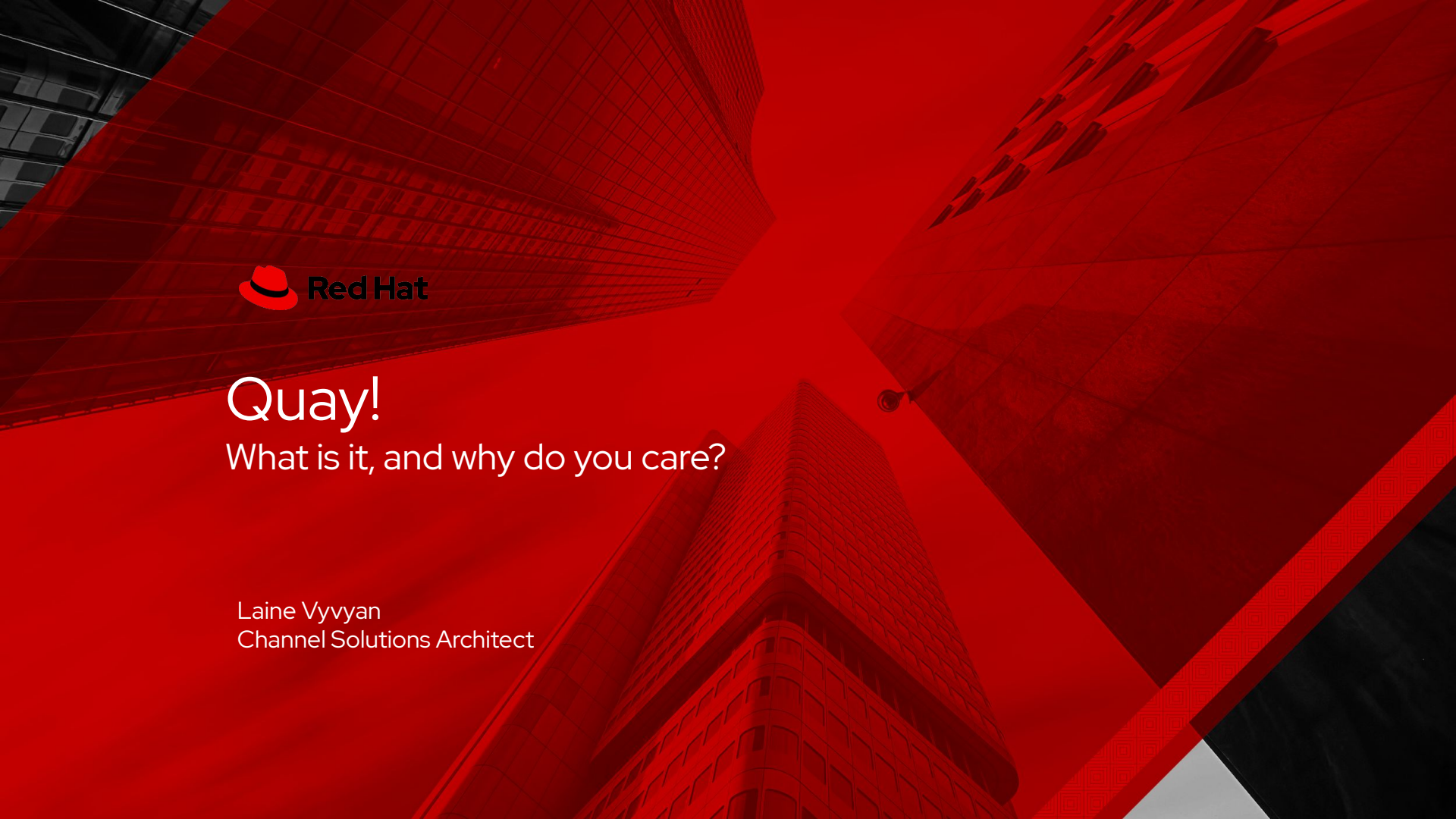


Quay!

What is it, and why do you care?

Laine Vyvyan
Channel Solutions Architect

Hi! I’m Laine Vyvyan.

I’m a Channel Solutions Architect, covering the Midwest.

I live in Michigan.

My favorite color is glitter.

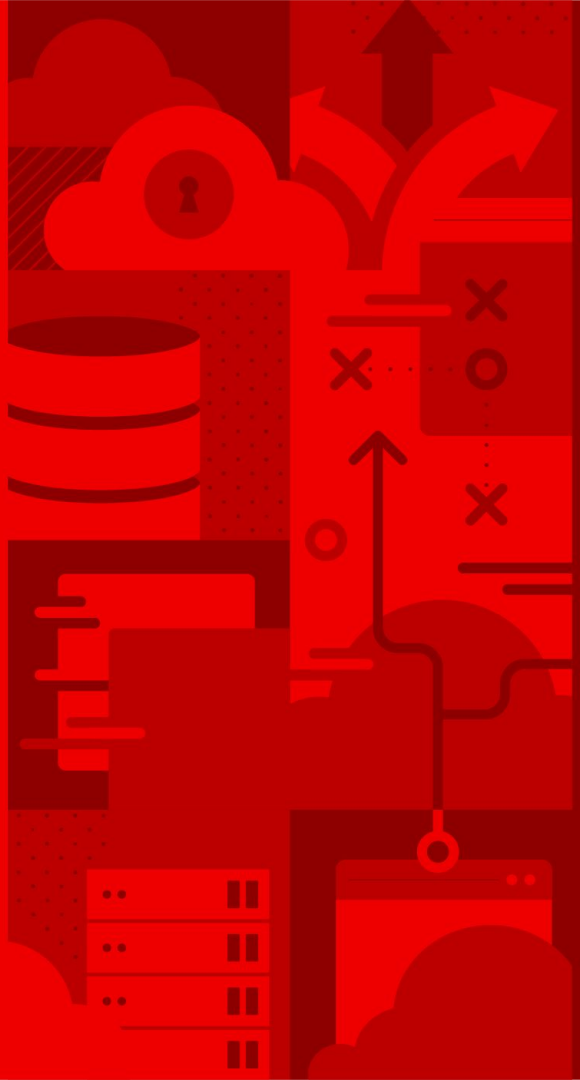
lvyvyan@redhat.com

@lainie_ftw

Red Hat

Agenda

- Review: OpenShift and Containers
- Review: OpenShift and Kubernetes - Core (Relevant) Technical Pieces
- Quay: What is it?
- Quay: Why do you care/why is it *cool*?
- Quay: How does it work?

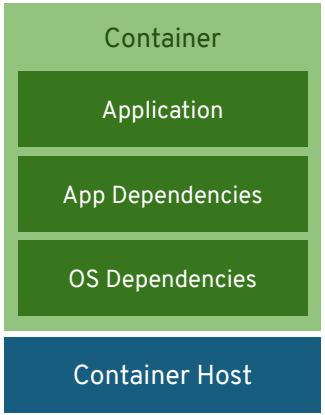


First, let's touch base on OpenShift and containers in general...

Red Hat

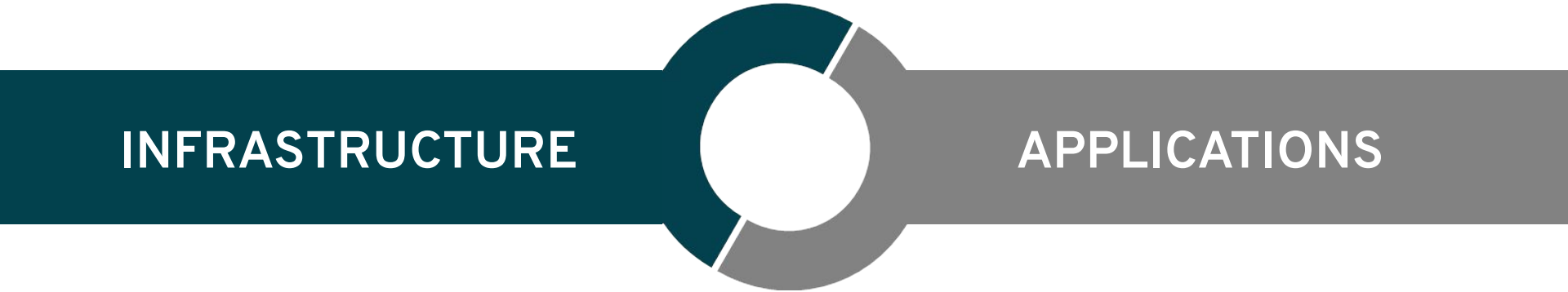
What is a container?

A container is an application, the application's dependencies/libraries/other binaries, and the configuration files that the application needs to run, all bundled into **one portable unit**.



Red Hat

Okay, but *why* containers?



- Application processes on a shared kernel
- Simpler, lighter, and denser than VMs
- **Portable** across different environments
- Dynamic **scalability** on demand

- Package apps with all dependencies
- Deploy to any environment in **seconds**
- Cloud-native application development
- **Flexibility** with language & runtime

Red Hat

What is Kubernetes?

Kubernetes is open source container orchestration – it automates **deployment**, **scaling**, and **management** of containers.

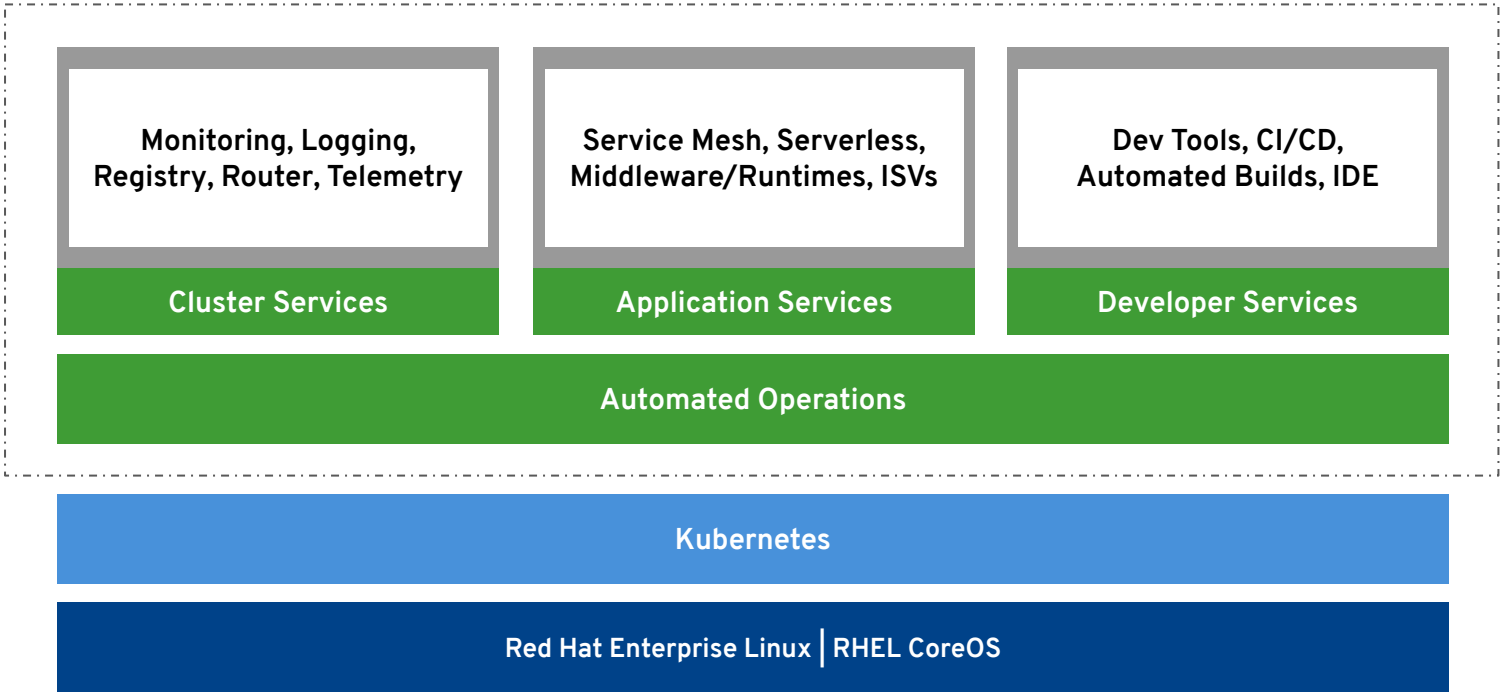
Red Hat

OpenShift 

Red Hat OpenShift is a Kubernetes-based, enterprise-ready container application platform.

