OpenShift on Microsoft Azure Workshop

Eric Webb, Cardinal Solutions

AGENDA

8:30 – 9:00	Breakfast
9:00 – 9:10	Welcome – Kevin McCauley, Red Hat
9:10 – 9:45	Intro to Docker, Containers, Kubernetes, Microservices – Eric Webb
9:45 – 10:00	Client tools setup and overview of labs 1-5 – Eric Webb
10:00 – 10:45	Labs 1-5
10:45 – 11:00	Break
11:00 – 11:30	OpenShift Core Concepts – Eric Webb
11:30 – 12:15	Lunch
12:15 – 12:40	Blackduck Container Security
12:40 – 1:00	OpenShift management with Cloud Forms
1:00 – 1:15	Break
1:15 – 1:20	Overview of labs 6-10
1:20 – 2:00	Labs
2:00 – 2:45	CI/CD with Jenkins – Eric Webb
2:45 – 2:55	Break
2:55 – 3:00	Overview of labs 11-15
3:00 - 3:40	Labs
3:40 - 3:50	Getting started w/ OpenShift on your Azure subscription – Eric Webb
3:50 - 4:00	Wrap-up, survey, & raffle - Eric Webb

redhat.

RED HAT ENTERPRISE LINUX– THE #1 LINUX OS FOR PUBLIC CLOUD DEPLOYMENTS

- Organizations are turning to Linux to deploy key public cloud workloads.

- Respondents’ requirements for enterprise Linux in public clouds are the same as those for on-premise deployments.

- Red Hat® Enterprise Linux continues as the Linux market leader in public cloud deployments

RED HAT ENTERPRISE LINUX

redhat

STATE OF LINUX IN THE PUBLIC CLOUD FOR ENTERPRISES

A study by Management Insight Technologies, sponsored by Red Hat

SOLUTION OVERVIEW

ACCORDING TO RESPONDENTS:

54%

of all applications running in public cloud infrastructure are running on Linux virtual machines (VMs)

65%

of enterprise Linux operating system deployments in the public cloud are paid

#1

commercial Linux distribution in the public cloud is Red Hat Enterprise Linux

SUMMARY

Linux® has become the de facto standard for highly available, reliable, and critical workloads in datacenters and cloud computing environments. It supports multiple use cases, target systems, and devices, depending on user needs and workloads. According to the Linux Foundation, nine of the top ten public clouds run on Linux[1]. Every major public cloud provider–including Amazon Web Services (AWS), Microsoft Azure, Google Cloud Platform (GCP), and Alibaba Cloud–offers multiple distributions of Linux in their marketplaces. In fact, nearly 30% of virtual machines (VMs) running on Microsoft Azure are Linux-based[2].

A 2017 study conducted by Management Insight Technologies and sponsored by Red Hat examined the preferences and characteristics in a Linux operating system (OS) distribution for public clouds. This survey included more than 500 cloud IT decision makers across North America and Europe, within organizations with 500 or more employees and across various industries.

According to this study, Linux has grown in importance in the public cloud among respondents, providing many of the same capabilities that have made it so popular in on-premise enterprise IT.

STUDY HIGHLIGHTS

Organizations are turning to Linux to deploy key public cloud workloads.

- Most respondents’ enterprises plan to deploy one to two public cloud platforms (Figure 4).
- Just over half of all respondents’ public cloud applications are running on Linux VMs (Figure 3).
- The most popular public cloud workloads last deployed on Linux by respondents were structured databases and web applications (Figure 5).

Respondents’ requirements for enterprise Linux in public clouds are the same as those for on-premise deployments.

- Reliability, security, ease of deployment, and maintainability are the top rated capabilities for deploying a Linux operating system on public cloud infrastructure (Figure 6).
- 78% of respondents prefer paid, commercially supported Linux for their critical production workloads (Figure 7).
- 65% of respondents’ enterprise Linux deployments in public clouds are paid and commercially supported (Figure 8).

facebook.com/redhatinc
@redhatnews
linkedin.com/company/red-hat

redhat.com

[1] "2017 State of Linux Kernel Development." The Linux Foundation. https://www.linuxfoundation.org/2017-linux-kernel-report-landing-page/

[2] "Microsoft by the Numbers." Microsoft Story Labs. https://news.microsoft.com/bythenumbers/azure-virtual

Microsoft + Red Hat: Stronger Together

Wide **availability** of Red Hat solutions whether PAYG or BYOS, across all Azure regions.

