tag Expiration (Retention Policy)

Users can add a `quay.expires-after=20h` label via the Dockerfile `LABEL` command. The values can be things like 1h, 2d, 3w for hour, day and weeks, respectively

Additionally, there's a column in the UI that can be set entitled "Expires."

The tag/image version will be deleted from the repo when the expiration time is reached.

Repository Tags

TAG	LAST MODIFIED	SECURITY SCAN	SIZE	EXPIRES	MANIFEST
3.2.9-2	40 minutes ago	21 High · 78 fixable	85.7 MB	Never	SHA256 00903c1a6dfc
3.2.9	40 minutes ago	21 High · 78 fixable	85.7 MB	Never	SHA256 2fd44ebea0ba
3.2.7-2	40 minutes ago	21 High · 78 fixable	85.7 MB	Never	SHA256 2fae884bfac3
3.2.7	40 minutes ago	21 High · 78 fixable	85.7 MB	Never	SHA256 4d910c7a6581
3.2.5-2	40 minutes ago	24 High · 81 fixable	82.4 MB	Never	SHA256 e7dbc3be10ad
3.2.5	40 minutes ago	24 High · 81 fixable	82.4 MB	Never	SHA256 698f3850983c
3.2.26-36					

Change Tags Expiration

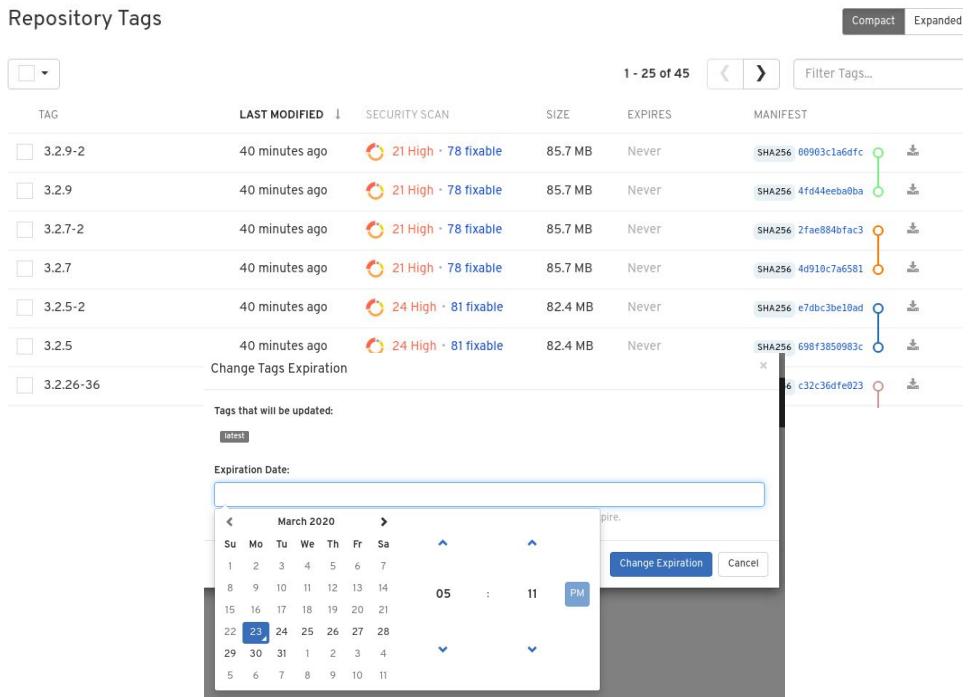
Tags that will be updated:

Expiration Date:

March 2020

Su	Mo	Tu	We	Th	Fr	Sa
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31	1	2	3	4
5	6	7	8	9	10	11

Change Expiration Cancel





Repository Mirroring

Repository Mirroring

This feature will convert [devtable/complex](#) into a mirror. Changes will be duplicated here. While enabled, users will be unable to...

External Repository

Registry URL	quay.io
User or Organization	coreos
Repository Name	etcd
Sync Interval	0
Robot User	

Credentials

Required if the external repository...

