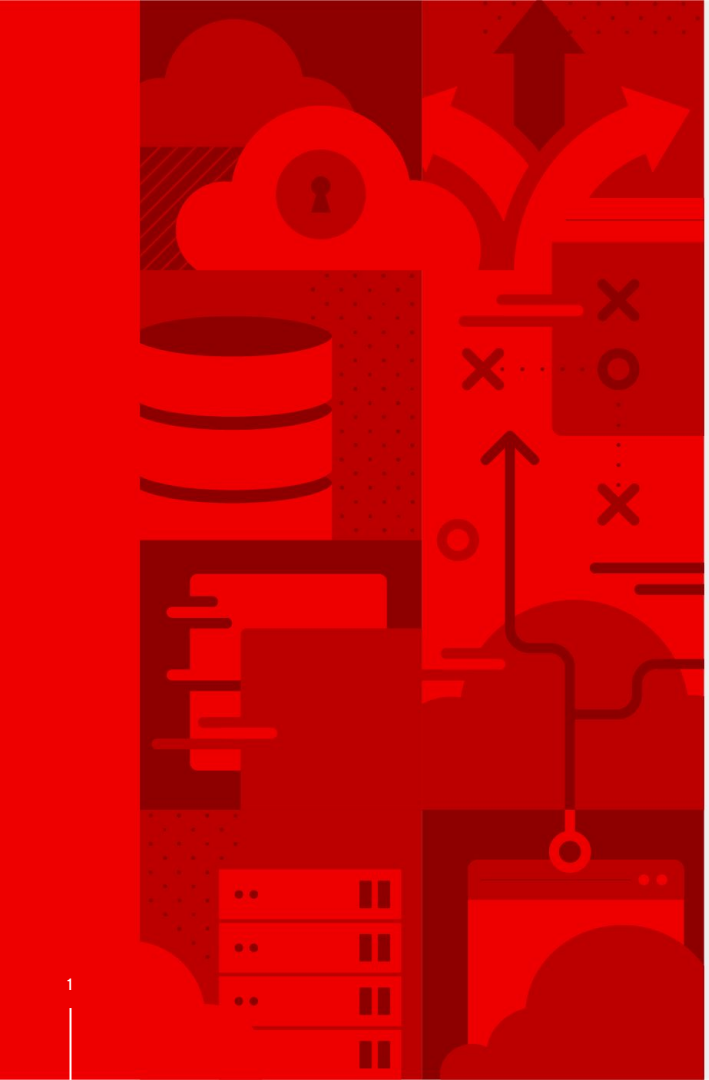


OPENSHIFT CONTAINER PLATFORM

TECHNICAL OVERVIEW

linkedin.com/company/red-hat

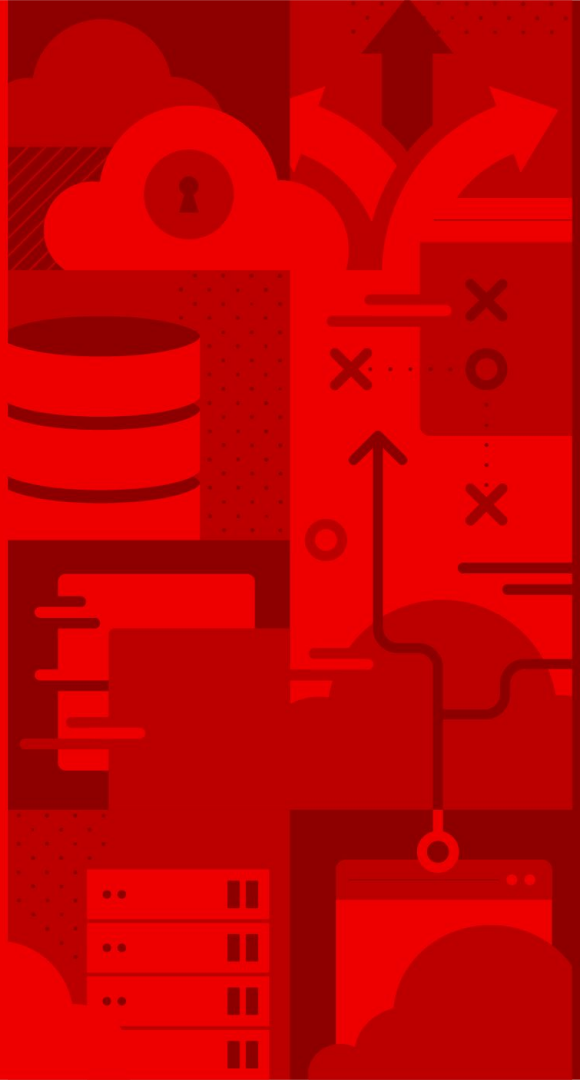
facebook.com/redhatinc

youtube.com/user/RedHatVideos

twitter.com/RedHat

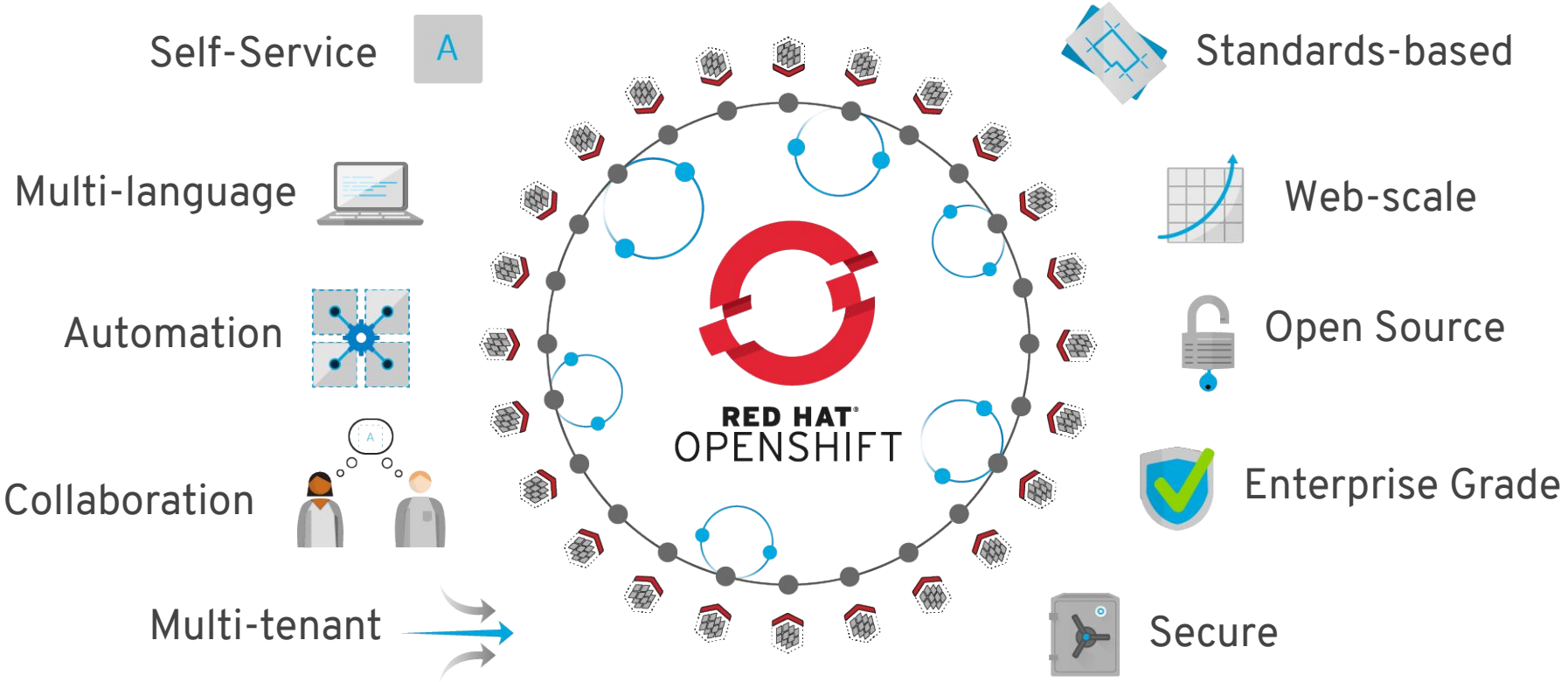
Alfred Bach
Partner ENablement Manager
Okt 2019

Red Hat

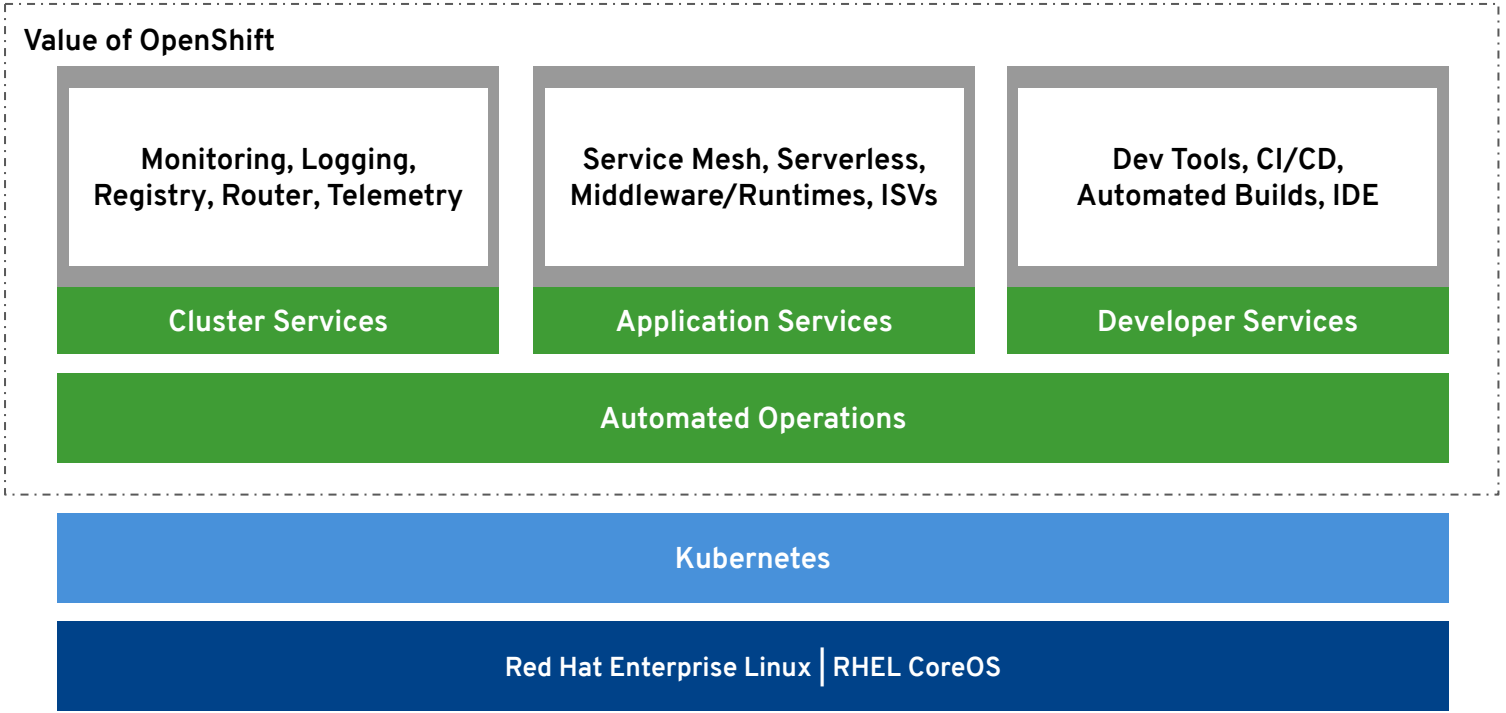


Functional overview

Red Hat


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

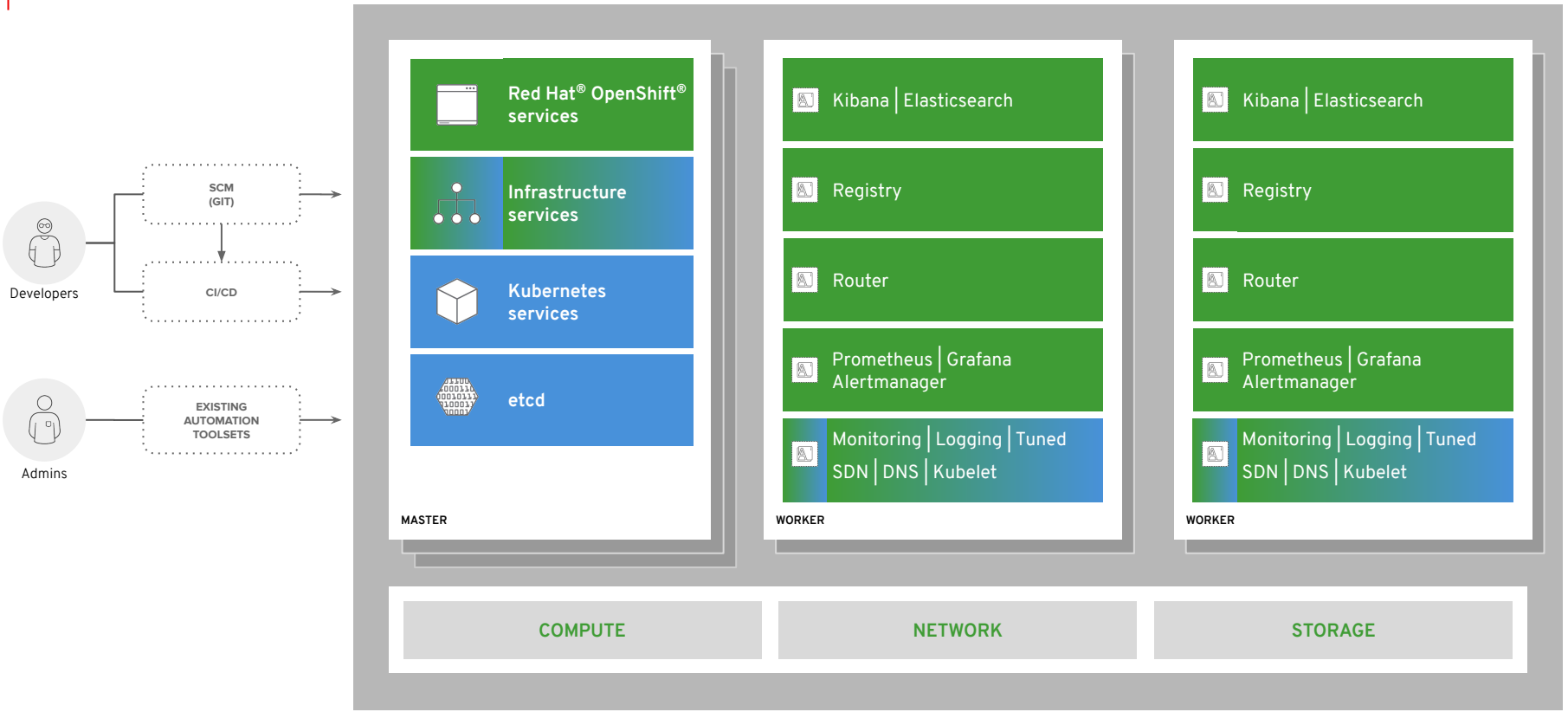
Red Hat



Best IT Ops Experience

CaaS ⟷ PaaS ⟷ FaaS

Best Developer Experience


SCM (GIT)
CI/CD
Developers
EXISTING AUTOMATION TOOLSETS
Admins
Red Hat® OpenShift® services
Infrastructure services
Kubernetes services
etcd
MASTER
Kibana | Elasticsearch
Registry
Router
Prometheus | Grafana Alertmanager
Monitoring | Logging | Tuned
SDN | DNS | Kubelet
WORKER
Kibana | Elasticsearch
Registry
Router
Prometheus | Grafana Alertmanager
Monitoring | Logging | Tuned
SDN | DNS | Kubelet
WORKER
COMPUTE
NETWORK
STORAGE

Red Hat

CNCF Cloud Native Landscape
2019-07-12T03:51:02Z 024e93b

Overwhelmed? Please see the CNCF Trail Map. That and the interactive landscape are at l.cncf.io

Greyed logos are not open source

App Definition and Development

Database

Streaming & Messaging

Application Definition & Image Build

Continuous Integration & Delivery

Orchestration & Management

Scheduling & Orchestration

Coordination & Service Discovery

Remote Procedure Call

Service Proxy

API Gateway

Service Mesh

Runtime

Cloud-Native Storage

Container Runtime

Cloud-Native Network

Provisioning

Automation & Configuration

Container Registry

Security & Compliance

Key Management

Cloud

Public

Special

Kubernetes Certified Service Provider

Platform

Certified Kubernetes - Distribution

Certified Kubernetes - Hosted

Certified Kubernetes - Installer

PaaS/Container Service

Kubernetes Training Partner

Observability and Analysis

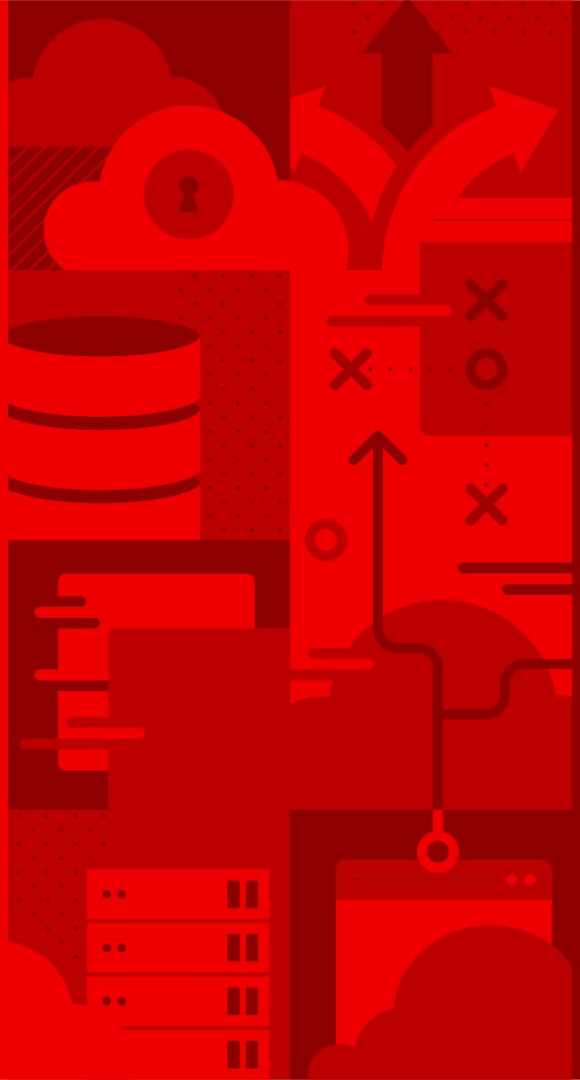
Monitoring

Logging

Tracing

Chaos Engineering

Serverless

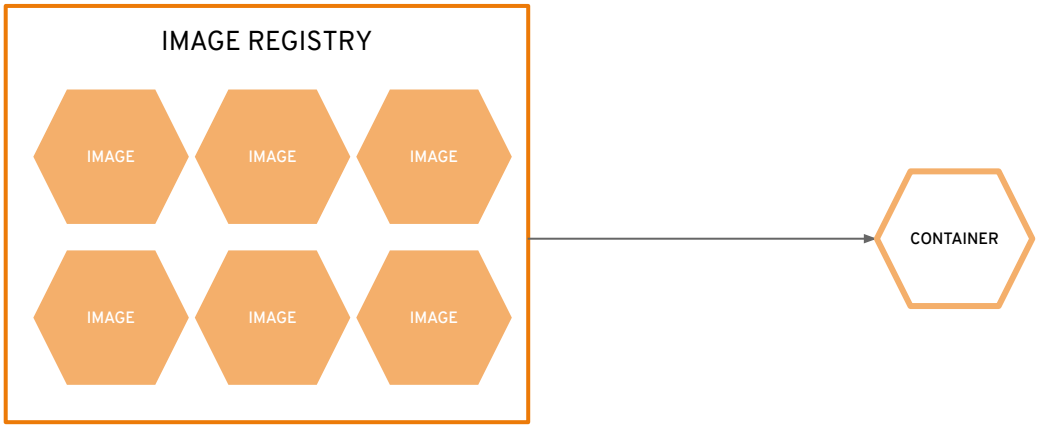


OpenShift and Kubernetes core concepts

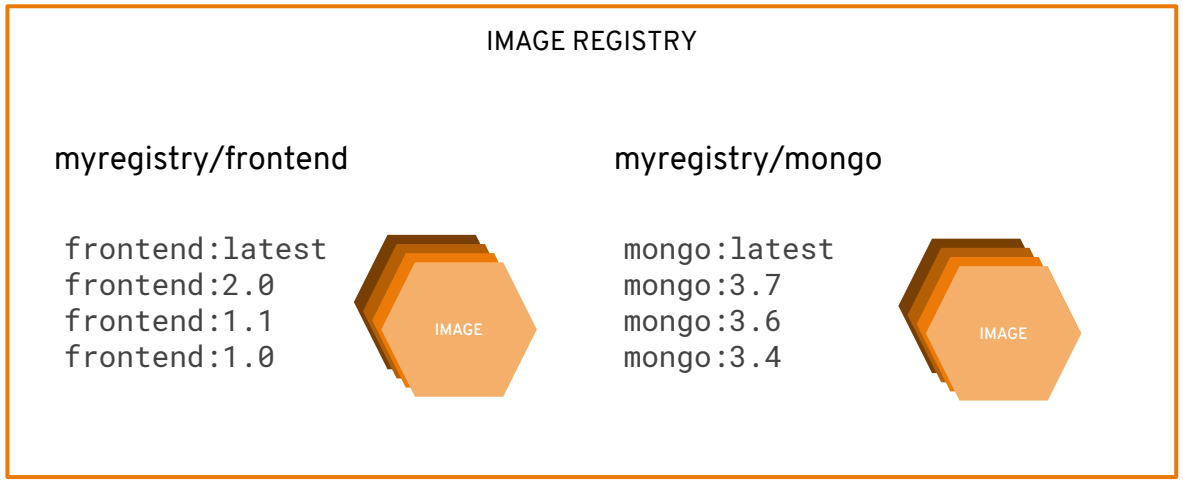
a container is the smallest compute unit