Microsoft Azure participation in Red Hat Cloud & Service Provider Program (CCSP)

Developers can easily create and **deploy** apps with a .NET front-end on Windows and a MySQL database on Red Hat Enterprise Linux through OpenShift Container Platform.

.

Secure, **manageable** and well-supported Red Hat solutions in the Microsoft cloud, including Red Hat Enterprise Linux, OpenShift, CloudForms, Ansible and JBOSS middleware.

Integrated enterprise-grade support spanning hybrid cloud, including co-located support resources.

Released, & Coming Soon

Released

- SQL 2017 on RHEL
- SAP, SAP HANA on RHEL

Spring

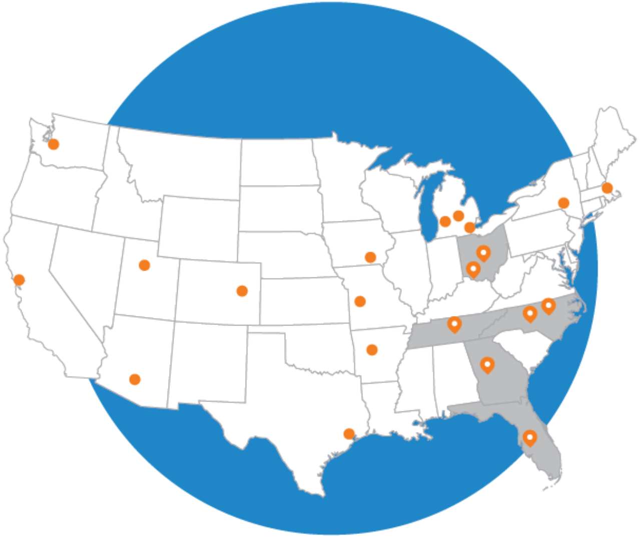
- OpenShift Dedicated for Azure

Summer

- Windows Server Container in OpenShift
- Support for RHEL on Azure Stack

cardinal

Cardinal provides creative technology solutions that transform client visions into compelling business and customer experiences.



1996 EST. IN

7 LOCATIONS

450 CONSULTANTS

65+M REVENUE

NATIONAL FOOTPRINT

Cloud

Mobile

Data

Web

Microsoft cloudera Pivotal redhat sitecore MicroStrategy tableau

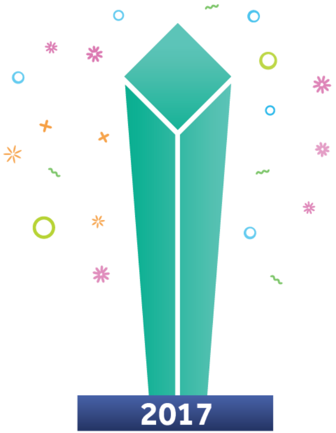
MICROSOFT PARTNERSHIP

Microsoft Partner Since 2001

Microsoft National Solution Provider Program

Competencies

- Application Development
- Cloud Platform
- Cloud Productivity
- Collaboration and Content
- Data Analytics
- Data Platform
- DevOps
- Enterprise Mobility
- Project, Program, Portfolio Mgmt



2017 Partner of the Worldwide Year

Open Source
on Azure

2017 East Region SMS&P Partner of the Year

2015 Central Region Partner of the Year

Build the
Intelligent Cloud

CONTAINERS AND MICROSERVICES: BUSINESS VALUE OF DEVOPS

Digital Transformation

There is Evolution in...