Username	
Password	

Advanced Settings

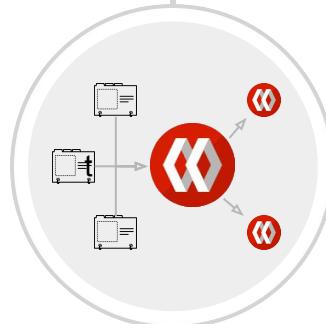
Start Date	July 31, 2019 12:07
------------	---------------------

Verify TLS	<input checked="" type="checkbox"/>
Require HTTPS and verify certificates when talking to the external registry.	

Proxy URL	
-----------	--

Tags	Comma-separated list of tag patterns to synchronize.
Examples: latest	

[Enable Mirror](#)



<https://github.com/containers/skopeo>

Repository Mirroring

Allows to continually synchronize repositories from external source registries into Quay (content ingress point)

Allows to mirror a subset of the entire registry content to distributed registry deployments

Filters could be applied to sync only a subset of a repository

New: API calls available to automate mirroring configuration

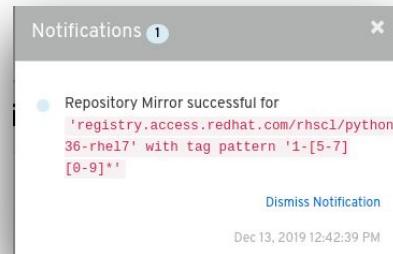
Introduced as Dev Preview in already in Quay 3.1 - now **GA**

Notifications

In addition to pre-existing notification events we added 3 repo mirroring related events which can be configured inside the Settings tab for each repository.

New Notification Events:

- Repository Mirror Started
- Repository Mirror Success
- Repository Mirror Unsuccessful

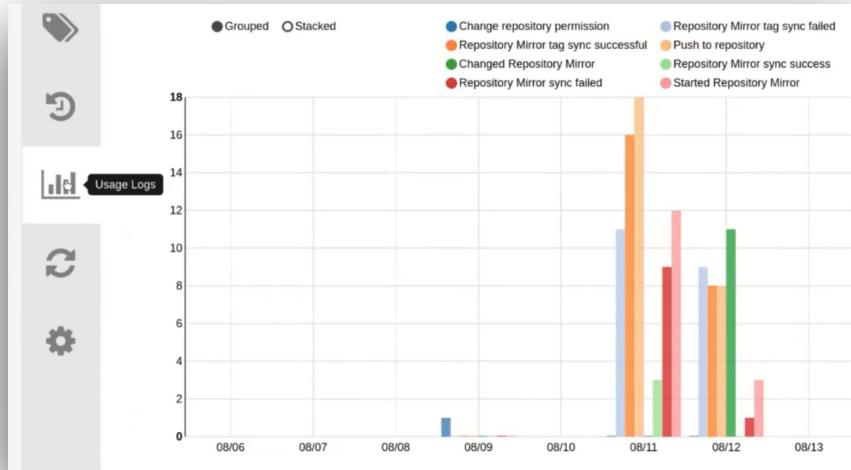
A screenshot of the Red Hat Quay interface. At the top, it shows the Red Hat logo, 'RED HAT® QUAY', 'EXPLORE', and 'REPOSITORIES'. Below that, it shows a repository path 'dherrman / python3-mi' and a 'Create repository' button. The main area is titled 'When this event occurs' with a dropdown menu set to 'Please select the event'. A list of notification types is shown in a dropdown menu:

- Push to Repository
- Repository Mirror Started
- Repository Mirror Success
- Repository Mirror Unsuccessful

30

 Red Hat

Logging and Auditing



> ○ Push of daily-2019-08-03	Tue, Aug 13, 2019 9:32 AM
> ● Mirror of daily-2019-08-02 successful	Tue, Aug 13, 2019 9:32 AM
> ○ Push of daily-2019-08-02	Tue, Aug 13, 2019 9:32 AM
> ● Mirror of daily-2019-08-01 successful	Tue, Aug 13, 2019 9:32 AM
> ○ Push of daily-2019-08-01	Tue, Aug 13, 2019 9:32 AM
> ● Mirror of daily successful	Tue, Aug 13, 2019 9:32 AM
> ○ Push of daily	Tue, Aug 13, 2019 9:32 AM
> ● Mirror started for 'quay.io/outline/shadowbox' with tag pattern 'daily,daily-2019-08-*'	Tue, Aug 13, 2019 9:32 AM
> ● Immediate mirror scheduled	Tue, Aug 13, 2019 9:32 AM
> ● Mirror of daily-2019-08-12 failure	Tue, Aug 13, 2019 9:30 AM
> ● Mirror of daily-2019-08-11 failure	Tue, Aug 13, 2019 9:30 AM