Red Hat

OpenShift

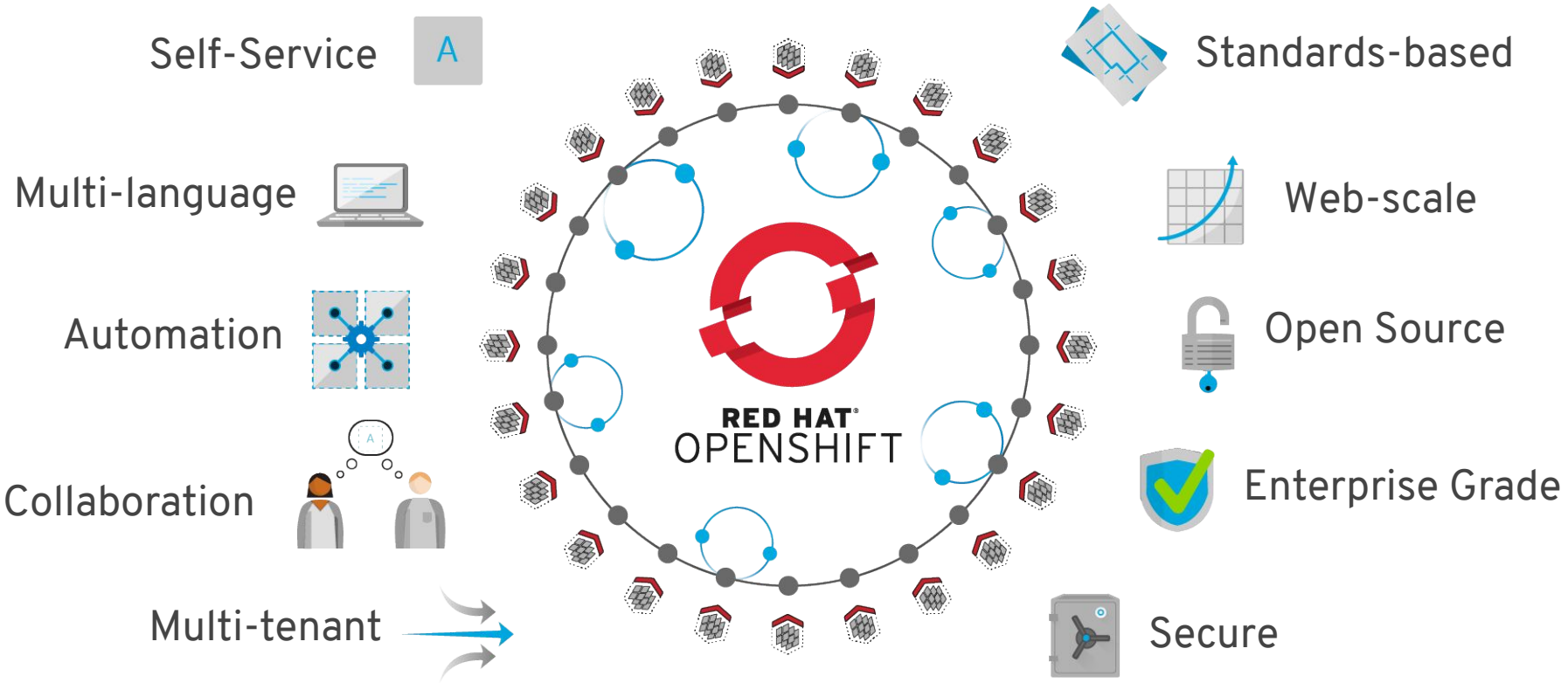


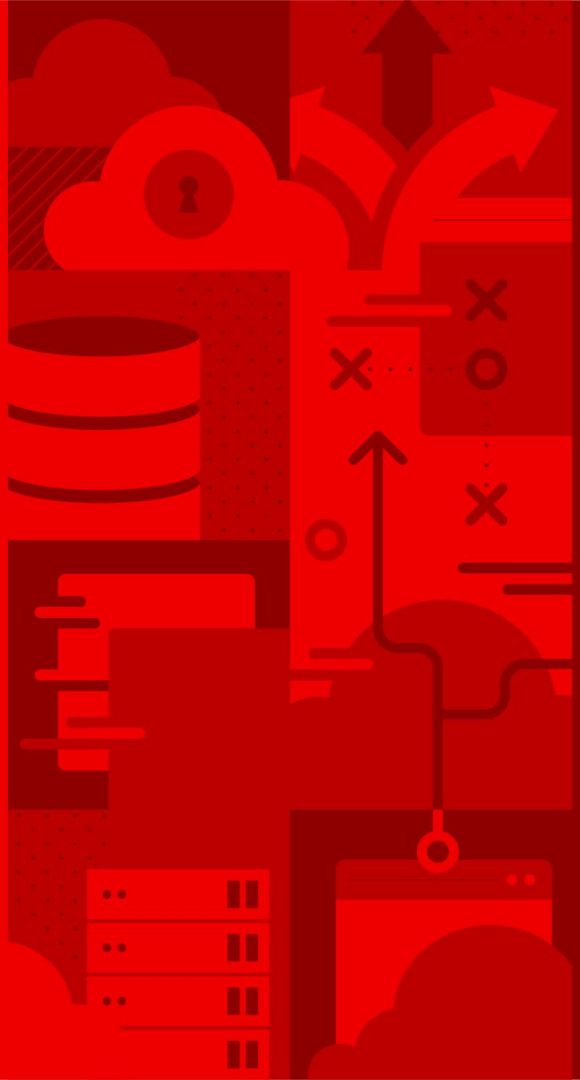
Best IT Ops Experience

CaaS ⟷ PaaS ⟷ FaaS

Best Developer Experience

Red Hat


Self-Service
Multi-language
Automation
Collaboration
Multi-tenant
RED HAT
OPENSHIFT
Standards-based
Web-scale
Open Source
Enterprise Grade
Secure



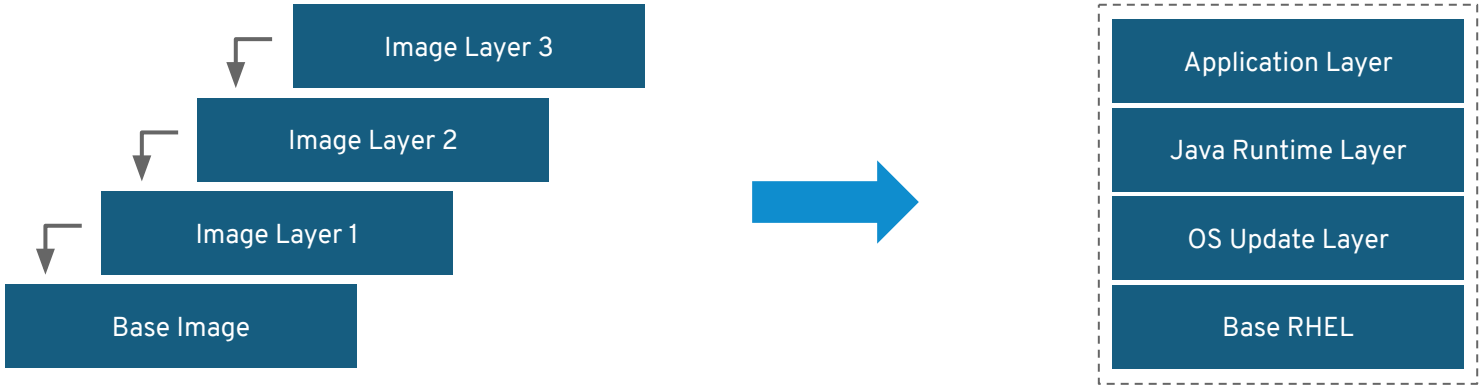
OpenShift and Kubernetes: Core (Relevant) Technical Pieces

Red Hat

a container is the smallest compute unit

containers are created from container `images`

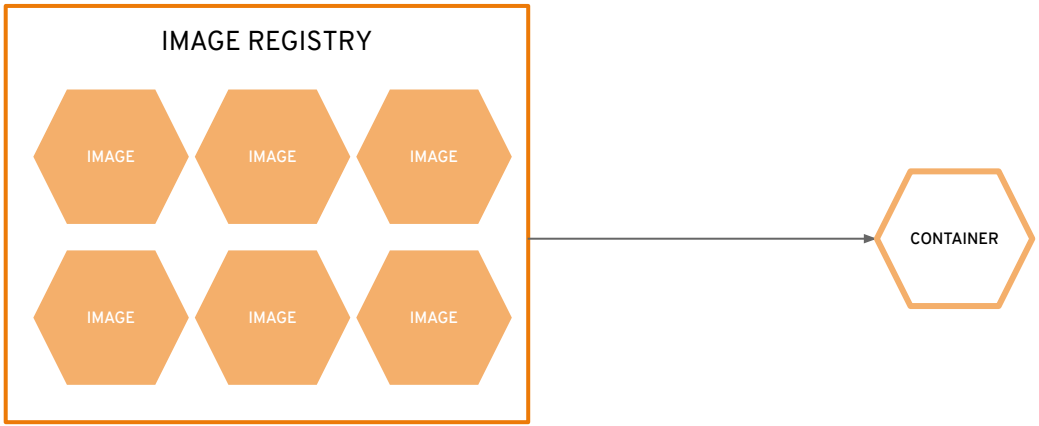
container `images` are built in layers



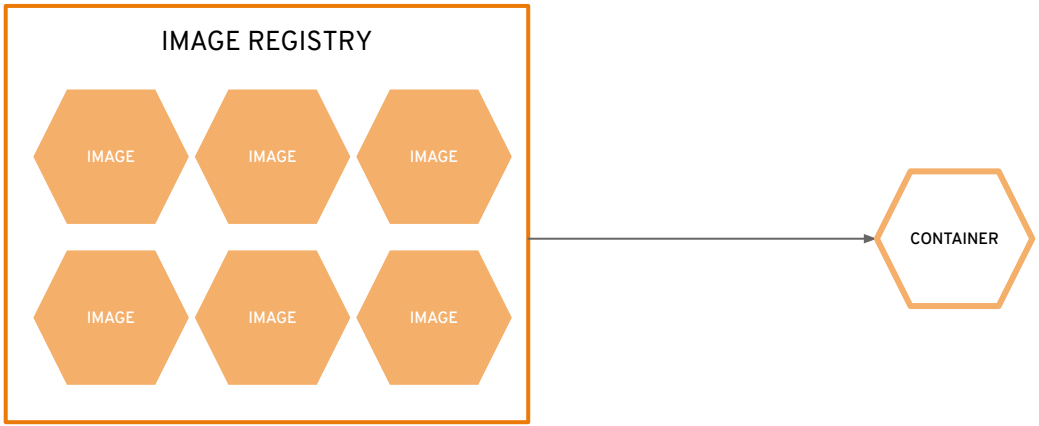
Container Image Layers

Example Container Image

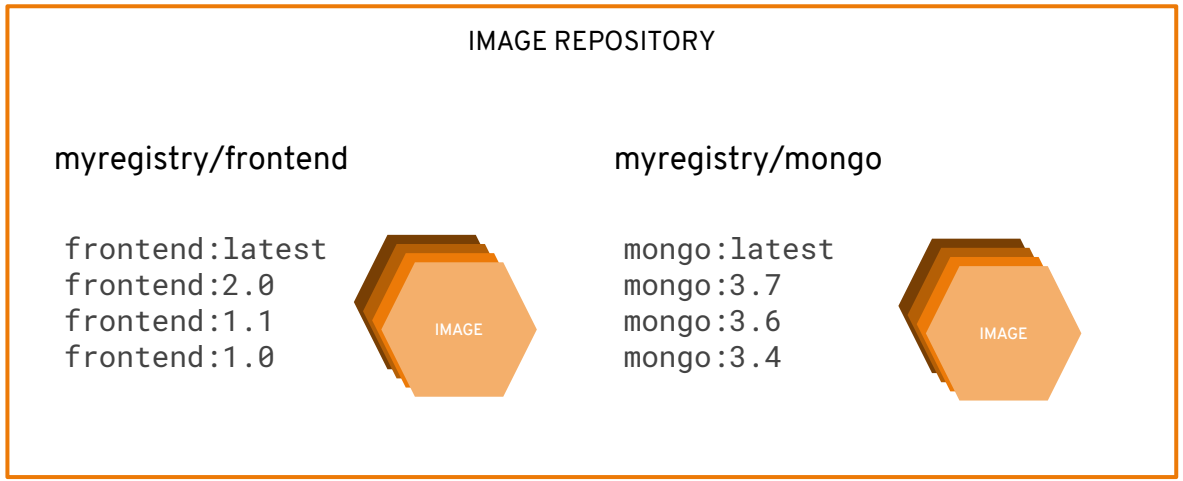
...and are stored in an `image registry`