APPLICATIONS

New ways of developing, delivering and integrating applications

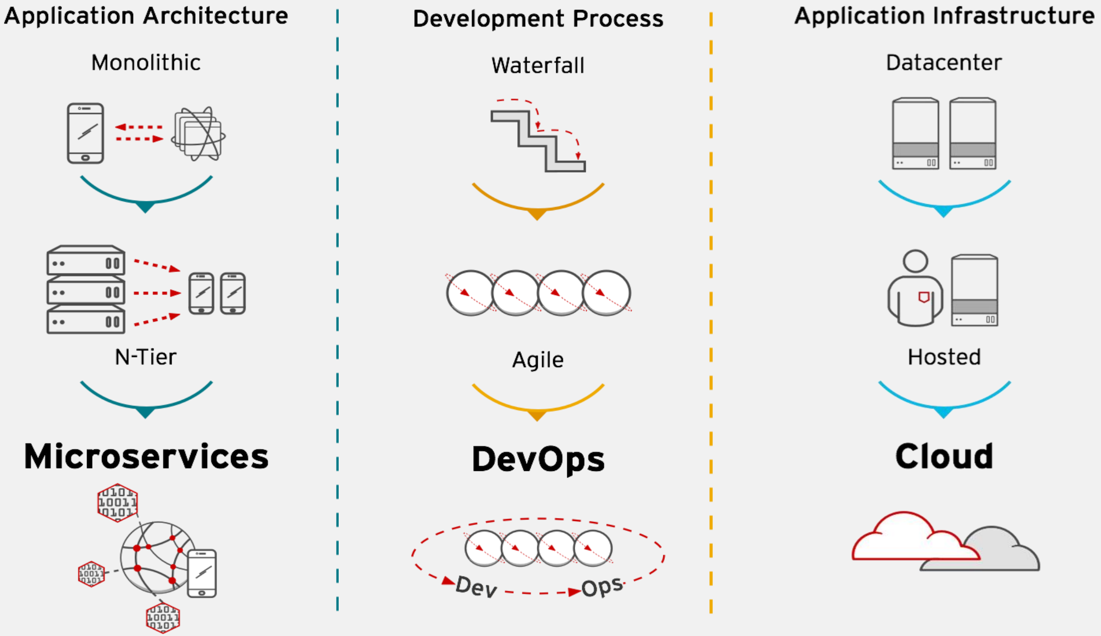
PROCESS

More agile processes across both IT and the business

INFRASTRUCTURE

Modernize existing and build new cloud based infrastructure

EVOLUTION OF APPS, INFRASTRUCTURE & PROCESS



Application Architecture

Monolithic

N-Tier

Microservices

Development Process

Waterfall

Agile

DevOps

Application Infrastructure

Datacenter

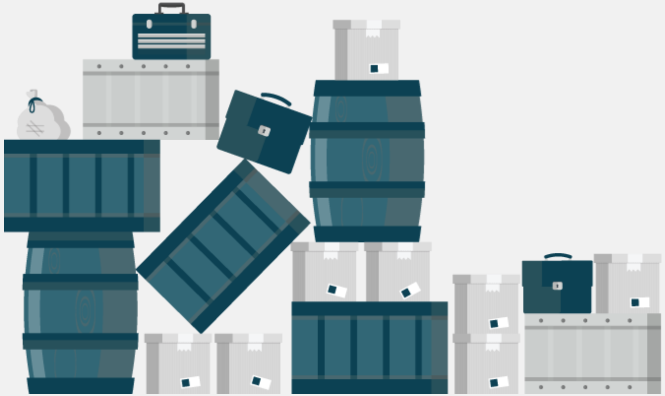
Hosted

Cloud

THE PROBLEM

Existing applications require complicated installation and integration every time they are deployed leading to