- Detailed auditing and logging including the skopeo log information

API Calls

mirror

GET /api/v1/repository/{repository}/mirror

PUT /api/v1/repository/{repository}/mirror

POST /api/v1/repository/{repository}/mirror

POST /api/v1/repository/{repository}/mirror/sync-now

POST /api/v1/repository/{repository}/mirror/sync-cancel

PUT /api/v1/repository/{repository}/mirror/rules

repository - The full path of the repository e.g. namespace/repo

repository - The full path of the repository e.g.

body - Request body contents.

Request Value - Model

```
{ "is_enabled": true, "external_registry_url": "Unknown Type: string.null", "external_registry_username": "Unknown Type: string.null", "external_registry_password": "Unknown Type: string.null", "internal_registry_name": "Unknown Type: string.null", "internal_registry_url": "Unknown Type: string.null", "internal_registry_username": "Unknown Type: string.null", "internal_registry_password": "Unknown Type: string.null", "rules": [ { "rule": "string", "action": "string", "order": 1 } ] }
```

Response Value - Model

Responses

Code Description

200 Successful invocation

400 Bad Request

401 Session required

- All repository mirroring configuration can be done via Quay API
- Documentation via Swagger UI (see Quay API section for further details)



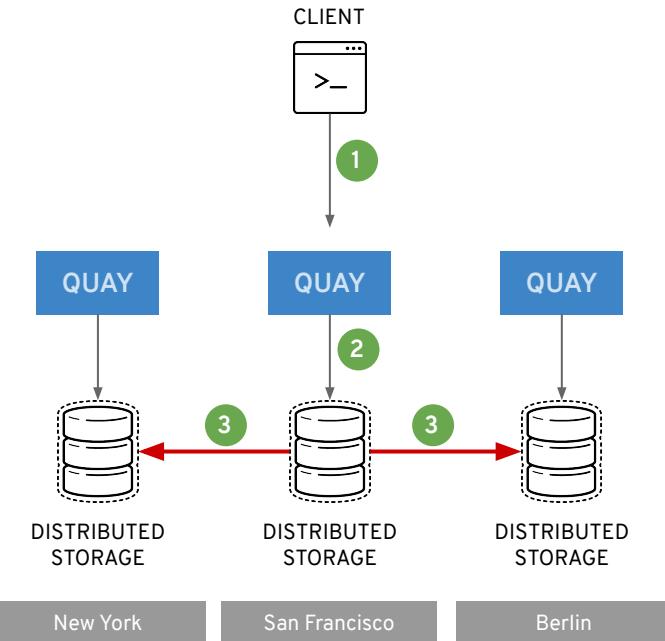
Geo-Replication

Quay Geo-Replication

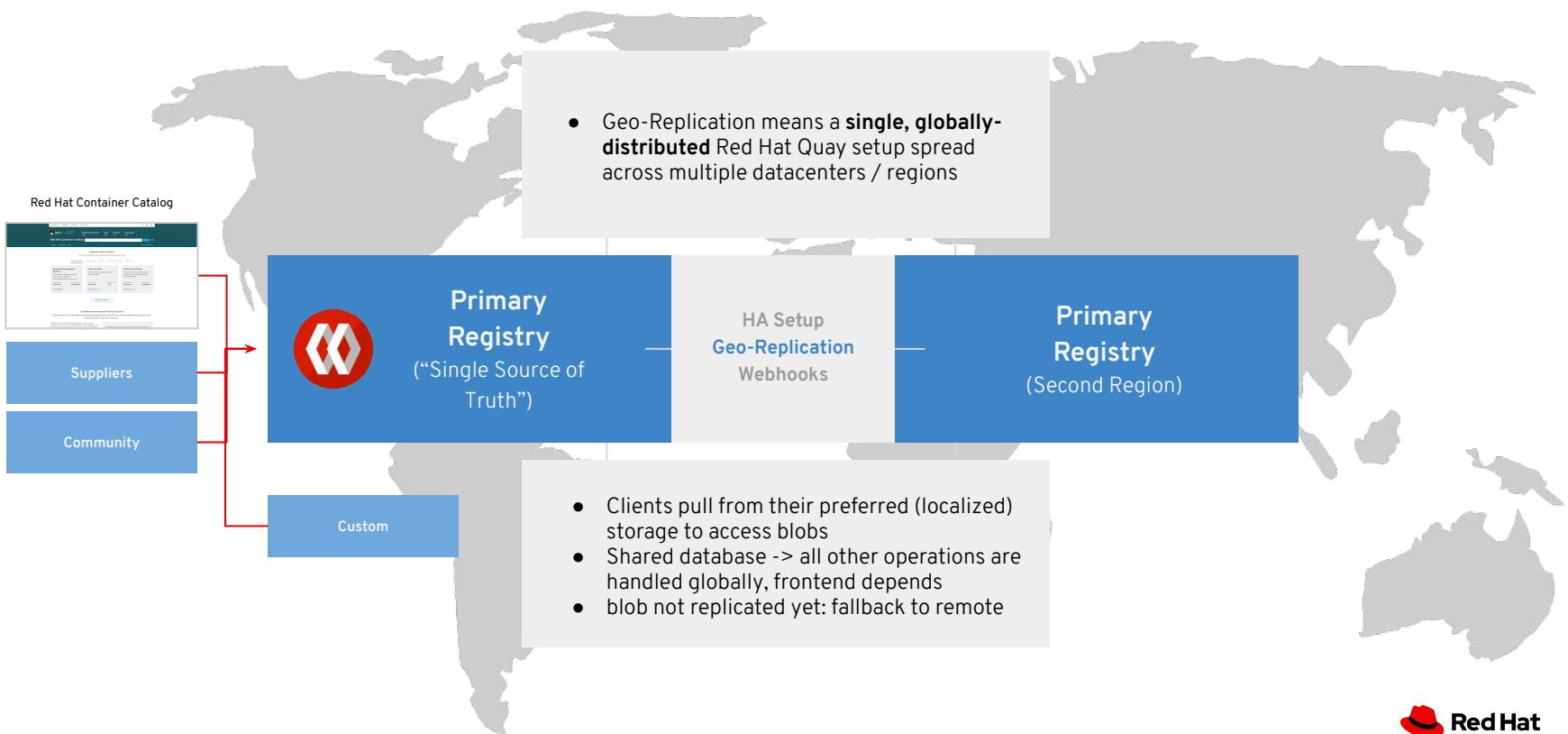
Geo-replication allows for a single globally-distributed Quay Enterprise to serve container images from localized storage.

How it Works:

- Image data will be asynchronously replicated in the background
- By default *all* images are replicated to *all* storage engines configured