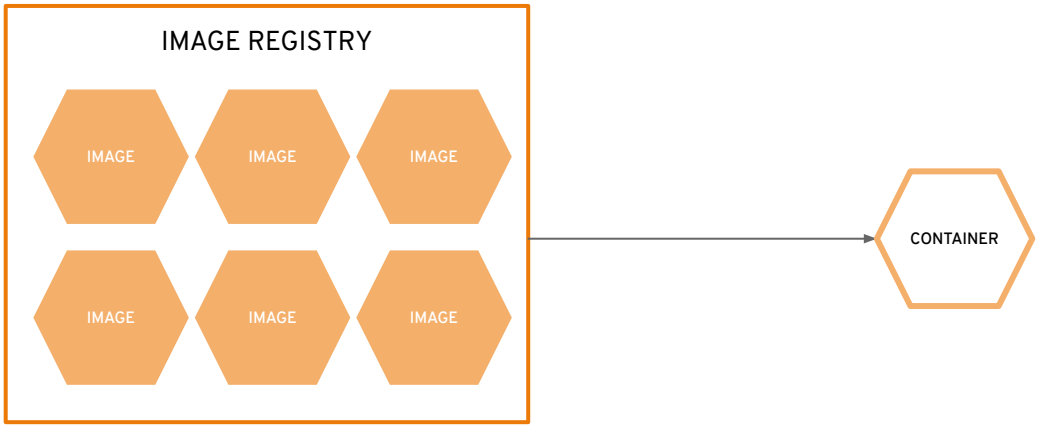
an `image registry` is the basic *concept* behind image storage and management



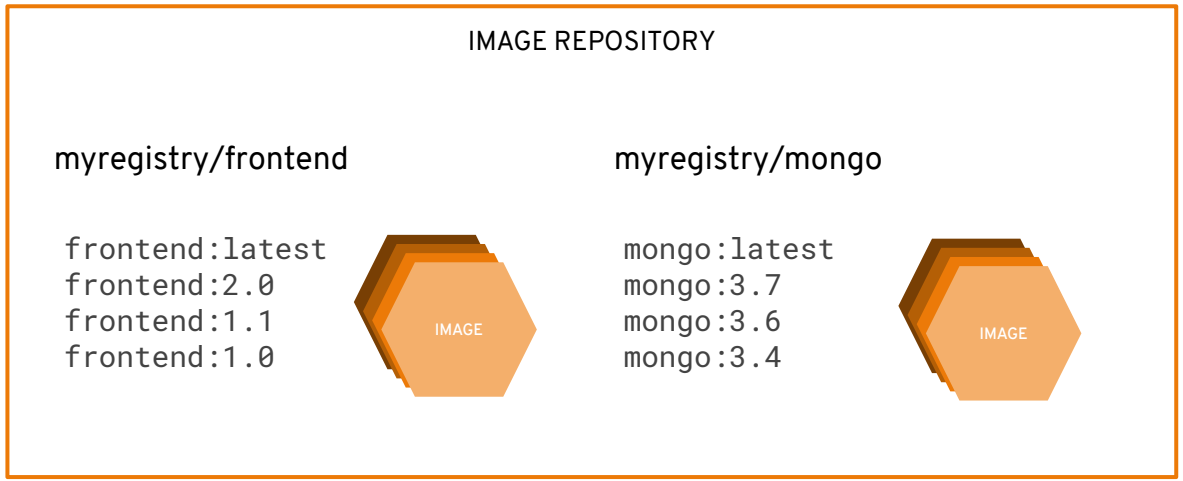
an `image repository` contains all versions (`tags`) of an image

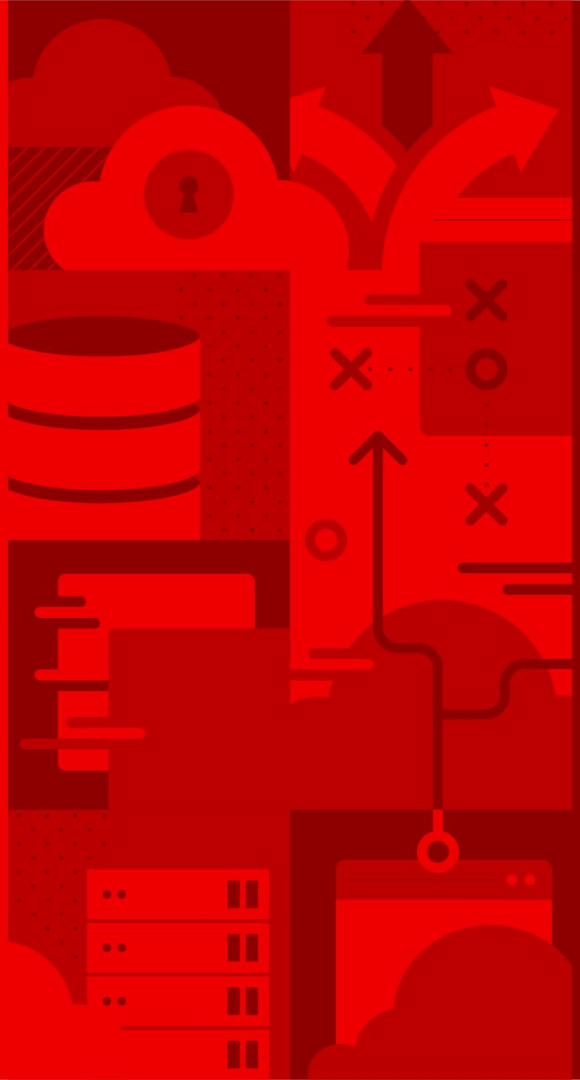


`image registry` = concept



`image repository` = *implementation*





Red Hat Quay

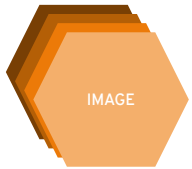
What is it?

Red Hat Quay

IMAGE REPOSITORY

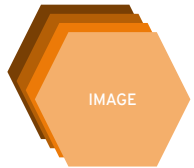
myregistry/frontend

frontend:latest
frontend:2.0
frontend:1.1
frontend:1.0

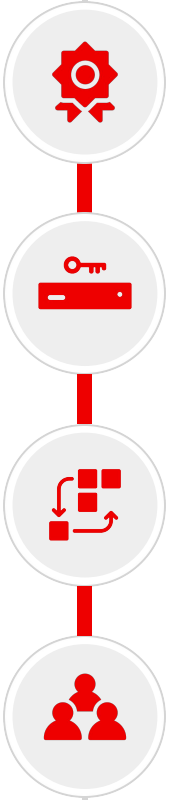


myregistry/mongo

mongo:latest
mongo:3.7
mongo:3.6
mongo:3.4



Red Hat Quay

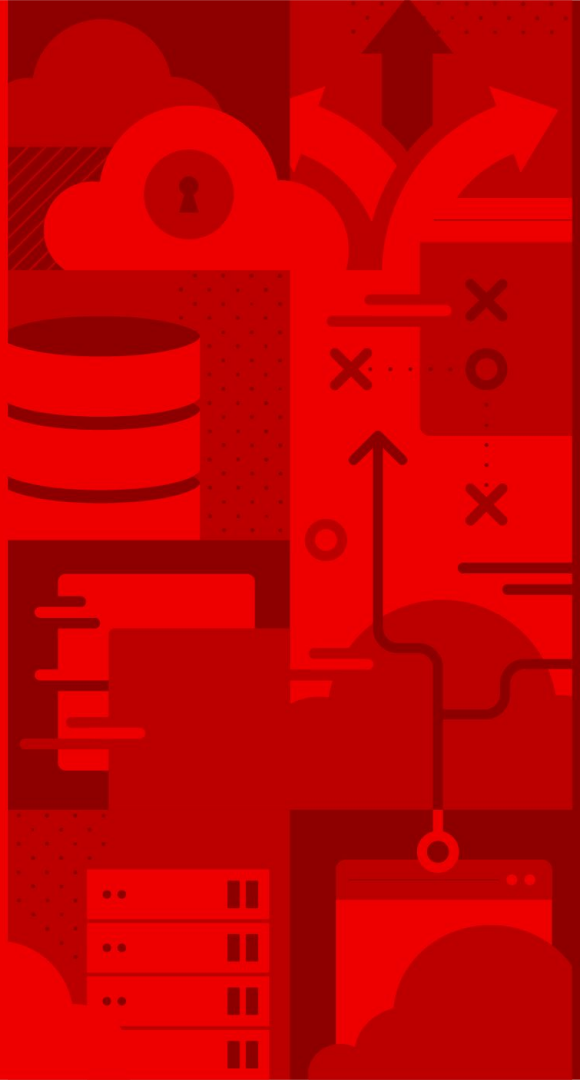


Industry-leading, **trusted, and open source registry platform** operating at scale since 2014

Built to **efficiently manage content** with governance and global (across an org) **security** controls

Runs **everywhere**, easy to **integrate** and **automate** - but it works best *with* **OpenShift**

Developed in **collaboration** with a broad open source, customer, and ecosystem **community**



Red Hat Quay

Why do you care?

Red Hat Quay Key Features

SCALABILITY

Massive Scale Testing Quay.io
Real Time Garbage Collection

BUILD AUTOMATION

Seamless Git Integration
Build Workers
Webhooks

INTEGRATION

Extensible API
Webhooks, OAuth
Robot Accounts

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

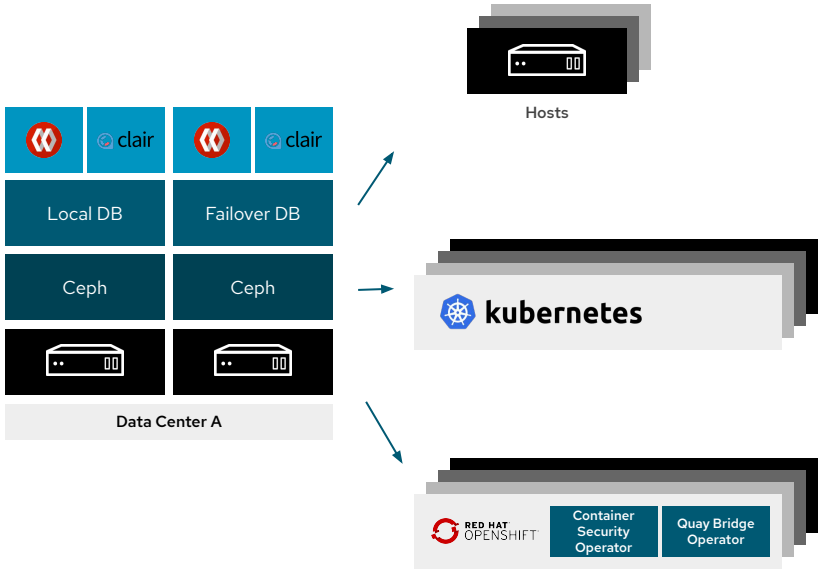
ACCESS CONTROL

Authentication Providers
Fine-Grained RBAC
Organizations & Teams

Okay, but...what *business* problems does it solve?

