



# OPENSIFT CONTAINER PLATFORM

## TECHNICAL OVERVIEW

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

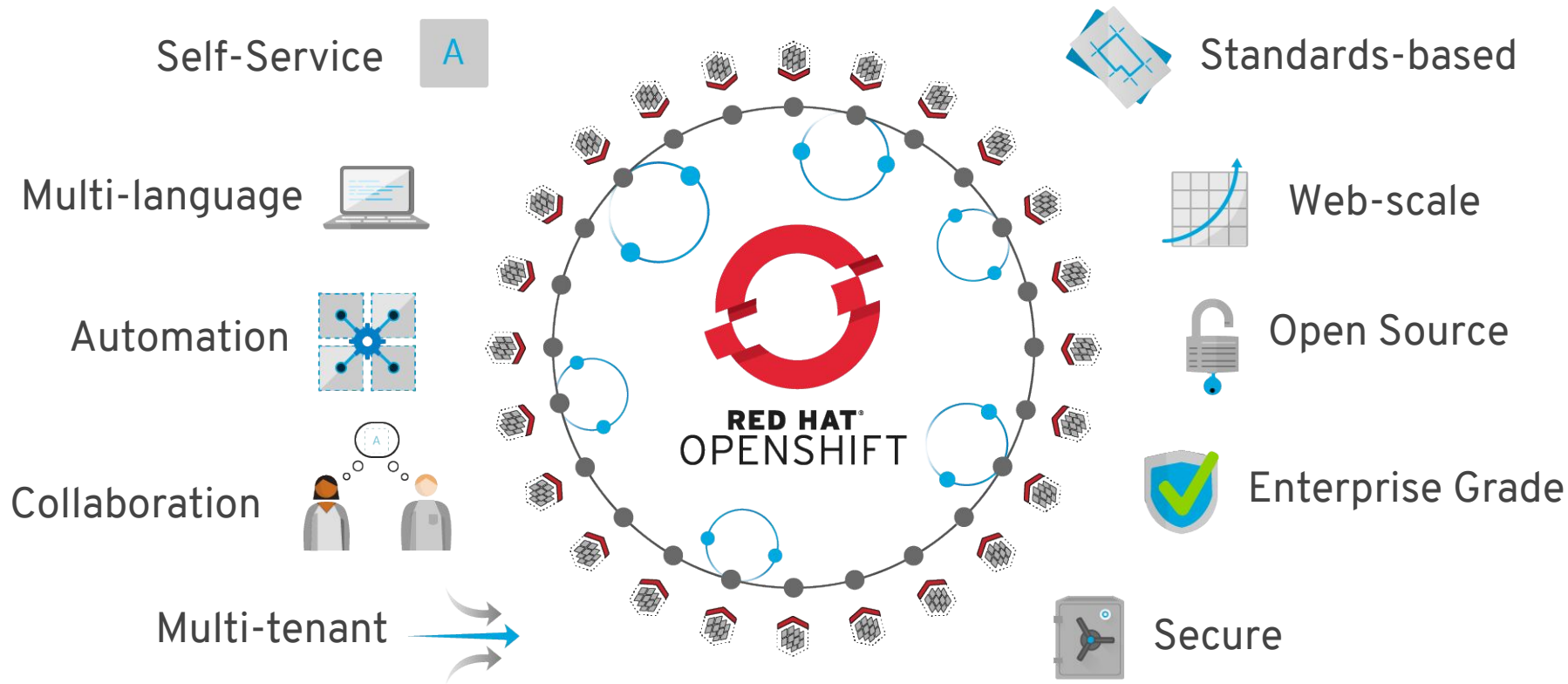
 [facebook.com/redhatinc](https://www.facebook.com/redhatinc)

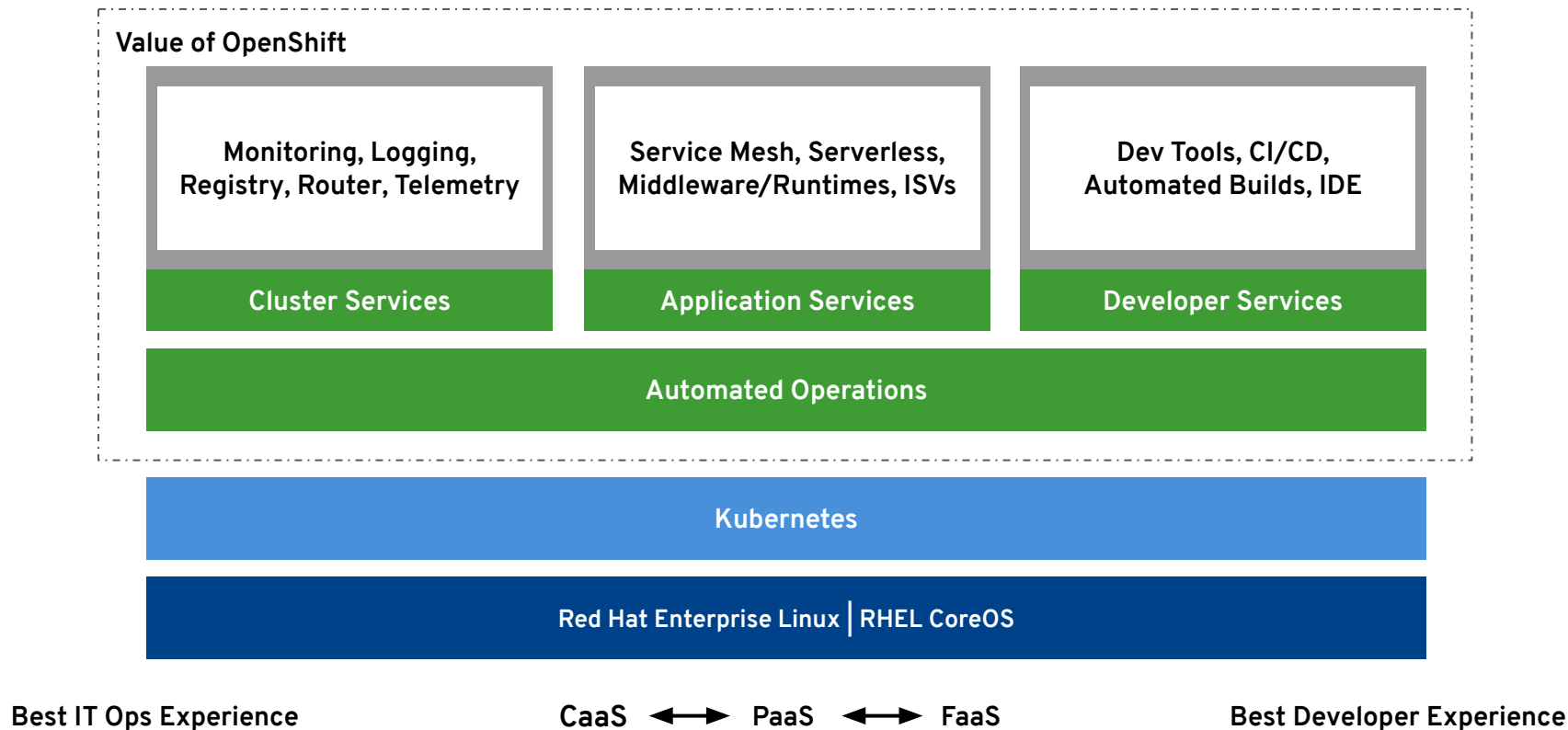
 [youtube.com/user/RedHatVideos](https://www.youtube.com/user/RedHatVideos)

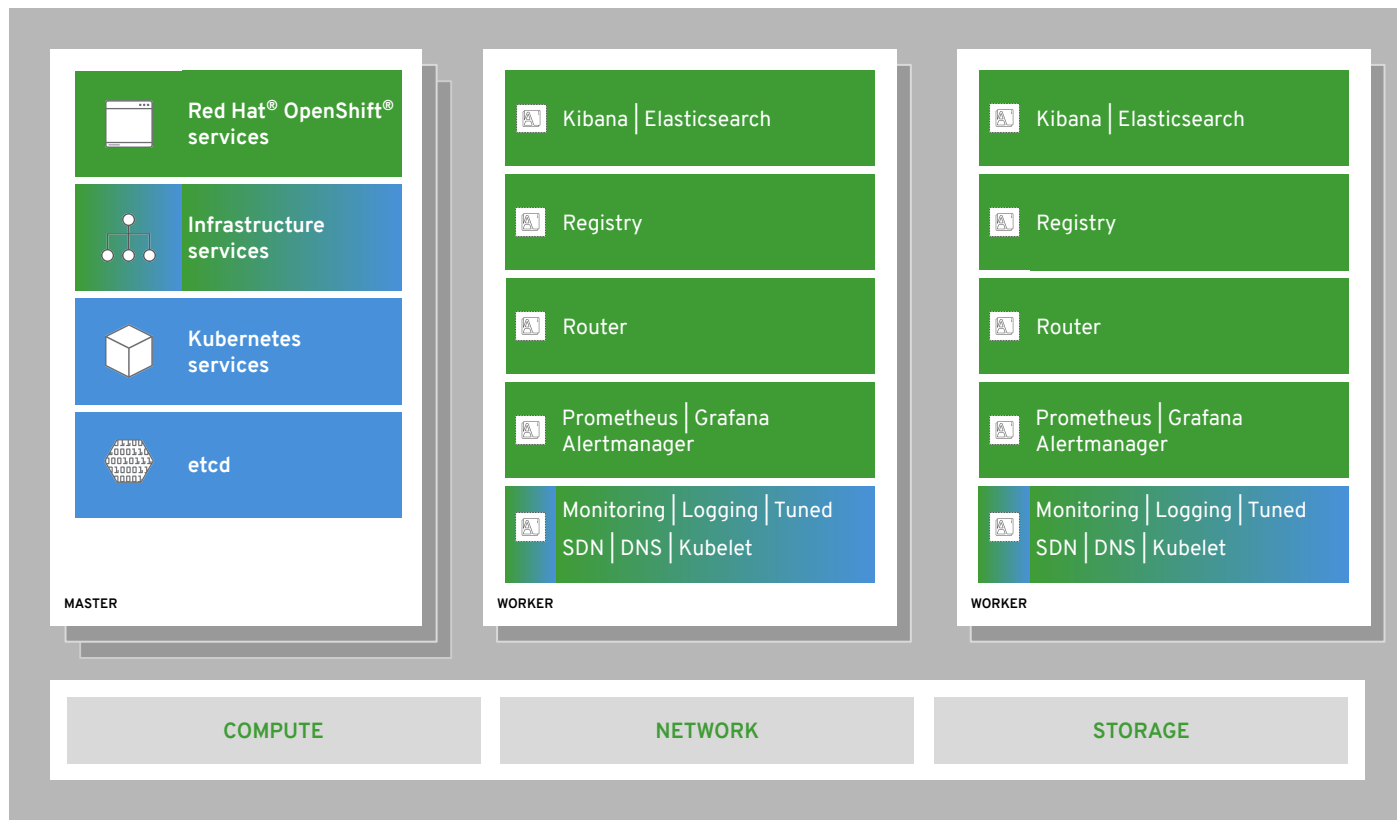
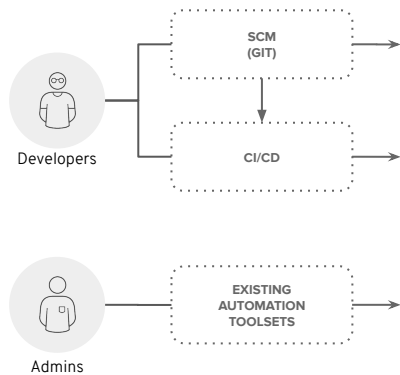
 [twitter.com/RedHat](https://twitter.com/RedHat)