containers are created from container images

container images are stored in an image registry

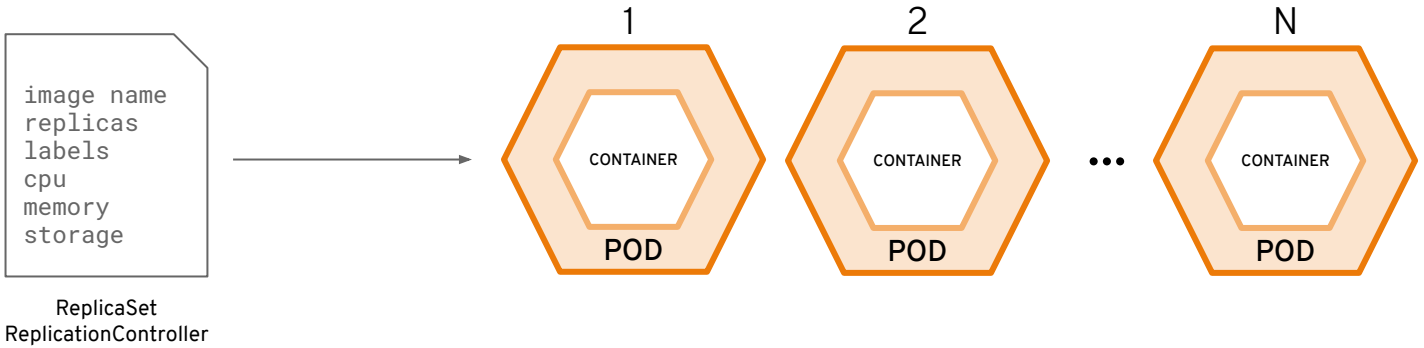


an image repository contains all versions of an image in the image registry

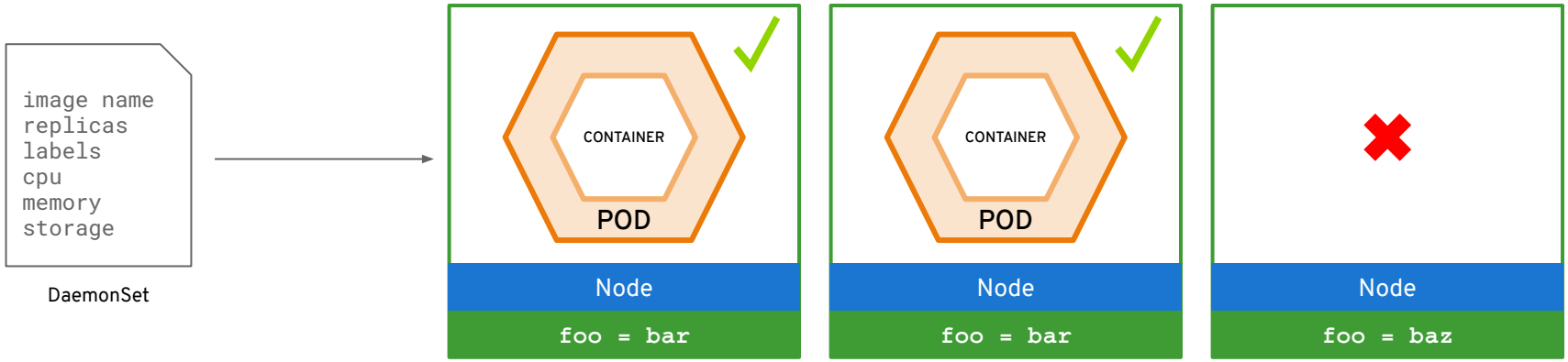


containers are wrapped in pods which are units of deployment and management

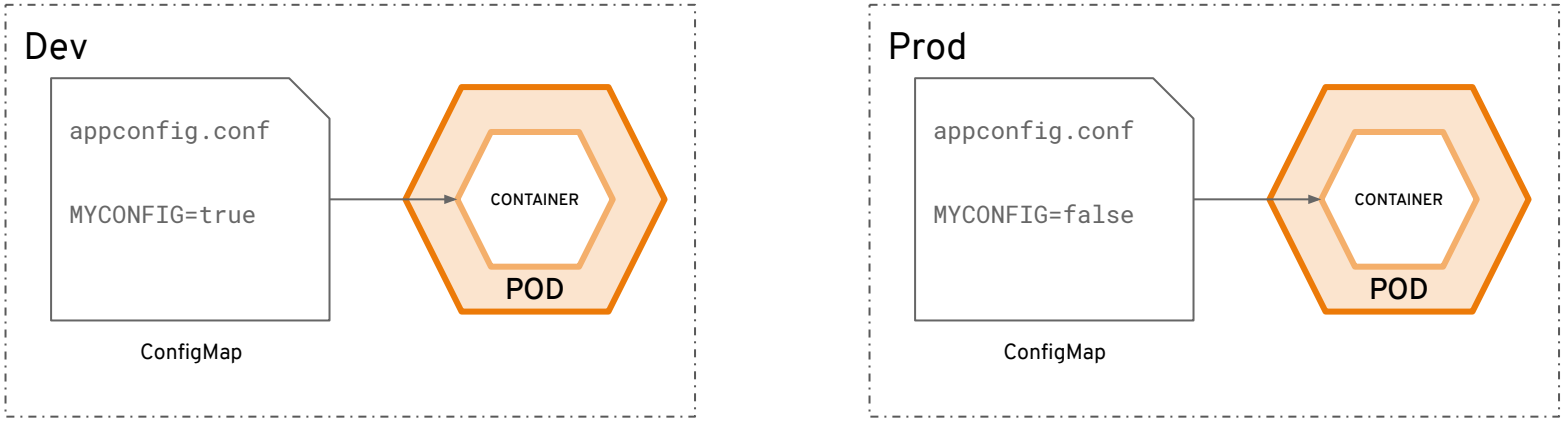
`ReplicationControllers` & `ReplicaSets` ensure a specified number of pods are running at any given time



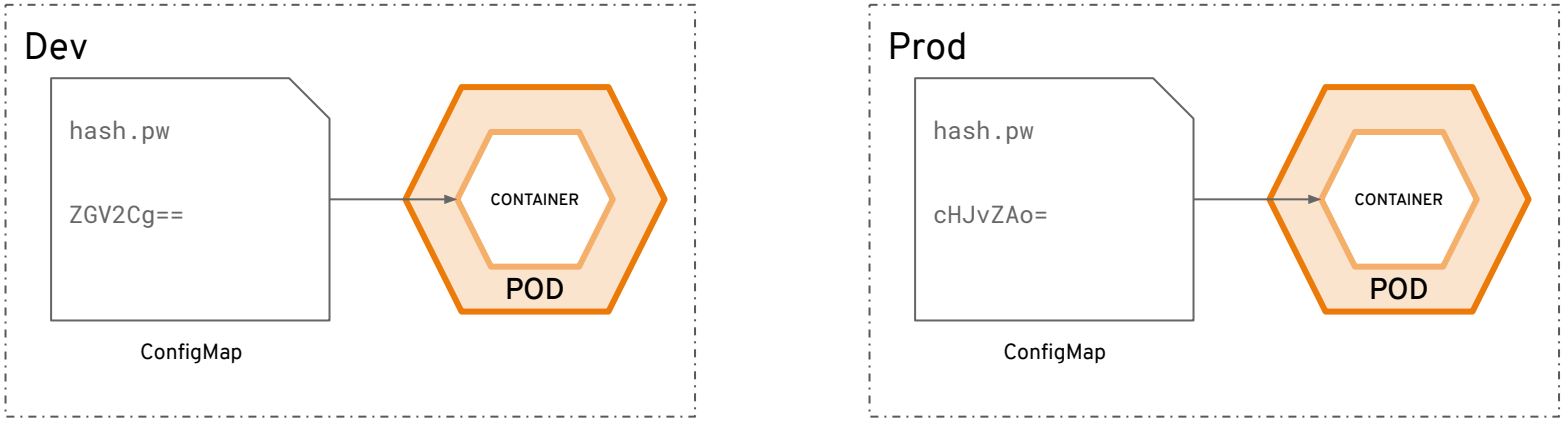
a daemonset ensures that all (or some) nodes run a copy of a pod



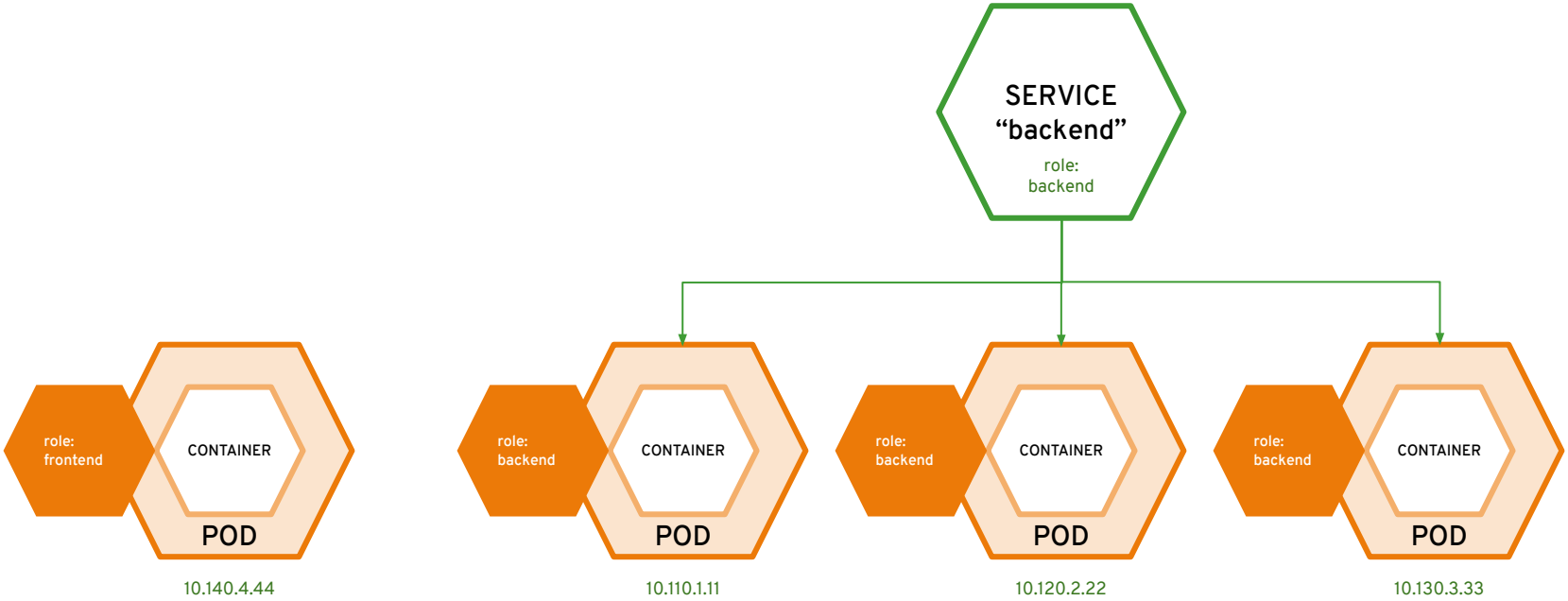
`configmaps` allow you to decouple configuration artifacts from image content



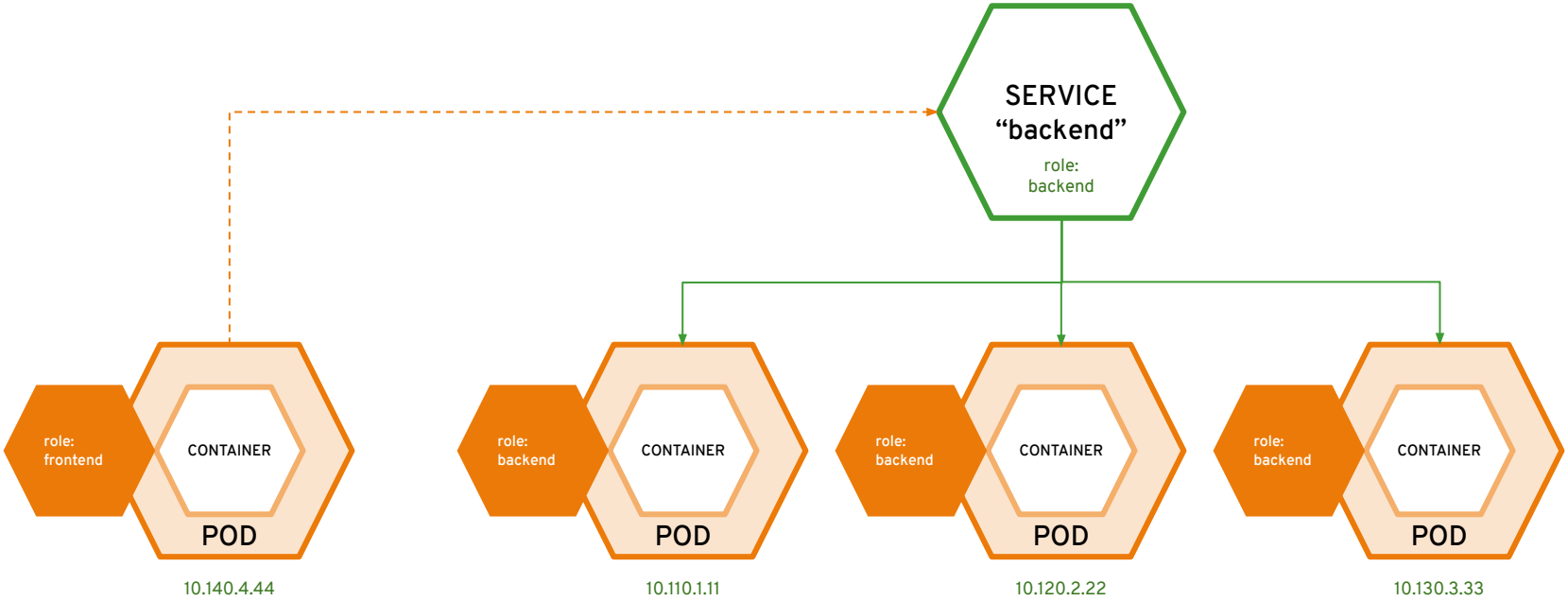
`secrets` provide a mechanism to hold sensitive information such as passwords



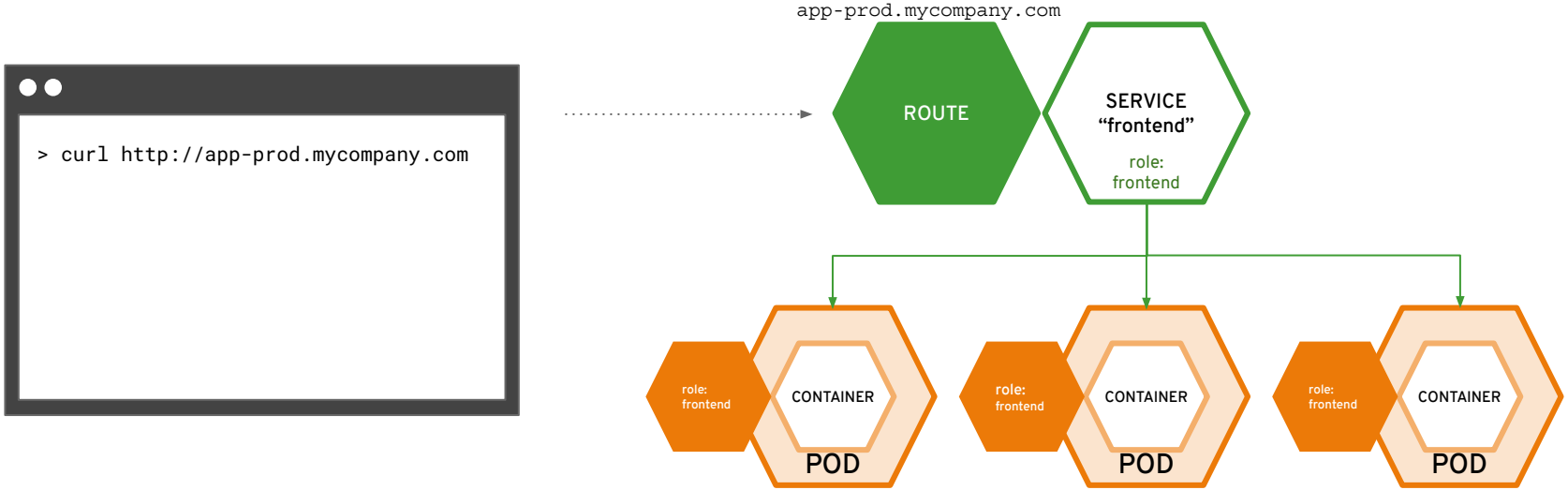
services provide internal load-balancing and service discovery across pods



apps can talk to each other via services

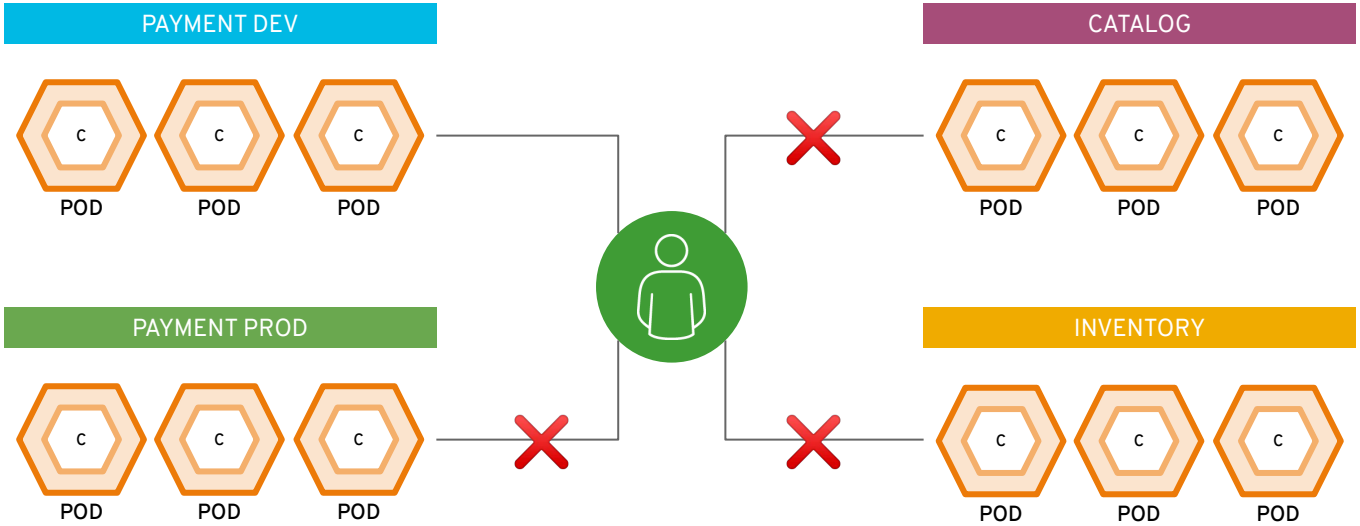


routes make services accessible to clients outside the environment via real-world urls