- Large-scale or distributed environments (think thousands of users, and/or thousands of images)
- Images shared across multiple OpenShift clusters
  - Dev + Prod
  - Dev + Prod + Prod 2 + Prod n
  - East + West + Europe
  - AWS + Azure + On-prem
  - ...etc
- Governance/security of container images
- High image maintenance and automation requirements
- “Source of truth” tailored to container images

Quay Deployment Models



Red Hat Quay can serve content to:

- Any container runtime or host
- Any orchestration platform

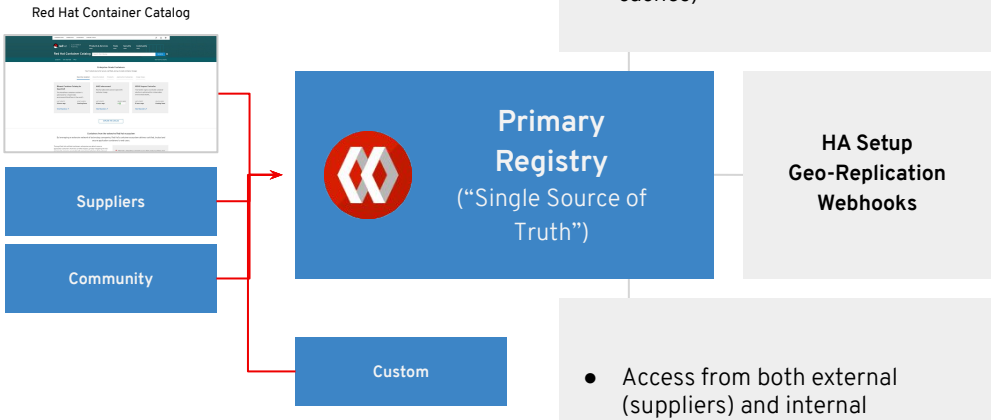
Typically Quay is serving content to many clients

- With different technologies / runtimes
- In different datacenters, VPCs, regions

The only requirement is that the client is compatible with the protocols and specs Quay supports
(Docker Registry API, OCI distribution spec)

Red Hat

Model #1 – Buffer/Source of Truth



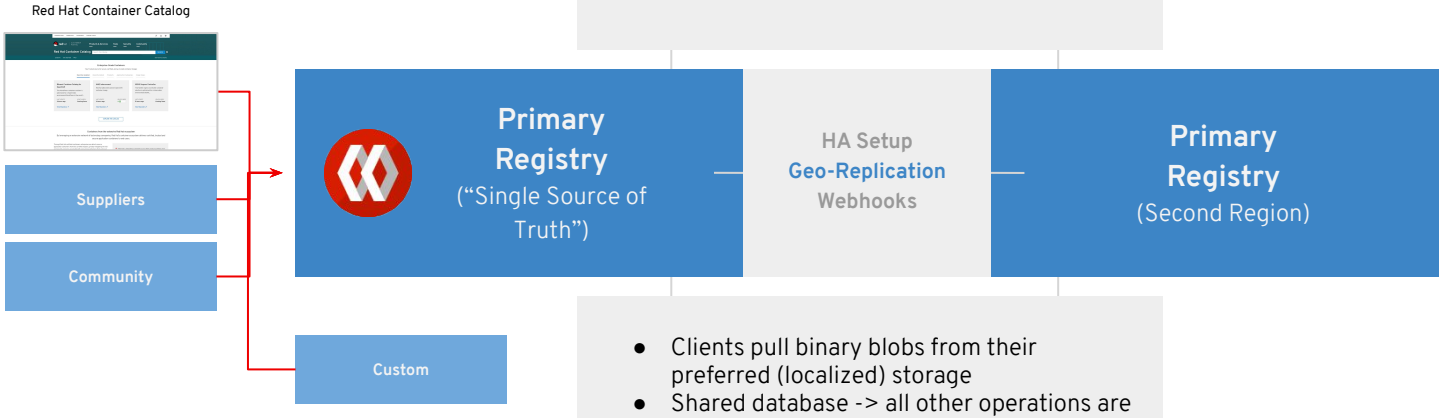
- Explicitly **whitelisted** content from external sources (in contrast to arbitrary proxy caches)

- Access from both external (suppliers) and internal (private repos, RBAC)

Red Hat

Model #2 - Source of Truth + Geo-Replication/Multi-Region

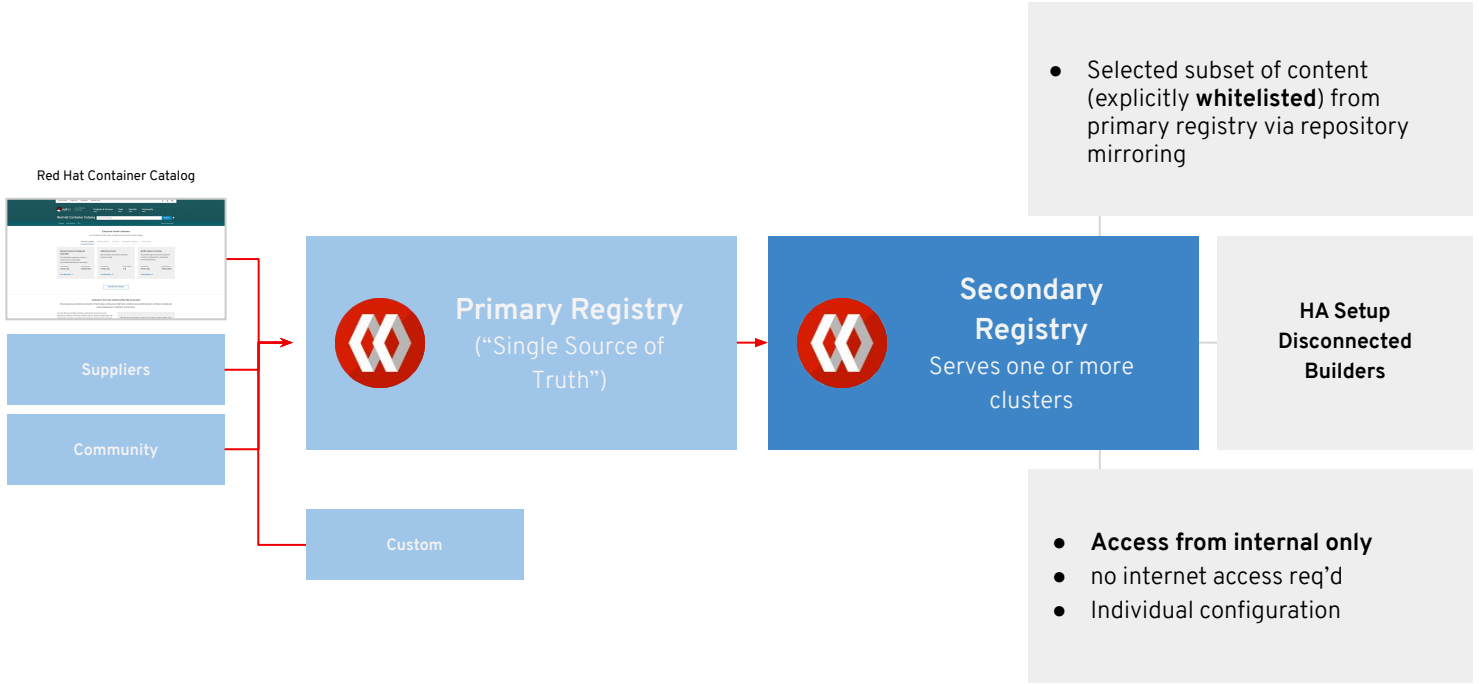
- Geo-Replication means a **single, globally-distributed** Red Hat Quay setup spread across multiple datacenters / regions



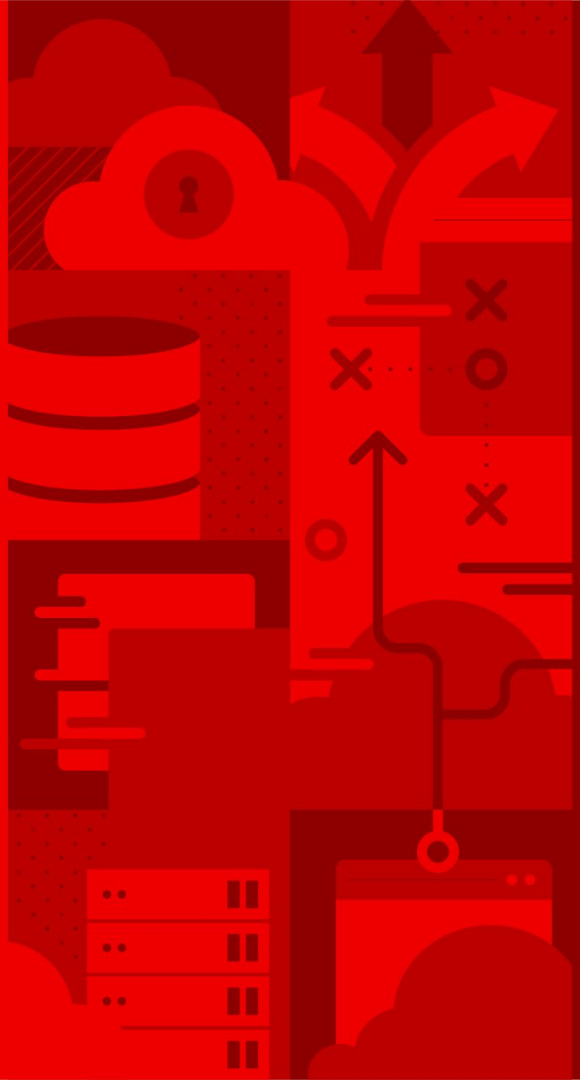
- Clients pull binary blobs from their preferred (localized) storage
- Shared database -> all other operations are handled globally, frontend depends
- blob not replicated yet: fallback to remote

Red Hat

Model #3 – Source of Truth + Mirror(s)