Multi-Region Quay Setup



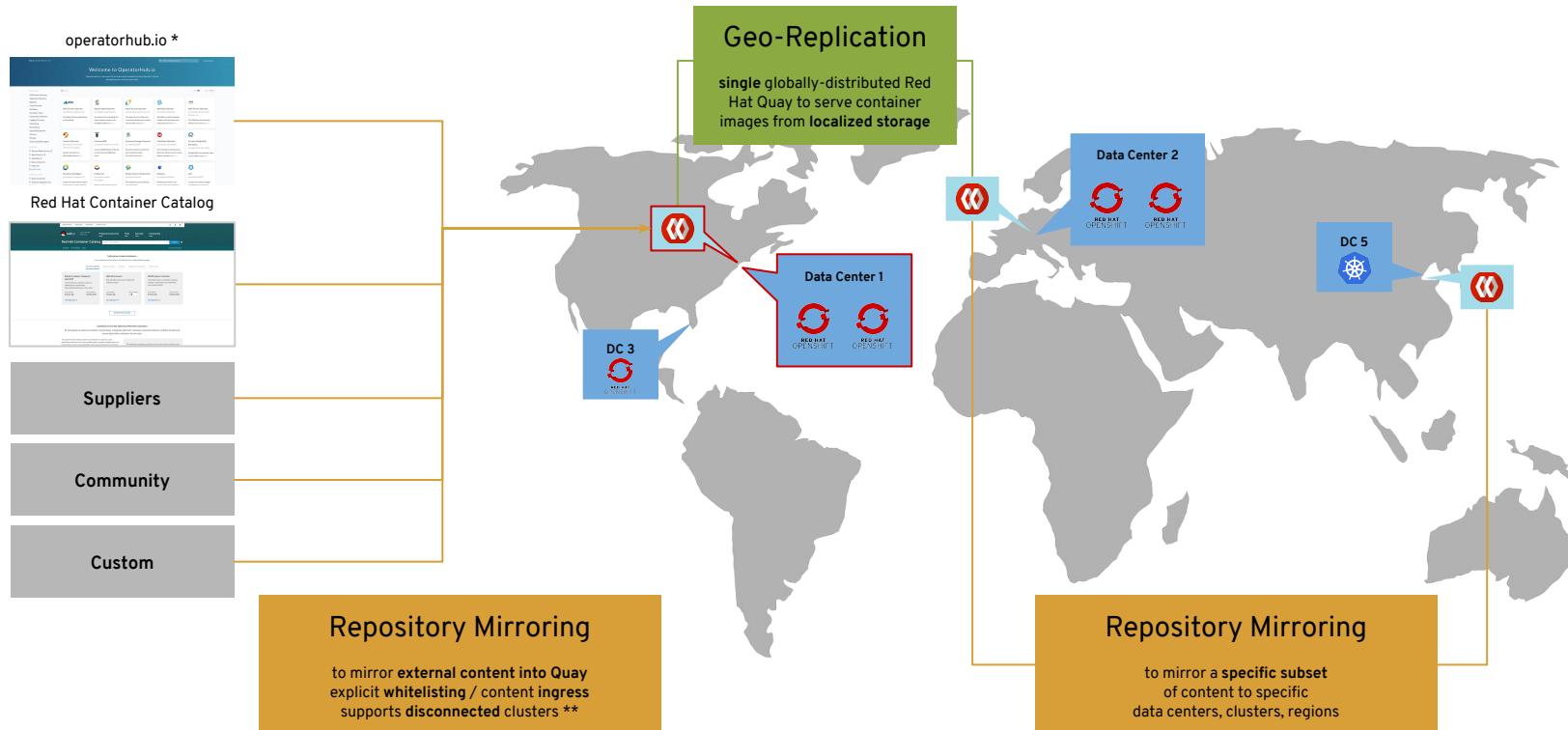


Repository Mirroring vs/and Geo-Replication

Quay Repository Mirroring vs. Geo-Replication

Feature / Capability	Geo-Replication	Repository Mirroring
Feature is designed for	a shared, global registry	Distinct, different registries
If replication / mirroring hasn't been completed yet then...	The remote copy is used (slower)	No image is served
Access to all storage backends in both regions required	Yes (all Quay nodes)	No (distinct storage)
Users can push images from both sites to the same repository	yes	no
whole registry content and configuration is identical across all regions (shared database)	yes	no
users can select individual namespaces / repositories to be mirrored	no	yes
users can apply filters (tag, tag range, etc.) to synchronization rules	no	yes
allows individual / different RBAC configurations in each region	no	yes

Content (Re)Distribution across the Globe



* Mirroring of operator repositories from operatorhub.io and expose them as a private catalog source to the embedded operatorhub in OpenShift planned for future releases of Red Hat Quay

** Quay connected but clusters only have access to Quay - full support for running Quay deployments in air-gapped environments planned for future releases of Red Hat Quay



High-Availability

Quay High-Availability Setup

High-availability reference architecture prevents critical single PoF by running multiple instances of Quay.

How it Works:

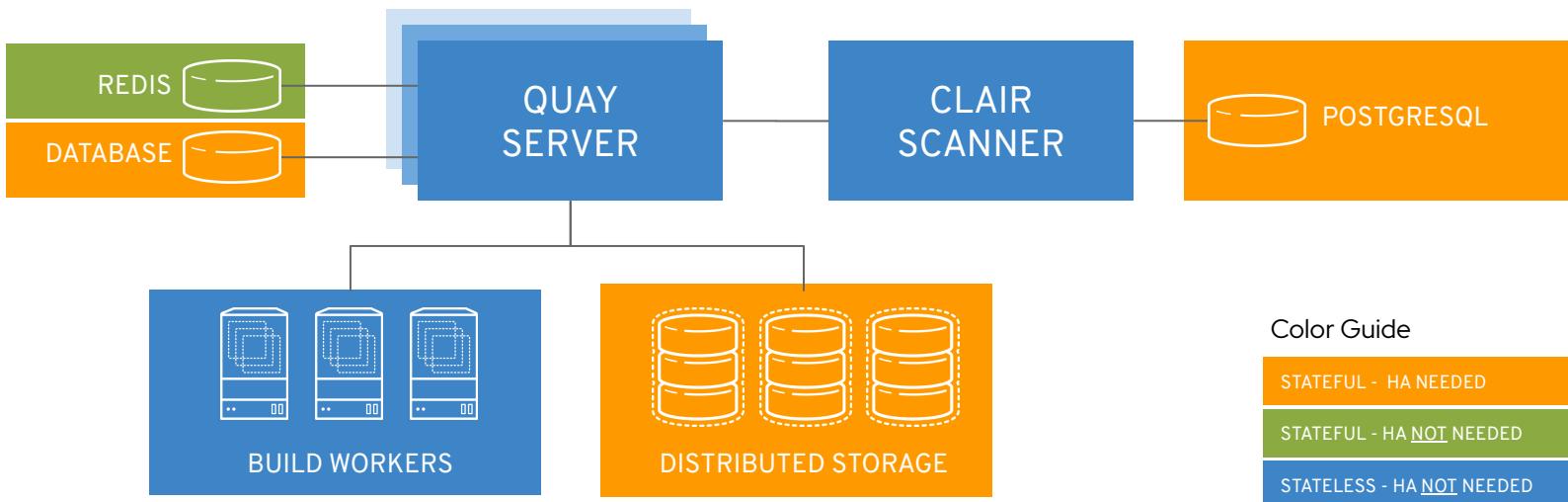
- Stateful components in HA mode
- Stateless components can be horizontally scaled arbitrarily



Note: Scaling out stateless components will add load to the stateful components, which must be accounted for in capacity planning.

https://access.redhat.com/documentation/en-us/red_hat_quay/3/html/deploy_red_hat_quay - high_availability/

Quay Components for HA





Built-In Vulnerability Scanning via Clair

Clair Overview

- Clair is an open source tool for **static analysis of vulnerabilities** in application containers
- Developed by CoreOS for Quay
- Used by various other projects and third party products
- Upstream Repository:
<https://github.com/quay/clair>



RED HAT Quay.io EXPLORE TUTORIAL PRICING

search SIGN IN

29abc8c5a3b2

Quay Security Scanner has detected 61 vulnerabilities.
Patches are available for 61 vulnerabilities.

14 High-level vulnerabilities.
33 Medium-level vulnerabilities.
14 Low-level vulnerabilities.

Vulnerabilities

CVE	SEVERITY	PACKAGE	CURRENT VERSION	FIXED IN VERSION	INTRODUCED IN LAYER
RHSA-2019-0710	High	python-lbs	2.7.5-68.el7	0:2.7.5-77.el7_5	
RHSA-2019-1587	High	python-lbs	2.7.5-68.el7	0:2.7.5-80.el7_6	
RHSA-2019-0368	High	systemd-lbs	219-67.el7	0:219-62.el7_5.5	
RHSA-2019-0049	High	systemd-lbs	219-67.el7	0:219-62.el7_5.2	
RHSA-2019-0679	High	libssh2	1.4.3-10.el7_2.1	0:1.4.3-12.el7_5.2	
RHSA-2019-2285	High	yun-plugin-ovl	1.1.31-45.el7	0:1.1.31-46.el7_5	
RHSA-2019-2141	High	awnumd	3.0.20-4.el7	0:3.0.20-5.el7	

Filter Vulnerabilities... Only show fixable

Clair - Supported Content Types

Clair v2 limited to OS level package managers including support for:

- Red Hat Enterprise Linux
- Debian Linux
- Ubuntu Linux
- Oracle Linux
- SuSE Linux
- Alpine Linux