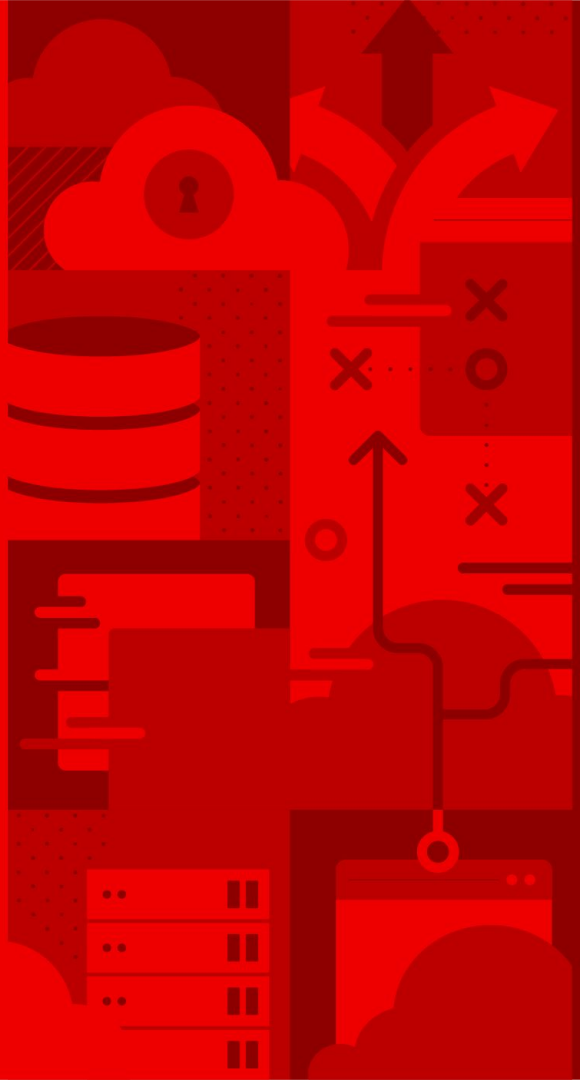


Red Hat

projects isolate apps across environments, teams, groups and departments



Red Hat



OpenShift lifecycle, installation & upgrades

Red Hat

OpenShift 3.11 Installation

Two new paradigms for deploying clusters

Red Hat

Host Preparation

Pre Requisits

Infrastructure

IP Address Segment
DNS Server
Subscriptions
Internet Access

X86 or Power 8/9
Hardware

Master Nodes

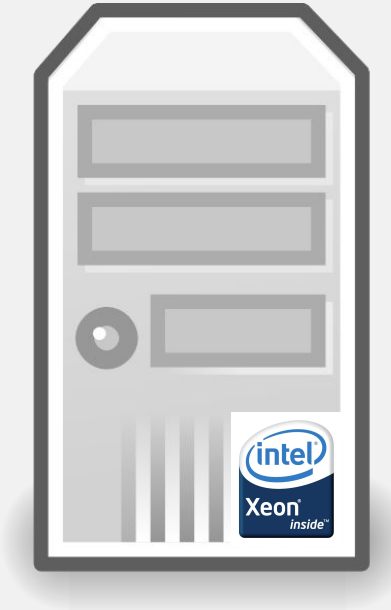
16 GB RAM
QUAD Core CPU
2 Disks / min 50 GB

Nodes

8 GB RAM
Dual Core CPU
2 Disks / min 50 GB

Prep. Installation all Node(s)

Install OPENSHIFT on RHEL 7.5 - 7.7



Subscribe to the required cannels

**—enable=„rhel-7-server-rpms" **
**—enable=„rhel-7-server-extras-rpms" **
**—enable=„rhel-7-server-ose-3.11-rpms" **
**—enable=„rhel-7-fast-datapath-rpms" **
--enable="rhel-7-server-ansible-2.5-rpms"

Install Bases Packages

yum install wget git net-tools bind-utils iptables-services bridge-utils bash-completion kexec-tools sos psacct

If you want to use the RPM Installer

yum install atomic-openshift-utils

Install the following package, which provides RPM-based OpenShift Container Platform installer utilities and pulls in other tools required by the quick and advanced installation methods, such asAnsible and related configuration files:

yum install atomic-openshift-utils

Prep. Installation all Node(s) 2/3
Install and Configure Docker

Install Docker

yum install docker-1.13.1

Configure insecure registiry

Edit the /etc/sysconfig/docker file and add --insecure-registry 172.30.0.0/16

Configure Docker Storage

```
cat <<EOF > /etc/sysconfig/docker-storage-setup
DEVS=/dev/vdc
VG=docker-vg
EOF
run docker-storage-setup
```

Start Docker

```
systemctl enable docker
systemctl start docker
```

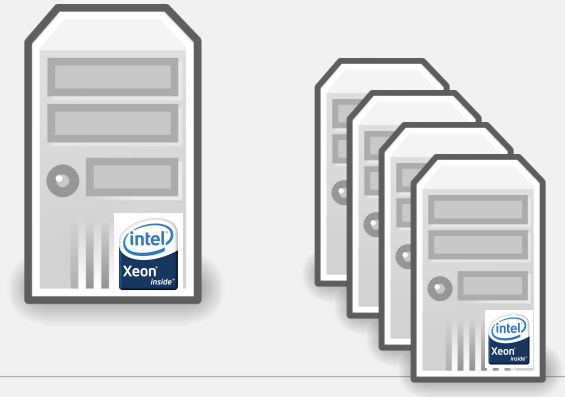
Prep.Installation on one Node only (Install Node) Ensuring Host Access

The quick and advanced installation methods require a user that has access to all hosts. If you want to run the installer as a non-root user, passwordless sudo rights must be configured on each destination host.

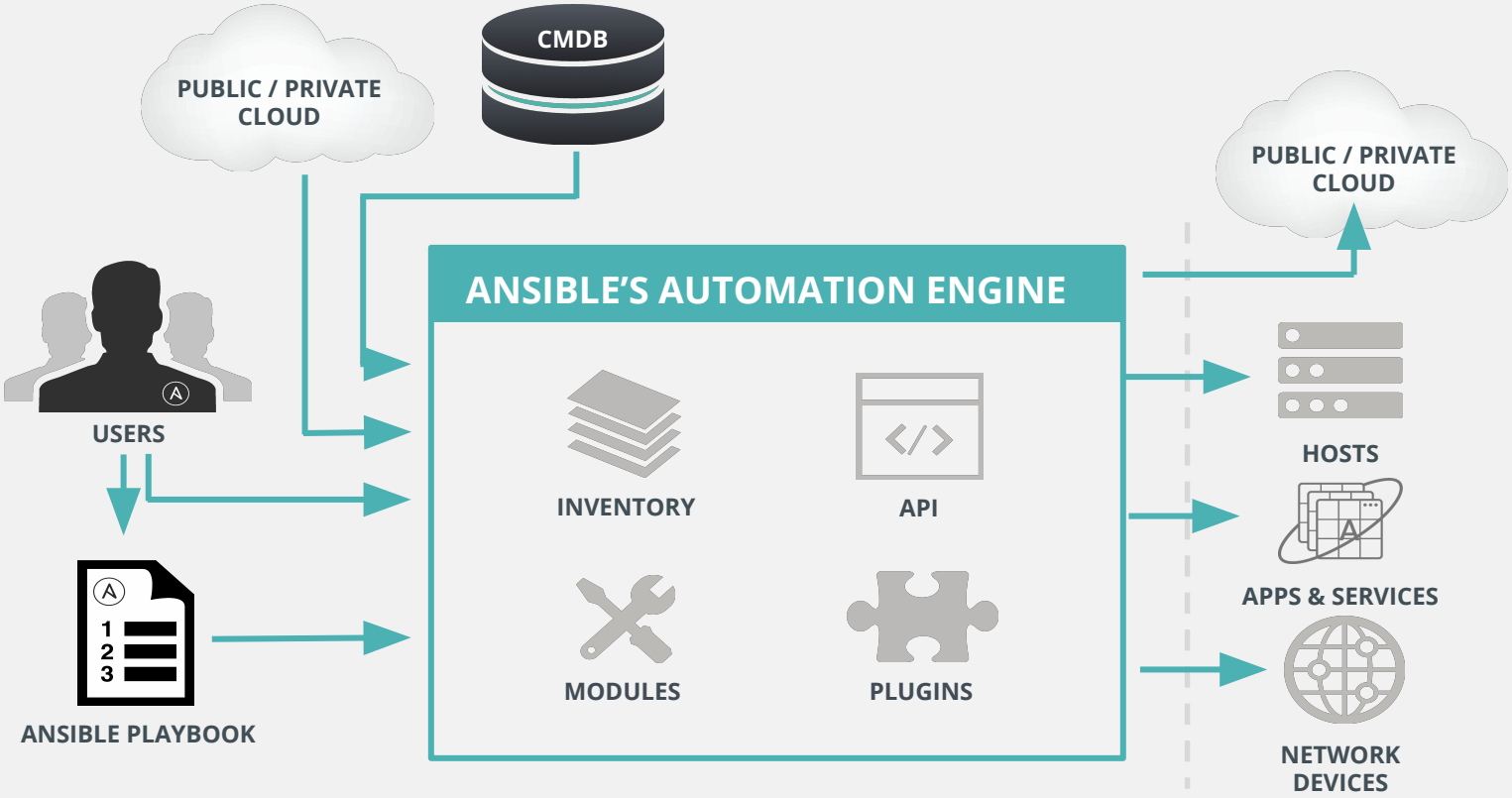
-> ssh-keygen ->Do not use a password.

An easy way to distribute your SSH keys is by using a bash loop:

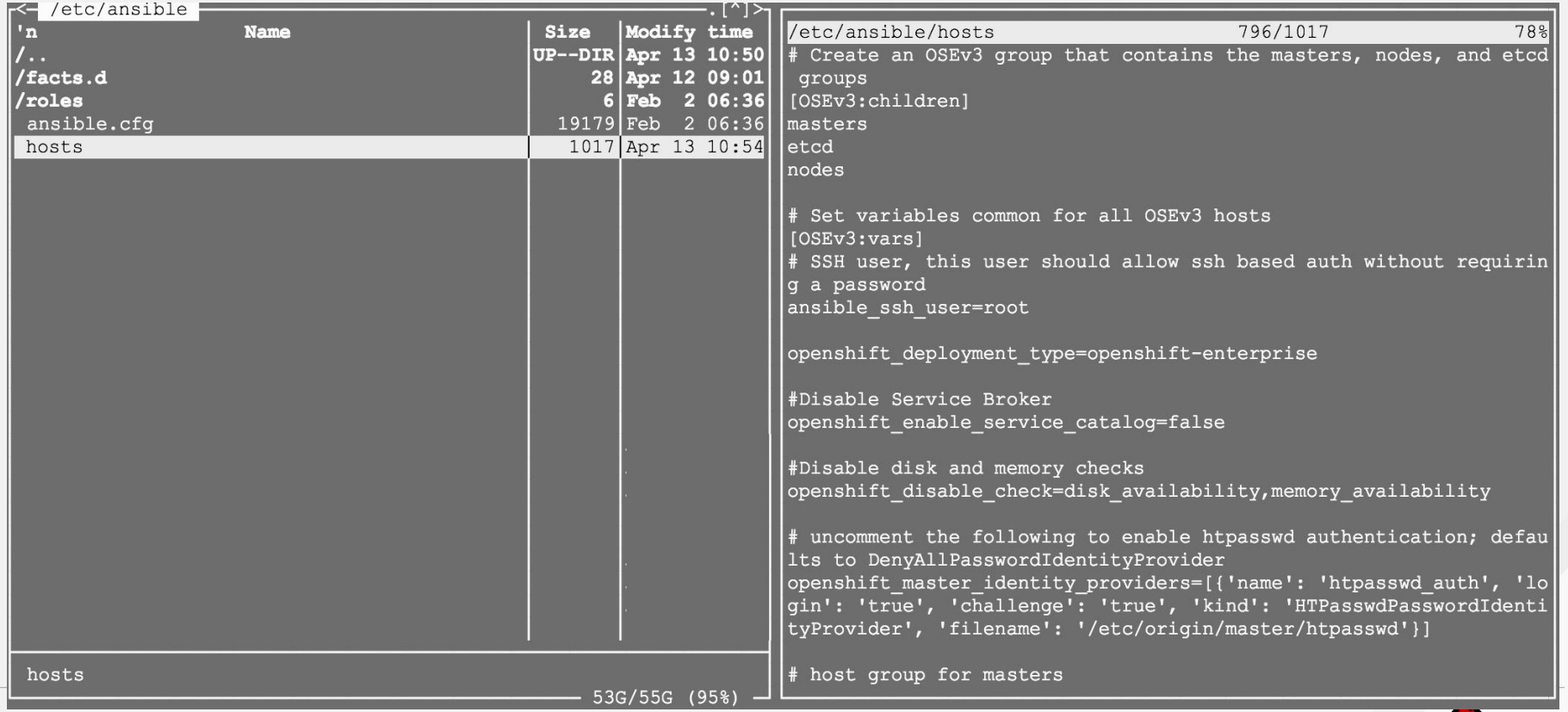
```
for host in master.example.com \
    node1.example.com \
    node2.example.com; \
    do ssh-copy-id -i ~/.ssh/id_rsa.pub $host; \
done
```



HOW ANSIBLE WORKS



/etc/ansible/hosts



Example Inventory Files (Single Master)

Host Name	Infrastructure Component to Install
master.example.com	Master, etcd, and node
node1.example.com	Node
node2.example.com	
infra-node1.example.com	Node (with `region=infra` label)
infra-node2.example.com	

Example Inventory Files (Single Master)

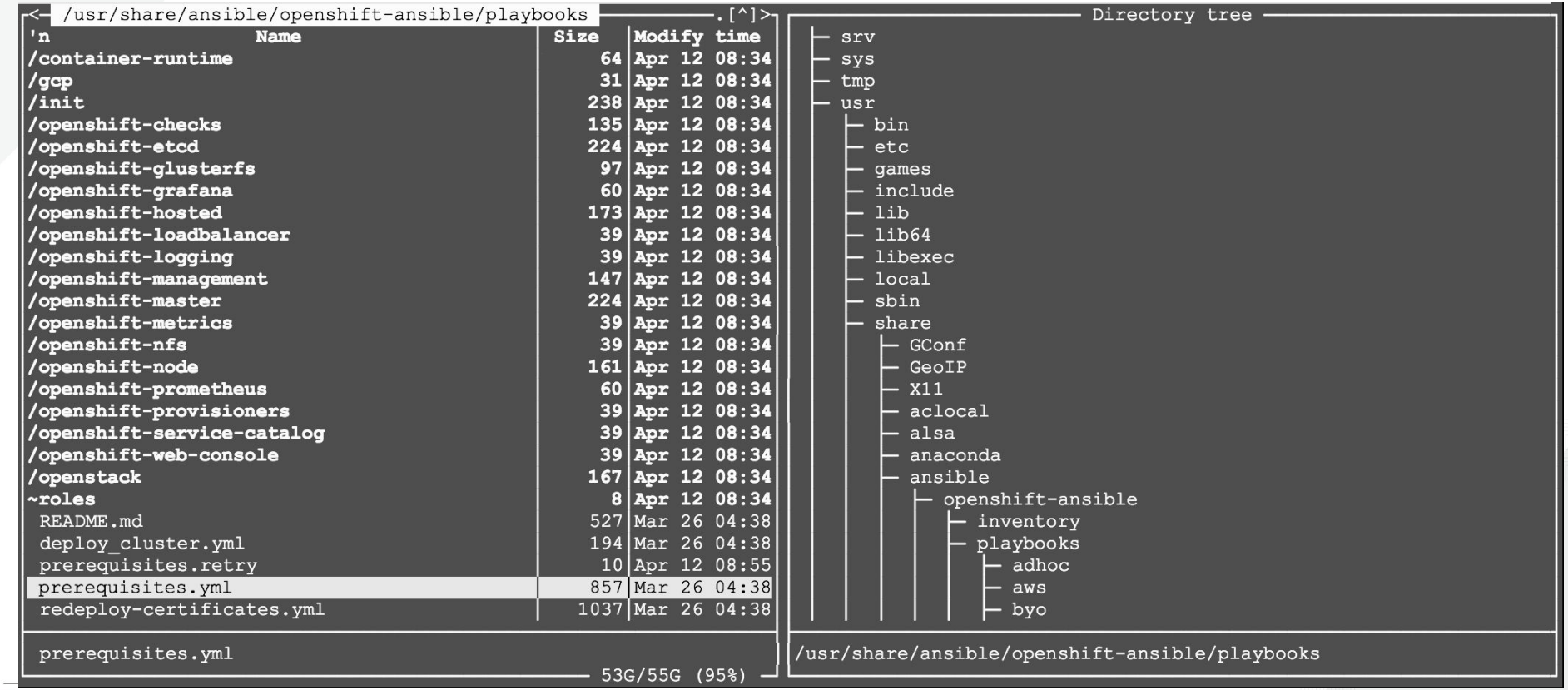
```
# Create an OSEv3 group that contains the masters, nodes, and etcd groups
[OSEv3:children]
masters
nodes
etcd
# Set variables common for all OSEv3 hosts

[OSEv3:vars]
# SSH user, this user should allow ssh based auth without requiring a password
ansible_ssh_user=root
# If ansible_ssh_user is not root, ansible_become must be set to true
# ansible_become=true
openshift_deployment_type=openshift-enterprise
# uncomment the following to enable htpasswd authentication; defaults to
# DenyAllPasswordIdentityProvider
openshift_master_identity_providers=[{'name': 'htpasswd_auth', 'login':
'true', 'challenge': 'true', 'kind': 'HTPasswdPasswordIdentityProvider',
'filename': '/etc/origin/master/htpasswd'}]
```

Example Inventory Files (Single Master) (cont)

```
# host group for masters
[masters]
master.example.com
# host group for etcd
[etcd]
master.example.com
# host group for nodes, includes region info
[nodes]
node1.example.com openshift_node_labels="{'region': 'primary', 'zone':
'east'}"
node2.example.com openshift_node_labels="{'region': 'primary', 'zone':
'west'}"
infra-node1.example.com openshift_node_labels="{'region': 'infra', 'zone':
'default'}"
infra-node2.example.com openshift_node_labels="{'region': 'infra', 'zone':
'default'}"
```

Ansible Playbook(s)



Host & Playbook

Preflight check:
ansible-playbook /usr/share/ansible/openshift-ansible/playbooks/prerequisites.yml

Install:
Ansible-playbook /usr/share/ansible/openshiftansible/playbooks/deploy_cluster.yml

You end up with ...

Single Master
& ETCD

InfraNode

InfraNode

Worker Node
(west)

Workernode
(east)

H/A for Master ?