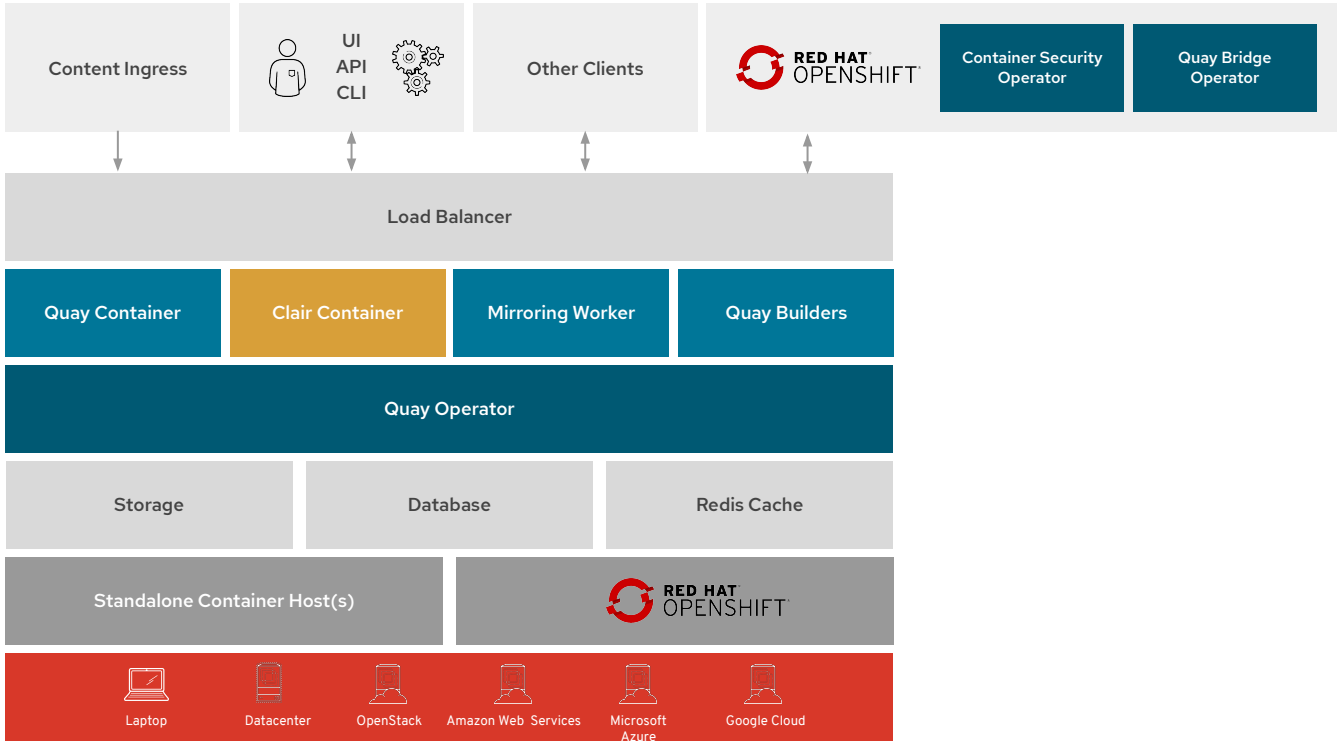
Red Hat



Red Hat Quay

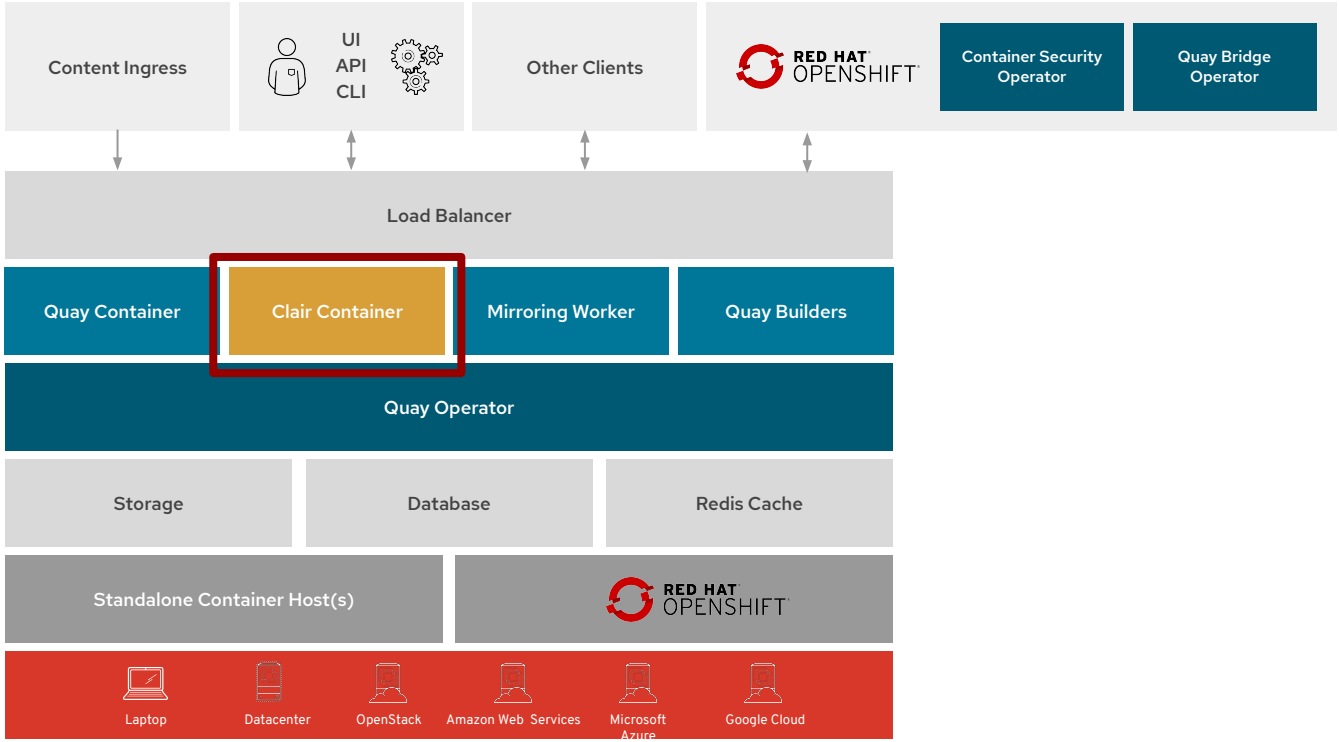
How does it work? (Quay Architecture)

Red Hat Quay Architecture



Red Hat

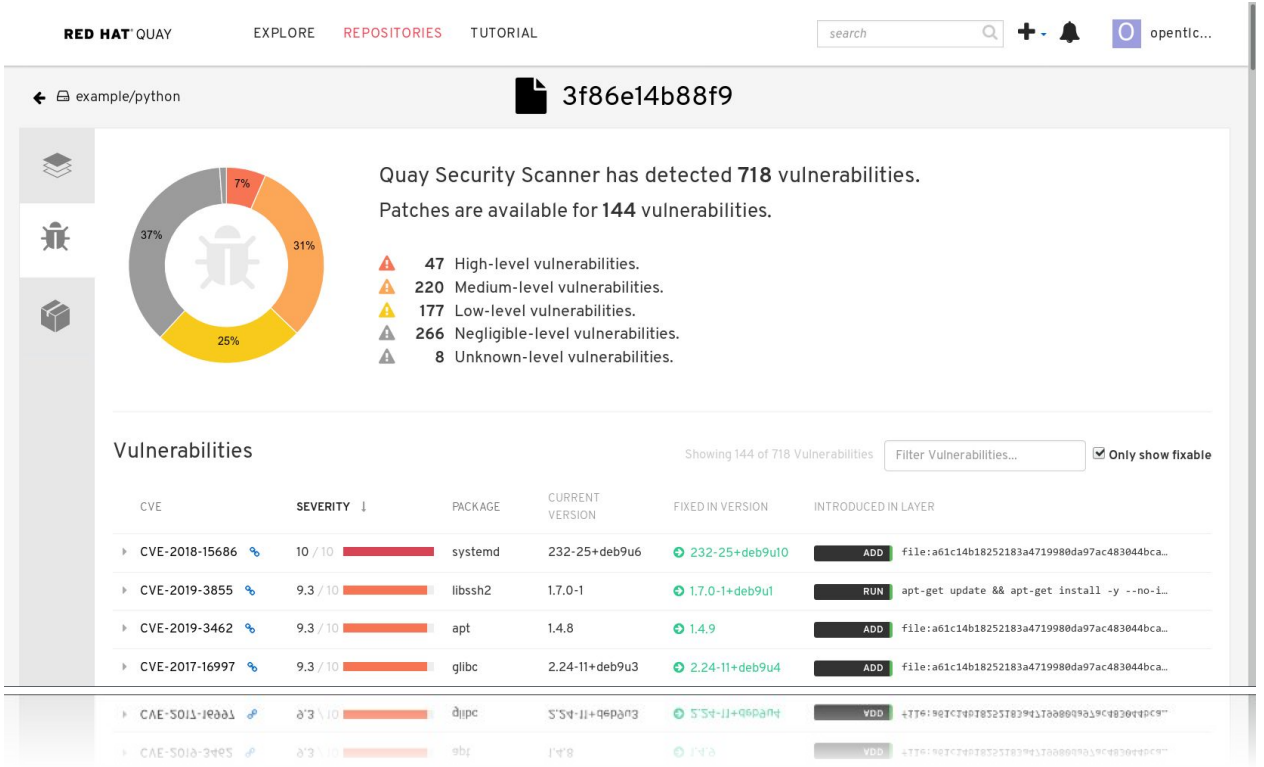
Red Hat Quay Architecture



Red Hat

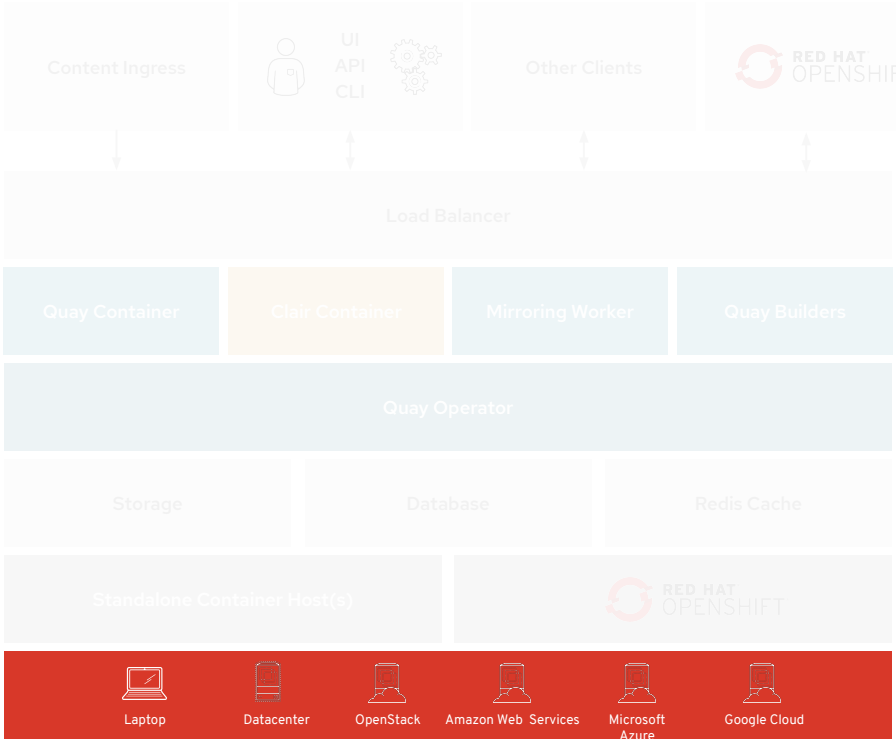
Clair: Integrated Container Vulnerability Scanning

Quay integrates with Clair to **continually scan** your containers for vulnerabilities.