# Functional overview







Database

Streaming & Messaging

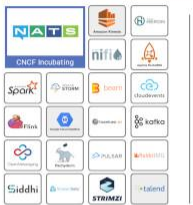
Application Definition & Image Build

Continuous Integration & Delivery

Platform

Observability and Analysis

App Definition and Development



Orchestration & Management



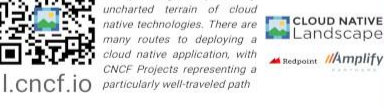
Runtime




Provisioning



Cloud





# OpenShift and Kubernetes core concepts

# a container is the smallest compute unit

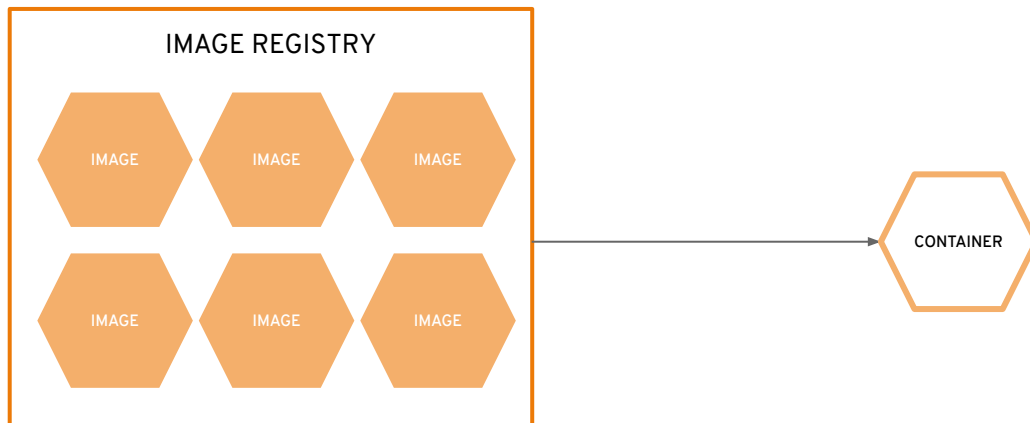




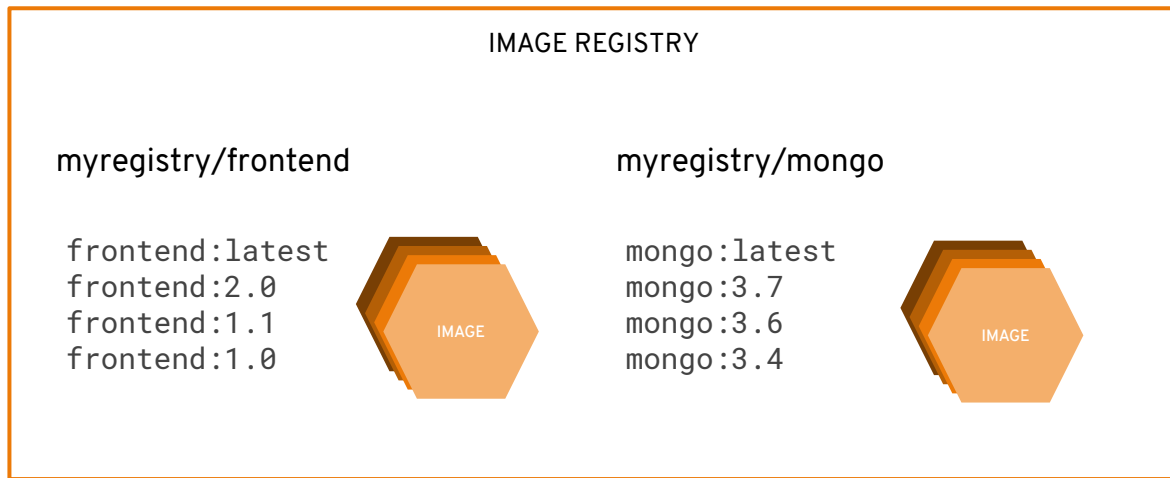
# containers are created from container images



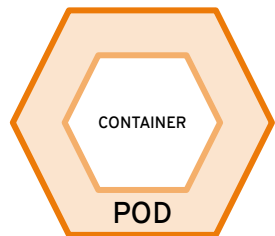
# container images are stored in an image registry



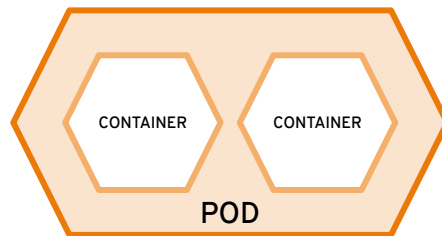
# an image repository contains all versions of an image in the image registry



# containers are wrapped in pods which are units of deployment and management

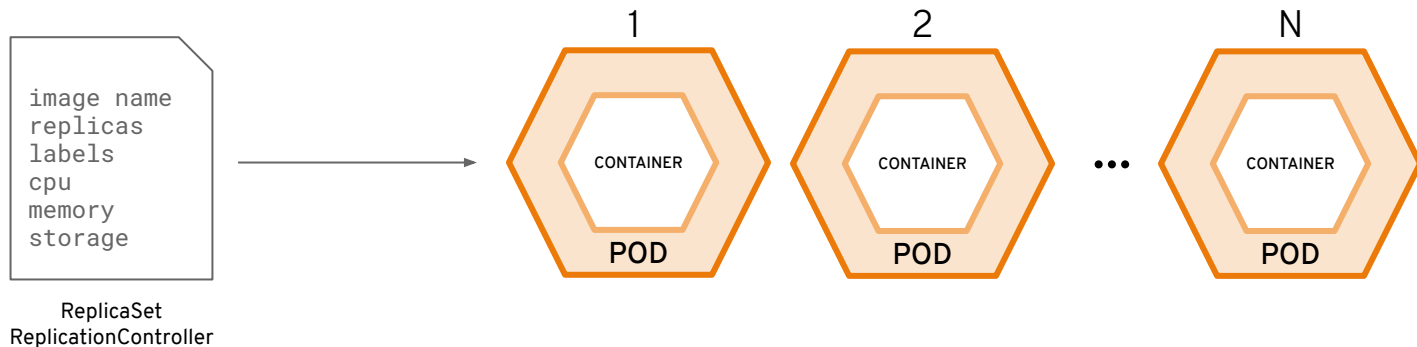


10.140.4.44

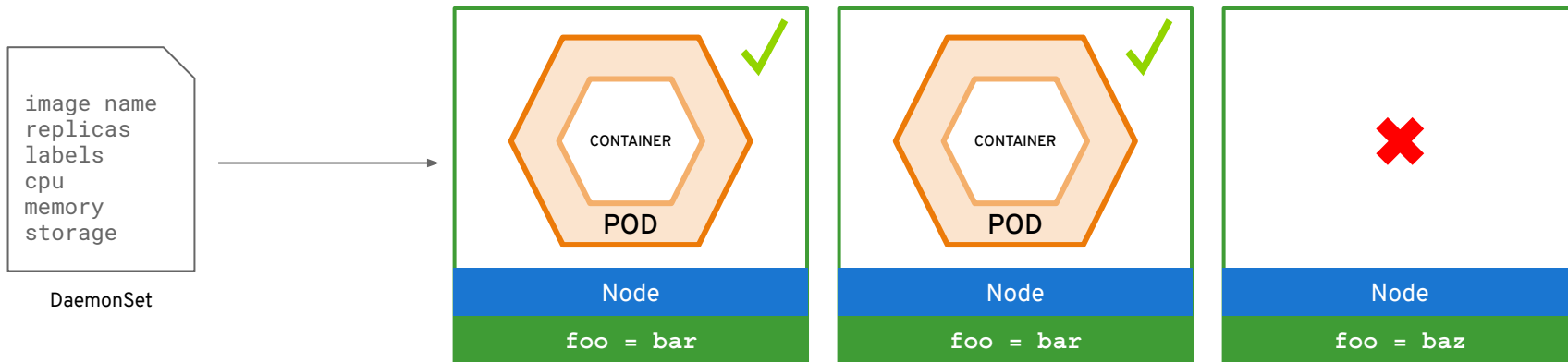


10.15.6.55

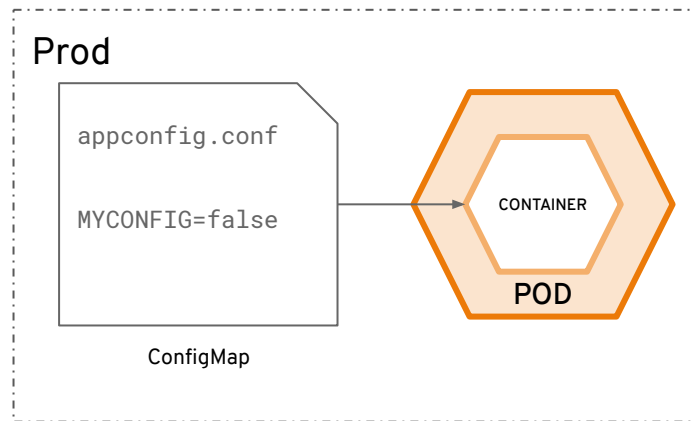
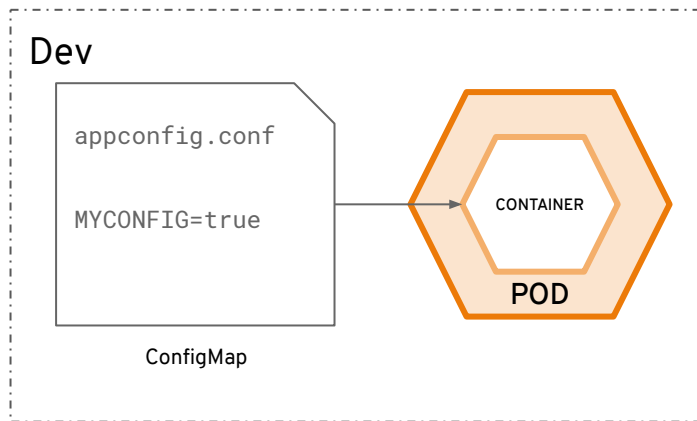
# ReplicationControllers & ReplicaSets ensure a specified number of pods are running at any given time



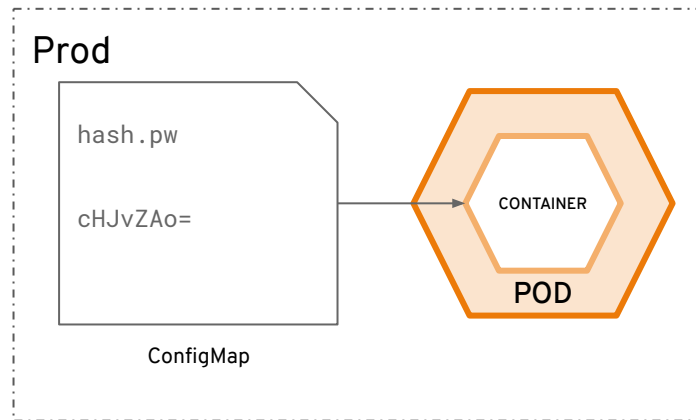
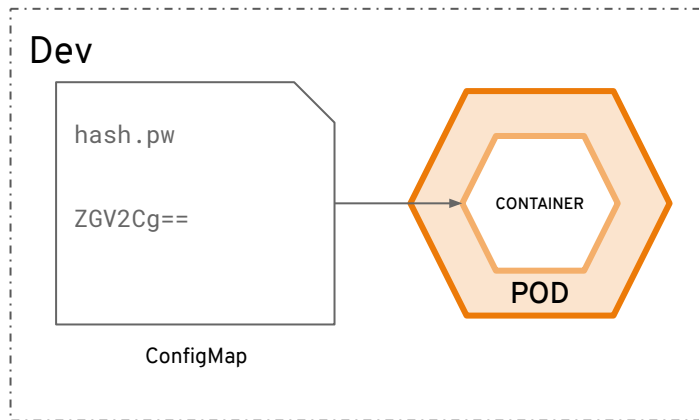
a daemonset ensures that all  
(or some) nodes run a copy of a  
pod