Example Inventory Files (Single Master) Add HA for Master Node

```
# host group for masters
[masters]
Master.example.com
master2.example.com
master3.example.com
# host group for etcd
[etcd]
Master.example.com
master2.example.com
master3.example.com
# host group for nodes, includes region info
[nodes]
node1.example.com openshift_node_labels="{'region': 'primary', 'zone':'east'}"
node2.example.com openshift_node_labels="{'region': 'primary', 'zone':'west'}"
infra-node1.example.com openshift_node_labels="{'region': 'infra', 'zone':'default'}"
infra-node2.example.com openshift_node_labels="{'region': 'infra', 'zone':'default'}"
```

ScaleUp

Ansible-playbook
/usr/share/ansible/openshiftansible/playbooks/openshift_master/scaleup.yml

You end up with ...

Need more nodes ?

Example Inventory Files (Single Master) Add HA for Master Node

```
# host group for masters
[masters]
master.example.com
master2.example.com
master3.example.com
# host group for etcd
[etcd]
aster.example.com
master2.example.com
master3.example.com
# host group for nodes, includes region info
[nodes]
node1.example.com openshift_node_labels="{'region': 'primary', 'zone':'east'}"
node2.example.com openshift_node_labels="{'region': 'primary', 'zone':'west'}"
node3.example.com openshift_node_labels="{'region': 'primary', 'zone':'north'}"
node4.example.com openshift_node_labels="{'region': 'primary', 'zone':'south'}"
infra-node1.example.com openshift_node_labels="{'region': 'infra', 'zone':'default'}"
infra-node2.example.com openshift_node_labels="{'region': 'infra', 'zone':'default'}"
```

ScaleUp

Ansible-playbook /usr/share/ansible/openshiftansible/playbooks/openshift_node/scaleup.yml

You end up with ...

Master &
ETCD

Master
& ETCD

Master
& ETCD

InfraNode

InfraNode

Worker Node
north

Worker node
south

Worker Node
west

Worker node
east

More Features (Options) ?

- Configure the automatically-deployed router.

2.6. ADVANCED INSTALLATION

2.6.1. Overview

A reference configuration implemented using Ansible playbooks is available as the *advanced installation* method for installing a OpenShift Container Platform cluster. Familiarity with Ansible is assumed, however you can use this configuration as a reference to create your own implementation using the configuration management tool of your choosing.

IMPORTANT

While RHEL Atomic Host is supported for running containerized OpenShift Container Platform services, the advanced installation method utilizes Ansible, which is not available in RHEL Atomic Host. The RPM-based installer must therefore be run from a RHEL 7 system. The host initiating the installation does not need to be intended for inclusion in the OpenShift Container Platform cluster, but it can be. Alternatively, a containerized version of the installer is available as a system container, which can be run from a RHEL Atomic Host system.

NOTE

To install OpenShift Container Platform as a stand-alone registry, see Installing a Stand-alone Registry.

2.6.2. Before You Begin

Before installing OpenShift Container Platform, you must first see the Prerequisites and Host Preparation topics to prepare your hosts. This includes verifying system and environment requirements per component type and properly installing and configuring Docker. It also includes installing Ansible version 2.4 or later, as the advanced installation method is based on Ansible playbooks and as such requires directly invoking Ansible.

OpenShift 4 Installation

Two new paradigms for deploying clusters

Red Hat

Installation Paradigms

OPENSHIFT CONTAINER PLATFORM

Full Stack Automated

Simplified opinionated "Best Practices" for cluster provisioning

Fully automated installation and updates including host container OS.

Red Hat Enterprise Linux CoreOS

Pre-existing Infrastructure

Customer managed resources & infrastructure provisioning

Plug into existing DNS and security boundaries

Red Hat Enterprise Linux CoreOS

Red Hat Enterprise Linux

HOSTED OPENSHIFT

Azure Red Hat OpenShift

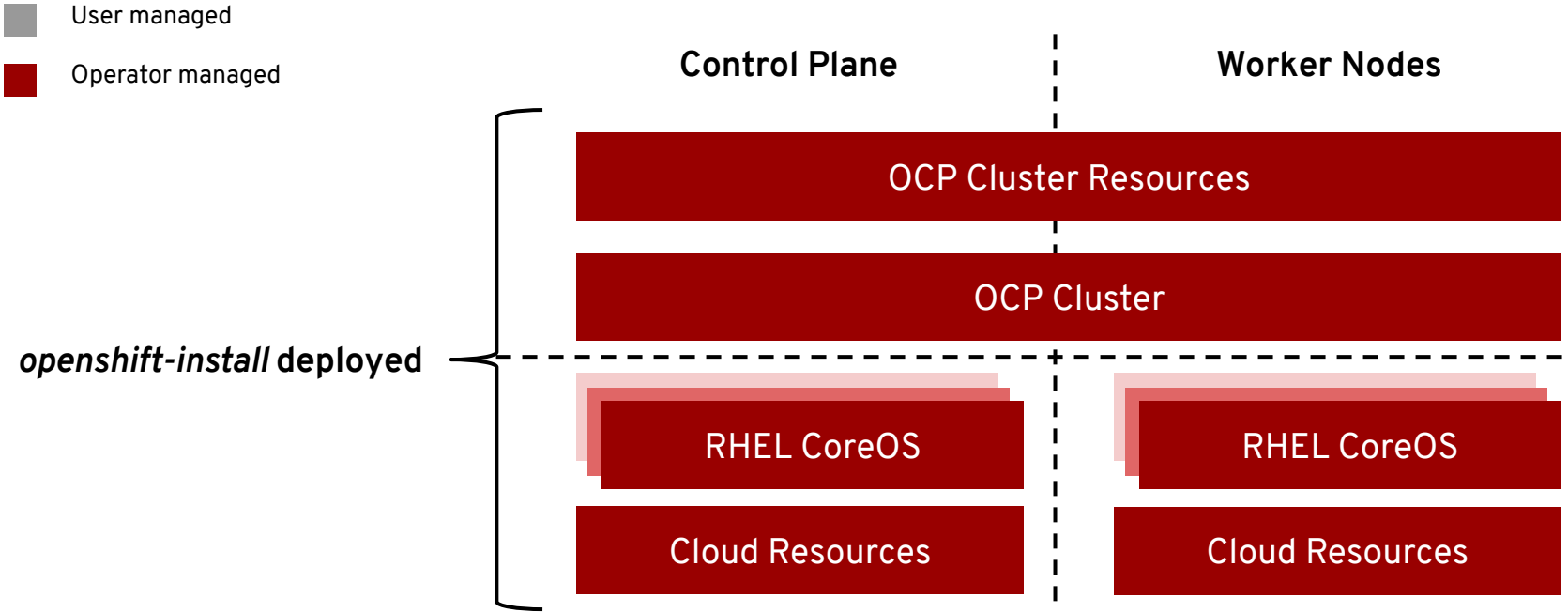
Deploy directly from the Azure console. Jointly managed by Red Hat and Microsoft Azure engineers.

OpenShift Dedicated

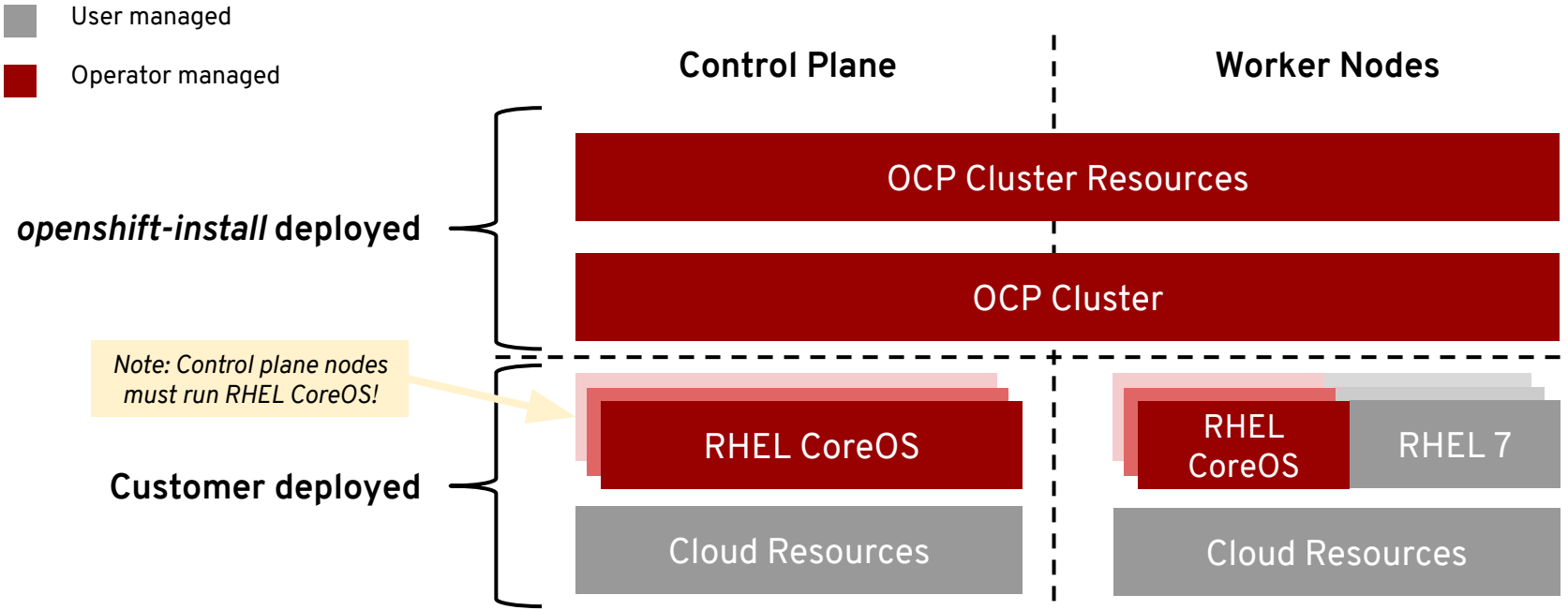
Get a powerful cluster, fully Managed by Red Hat engineers and support.

Red Hat

Full-stack Automated Installation



Pre-existing Infrastructure Installation



Comparison of Paradigms

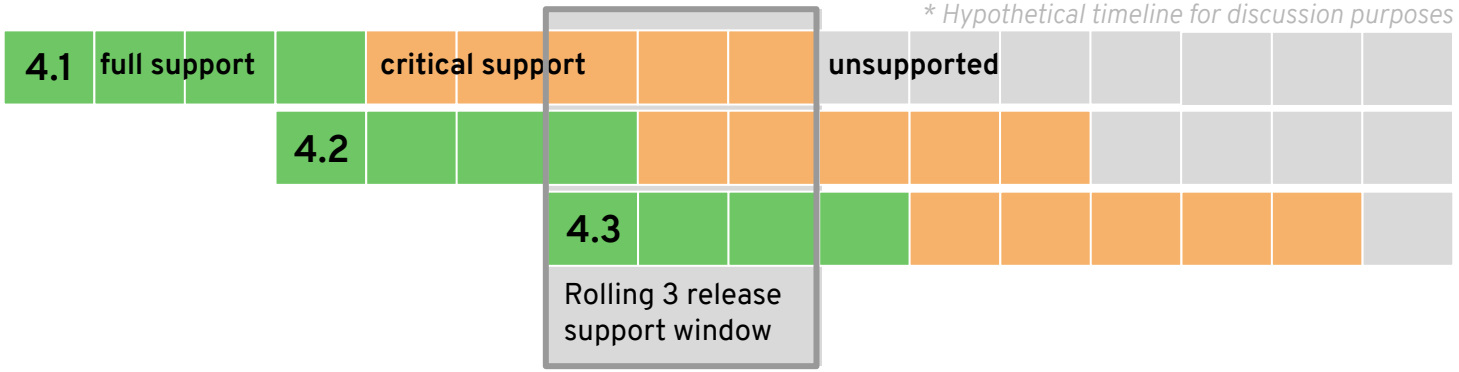
	Full Stack Automation	Pre-existing Infrastructure
Build Network	Installer	User
Setup Load Balancers	Installer	User
Configure DNS	Installer	User
Hardware/VM Provisioning	Installer	User
OS Installation	Installer	User
Generate Ignition Configs	Installer	Installer
OS Support	Installer: RHEL CoreOS	User: 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

Red Hat

Support Timelines



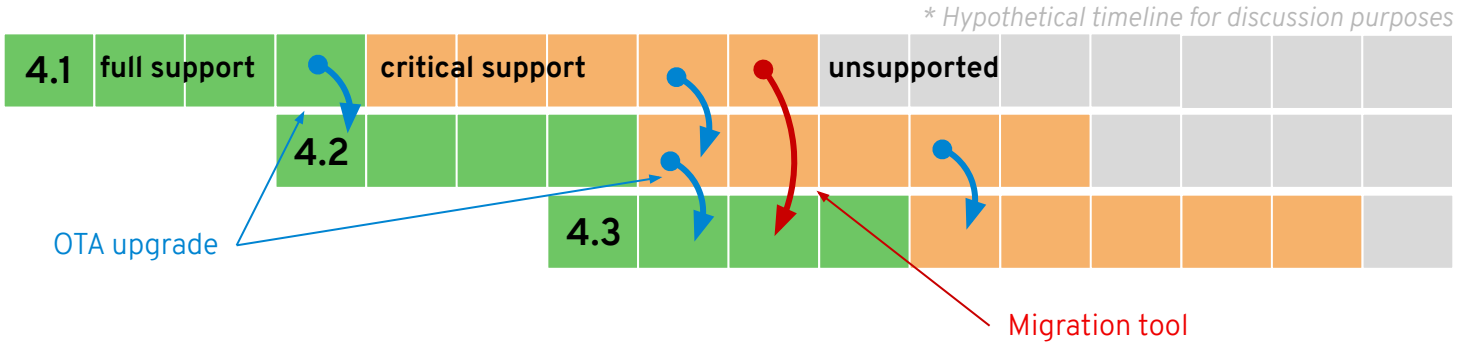
New model
Release based, not date based. Rolling three release window for support.

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



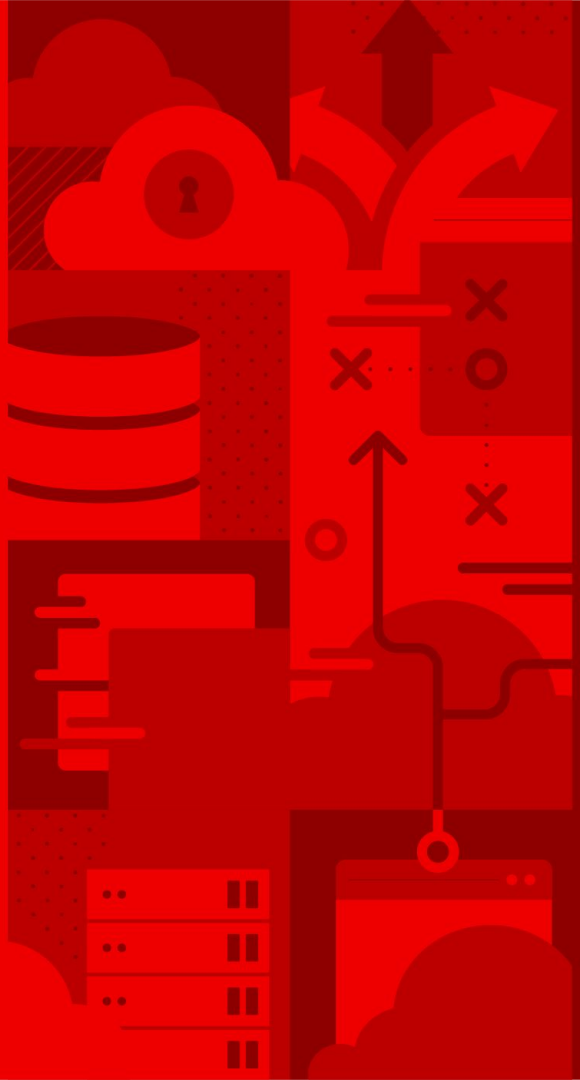
OTA Upgrades
Works between two minor releases in a serial manner.

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



Operations and infrastructure deep dive

Red Hat

Red Hat Enterprise Linux CoreOS

The OpenShift operating system

Red Hat

Red Hat Enterprise Linux

	RED HAT® ENTERPRISE LINUX®	RED HAT® ENTERPRISE LINUX CoreOS
	General Purpose OS	**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.

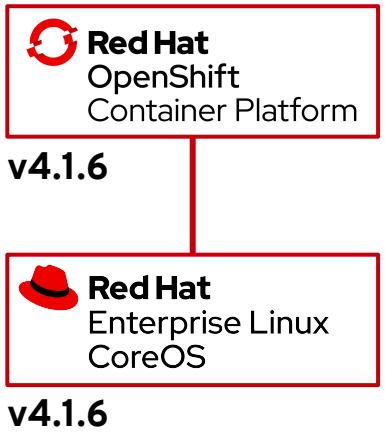
Red Hat Enterprise Linux CoreOS is managed by the cluster

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. 😀🙌



cri-o

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

CRI-O 1.12  Kubernetes 1.12  OpenShift 4.0 

CRI-O 1.13  Kubernetes 1.13  OpenShift 4.1 

CRI-O 1.14  Kubernetes 1.14  OpenShift 4.2 

podman

podman

A docker-compatible CLI for containers

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

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

Red Hat

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

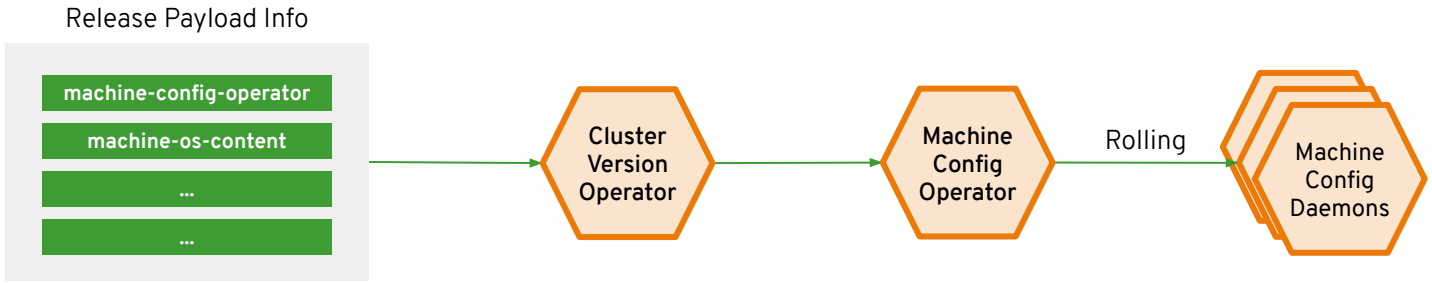
- MachineSets scoped to single AZ
- Installer stripes N machine sets across AZs by default
- Post-install best effort balance via cluster autoscaler