Red Hat

Underlying Infrastructure



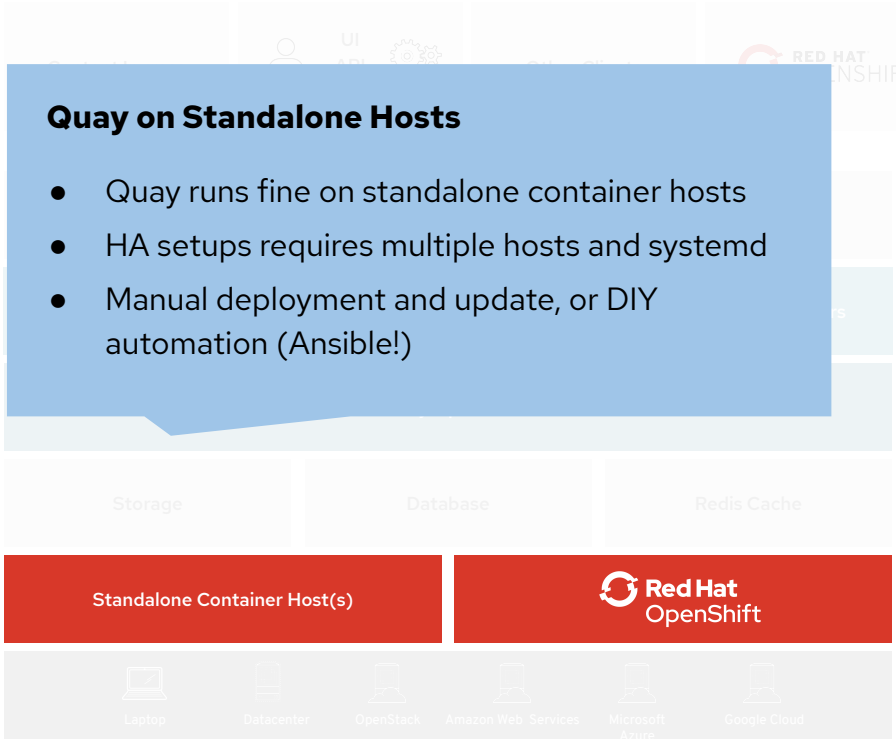
Quay runs on any* physical or virtual infrastructure, both on-premise or public cloud.**

Quay scales from an all-in-one setup on a developer laptop over running highly available on OpenShift up to geographically dispersed setup across multiple availability zones and regions

* Further details can be found in the Quay 3.x tested configuration matrix: https://access.redhat.com/articles/4067991
** Further details can be found in the Quay Support Policy: https://access.redhat.com/support/policy/updates/rhquay/policies

Red Hat

Container Runtime or Orchestration



Quay on Standalone Hosts

- Quay runs fine on standalone container hosts
- HA setups requires multiple hosts and systemd
- Manual deployment and update, or DIY automation (Ansible!)

Benefits of running Quay on OpenShift:

- Run Quay where you run all other container workloads and services
- Seamless deployment and day2 management of Red Hat Quay via the Quay operator and OLM
- Leverage all orchestration and management capabilities of Red Hat OpenShift incl. Operator Lifecycle Mgt (OLM), monitoring, dashboards, ...

Full list of tested and supported configurations can be found inside the Red Hat Quay Tested Integrations Matrix: https://access.redhat.com/articles/4067991

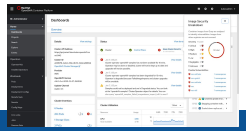
Red Hat

Red Hat Quay + OpenShift = ❤

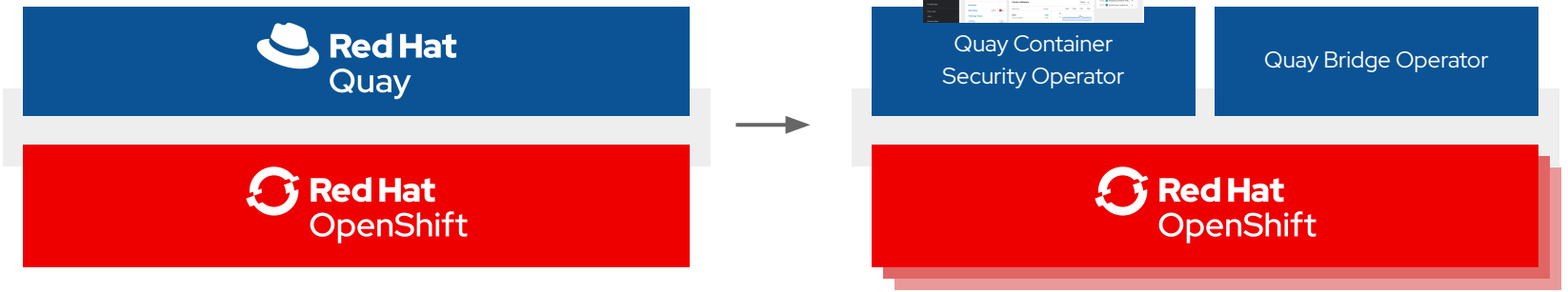
Red Hat Quay runs on any infrastructure but **runs best on OpenShift.**

The **Quay Operator** ensures seamless deployment and management of Quay on OpenShift.

CSO brings Quay / Clair vulnerability data into the OpenShift Console



The **Quay Bridge Operator** ensures seamless integration and user experience for using Quay as the internal OpenShift registry.



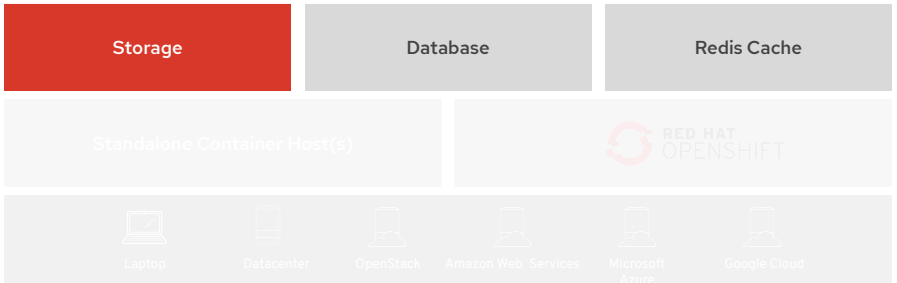
Quay serves content to **one or many OpenShift clusters**, wherever they're running.

Red Hat

Storage

How it's used

- Quay stores all binary blobs in its storage backend
- Local storage and NFS **only** for PoC / test setups, *not* supported for a Production configuration



Supported On-Prem Storage Types

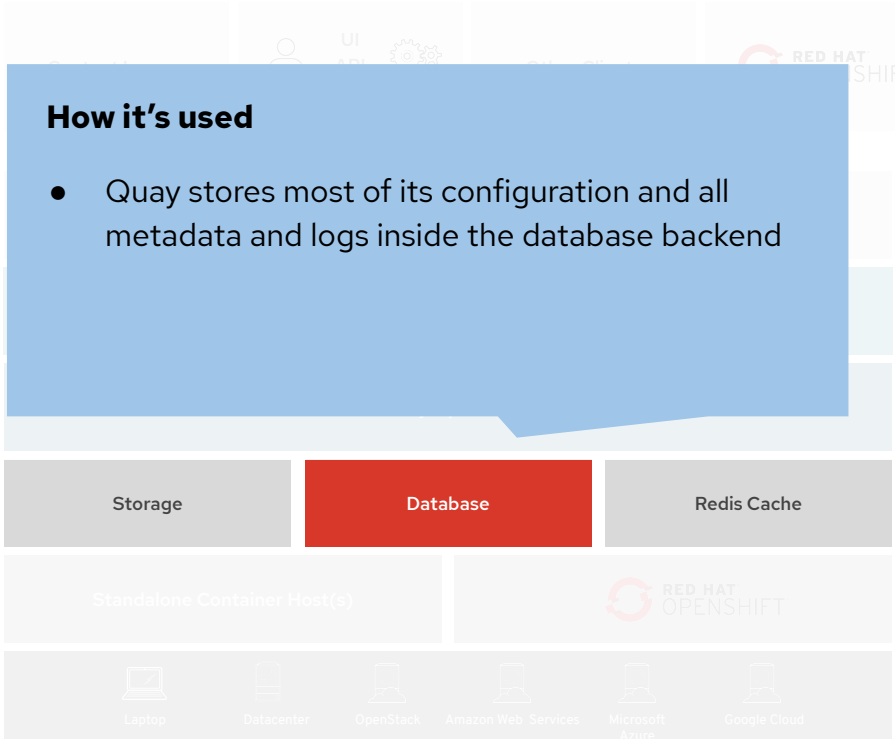
- Ceph Rados RGW
- OpenStack Swift
- RHOCS 4 (via NooBaa)

Supported Public Cloud Storage Types

- AWS S3
- Google Cloud Storage
- Azure Blob Storage

Note: Technically any S3 compatible storage solution works with Quay and is used by several customers. Support limitations might apply though.
Full list of tested and supported configurations can be found inside the Red Hat Quay Tested Integrations Matrix: https://access.redhat.com/articles/4067991

Database Backend



How it's used

- Quay stores most of its configuration and all metadata and logs inside the database backend

- **PostgreSQL** is the preferred database backend since it can be used for *both* Quay and Clair
- If Quay runs on **public cloud infrastructure** we recommend to use the PostgreSQL services provided by your cloud provider
- Typically runs off-cluster but if DB runs on k8s / **OpenShift** we recommend to use an **operator** such as Crunchy Data PostgreSQL Operator
- Logs *can* be pushed into **ElasticSearch** instead

Full list of tested and supported configurations can be found inside the Red Hat Quay Tested Integrations Matrix: https://access.redhat.com/articles/4067991

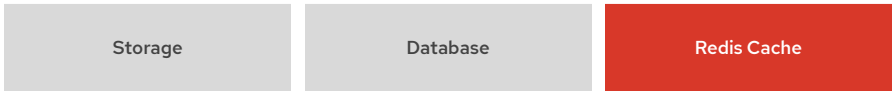
Red Hat

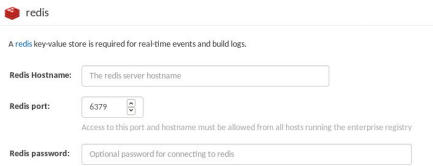
Redis Cache

How it's used

- Quay stores builder logs and the Quay tutorial inside a Redis cache
- Data stored is ephemeral in nature

- Redis via **Red Hat Software Collections** or any other redis works, too
- If Redis goes down you will lose access to:
  - Live build logs
  - Tutorial





Full list of tested and supported configurations can be found inside the Red Hat Quay Tested Integrations Matrix: https://access.redhat.com/articles/4067991

How to Update / Upgrade Quay

- Quay is designed for **zero-downtime upgrades** at both the infrastructure and application levels
- Individual nodes running Quay containers can be upgraded in a rolling fashion without overall registry downtime
- Similarly, Quay itself can typically be upgraded in a rolling fashion without causing any downtime
- For Quay upgrades requiring a database migration, these may be typically performed with minimal (on the order of minutes) or no downtime.

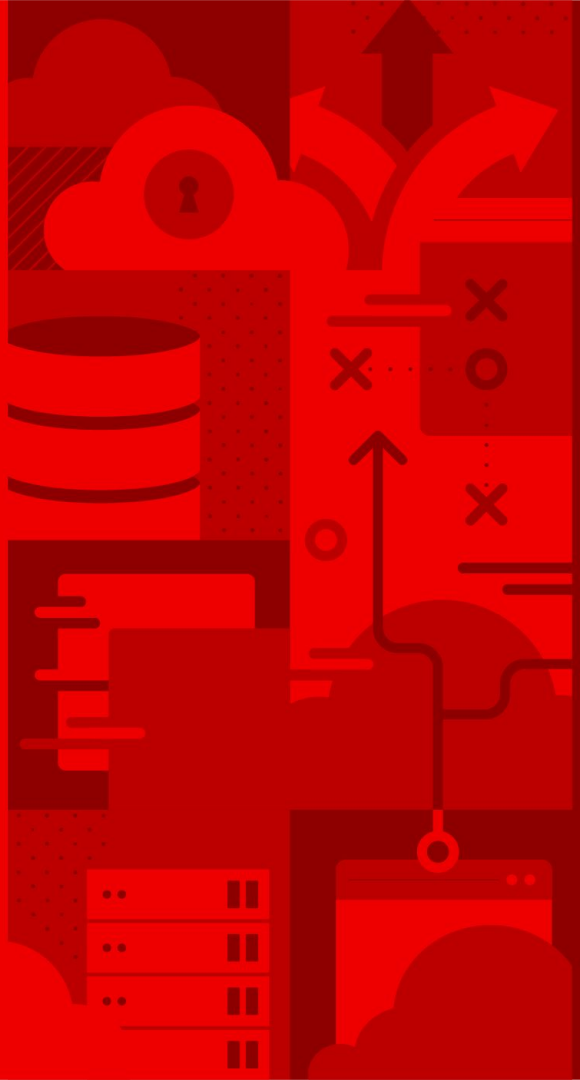
Updates can be performed using the Quay operator and oc cmd:

```
$ oc edit quayecosystem/quayecosystem
```

Find and update the following entries:

```
image: quay.io/redhat/clair-jwt:vX.X.X
image: quay.io/redhat/quay:vX.X.X
```

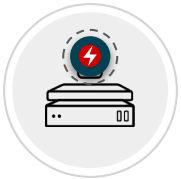
Note: The Quay operator will manage automatic updates in future versions of Red Hat Quay



Red Hat Quay

How does it work? (Running Quay + Features)

Red Hat Quay Operators



Quay Operator

Automates the initial deployment of Quay, Clair and backends on OpenShift

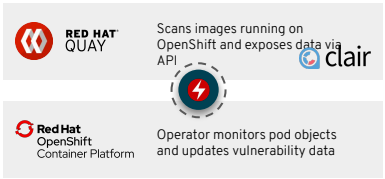
Simplifies Quay installation & Day 2 operations

Configures all relevant OpenShift objects (routes, secrets, certificates, etc.)