# configmaps allow you to decouple configuration artifacts from image content

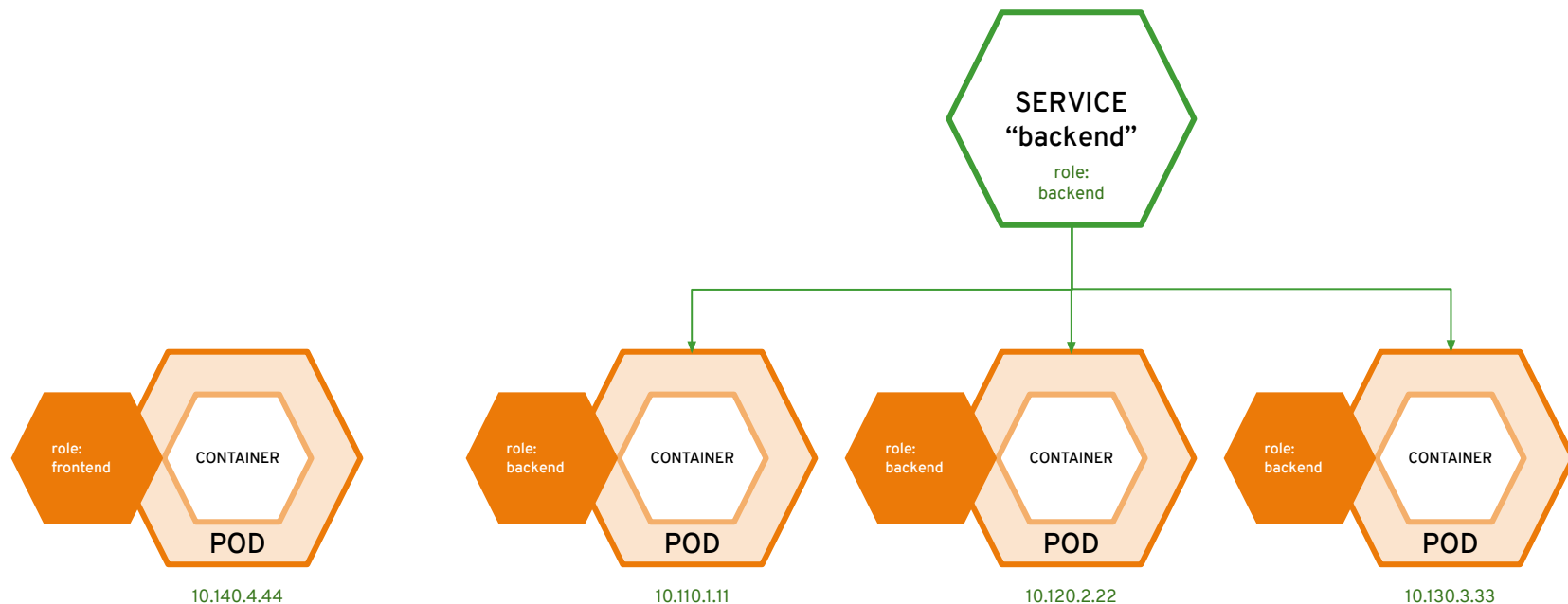


secrets provide a mechanism to hold sensitive information such as passwords

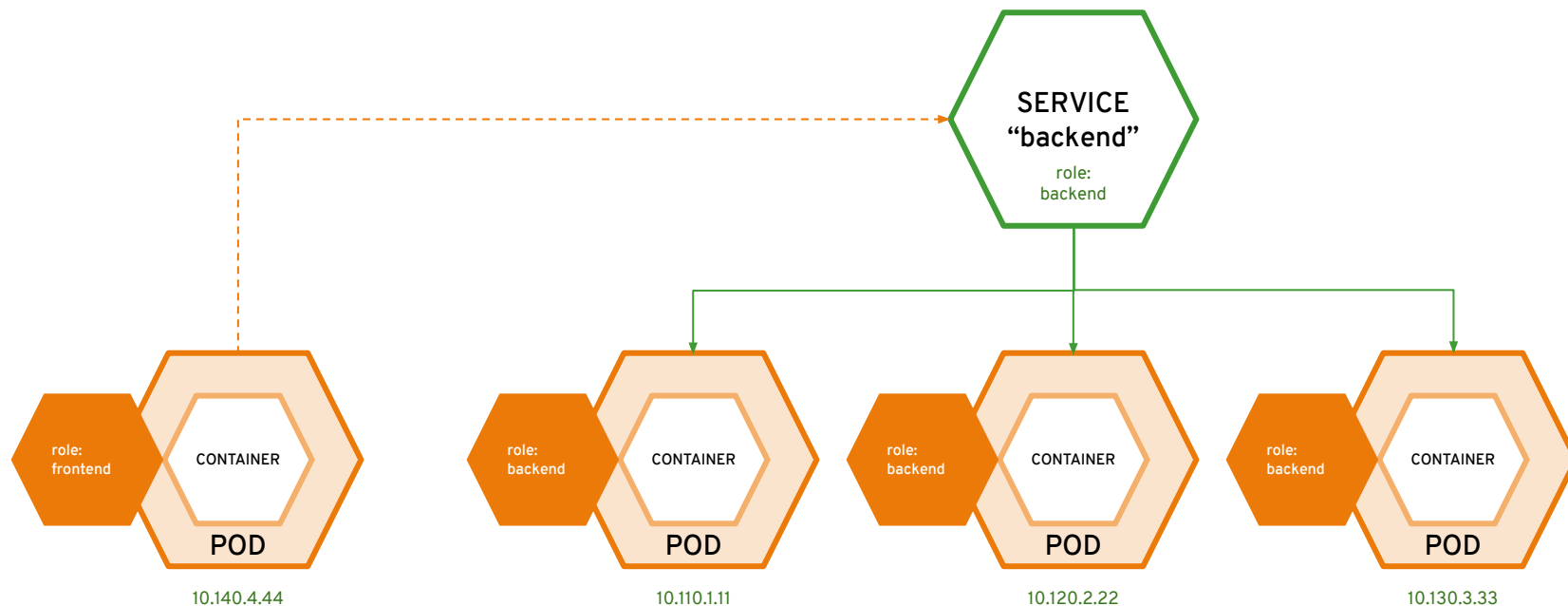




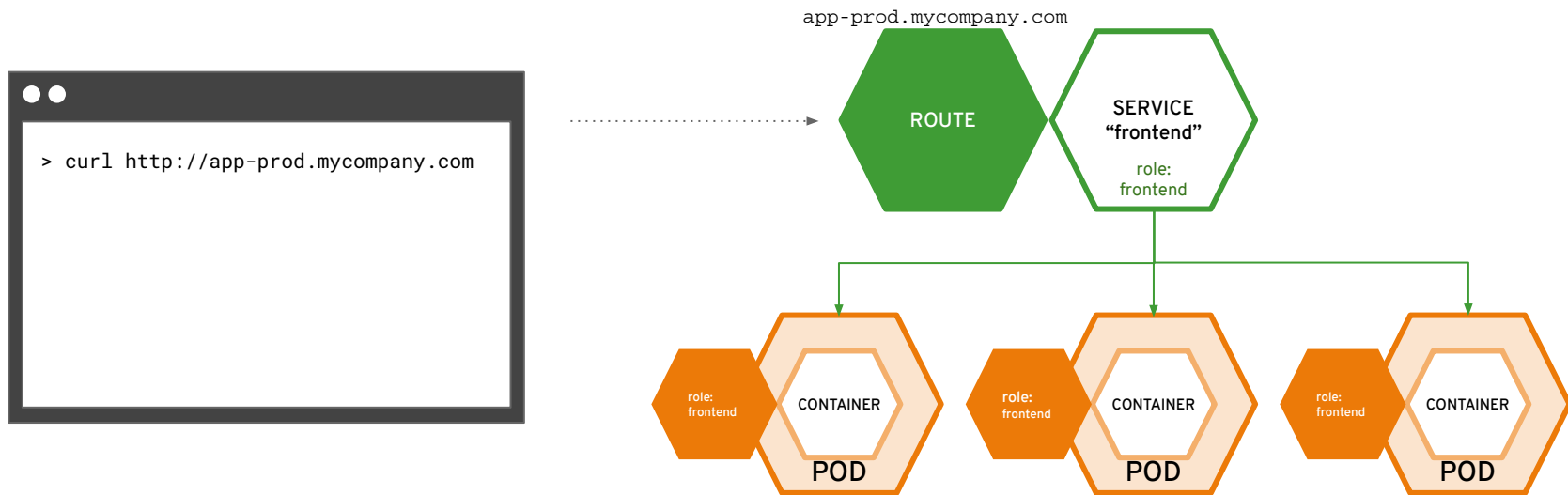
# services provide internal load-balancing and service discovery across pods



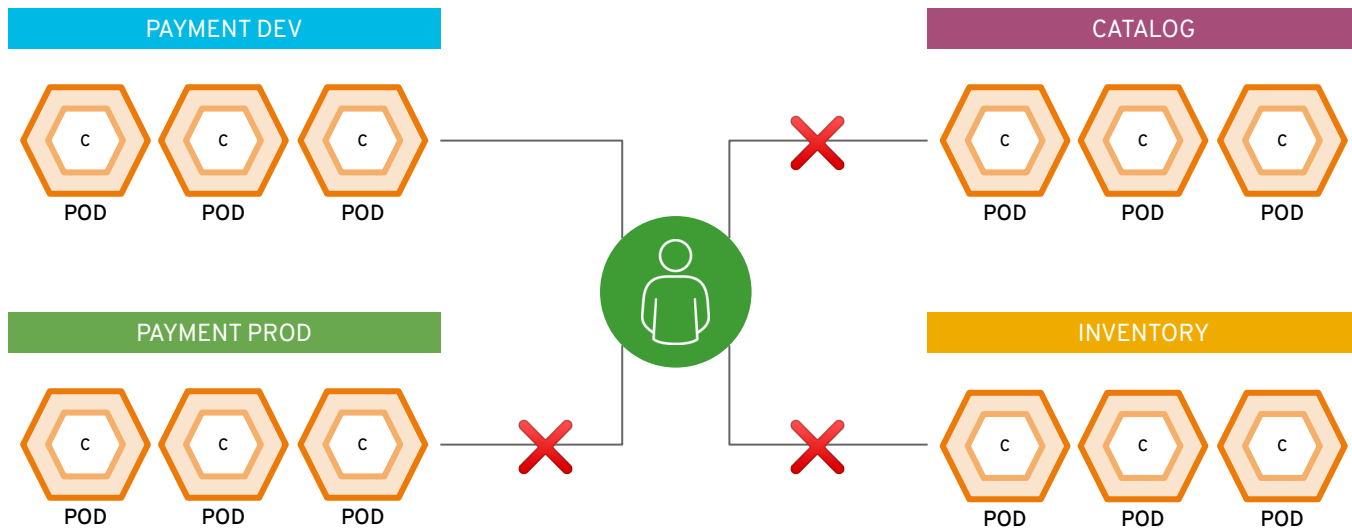
# apps can talk to each other via services



# routes make services accessible to clients outside the environment via real-world urls



projects isolate apps across environments,  
teams, groups and departments





# OpenShift lifecycle, installation & upgrades

# OpenShift 4 Installation

Two new paradigms for  
deploying clusters

# Installation Paradigms

## OPENSIFT CONTAINER PLATFORM

### Full Stack Automated

Simplified opinionated “Best Practices” for cluster provisioning

Fully automated installation and updates including host container OS.



### Pre-existing Infrastructure

Customer managed resources & infrastructure provisioning

Plug into existing DNS and security boundaries



## HOSTED OPENSIFT

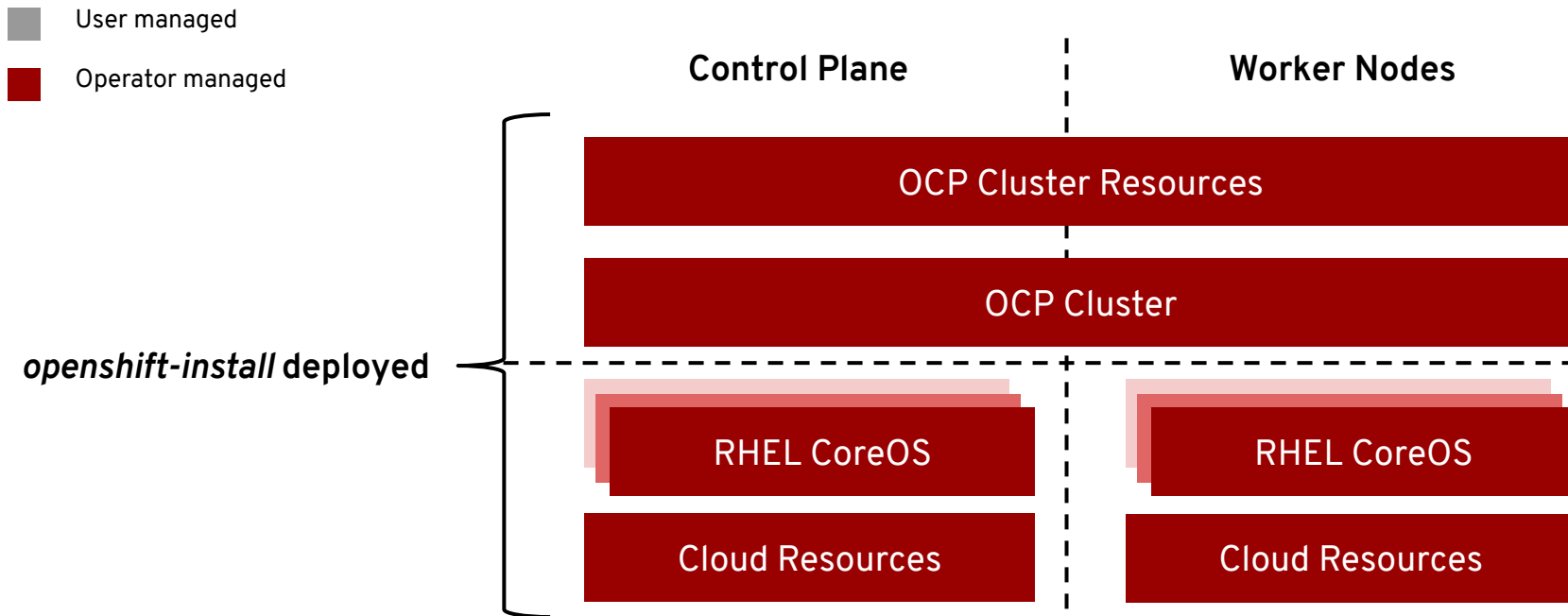
### Azure Red Hat OpenShift

Deploy directly from the Azure console. Jointly managed by Red Hat and Microsoft Azure engineers.