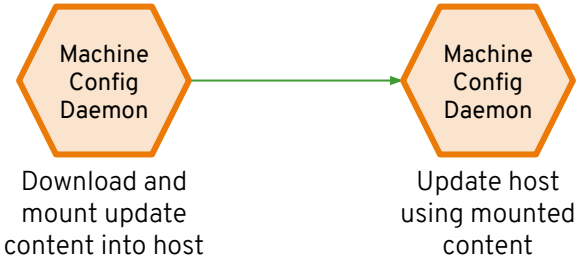
OpenShift 4 Cluster Management

Powered by Operators, OpenShift 4 automates many cluster management activities

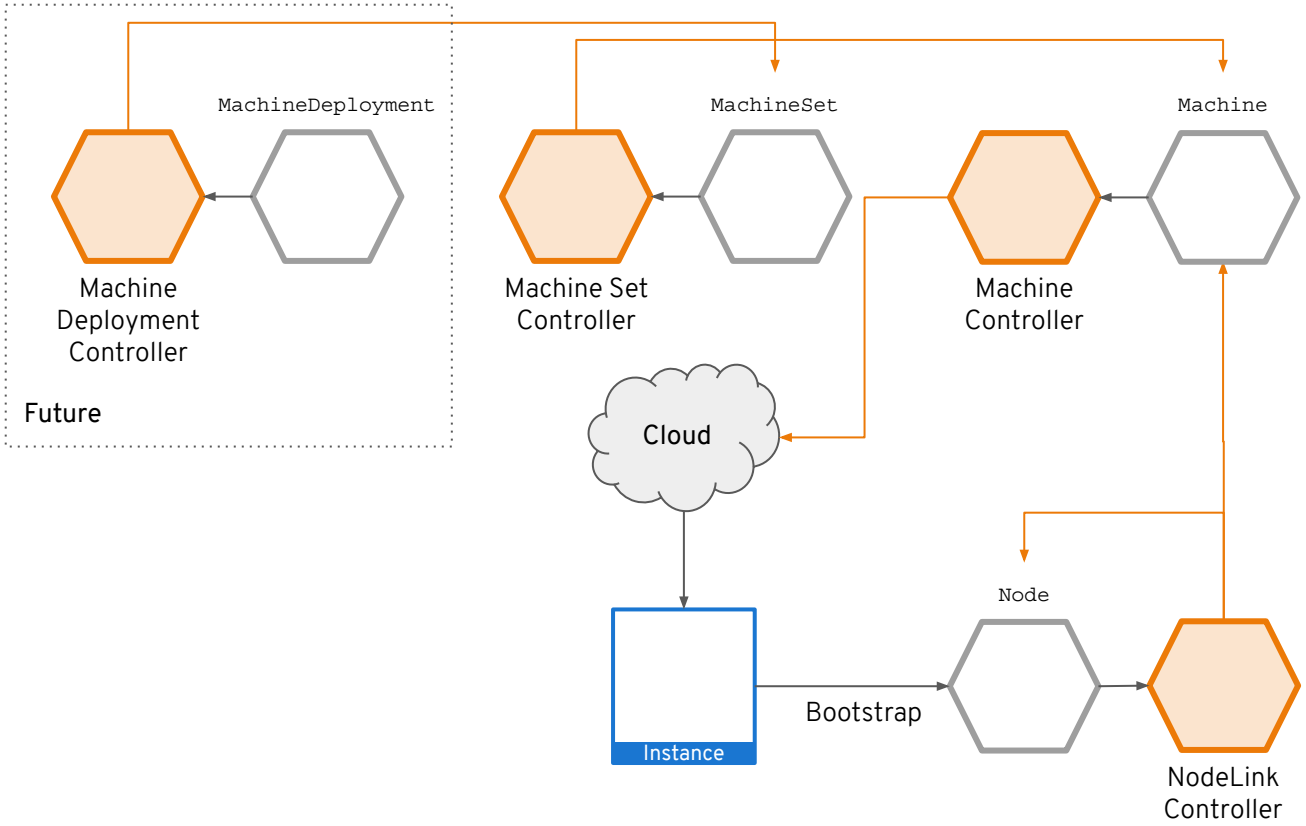
Red Hat

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

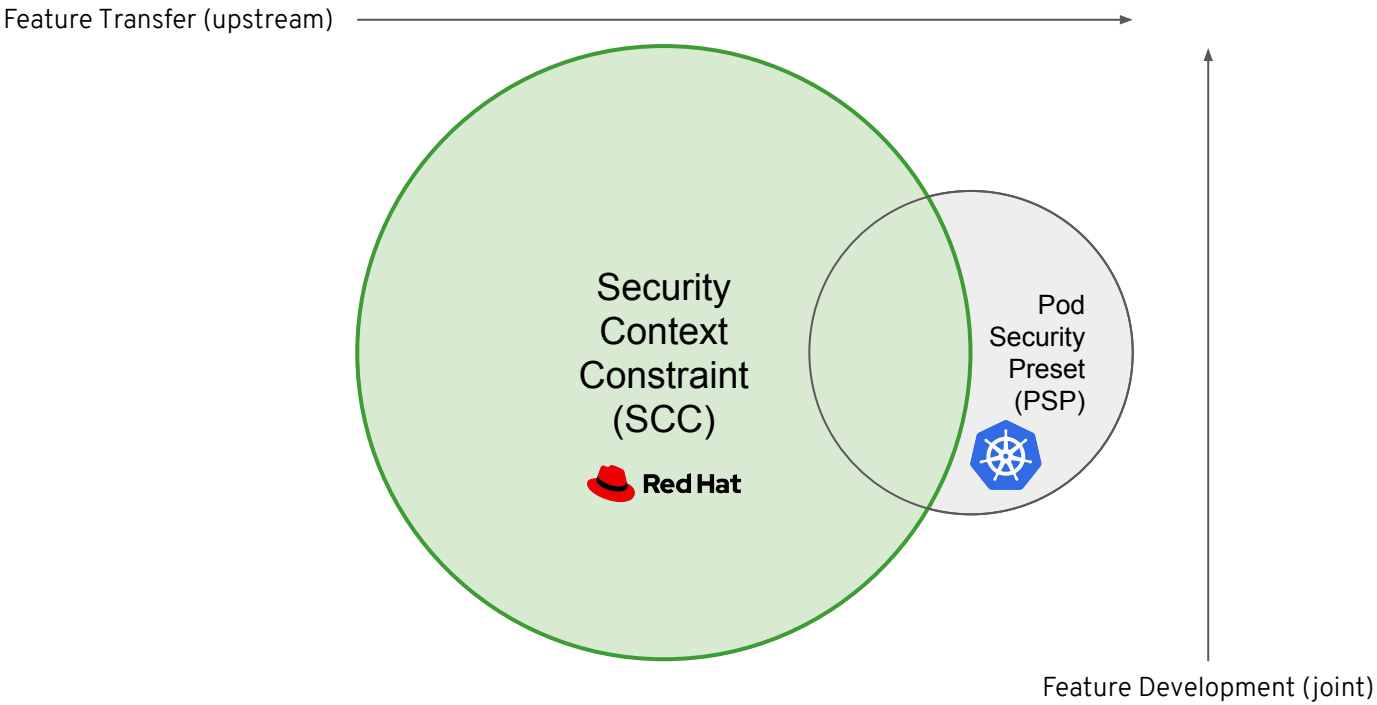
DEFEND
Infrastructure

- Container Platform
- Container Host Multi-tenancy
- Network Isolation
- Storage
- Audit & Logging
- API Management

EXTEND

- Security Ecosystem

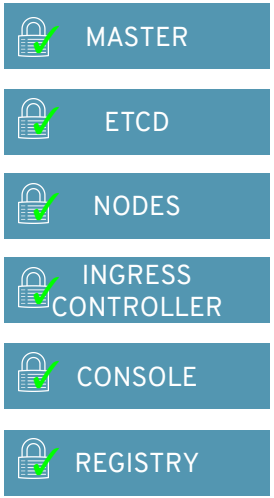
Extended Depth of Protection



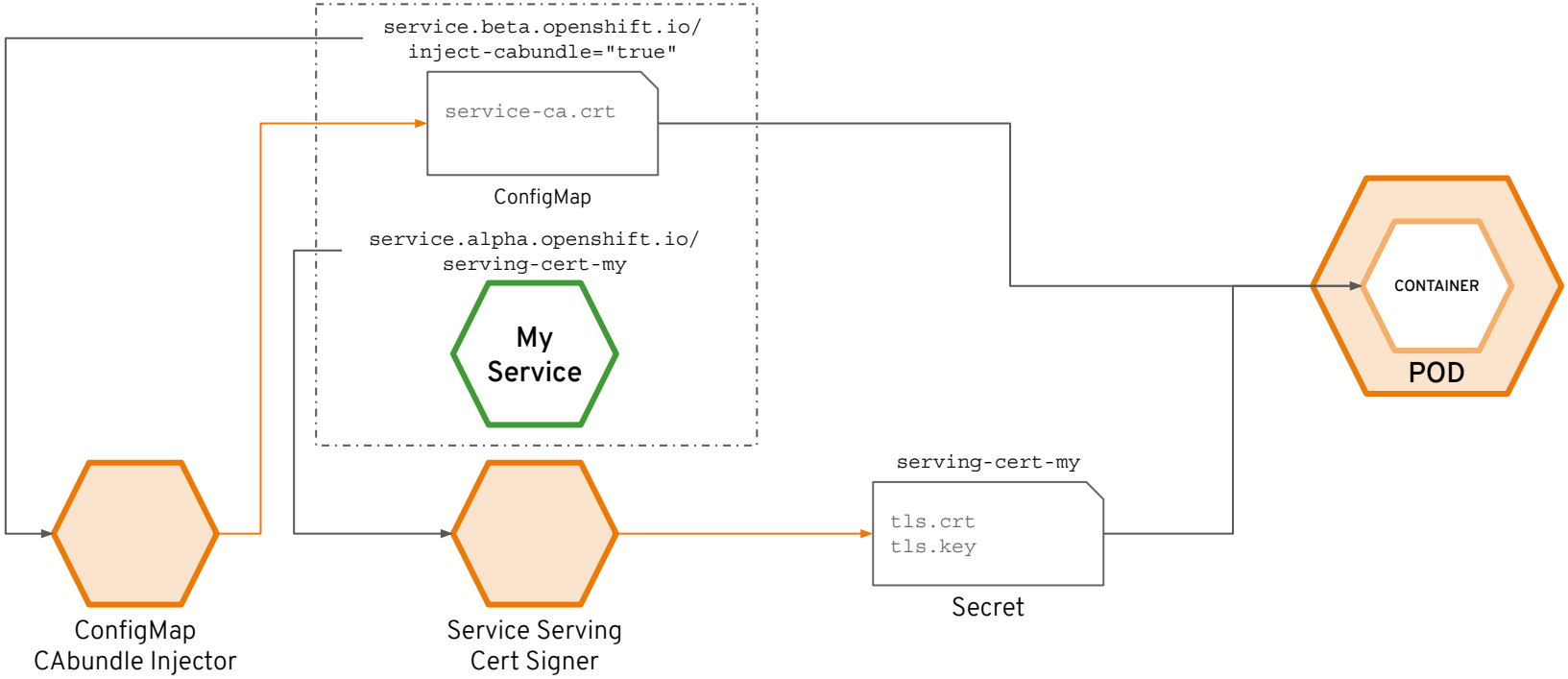
Red Hat

Certificates and Certificate Management

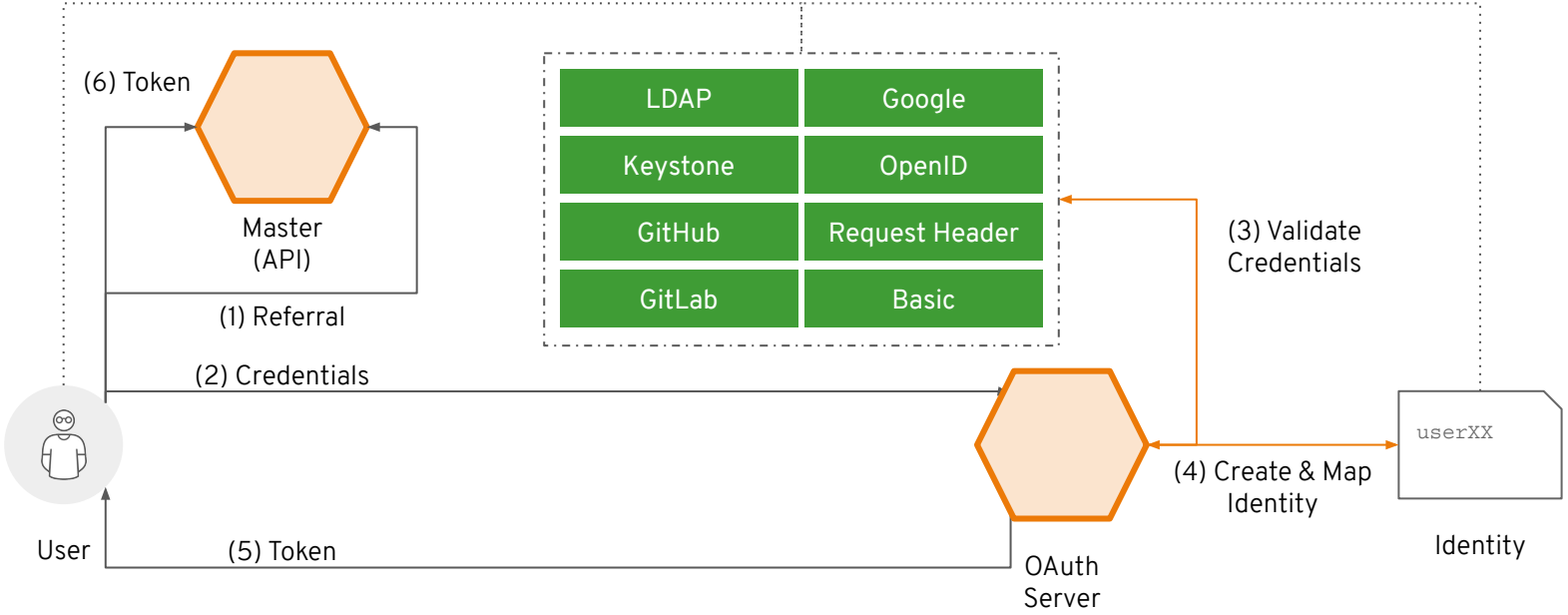
- OpenShift provides its own internal CA
- Certificates are used to provide secure connections to
  - master (APIs) and nodes
  - Ingress controller and registry
  - etcd
- Certificate rotation is automated
- Optionally configure external endpoints to use custom certificates



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

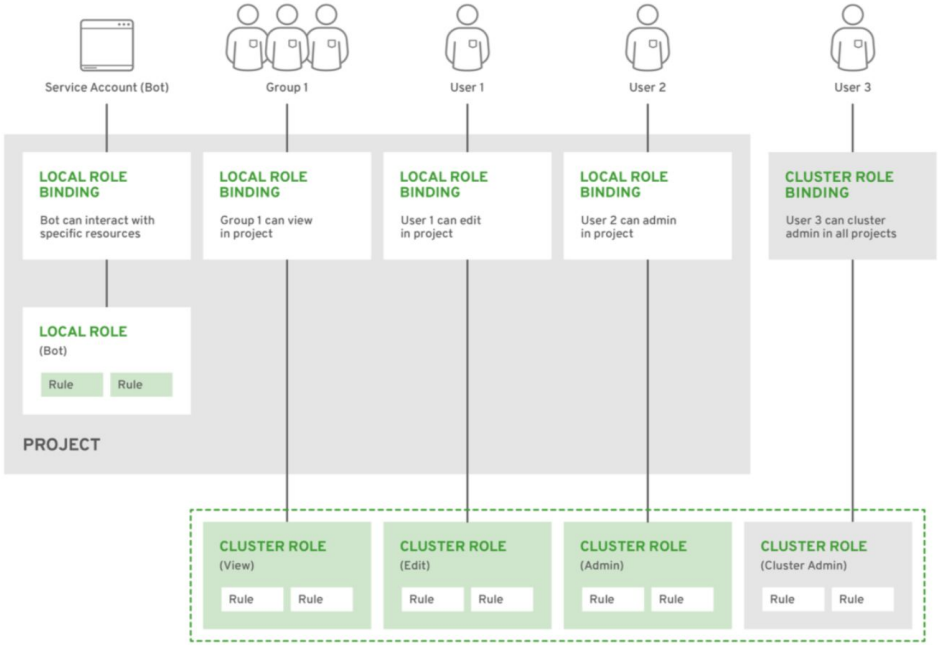


Figure 12 - Authorization Relationships

OpenShift Monitoring

An integrated cluster monitoring and alerting stack

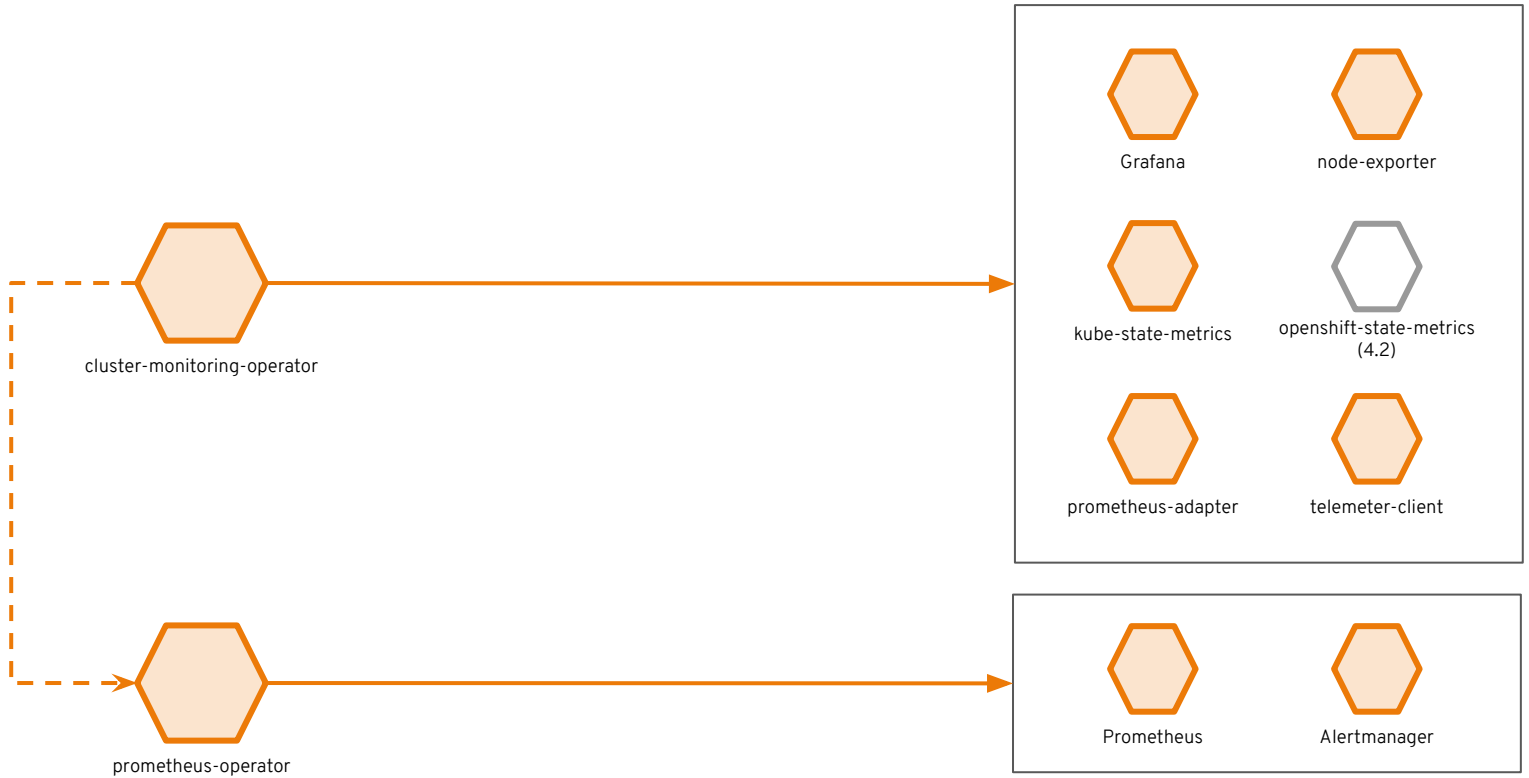
Red Hat

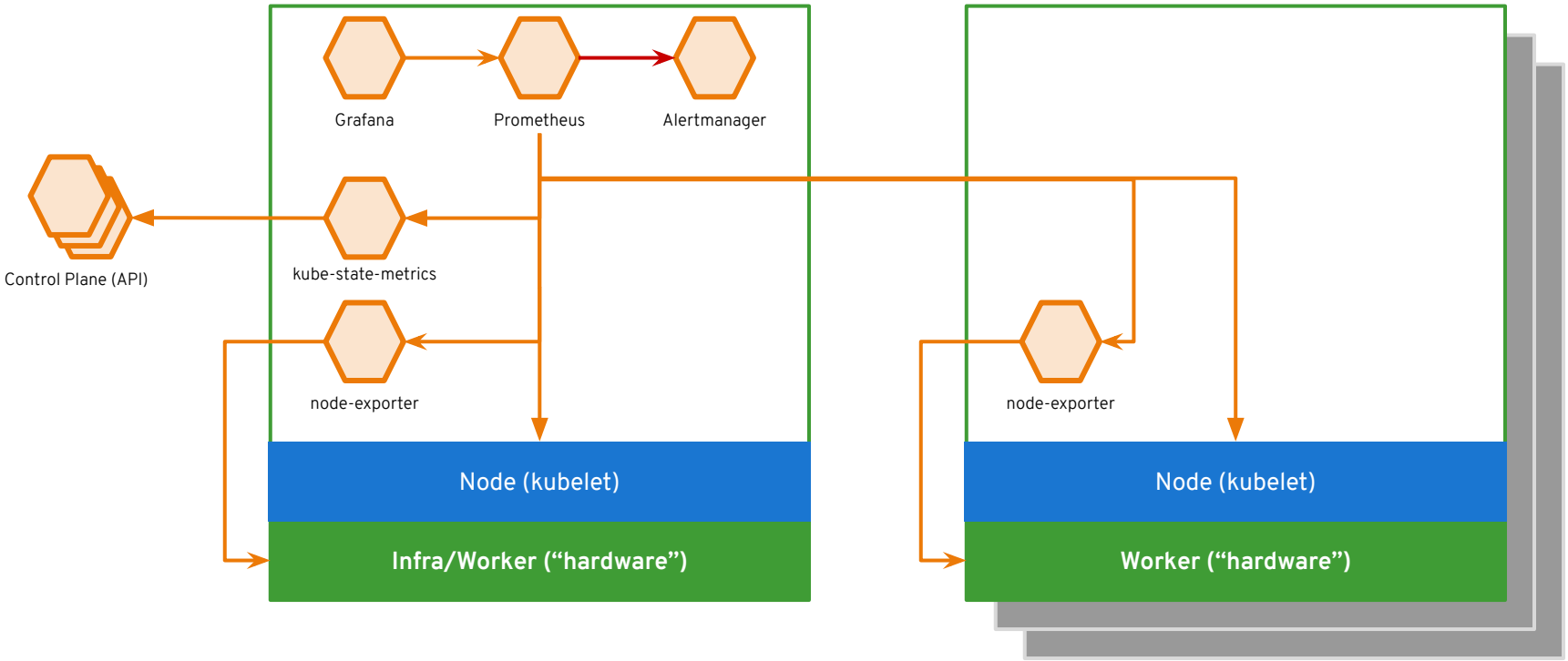
OpenShift Cluster Monitoring

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.