Runs (only) on the OpenShift Cluster Quay is running on

Not needed with Quay.io

Added in Quay 3.1



Container Security Operator

Operator which runs on OpenShift and fetches vulnerability data from Quay / Clair if Kubernetes pod objects change and stores it in CRs

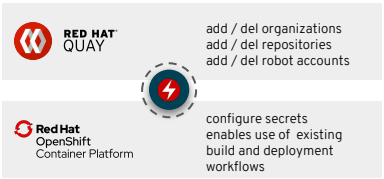
Synchronous Updates of vulnerability information

Vulnerability data shown in OpenShift Console

Runs on every OpenShift cluster Quay is serving content to

Works with Quay.io

Added in Quay 3.2



Quay Bridge Operator

Operator which runs on OpenShift and integrates Quay into OpenShift workflows similar to the existing internal registry experience.

Built in strong collaboration with Red Hat internal and customer / open source communities

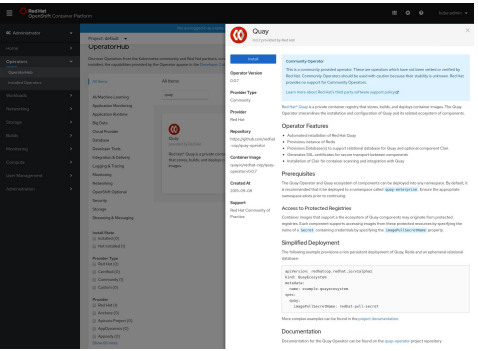
Runs on every OpenShift cluster Quay is serving content to

Does **not work** with Quay.io

added in Quay 3.3

Note: All Quay Operators are fully support on OpenShift 4 only due to OLM in Tech Preview on OCP 3

Red Hat



All Places > Communities at Red Hat > Applications

Container Community of Practice

Quay Operator

Automates the initial deployment of Quay and Clair

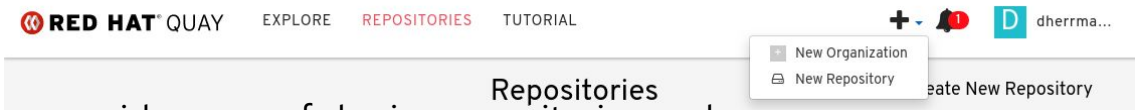
Configures all relevant OpenShift objects (routes, secrets, etc.)

Aids in certificate management

Automates Day 2 Management of Quay

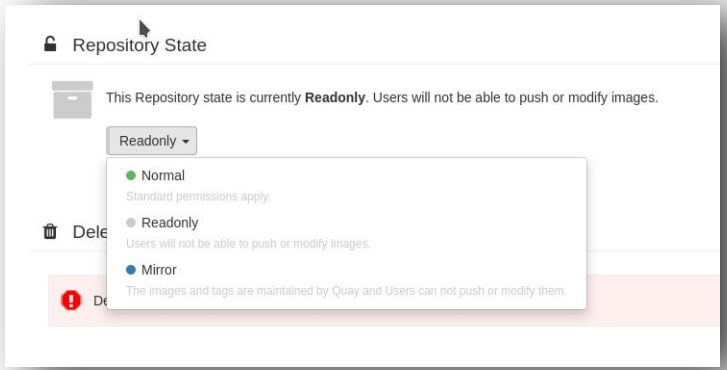
Red Hat

Quay Organizations



- **Organizations** provide a way of sharing repositories under a common namespace that does not belong to a single user, but rather to many users in a shared setting (such as a company).
- Organizations are organized into a set of **Teams** which provide access to a subset of the repositories under that namespace
- **Robot accounts** are managed inside the Robot Accounts tab and can belong only to **one** organization (but multiple **Teams**) while **Teams** and **Users** can belong to multiple organizations
- Time machine settings and OAuth applications are configured on an organization level
- Usage (audit) logs are shown on an organization level for all repositories inside the organization

Red Hat



Repository State: NORMAL

Repository Mirroring DISABLED

Default mode in Quay prior 3.1 and repos without mirroring enabled

Both push and pull permissions are defined via Quay RBAC model

Repository State: MIRRORED

Repository Mirroring **ENABLED**

All (user) pushes are disabled

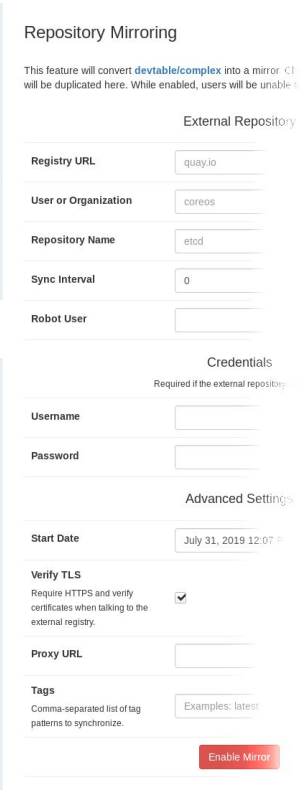
Pulls allowed (requires read permissions)

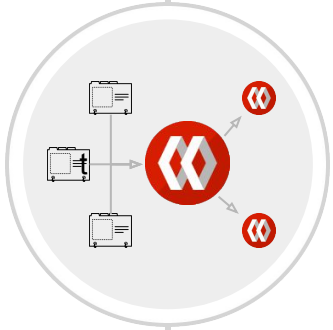
Repository State: READ-ONLY

Repository Mirroring **DISABLED**

No pushes allowed by users, robots and the mirroring workers

Primary use cases: archiving and (temporarily) “frozen repos”





Powered by

skopeo

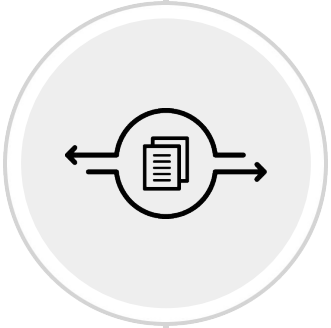
https://github.com/containers/skopeo

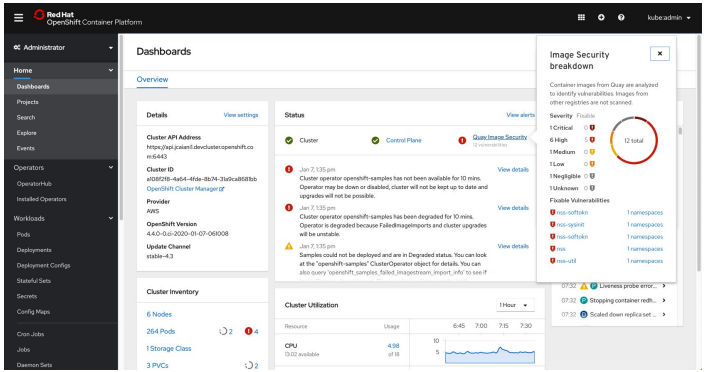
Repository Mirroring

Continually synchronize repositories from external source registries into Quay, on the schedule you choose.

Can mirror a *subset* of an entire registry to distributed registry deployments, or the entire repo.

Red Hat





RED HAT QUAY

Scans images running on OpenShift and exposes data via API

clair

Red Hat OpenShift Container Platform

Operator monitors pod objects and updates vulnerability data

Container Security Operator - Vulnerability Data in OpenShift

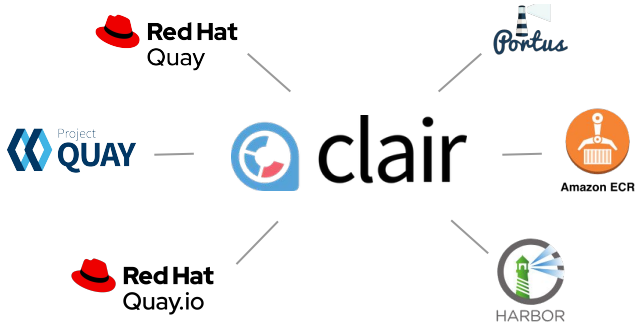
Operator which runs on OpenShift and fetches vulnerability from Quay / Clair if Kubernetes pod objects change

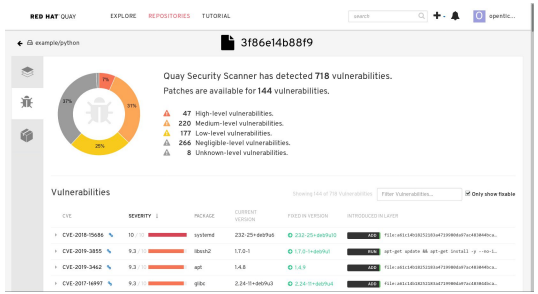
Synchronous Updates of vulnerability information

Prerequisite to leverage / show vulnerability data in OpenShift Console

Red Hat

clair v4



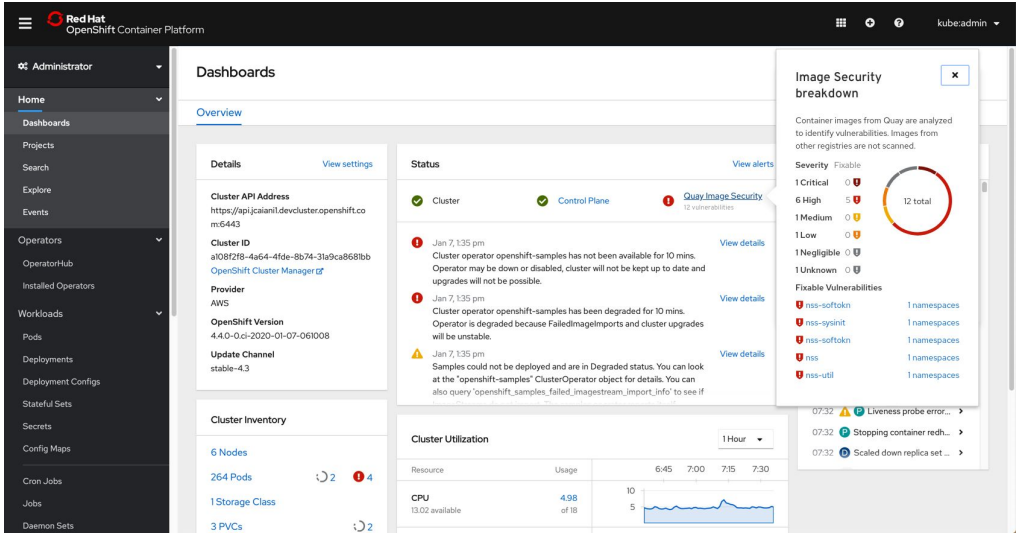


Clair v4 (Tech Preview with Quay 3.3)