### OpenShift Dedicated

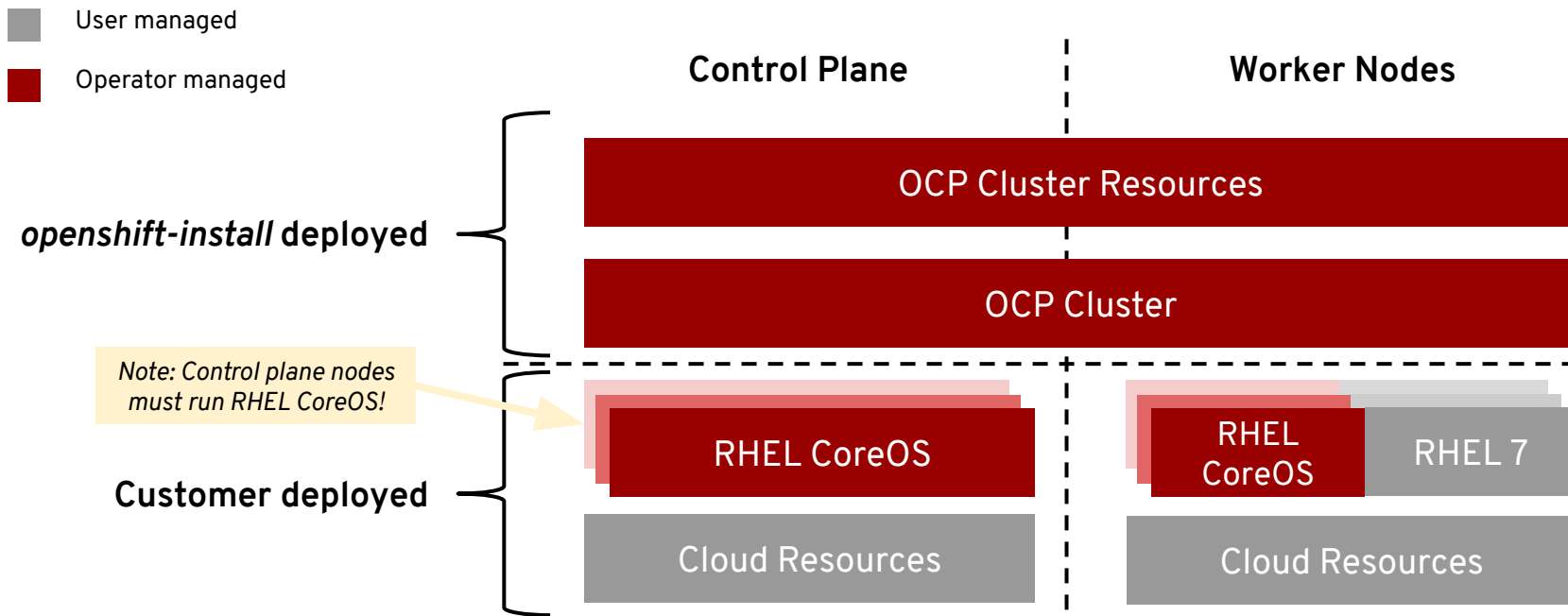
Get a powerful cluster, fully Managed by Red Hat engineers and support.

# Full-stack Automated Installation





## Pre-existing Infrastructure Installation



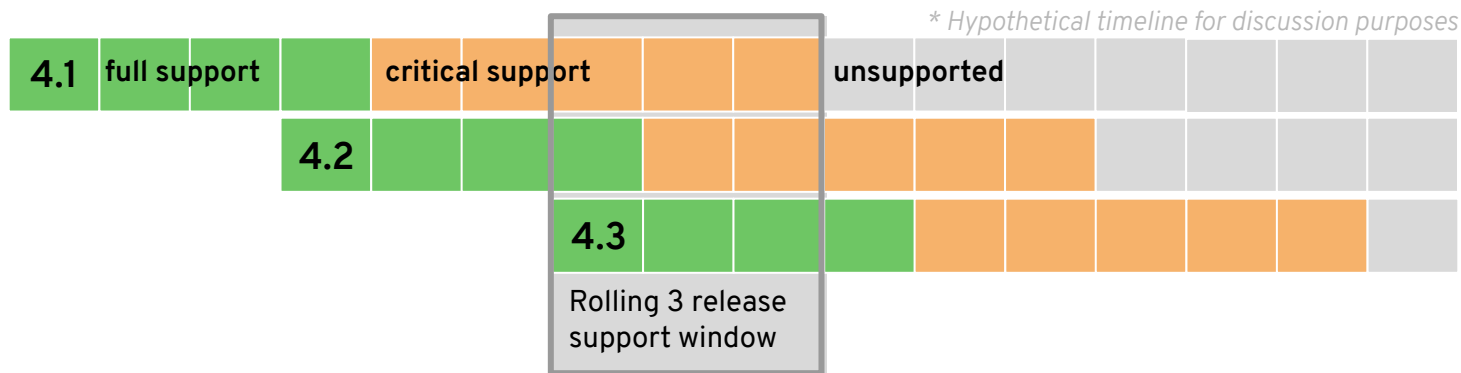
## Comparison of Paradigms

	Full Stack Automation	Pre-existing Infrastructure
Build Network	Installer	User
Setup Load Balancers	Installer	User
Configure DNS	Installer	User
Hardware/VM Provisioning	Installer	User
OS Installation	Installer	User
Generate Ignition Configs	Installer	Installer
OS Support	Installer: RHEL CoreOS	User: RHEL CoreOS + RHEL 7
Node Provisioning / Autoscaling	Yes	Only for providers with OpenShift Machine API support

# OpenShift 4 Lifecycle

Supported paths for  
upgrades and migrations

# Support Timelines



## New model

Release based, not date based. Rolling three release window for support.

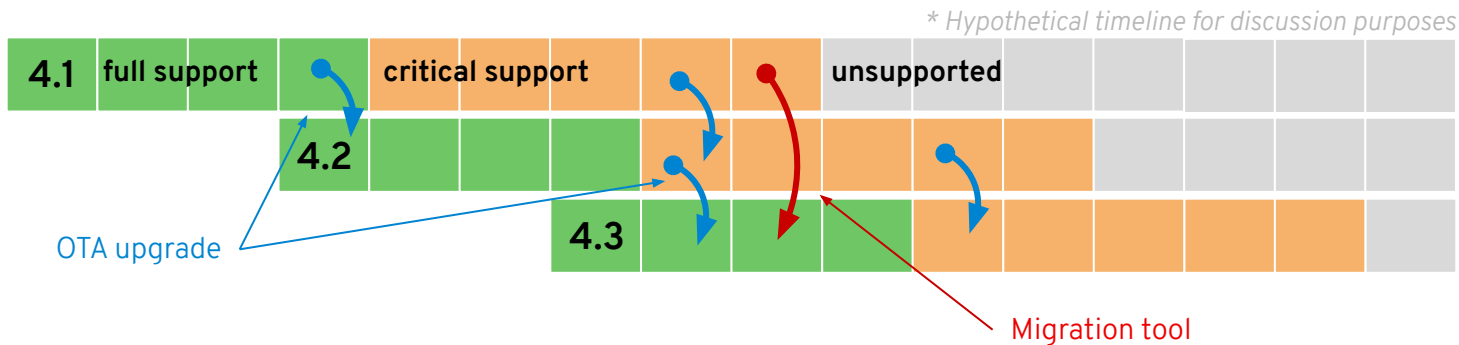
The overall 4 series will be supported for at least three years

- Minimum two years full support (likely more)
- One year maintenance past the end of full support

## EUS release planned

Supported for 14 months of critical bug and critical security fixes instead of the normal 5 months. If you stay on the EUS for its entire life, you must use the application migration tooling to move to a new cluster

# Upgrades vs. Migrations



## OTA Upgrades

Works between two minor releases in a serial manner.

## Happy path = migrate through each version

On a regular cadence, migrate to the next supported version.

## Optional path = migration tooling

If you fall more than two releases behind, you must use the application migration tooling to move to a new cluster.

## Current minor release

Full support for all bugs and security issues  
1 month full support overlap with next release to aid migrations

## Previous minor release

Fixes for critical bugs and security issues for 5 months

An abstract graphic on the left side of the slide, rendered in various shades of red. It features a vertical stack of server racks at the bottom, a cloud with a keyhole icon, a large upward-pointing arrow, and several 'X' marks and smaller arrows, suggesting a process or flow of data.

# Operations and infrastructure deep dive

# Red Hat Enterprise Linux CoreOS

The OpenShift operating  
system

# Red Hat Enterprise Linux

## RED HAT® ENTERPRISE LINUX®

General Purpose OS

## RED HAT® ENTERPRISE LINUX CoreOS

Immutable container host

### BENEFITS

- 10+ year enterprise life cycle
- Industry standard security
- High performance on any infrastructure
- Customizable and compatible with wide ecosystem of partner solutions

- Self-managing, over-the-air updates
- Immutable and tightly integrated with OpenShift
- Host isolation is enforced via Containers
- Optimized performance on popular infrastructure

### WHEN TO USE

When customization and integration with additional solutions is required

When cloud-native, hands-free operations are a top priority



# Immutable Operating System

## Red Hat Enterprise Linux CoreOS is versioned with OpenShift

CoreOS is tested and shipped in conjunction with the platform. Red Hat runs thousands of tests against these configurations.

## Red Hat Enterprise Linux CoreOS is managed by the cluster

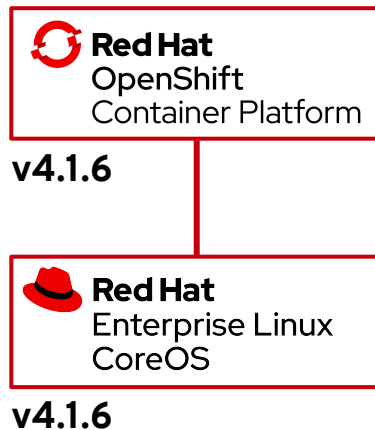
The Operating system is operated as part of the cluster, with the config for components managed by Machine Config

Operator:

- CRI-O config
- Kubelet config
- Authorized registries
- SSH config

RHEL CoreOS admins are responsible for:

Nothing. 😊 🙌





A lightweight, OCI-compliant container runtime

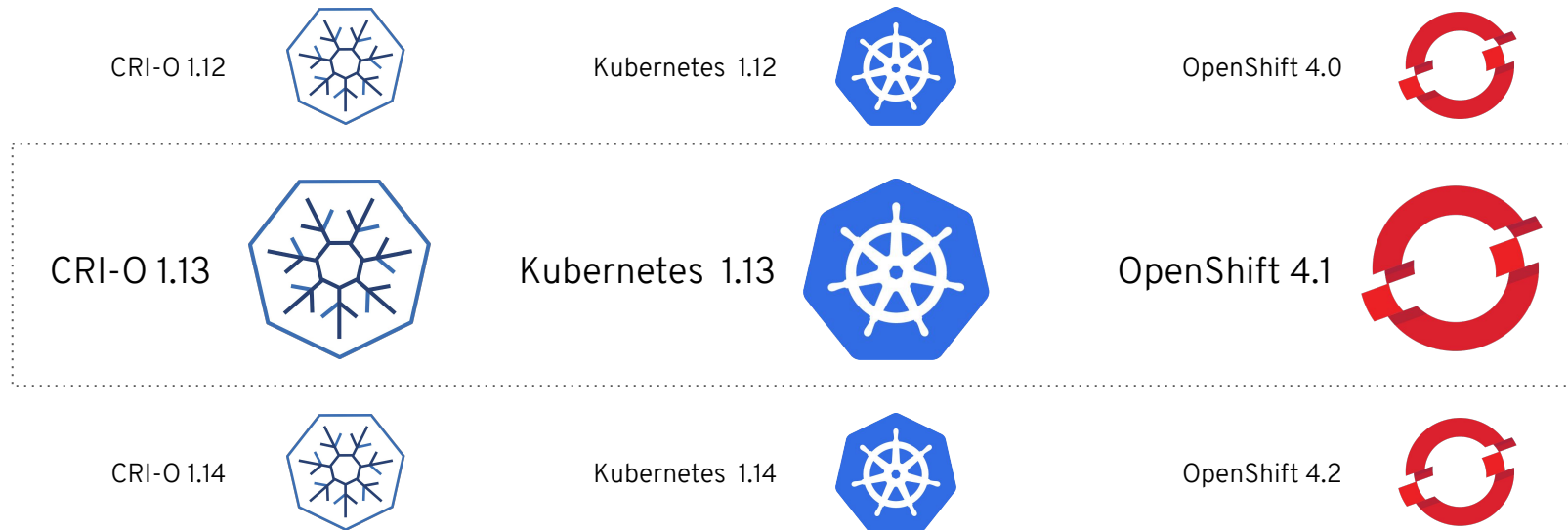
Minimal and Secure  
Architecture

Optimized for  
Kubernetes

Runs any  
OCI-compliant image  
(including docker)