45

Coming soon: Clair v4 (alongside Quay 3.4)



clair

Clair v4.0: Support for programming language package managers

Clair v4.x: Broader RH content coverage to cover non-RHEL RPM packages

Clair 2.9: support for various OS level package managers / distributions



Container Security Operator and OpenShift Console Integration

OpenShift Visualization of Vulnerabilities

The screenshot shows the Red Hat OpenShift Container Platform Admin Dashboard. The left sidebar includes sections for Administrator, Home, Dashboards, Projects, Search, Explore, Events, Operators, Workloads, and various sub-sections like OperatorHub, Installed Operators, and Cron Jobs. The main content area has tabs for Overview, Details, and View settings. Under Overview, there's a 'Status' section with icons for Cluster (green), Control Plane (green), and Quay Image Security (red, indicating 12 vulnerabilities). Below this are several alert cards: one for a Cluster operator being unavailable for 10 minutes, another for a Cluster operator being degraded for 10 minutes, and a third for samples failing to deploy. There's also a 'Cluster Inventory' section showing 6 Nodes, 264 Pods, 1 Storage Class, and 3 PVCs. A 'Cluster Utilization' chart shows CPU usage at 4.98 of 18 over a 1-hour period. A modal window titled 'Image Security breakdown' provides a circular chart of vulnerabilities by severity: 1 Critical (0), 6 High (5), 1 Medium (0), 1 Low (0), 1 Negligible (0), and 1 Unknown (0). It also lists 'Fixable Vulnerabilities' for namespaces nss-softokn, nss-sysinit, nss-softokn, nss, and nss-util.

- **Requires:**
 - Quay and Clair (on-cluster or off-cluster) and / or Quay.io used
 - CSO deployed and running on-cluster
- Container Security Operator (CSO) fetches vulnerability information from Clair
- Overview shown on OpenShift Cluster Admin Dashboard (Cluster Health Card)

Vulnerability Information in OpenShift Console

The screenshot shows the Red Hat OpenShift Container Platform console with the Quay Container Security Operator (CSO) integrated. The dashboard includes:

- A pie chart showing the distribution of vulnerabilities: 54% High, 27% Medium, 19% Low.
- A list of detected vulnerabilities:
 - RHSA-2019-0710 % (High)
 - RHSA-2019-1587 % (High)
 - RHSA-2019-0386 % (High)
 - RHSA-2019-0049 % (High)
 - RHSA-2019-0970 % (High)
 - RHSA-2018-2280 % (High)
 - RHSA-2018-0141 % (Info)
- A "Security breakdown" section with a summary of findings:

Severity	Count
High	14
Medium	33
Low	14

See all your Container Vulnerabilities right from the Console Dashboard

Link out to **Red Hat Quay** for more in depth information

Future Enhancements:

- data on a pod /project / cluster level
- Developer and admin console
- Reports incl. search filters / dashboards
- Notifications / Alerting



Quay Roadmap

Red Hat Quay / Quay.io / Quay Dedicated Roadmap

	Near Term (3.3)	Mid Term (3.4+)*	Long Term (3.5+)*
QUAY	<ul style="list-style-type: none">• Quay - OpenShift integration operator• LDAP filters / bindings• Usage (Quota) Reporting / Showback	<ul style="list-style-type: none">• Setup operator enhancem.• CSO and OCP Console enh.• Repo mirroring enhancem.• Geo-Replication redesign• Quota Alerting / Enforcement	<ul style="list-style-type: none">• Better air-gapped env support• Deeper integration into OCP• Quay Builder redesign• Content type Enhancements• App registry redesign
CLAIR	<ul style="list-style-type: none">• Clair v4 (Tech Preview):• App content scanning• RH content accuracy• 1-2 prog languages	<ul style="list-style-type: none">• Additional prog languages• Partner tooling integration• Clair air-gapped & multi-arch support	<ul style="list-style-type: none">• Support for Microsoft Windows images• Enhanced content coverage• Container Health Index
HOSTED	<ul style="list-style-type: none">• 2 SKUs for Quay.io (OF)	<ul style="list-style-type: none">• Quay.io usage in OSD	<ul style="list-style-type: none">• Quay Dedicated



Thank You!

 [linkedin.com/company/red-hat](https://www.linkedin.com/company/red-hat)

 [facebook.com/redhatinc](https://www.facebook.com/redhatinc)

 [youtube.com/user/RedHatVideos](https://www.youtube.com/user/RedHatVideos)

 twitter.com/RedHat