Grafana
node-exporter
kube-state-metrics
openshift-state-metrics
(4.2)
prometheus-adapter
telemeter-client
cluster-monitoring-operator
prometheus-operator
Prometheus
Alertmanager


Grafana
Prometheus
Alertmanager
Control Plane (API)
kube-state-metrics
node-exporter
node-exporter
Node (kubelet)
Node (kubelet)
Infra/Worker (“hardware”)
Worker (“hardware”)

OpenShift Logging

An integrated solution for exploring and corroborating application logs

Red Hat

Observability via log exploration and corroboration with EFK

Components

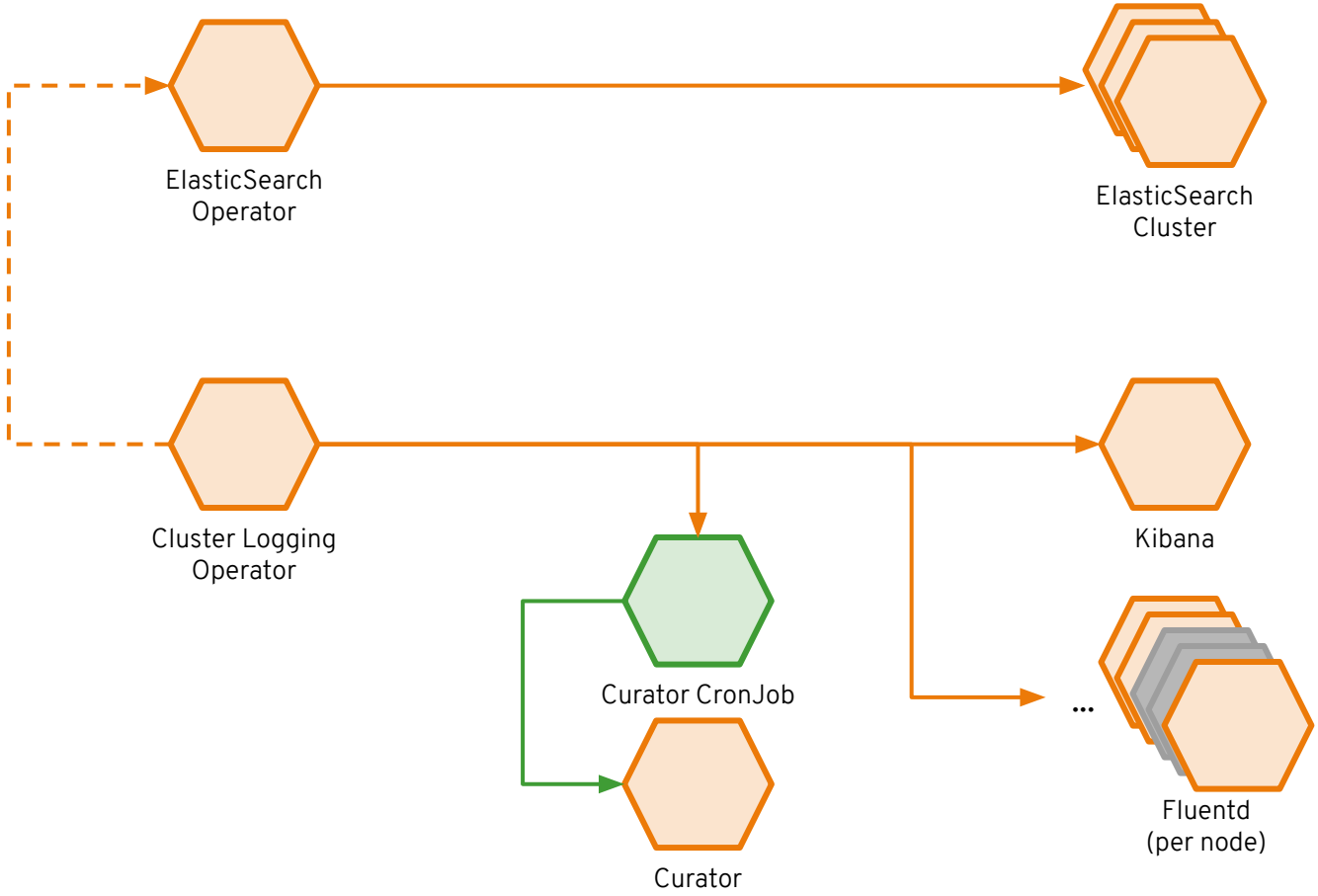
- **Elasticsearch:** a search and analytics engine to store logs
- **Fluentd:** gathers logs and sends to Elasticsearch.
- **Kibana:** A web UI for Elasticsearch.

Access control

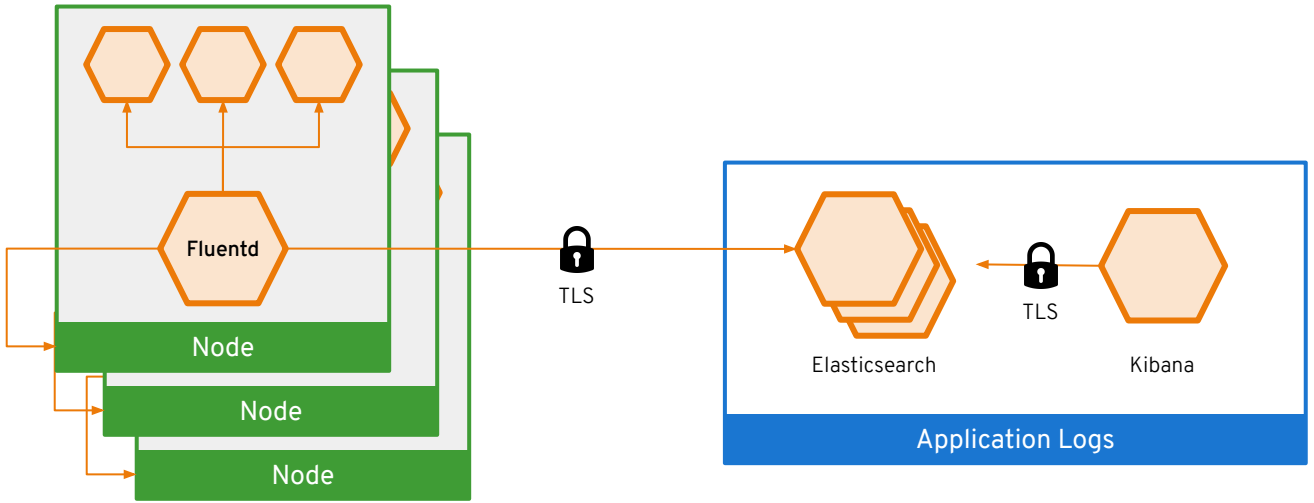
- Cluster administrators can view all logs
- Users can only view logs for their projects

Ability to forward logs elsewhere

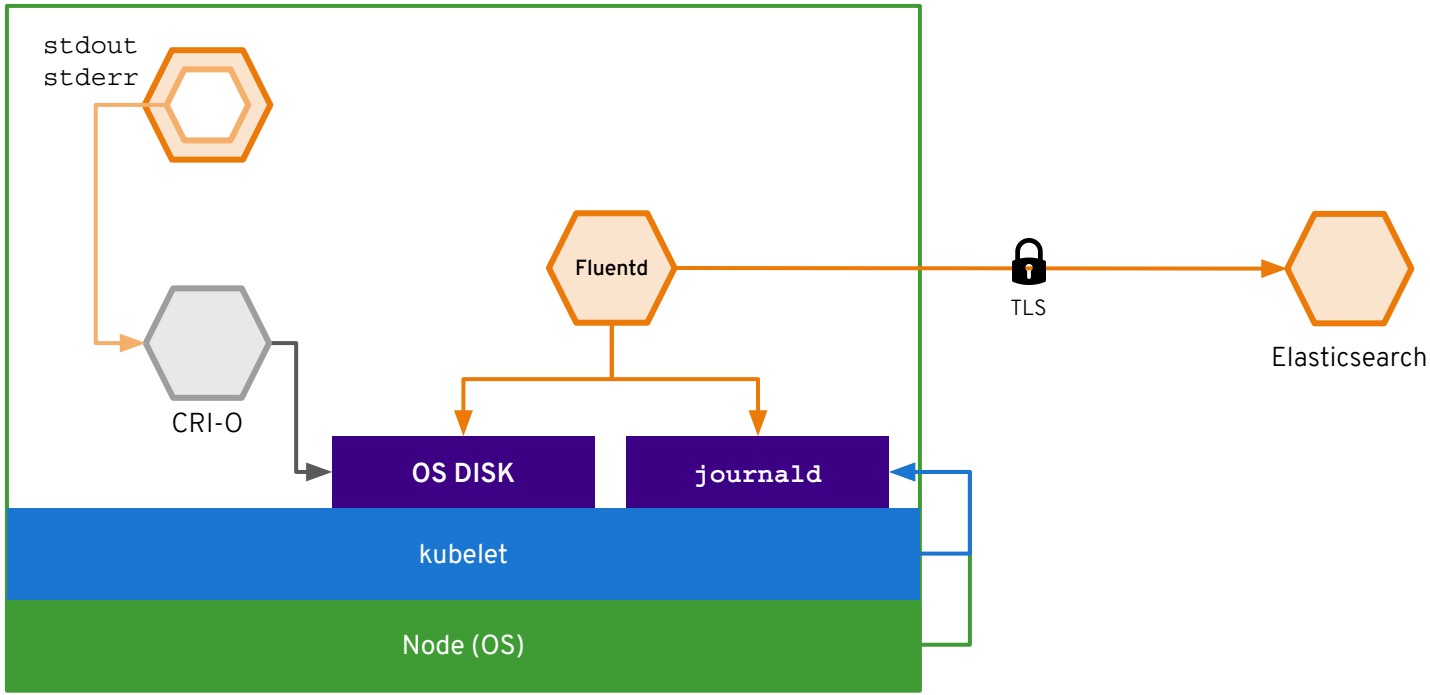
- External elasticsearch, Splunk, etc


ElasticSearch
Operator
ElasticSearch
Cluster
Cluster Logging
Operator
Kibana
Curator CronJob
...
Curator
Fluentd
(per node)

Log data flow in OpenShift



Log data flow in OpenShift



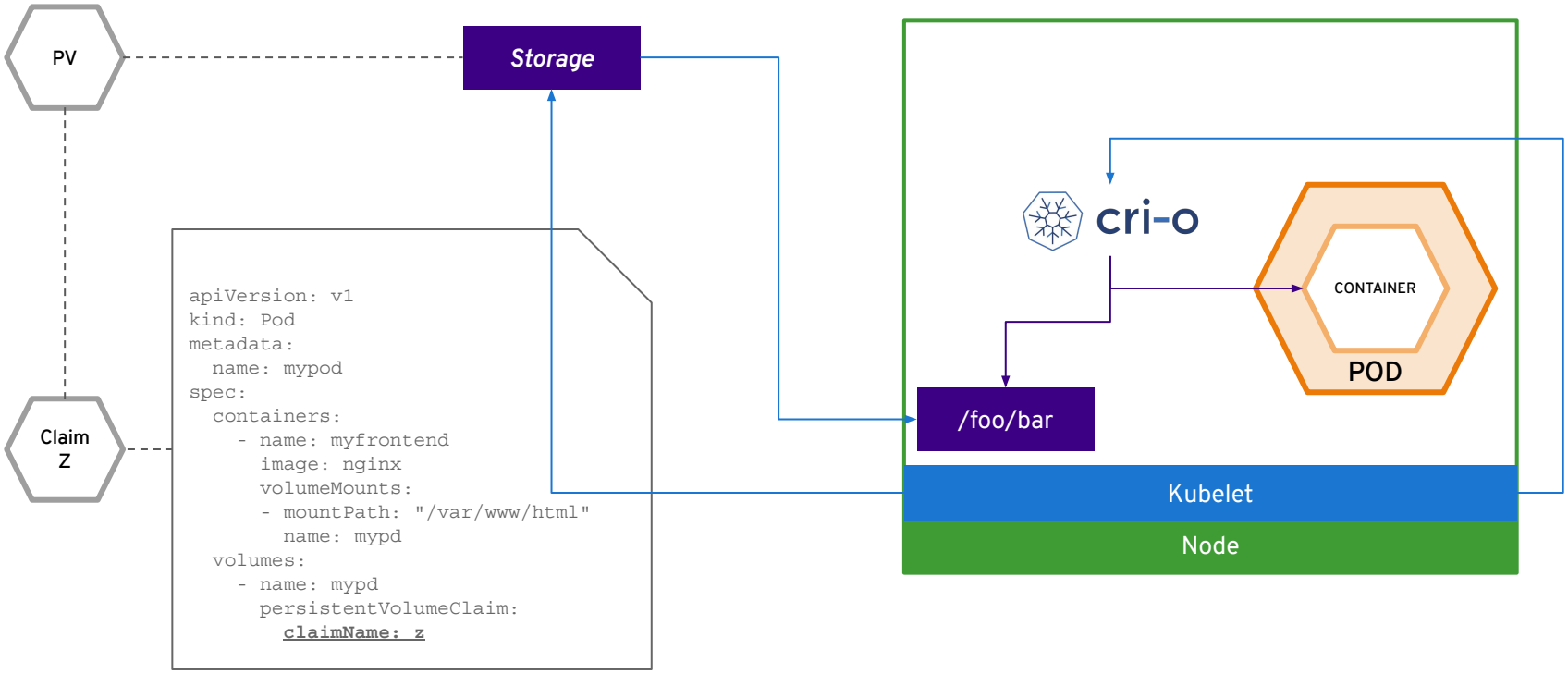
Persistent Storage

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

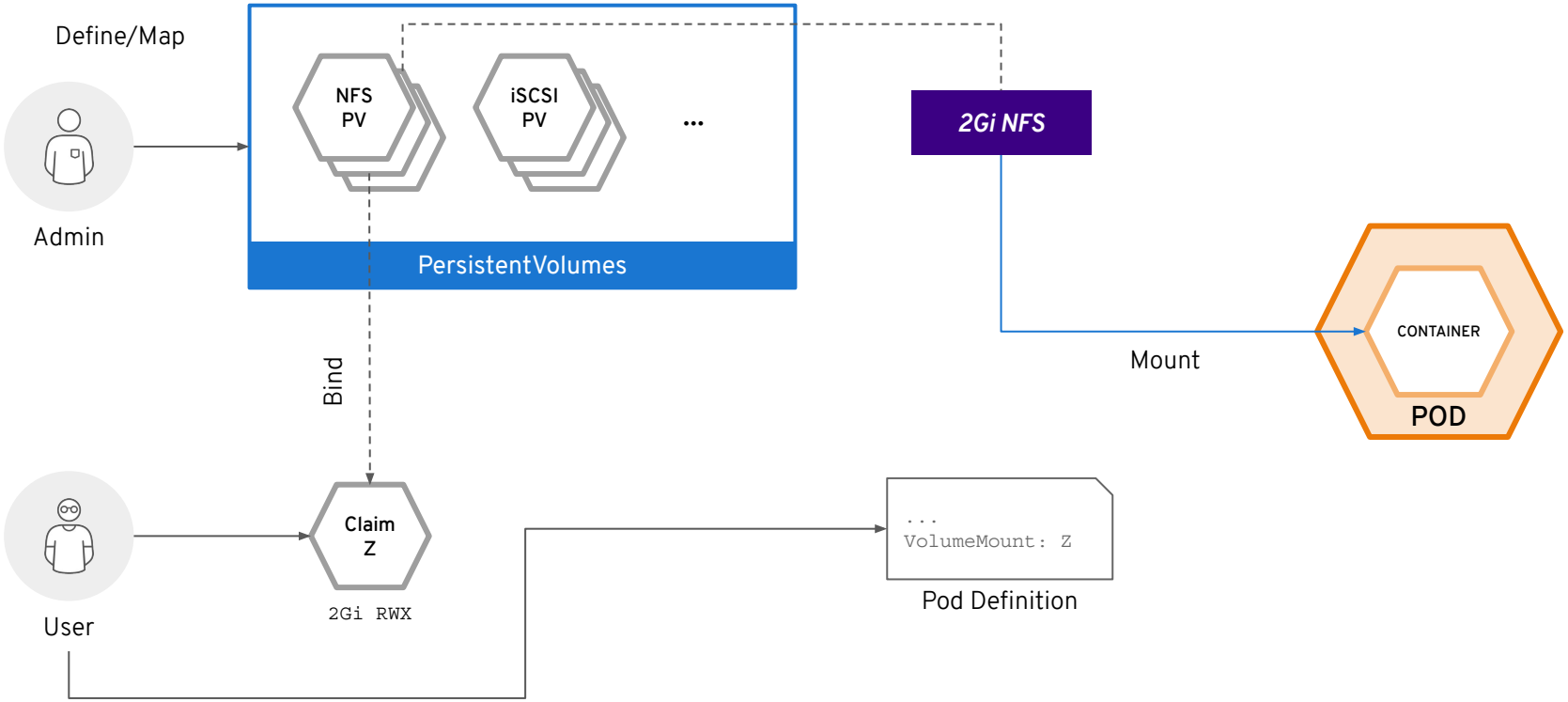
A broad spectrum of static and dynamic storage endpoints

NFS	OpenStack Cinder	iSCSI	Azure Disk	AWS EBS	FlexVolume
GlusterFS	Ceph RBD	Fiber Channel	Azure File	GCE Persistent Disk	VMWare vSphere VMDK
		NetApp Trident*	Container Storage Interface (CSI)**		