# CRI-O Support in OpenShift

CRI-O tracks and versions identical to Kubernetes, simplifying support permutations



# podman



A docker-compatible CLI  
for containers

- Remote  
management API  
via Varlink
- Image/container  
tagging
- Advanced  
namespace  
isolation

buildah



buildah

Secure & flexible OCI container builds

- Integrated into OCP build pods
- Performance improvements for knative enablement
- Image signing improvements

# OpenShift 4 installation

Installer and  
user-provisioned  
infrastructure, bootstrap,  
and more

# OpenShift Bootstrap Process: Self-Managed Kubernetes

## How to boot a self-managed cluster:

- OpenShift 4 is unique in that management extends all the way down to the operating system
- Every machine boots with a configuration that references resources hosted in the cluster it joins enabling cluster to manage itself
- Downside is that every machine looking to join the cluster is waiting on the cluster to be created
- Dependency loop is broken using a bootstrap machine, which acts as a temporary control plane whose sole purpose is bringing up the permanent control plane nodes
- Permanent control plane nodes get booted and join the cluster leveraging the control plane on the bootstrap machine
- Once the pivot to the permanent control plane takes place, the remaining worker nodes can be booted and join the cluster

## Bootstrapping process step by step:

1. Bootstrap machine boots and starts hosting the remote resources required for master machines to boot.
2. Master machines fetch the remote resources from the bootstrap machine and finish booting.
3. Master machines use the bootstrap node to form an etcd cluster.
4. Bootstrap node starts a temporary Kubernetes control plane using the newly-created etcd cluster.
5. Temporary control plane schedules the production control plane to the master machines.
6. Temporary control plane shuts down, yielding to the production control plane.
7. Bootstrap node injects OpenShift-specific components into the newly formed control plane.
8. Installer then tears down the bootstrap node or if user-provisioned, this needs to be performed by the administrator.

# How everything deployed comes under management

## Masters (Special)

- Terraform provisions initial masters\*
- Machine API adopts existing masters post-provision
- Each master is a standalone Machine object
- Termination protection (avoid self-destruction)

## Workers

- Each Machine Pool corresponds to MachineSet
- Optionally autoscale (min,max) and health check (replace if not ready > X minutes)

## Multi-AZ

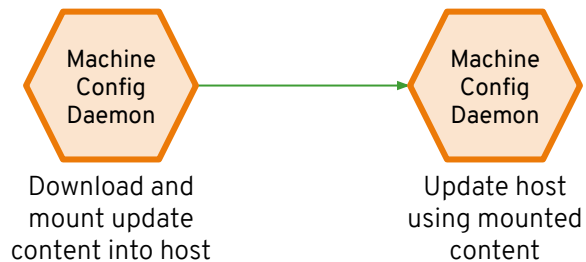
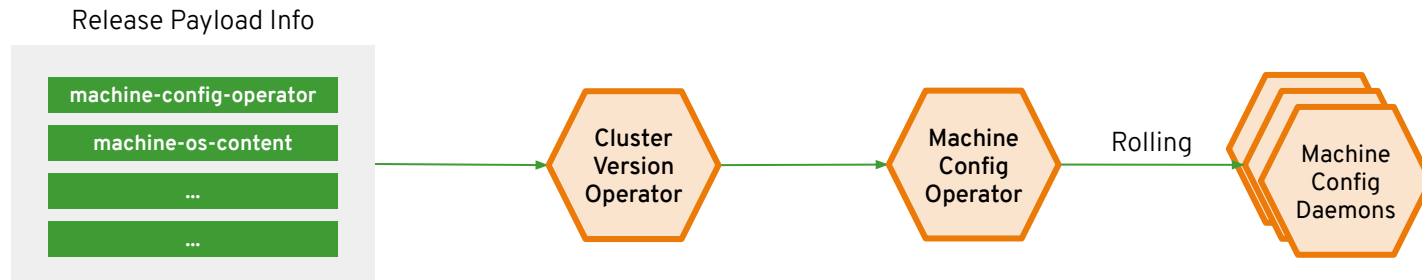
- MachineSets scoped to single AZ
- Installer stripes N machine sets across AZs by default
- Post-install best effort balance via cluster autoscaler



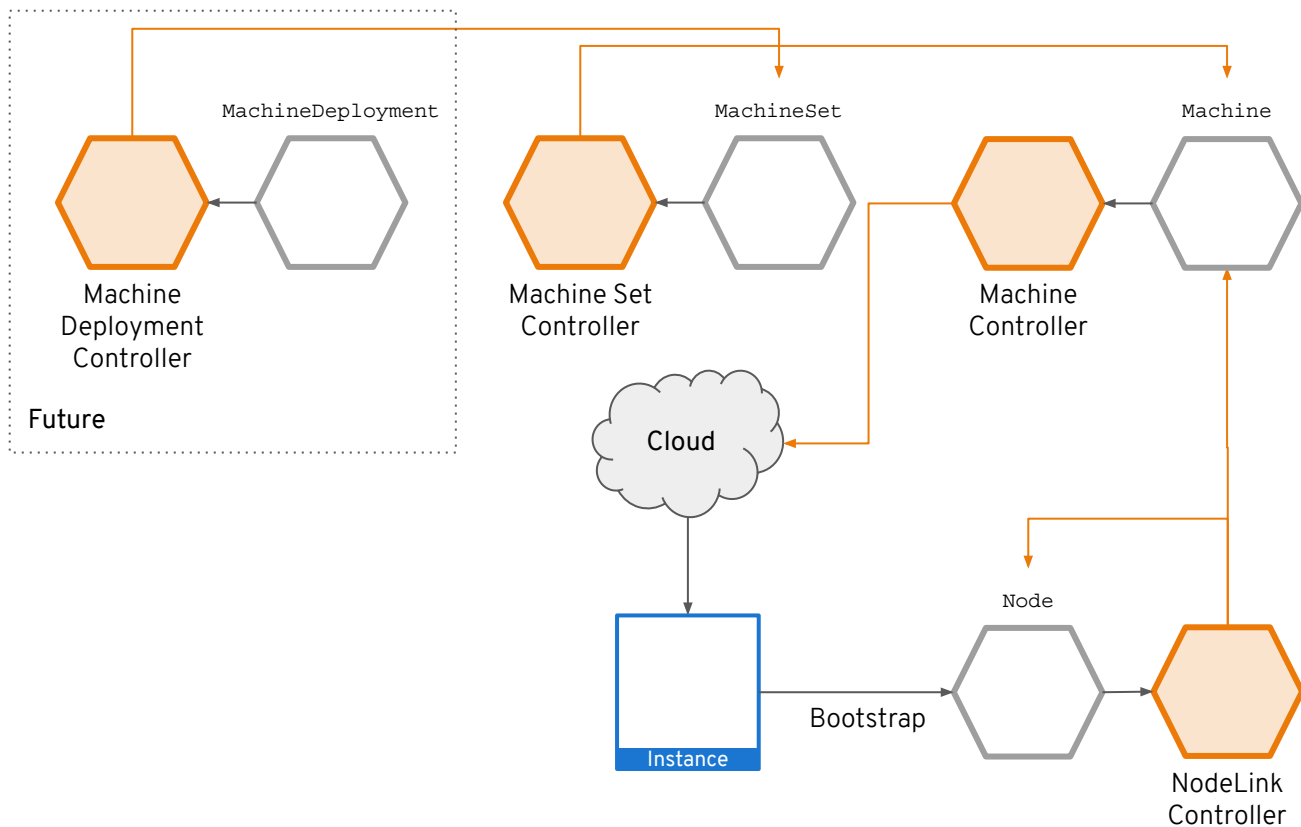
# OpenShift 4 Cluster Management

Powered by Operators,  
OpenShift 4 automates  
many cluster  
management activities