Clair v4 is the newest version of Clair after a massive refactoring in order to make several big enhancements possible. This includes:

- Support for programming language package managers (3.3: python)
- immutable data model & new manifest-oriented API
- Refocus on latest container specifications (OCI) (Content addressability)

Red Hat



RED HAT QUAY

Scans images running on OpenShift and exposes data via API

clair

Red Hat OpenShift Container Platform

Operator monitors pod objects and updates vulnerability data

OpenShift Console Enhancements for Clair Vulnerability Data

We extended the vulnerability information shown inside the OpenShift Console brought to Kubernetes via the Container Security Operator. This includes:

- Image Vulnerabilities Lists in the Administrator section
- Pod View for image vulnerabilities specific to a particular pod
- Enhanced information shown now including severity, advisories and versions
- Affected pods view to show all pods affected by a particular CVE

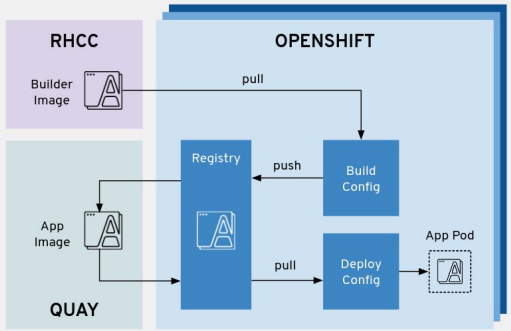
Red Hat

Using Quay With *or* Without Internal Registry

- Quay can be used as an external registry in front of an entire OpenShift cluster with its registry
- Quay also can be used *directly* without using the internal registry which requires a few configuration changes. These changes are **partially** managed automatically by QBO, with plans to expand this over time.

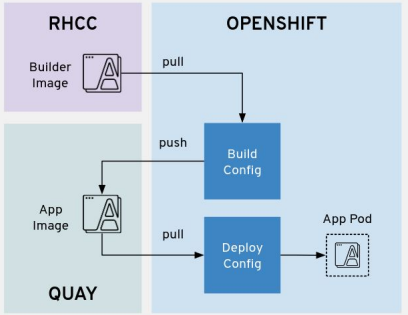
Quay as Upstream Registry with OpenShift

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

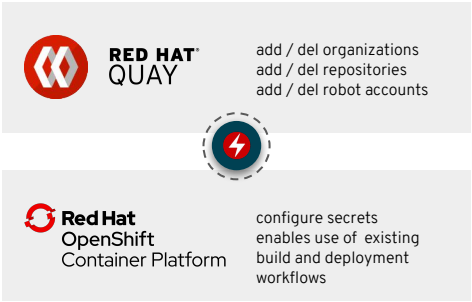


Quay as OpenShift Registry

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



Red Hat



Quay Bridge Operator

Operator which runs on OpenShift and integrates Quay into OpenShift workflows similar to the existing internal registry experience.

Built in strong collaboration with the Red Hat internal and customer communities

Supports multi-cluster setups, features OCP build integration

Red Hat

Quay and OpenShift – Bridge Operator

- Operator which runs on OpenShift and automates some integration pieces similar to our existing internal registry user experience
- Sample use cases:
  - New project in OpenShift -> new organization in Quay + robot accounts + configure pull and push secrets in OpenShift
  - New app in OpenShift -> build results pushed to Quay using the robot accounts and push secrets created earlier
  - New deployment in OpenShift -> pull image from Quay using the pull secret
  - Delete a project in OpenShift -> delete repositories, robot accounts and org in Quay
  - OpenShift cluster name = CRD config option -> supports multiple OCP clusters in Quay

Red Hat Quay Further Information

Product Docs

- Red Hat Quay Release Notes
- Deploy Red Hat Quay - Basic
- Deploy Red Hat Quay on OpenShift
- Deploy Red Hat Quay on OpenShift with Quay Setup Operator
- Deploy Red Hat Quay - High Availability
- Manage Red Hat Quay
- Upgrade Red Hat Quay
- Use Red Hat Quay
- Red Hat Quay API Guide

Knowledge Base

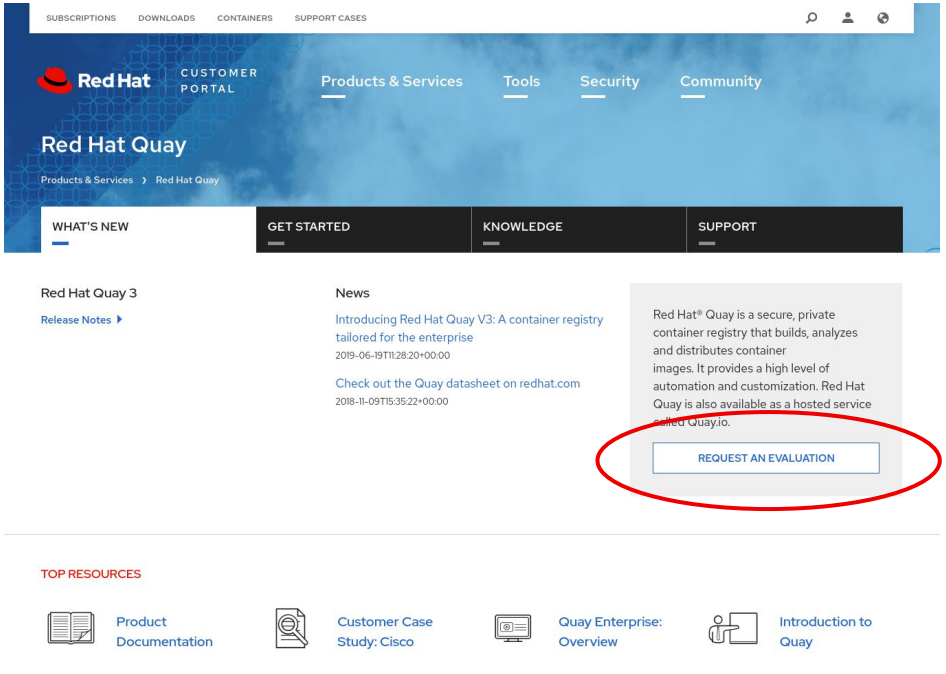
- Inside the Red Hat Customer Portal many knowledge base articles and solutions can be found around Red Hat Quay
- How to find them: enter your search term and select “Red Hat Quay” as the product
- Optional: preferred content type
- Sample Search URL

Other Information

- Community Mailing list (Quay SIG)
- Project Quay Community Page
- Source Code (Project Quay)
- Feature Development and Bugtracking in public Quay Jira
- Project Quay on Twitter
- Red Hat Quay.io (Hosted SaaS)

Try it out!

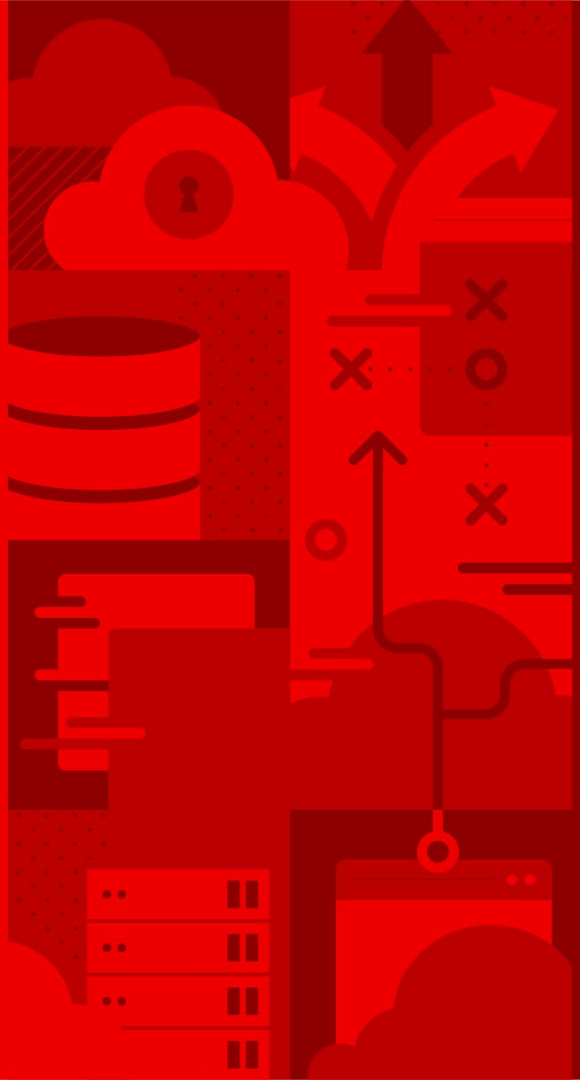
https://access.redhat.com/products/red-hat-quay



On all Quay product pages you can find an evaluation form which grants you access to the software for a 90 day trial period.

Alternatively you can signup **for free** on Quay.io

Red Hat

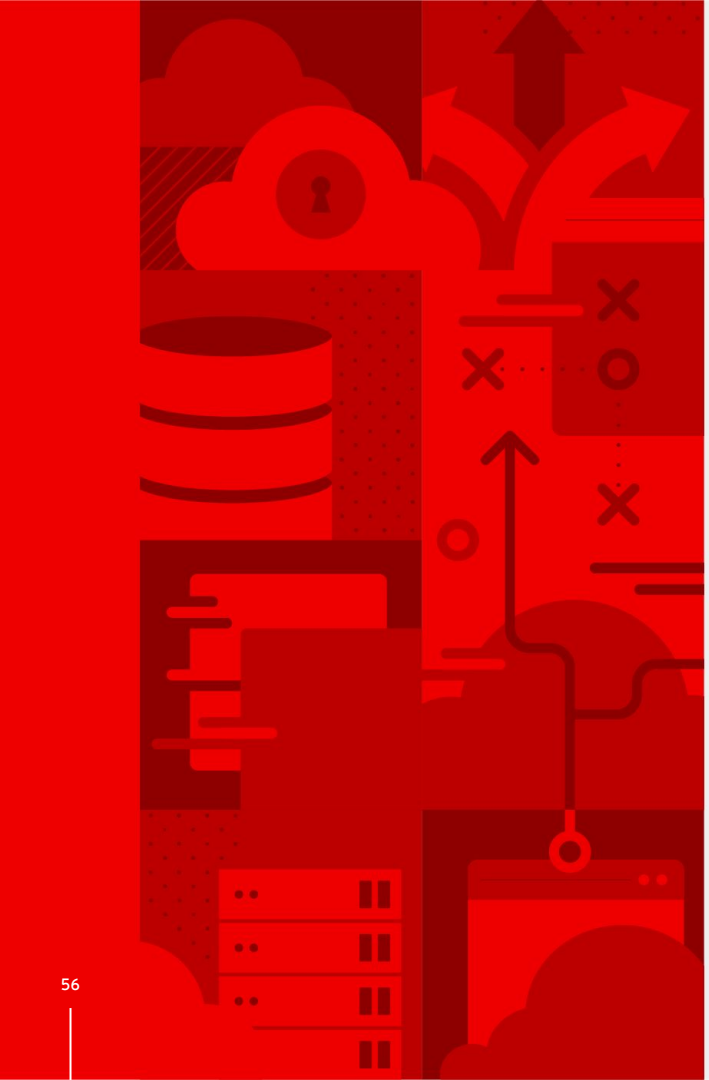


Red Hat Quay

DEMO TIME

https://github.com/lainie-ftw/demo-quay-on-openshift

Red Hat



Thank you!

linkedin.com/company/red-hat

facebook.com/redhatinc

youtube.com/user/RedHatVideos

twitter.com/RedHat

Red Hat