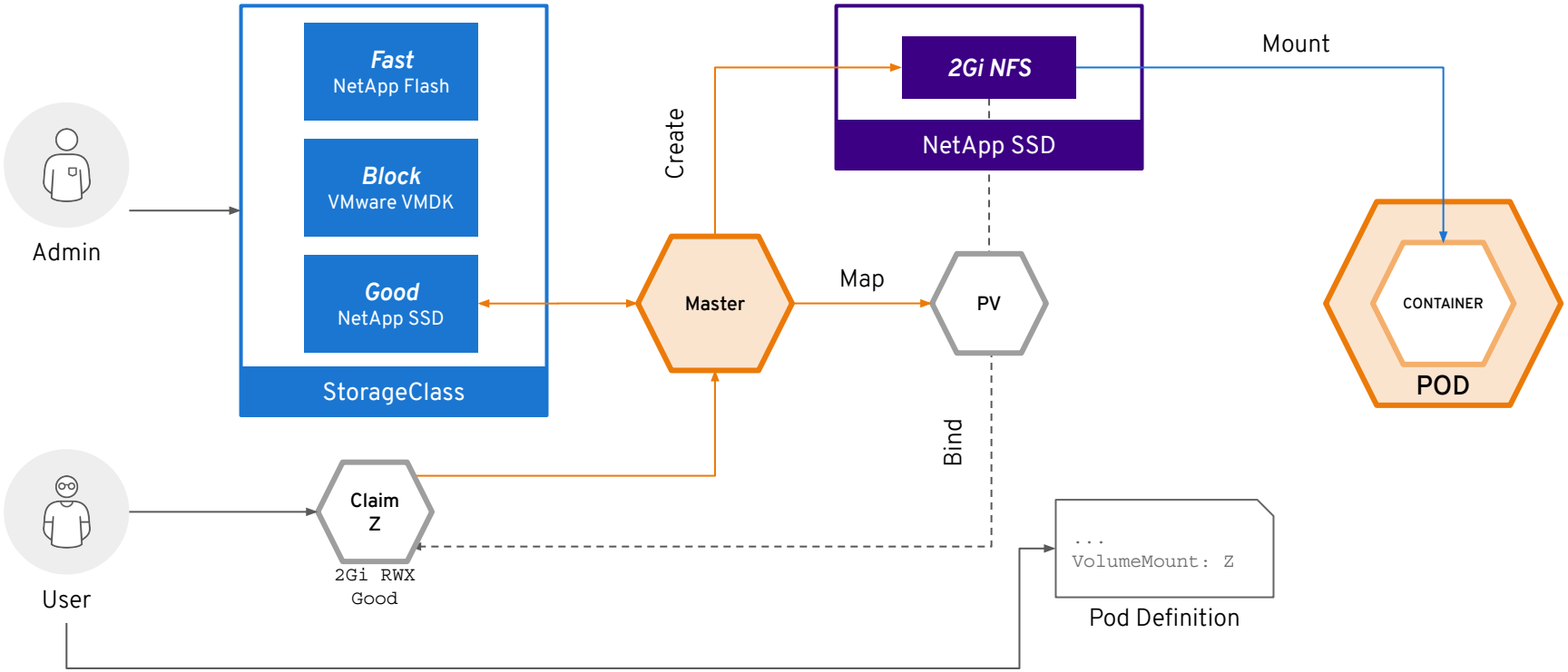
PV Consumption



Static Storage Provisioning



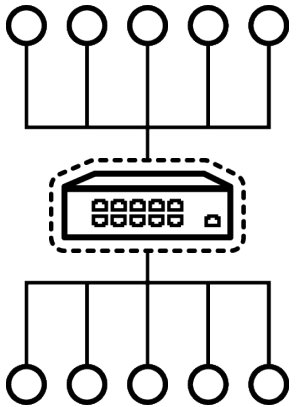
Dynamic Storage Provisioning



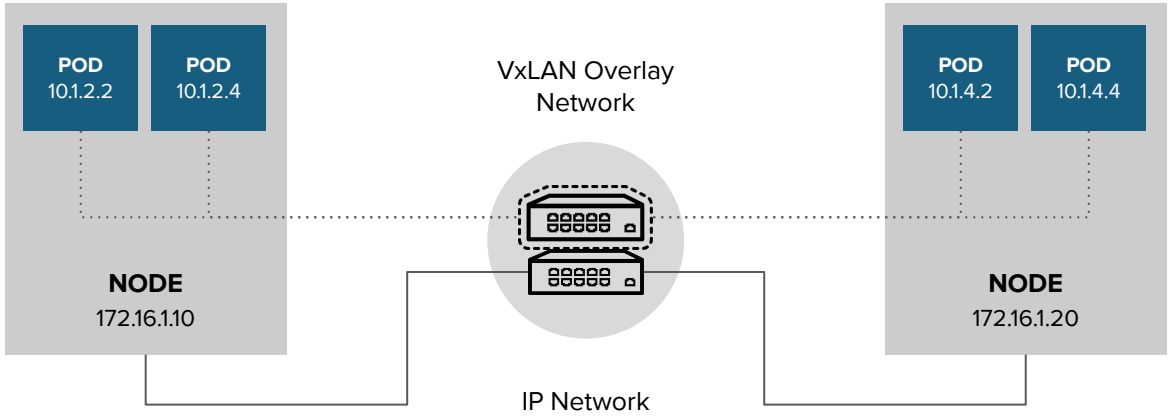
Networking

OPENSHIFT NETWORKING

- Built-in internal DNS to reach services by name
- Split DNS is supported via DNSmasq
  - Master answers DNS queries for internal services
  - Other name servers serve the rest of the queries
- Software Defined Networking (SDN) for a unified cluster network to enable pod-to-pod communication
- OpenShift follows the Kubernetes Container Networking Interface (CNI) plug-in model

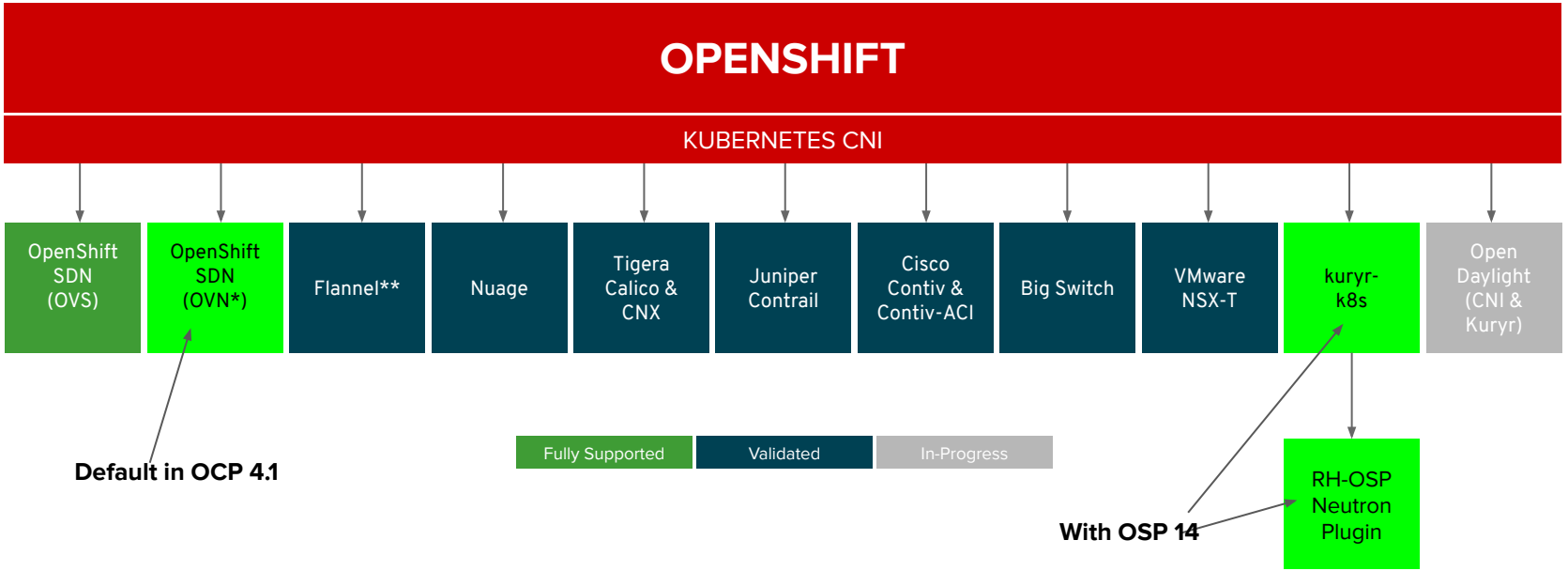


OPENSHIFT NETWORKING



Red Hat

OPENSHIFT NETWORK PLUGINS



Red Hat

OPENSHIFT SDN

FLAT NETWORK

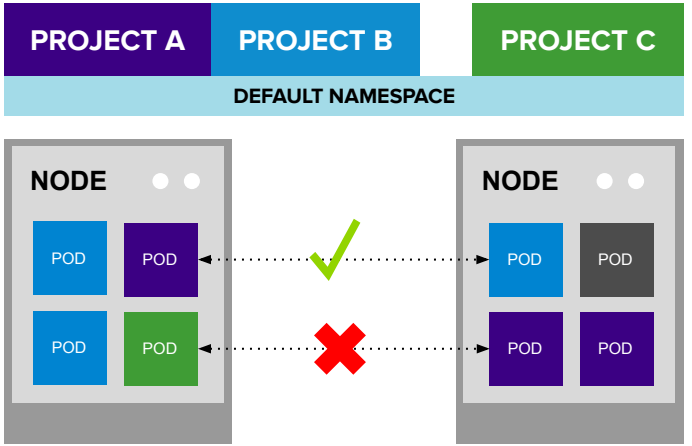
- All pods can communicate with each other across projects

MULTI-TENANT NETWORK

- Project-level network isolation
- Multicast support
- Egress network policies

NETWORK POLICY (Default)

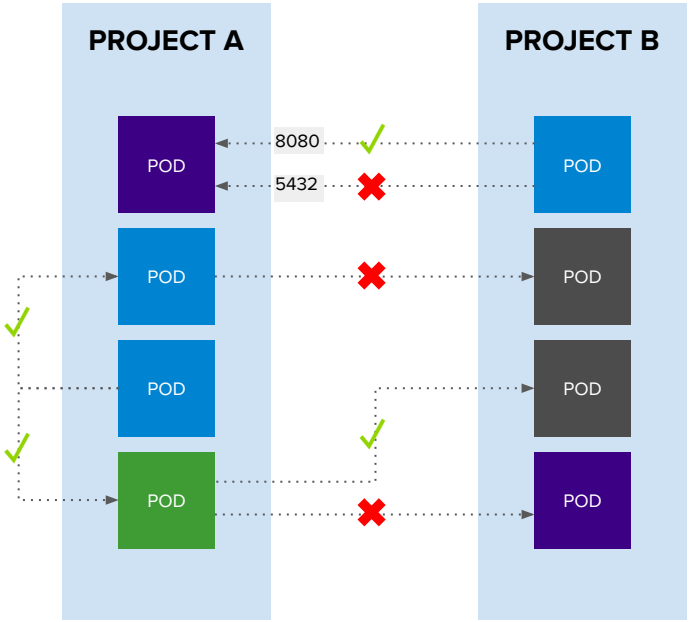
- Granular policy-based isolation



Multi-Tenant Network

Red Hat

OPENSHIFT SDN - NETWORK POLICY



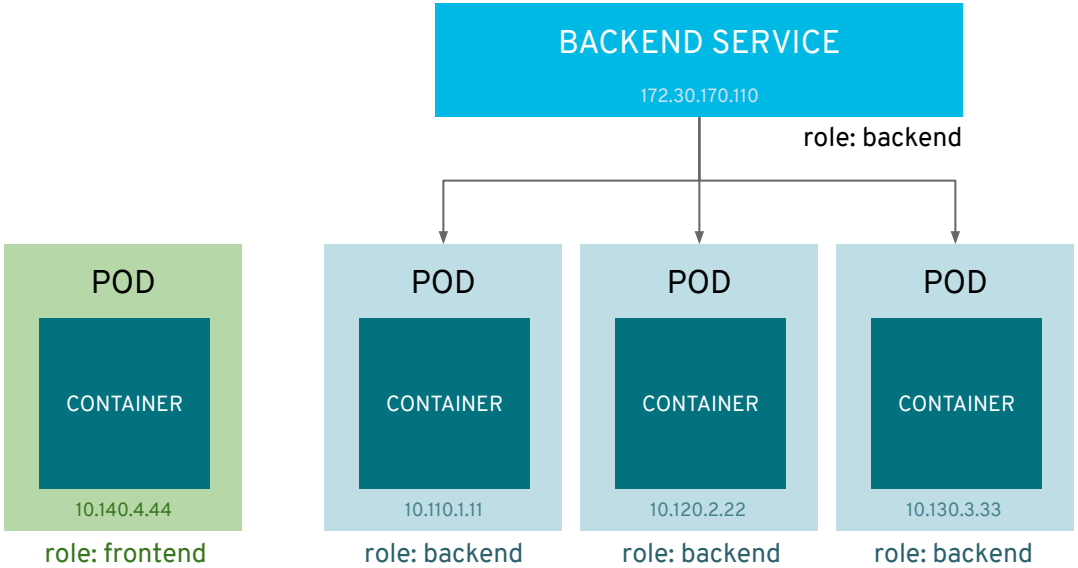
Example Policies

- Allow all traffic inside the project
- Allow traffic from green to gray
- Allow traffic to purple on 8080

```
apiVersion: extensions/v1beta1
kind: NetworkPolicy
metadata:
  name: allow-to-purple-on-8080
spec:
  podSelector:
    matchLabels:
      color: purple
  ingress:
  - ports:
    - protocol: tcp
      port: 8080
```

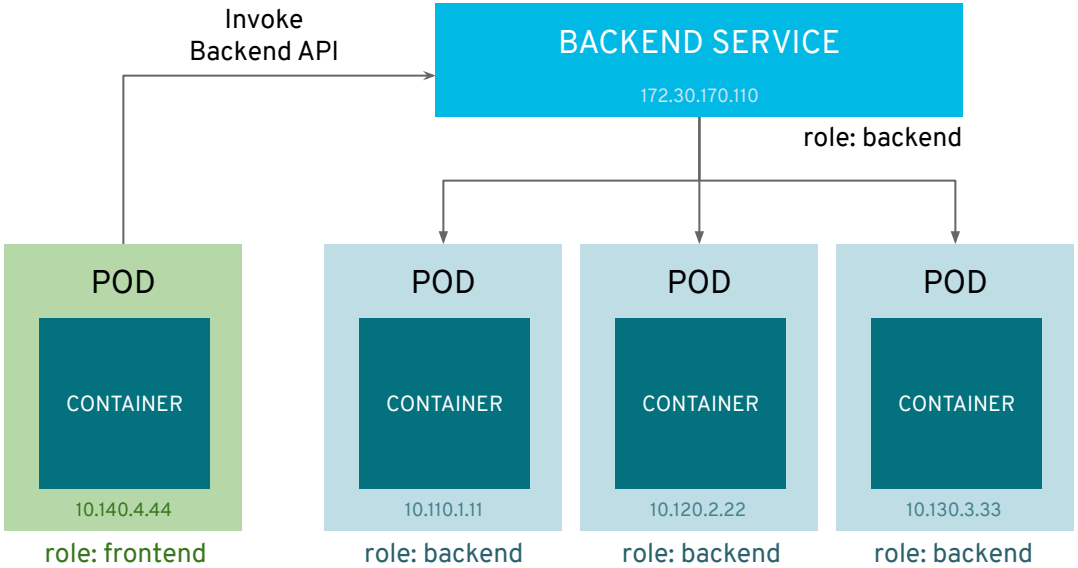

Red Hat

services provide internal load-balancing and service discovery across pods



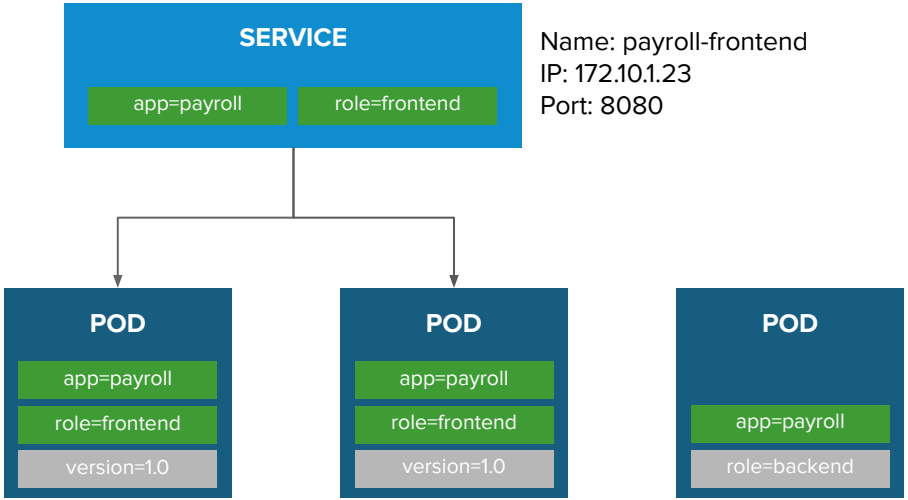
Red Hat

apps can talk to each other via services



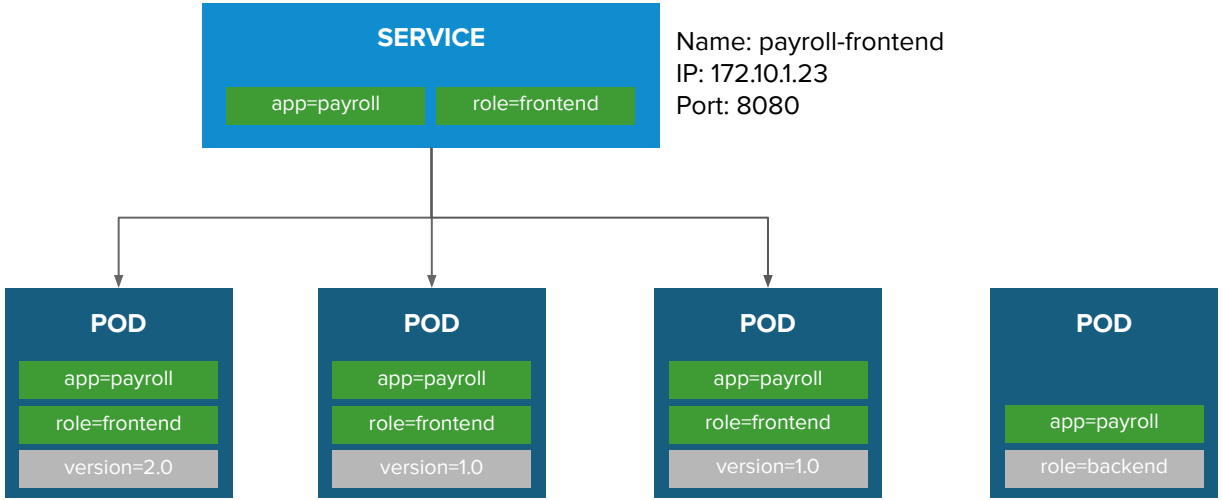
Red Hat

BUILT-IN SERVICE DISCOVERY
INTERNAL LOAD-BALANCING



Red Hat

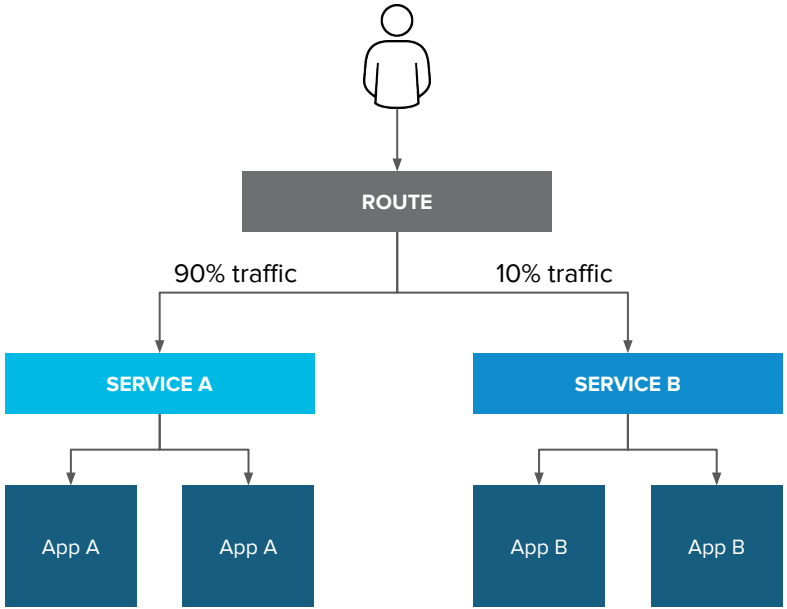
BUILT-IN SERVICE DISCOVERY
INTERNAL LOAD-BALANCING