# Over-the-air updates



# Cloud API



# OpenShift Security

Features, mechanisms  
and processes for  
container and platform  
isolation



## CONTROL

Application  
Security

Container Content

CI/CD Pipeline

Container Registry

Deployment Policies



## DEFEND

Infrastructure

Container Platform

Container Host Multi-tenancy

Network Isolation

Storage

Audit & Logging

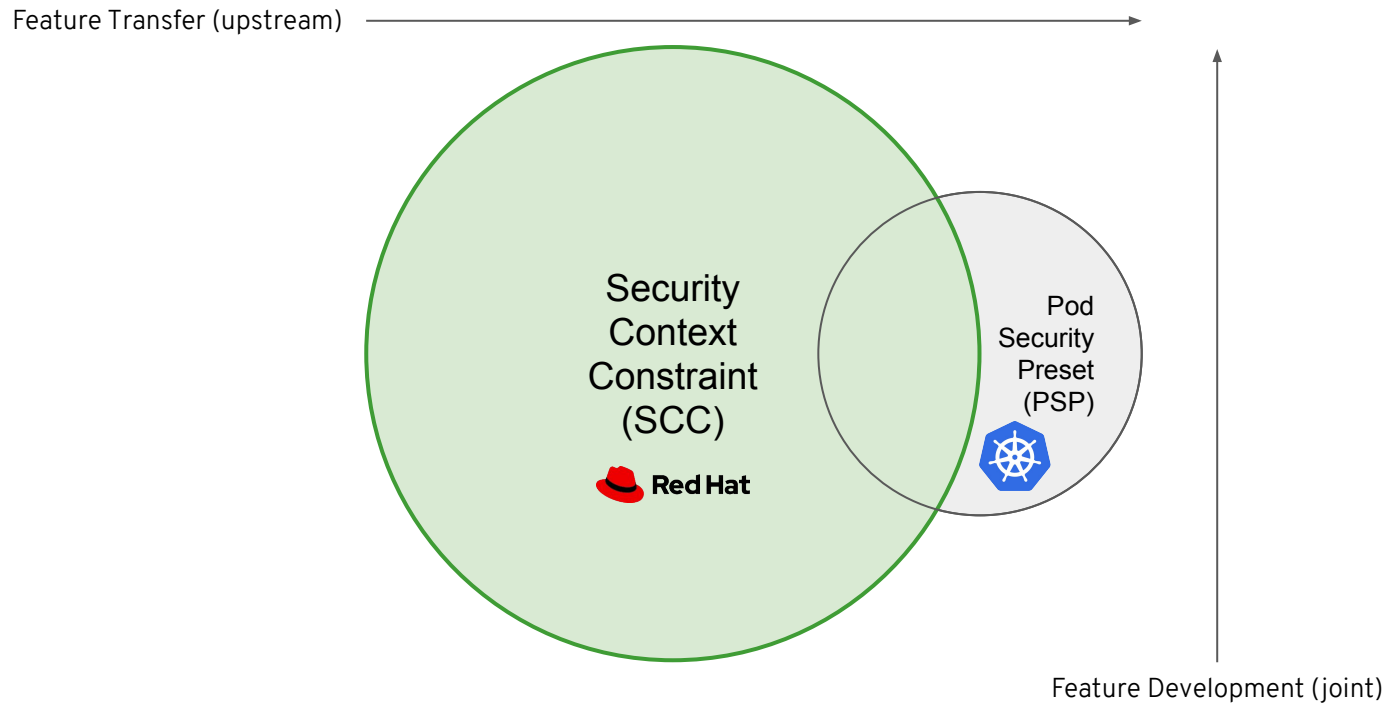
API Management



## EXTEND

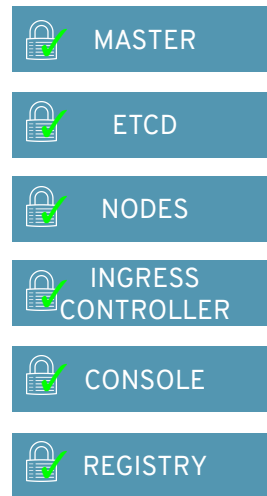
Security Ecosystem

## Extended Depth of Protection

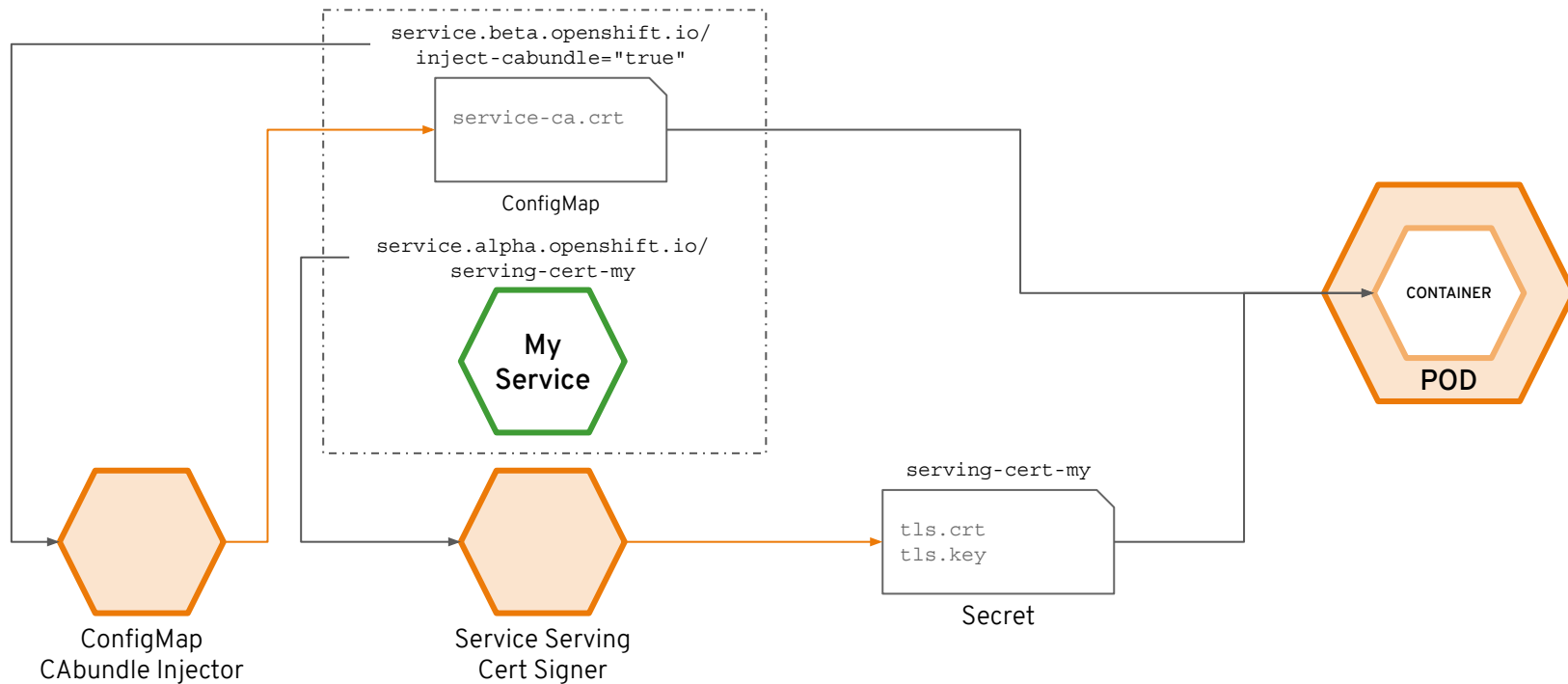


# Certificates and Certificate Management

- OpenShift provides its own internal CA
- Certificates are used to provide secure connections to
  - master (APIs) and nodes
  - Ingress controller and registry
  - etcd
- Certificate rotation is automated
- Optionally configure external endpoints to use custom certificates

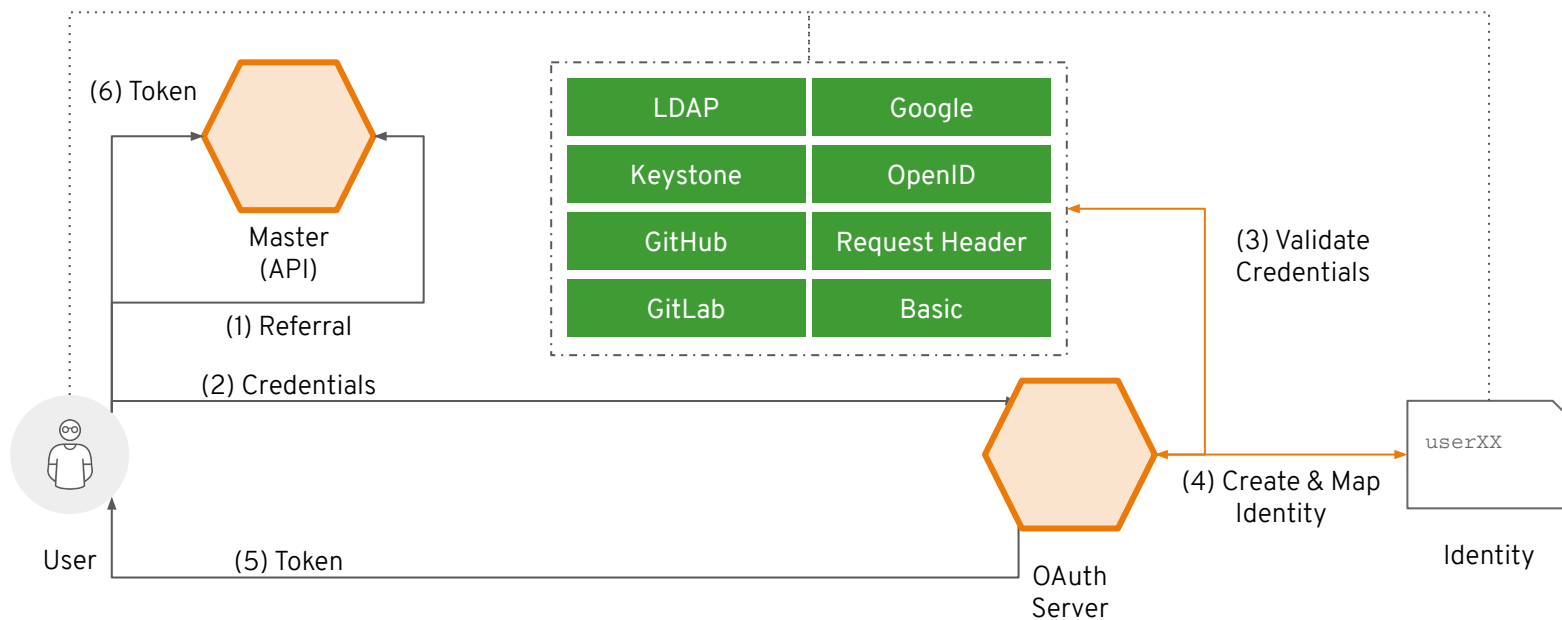


## Service Certificates





# Identity and Access Management



## Fine-Grained RBAC

- Project scope & cluster scope available
- Matches request attributes (verb,object,etc)
- If no roles match, request is denied ( deny by default )
- Operator- and user-level roles are defined by default
- Custom roles are supported

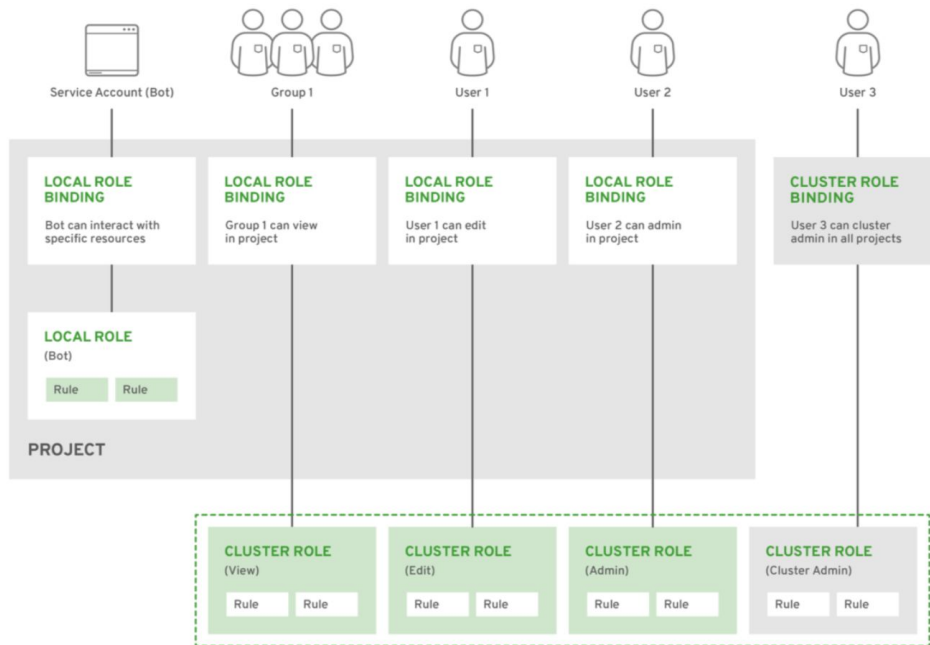


Figure 12 - Authorization Relationships

# OpenShift Monitoring

An integrated cluster  
monitoring and alerting  
stack

# OpenShift Cluster Monitoring



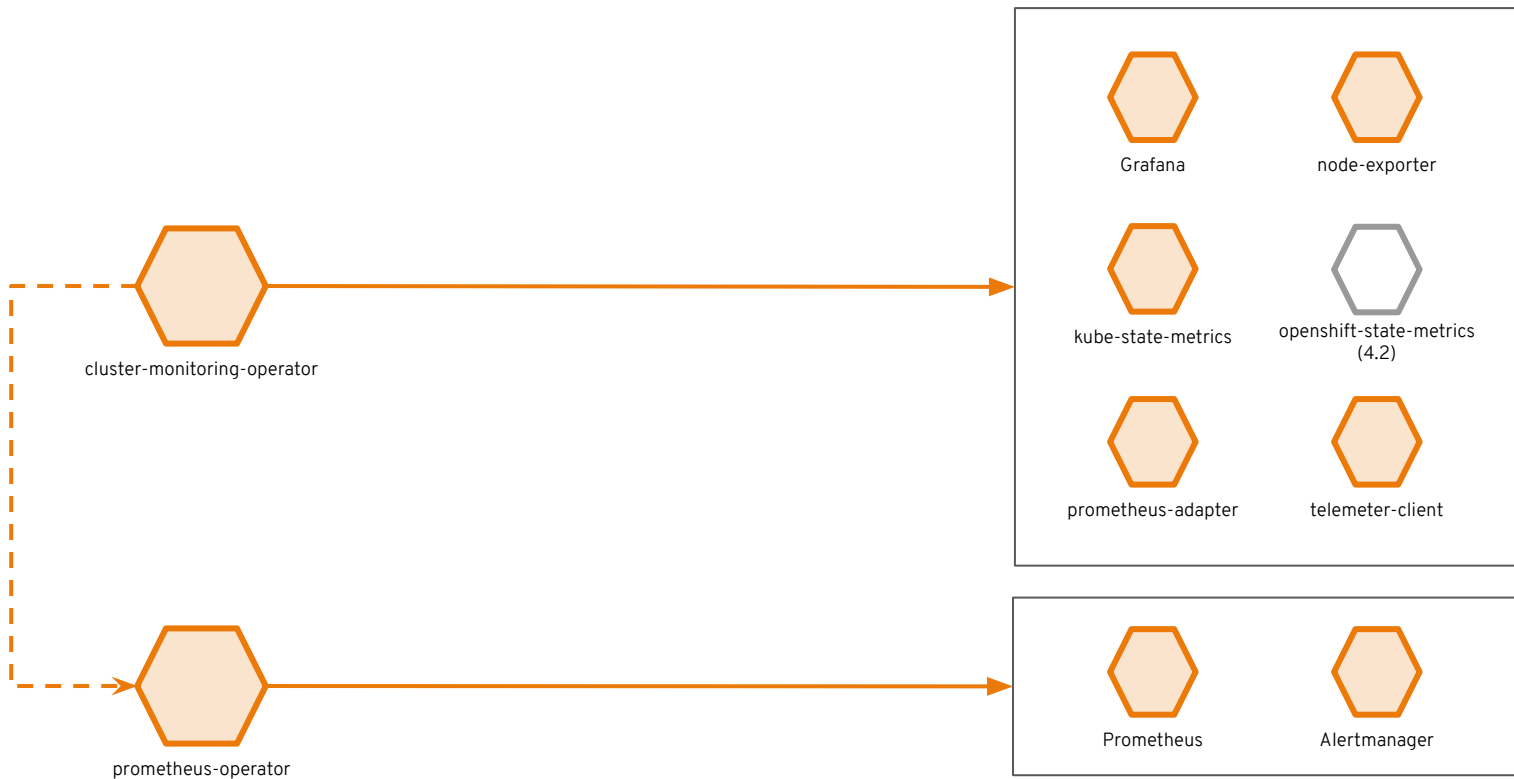
**Metrics collection and storage**  
via Prometheus, an  
open-source monitoring system  
time series database.

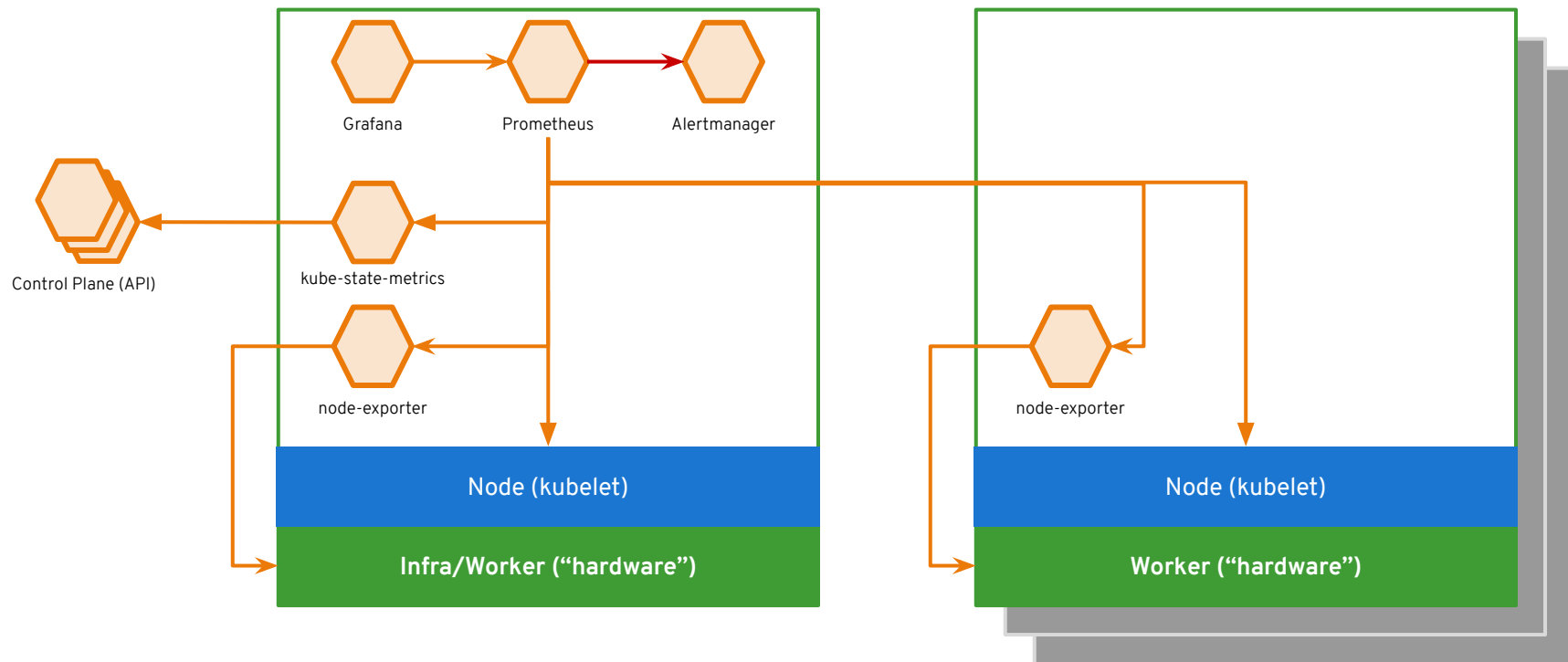


**Alerting/notification** via  
Prometheus' Alertmanager, an  
open-source tool that handles  
alerts send by Prometheus.



**Metrics visualization** via  
Grafana, the leading metrics  
visualization technology.





# OpenShift Logging

An integrated solution for  
exploring and  
corroborating application  
logs

# Observability via log exploration and corroboration with EFK

## Components

- **Elasticsearch:** a search and analytics engine to store logs
- **Fluentd:** gathers logs and sends to Elasticsearch.
- **Kibana:** A web UI for Elasticsearch.

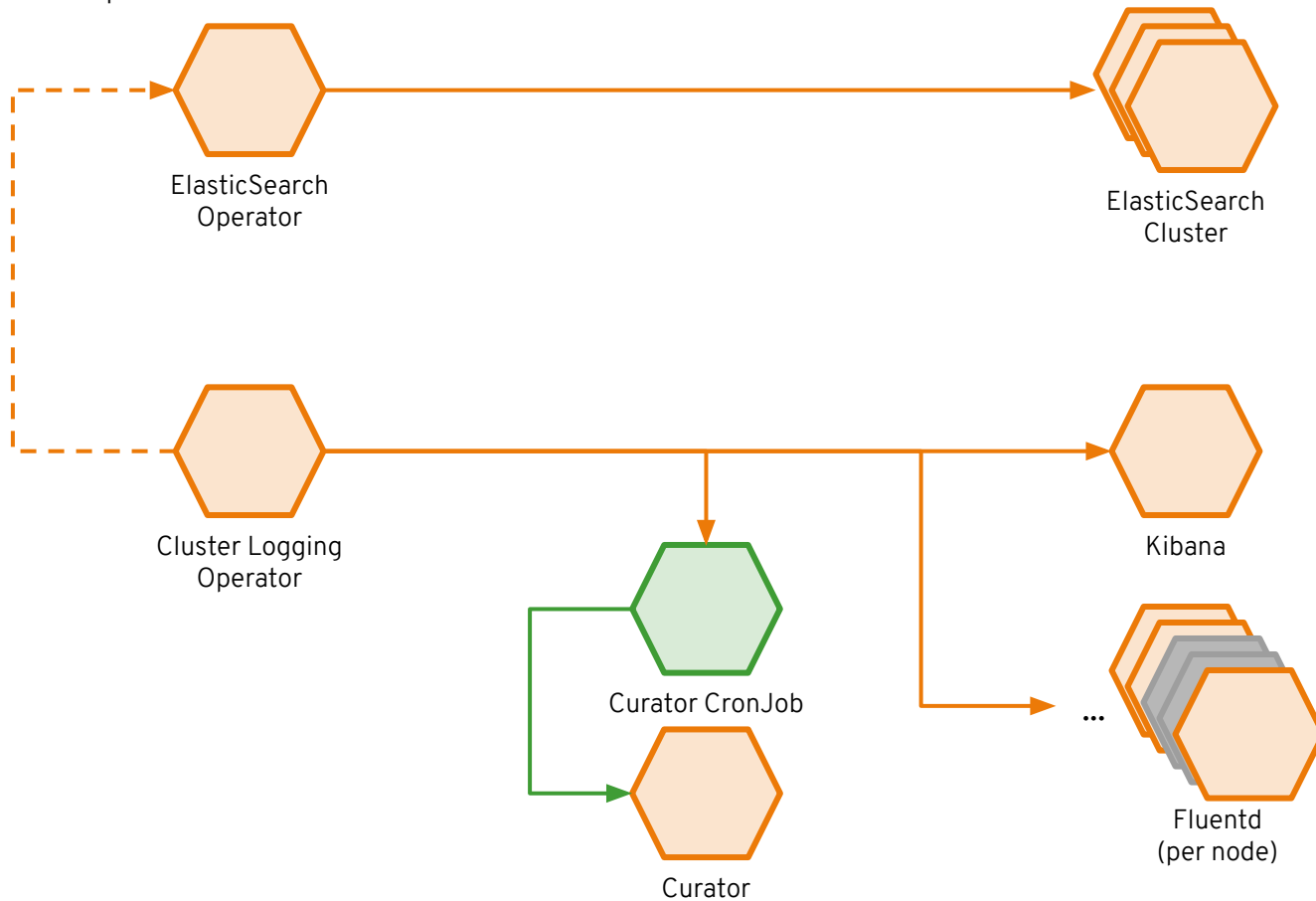
## Access control

- Cluster administrators can view all logs
- Users can only view logs for their projects