Red Hat

ROUTE SPLIT TRAFFIC

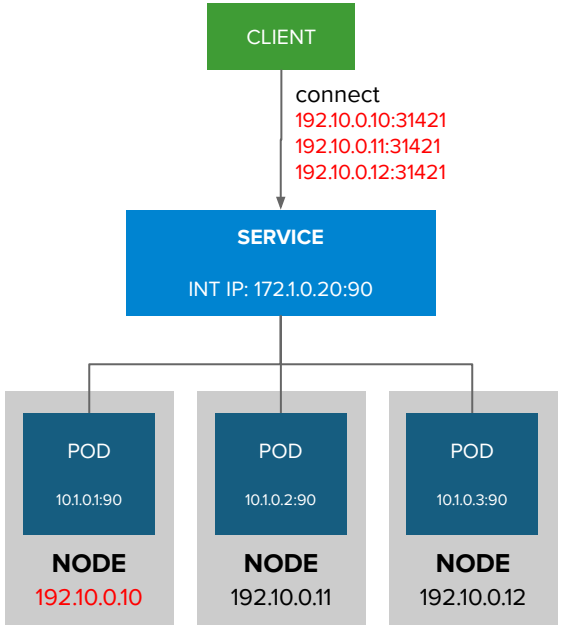
Split Traffic Between Multiple Services For A/B Testing, Blue/Green and Canary Deployments



Red Hat

EXTERNAL TRAFFIC TO A SERVICE ON A RANDOM PORT WITH NODEPORT

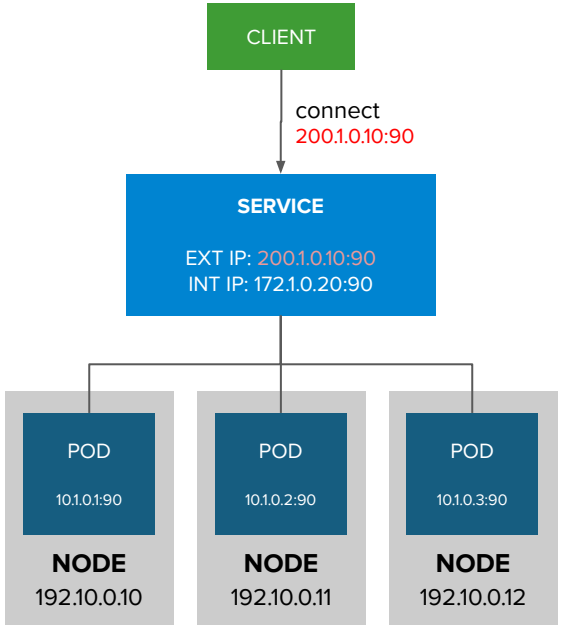
- NodePort binds a service to a unique port on all the nodes
- Traffic received on any node redirects to a node with the running service
- Ports in 30K-60K range which usually differs from the service
- Firewall rules must allow traffic to all nodes on the specific port



Red Hat

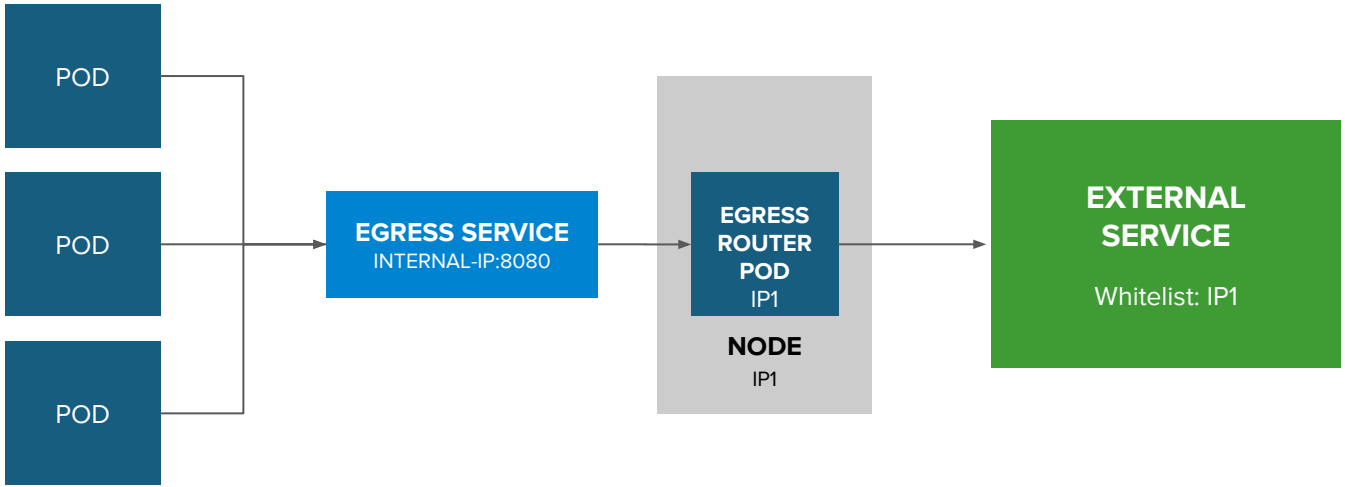
EXTERNAL TRAFFIC TO A SERVICE ON ANY PORT WITH INGRESS

- Access a service with an external IP on any TCP/UDP port, such as
  - Databases
  - Message Brokers
- Automatic IP allocation from a predefined pool using Ingress IP Self-Service
- IP failover pods provide high availability for the IP pool



Red Hat

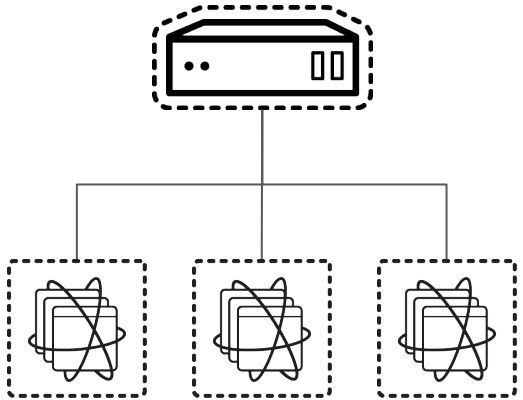
CONTROL OUTGOING TRAFFIC SOURCE IP WITH EGRESS ROUTER



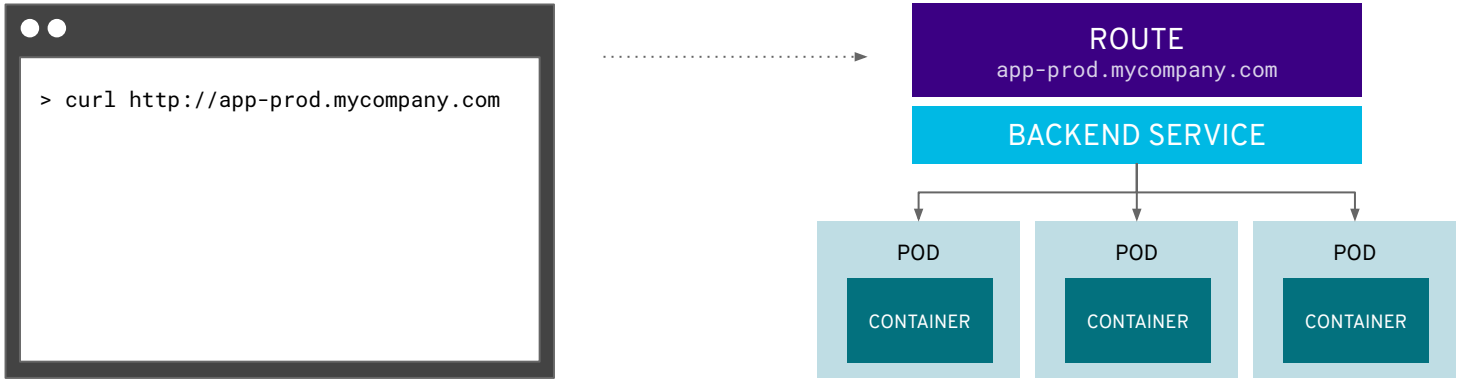
Red Hat

ROUTING AND EXTERNAL LOAD-BALANCING

- Pluggable routing architecture
  - HAProxy Router
  - F5 Router
- Multiple-routers with traffic sharding
- Router supported protocols
  - HTTP/HTTPS
  - WebSockets
  - TLS with SNI
- Non-standard ports via cloud load-balancers, external IP, and NodePort

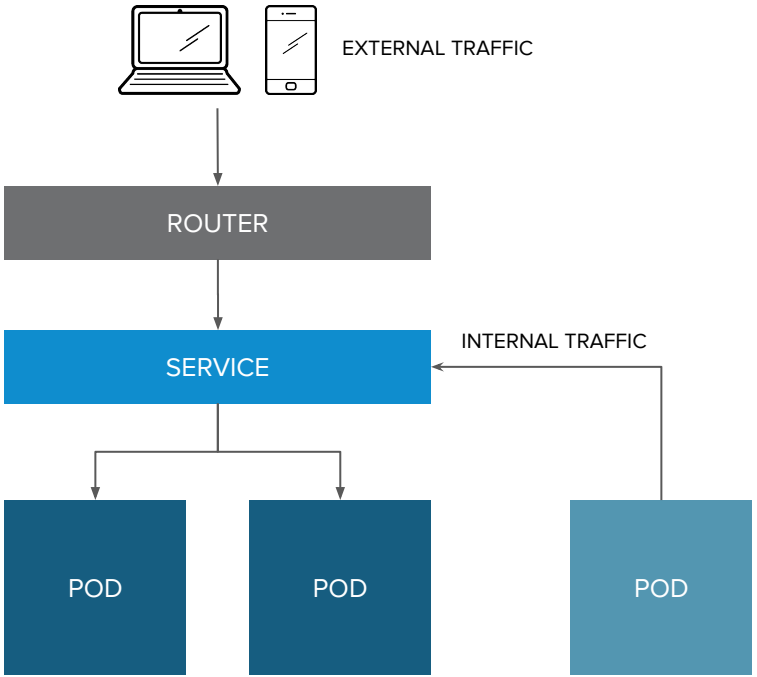


routes add services to the external load-balancer and provide readable urls for the app



Red Hat

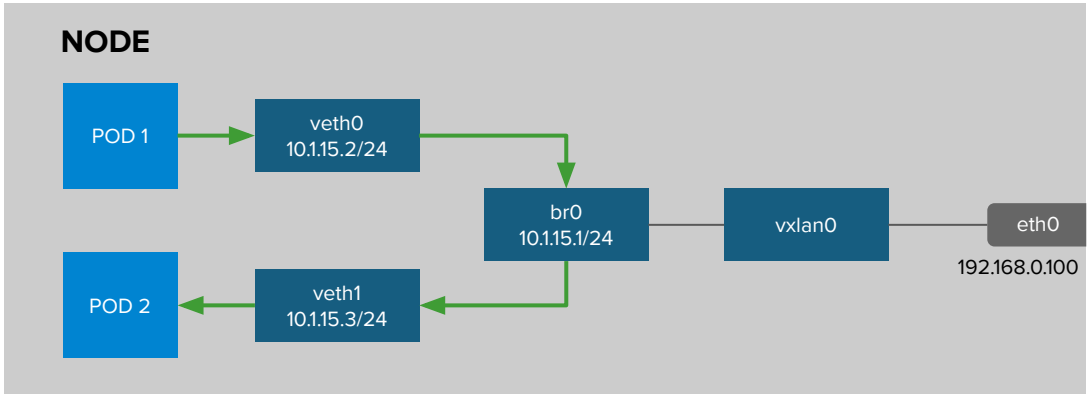
ROUTE EXPOSES SERVICES EXTERNALLY



Red Hat

OPENSHIFT SDN - OVS PACKET FLOW

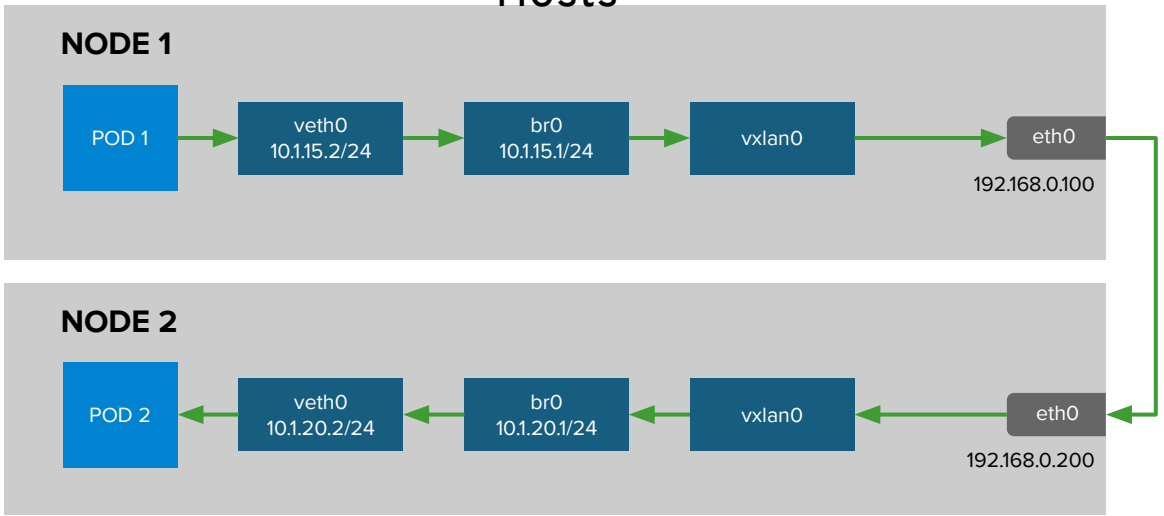
Container to Container on the Same Host



Red Hat

OPENSHIFT SDN - OVS PACKET FLOW

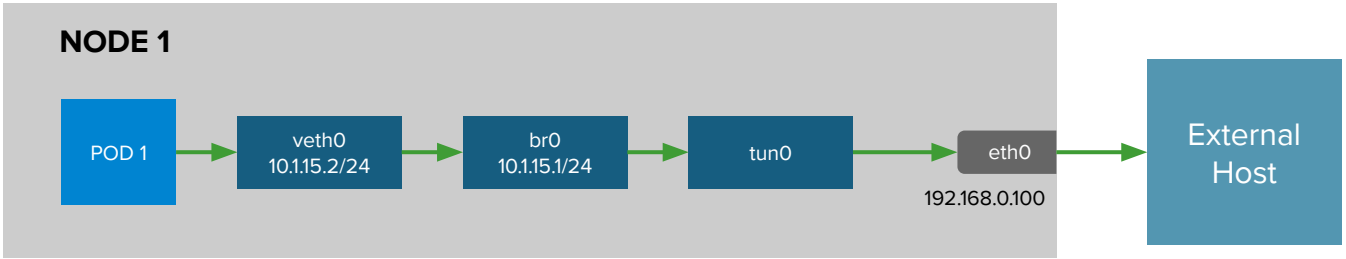
Container to Container on the Different Hosts



Red Hat

OPENSHIFT SDN - OVS PACKET FLOW

Container Connects to External Host



Red Hat

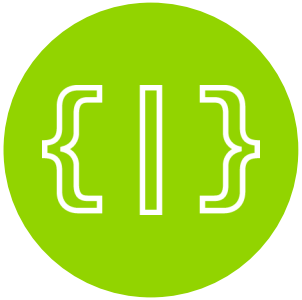
Build and Deploy Container Images

Tools and automation

that makes developers

productive quickly

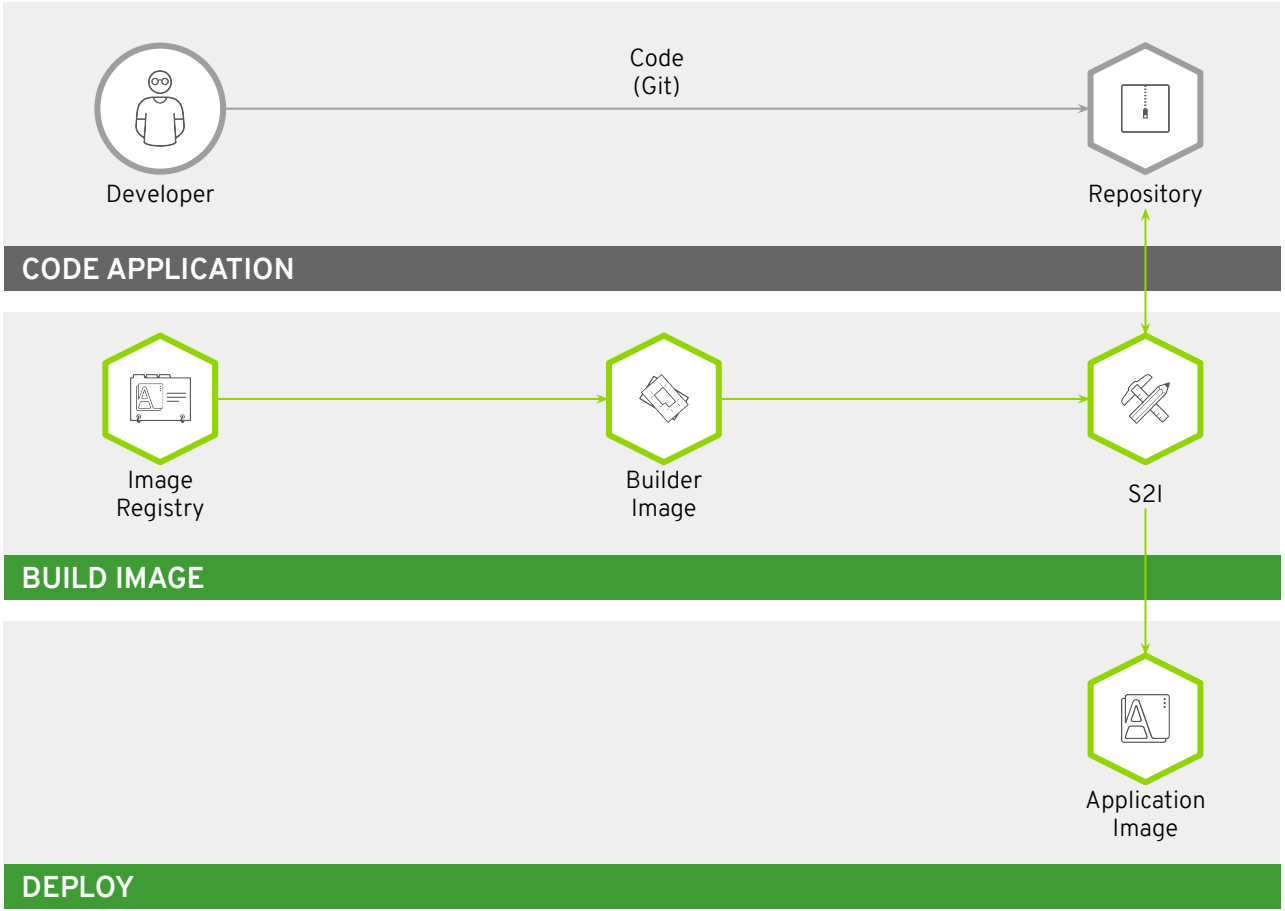
Red Hat



DEPLOY YOUR SOURCE CODE

DEPLOY YOUR APP BINARY

DEPLOY YOUR CONTAINER IMAGE


Code
(Git)
Developer
Repository
CODE APPLICATION
Image
Registry
Builder
Image
S2I
BUILD IMAGE
Application
Image
DEPLOY

Thank you

See you tomorrow

Red Hat