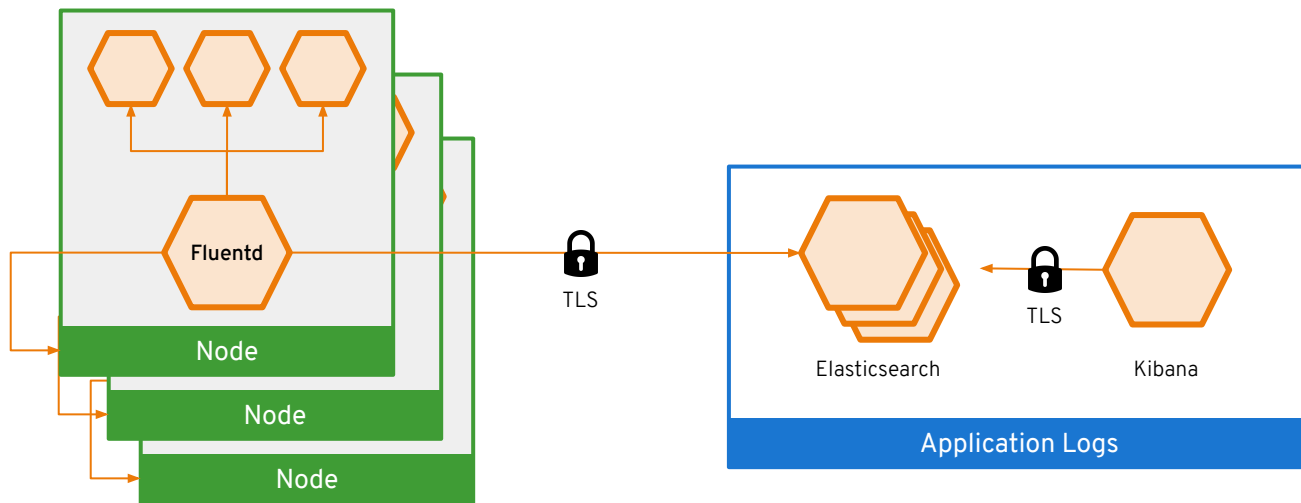
## Ability to forward logs elsewhere

- External elasticsearch, Splunk, etc

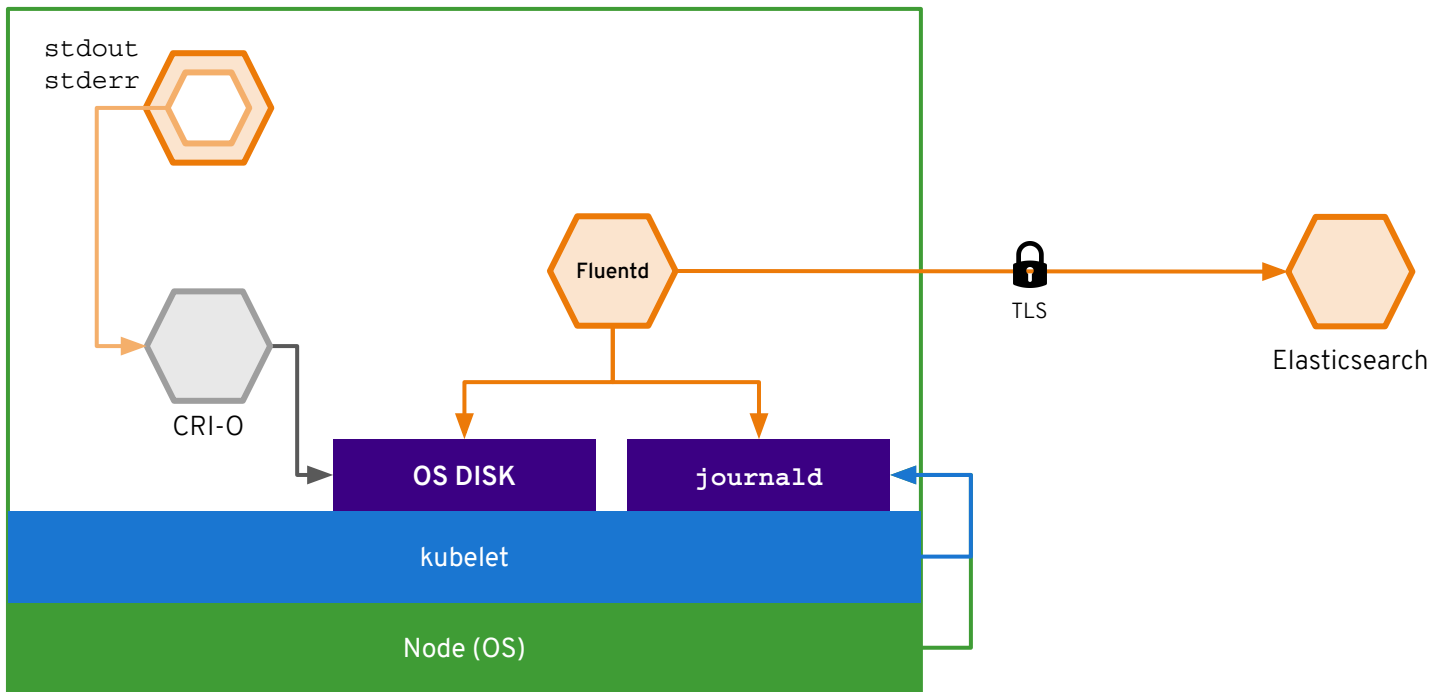




## Log data flow in OpenShift



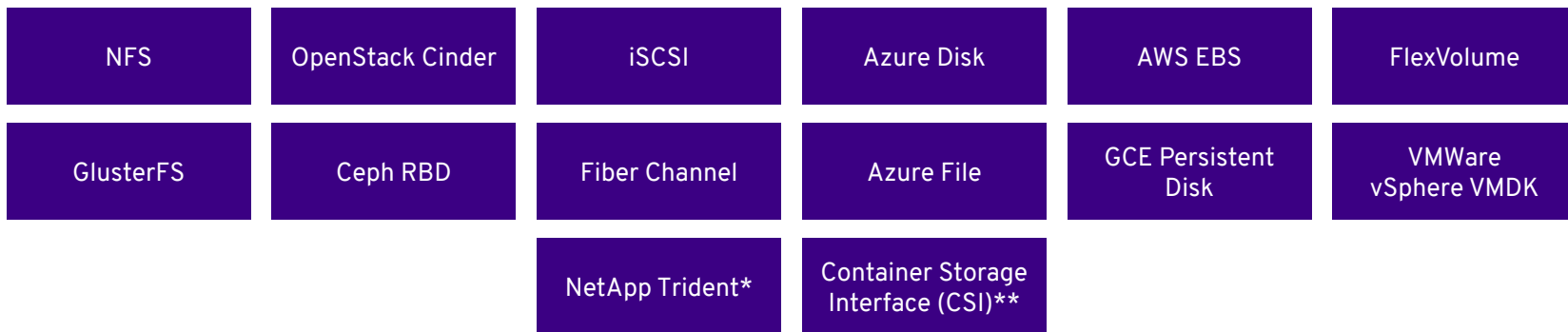
## Log data flow in OpenShift



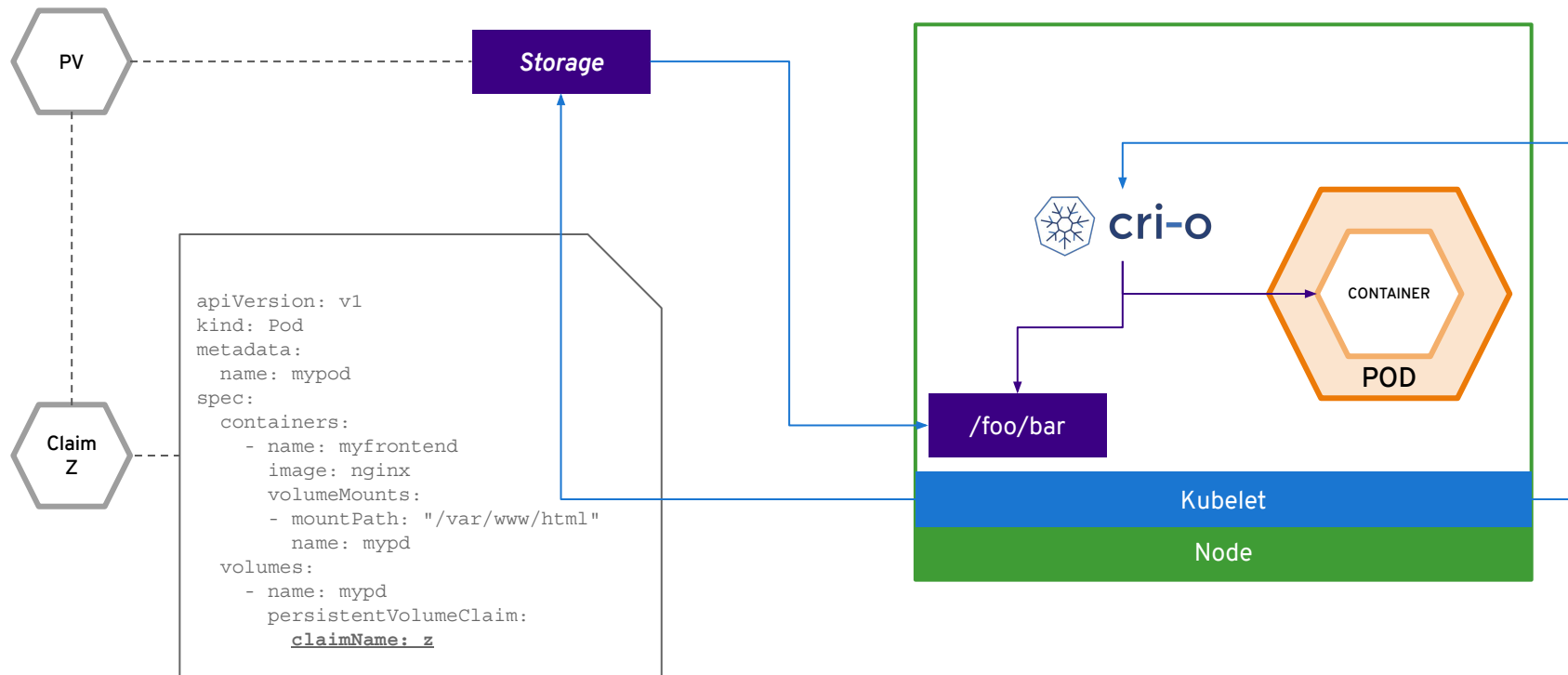
# Persistent Storage

Connecting real-world  
storage to your  
containers to enable  
stateful applications

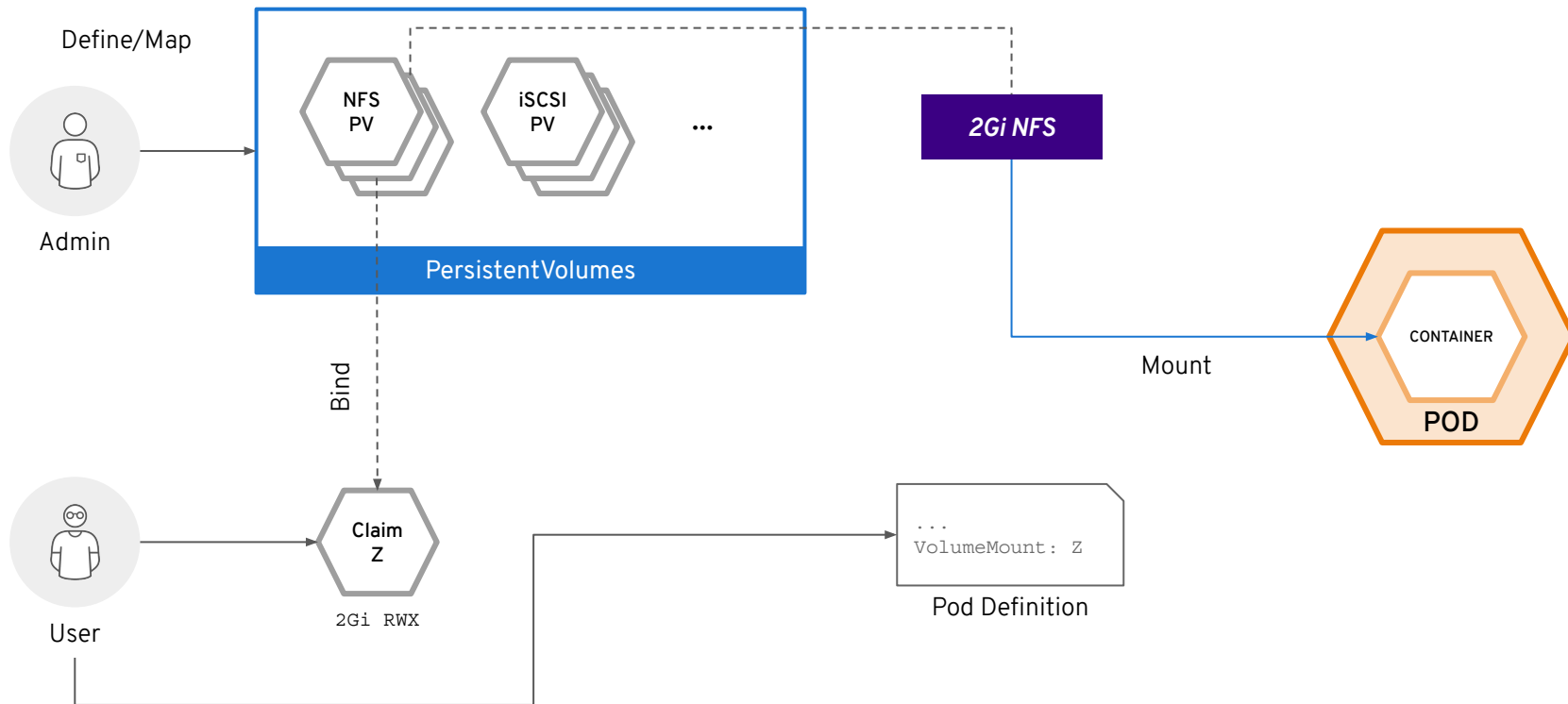
## A broad spectrum of static and dynamic storage endpoints



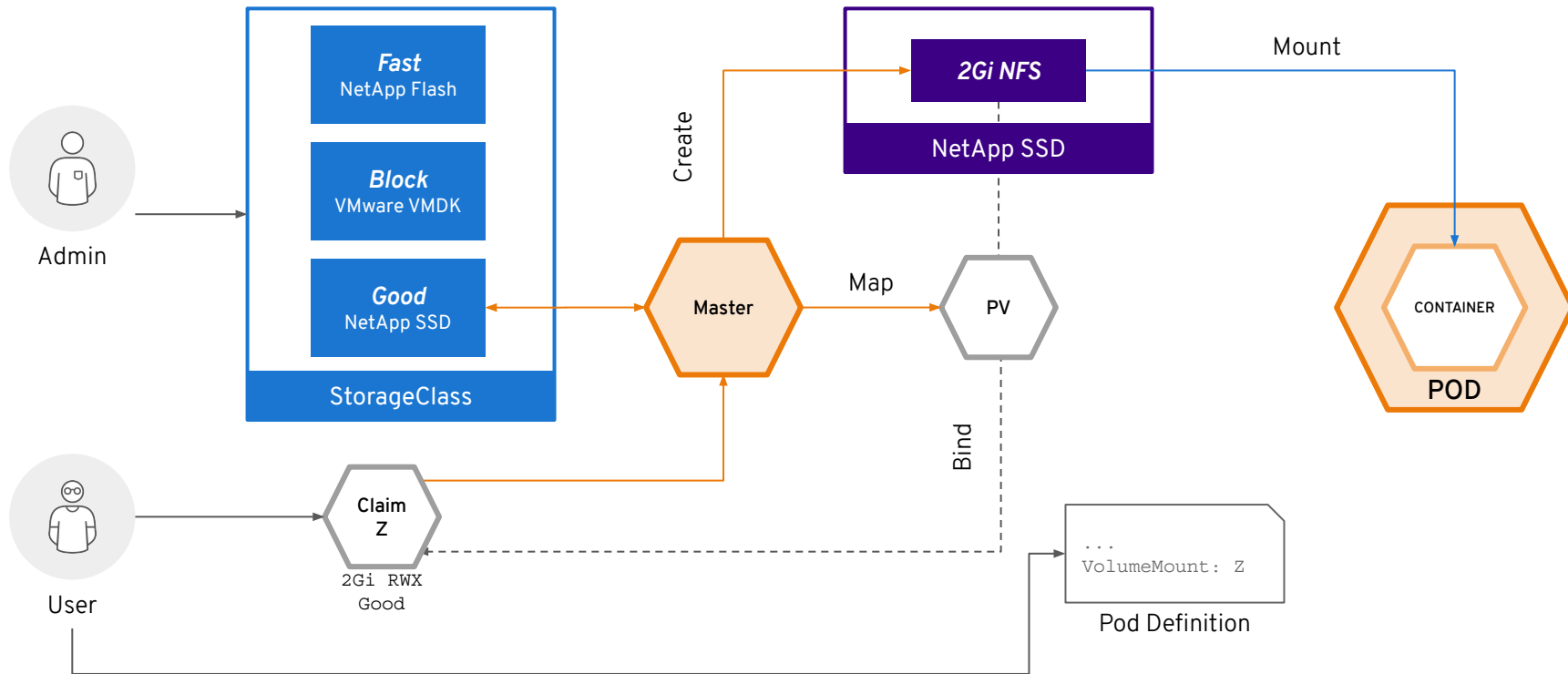
## PV Consumption



# Static Storage Provisioning



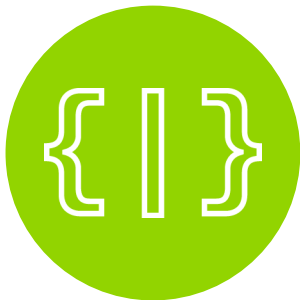
# Dynamic Storage Provisioning





# Build and Deploy Container Images

Tools and automation  
that makes developers  
productive quickly



**DEPLOY YOUR  
SOURCE CODE**



**DEPLOY YOUR  
APP BINARY**



**DEPLOY YOUR  
CONTAINER IMAGE**