- Slow service delivery
- Reduced service quality
- Frequent down times

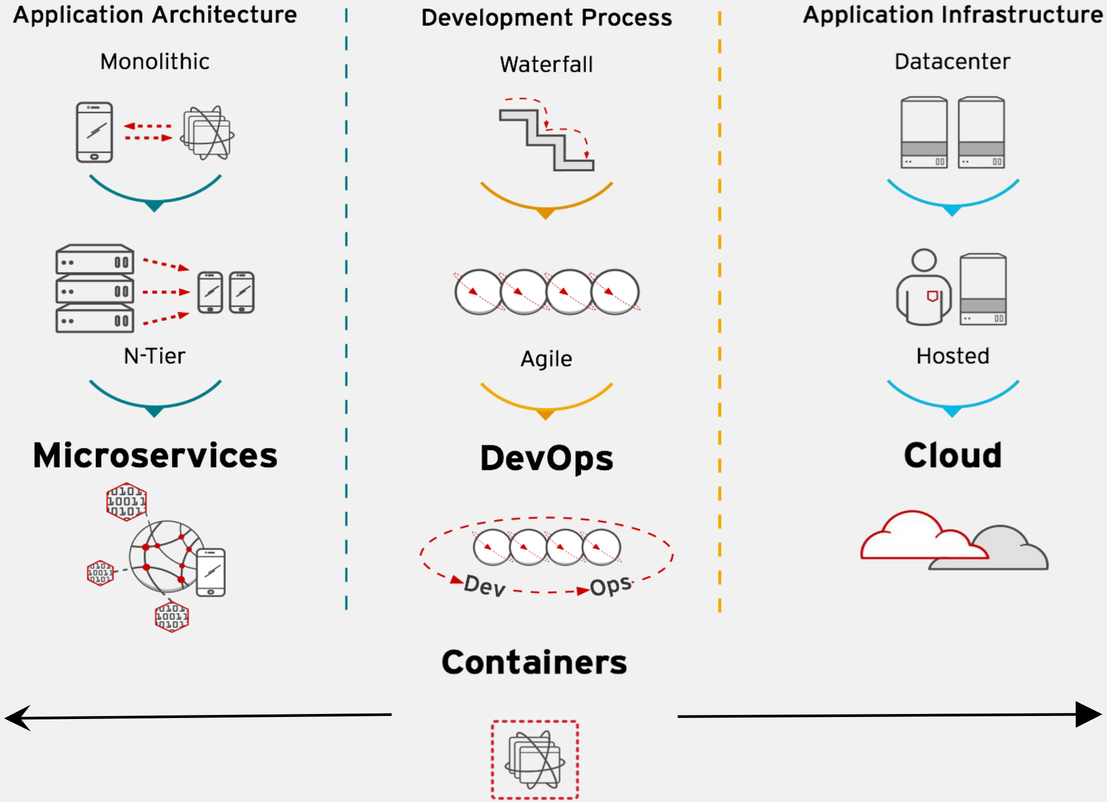


THE SOLUTION

Adopting a container strategy will allow applications to be easily shared and deployed

- Consistent env and tools
- Predictable building blocks
- Faster deployment

CONTAINERS - TRANSFORM APPS, INFRASTRUCTURE & PROCESS



Application Architecture

Monolithic

N-Tier

Microservices

Development Process

Waterfall

Agile

DevOps

Application Infrastructure

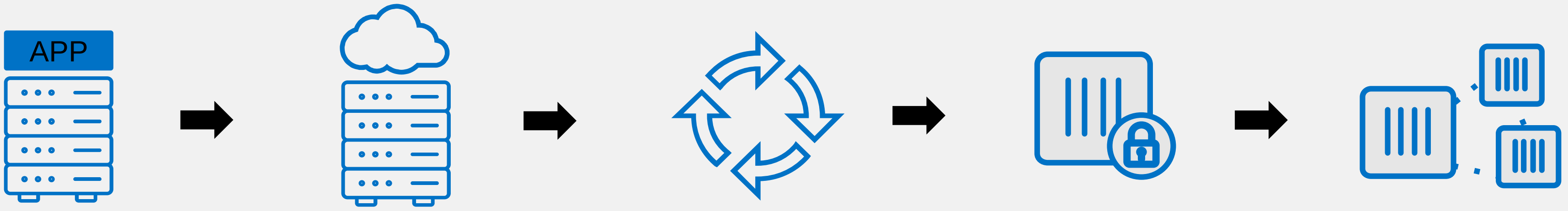
Datacenter

Hosted

Cloud

Containers

APP MODERNIZATION PATH



Existing Application

Modern Infrastructure

Move to the
cloud as VMs
or Containers
or refresh HW.

Modern Methodologies

Implement
CI/CD
and
automation.

Containerize Applications

Re-architect
apps for scale
with containers.

Modern Microservices

Add new services
or start peeling off
services from
monolithic code.

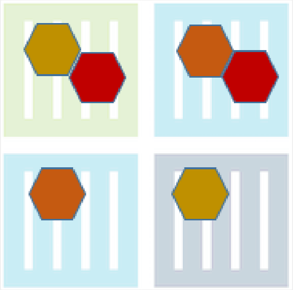
ONE MISCONCEPTION...

CONTAINERS & MICROSERVICES

Microservices ≠ containers

Microservices is a application design paradigm

Containers are a implementation detail that can help

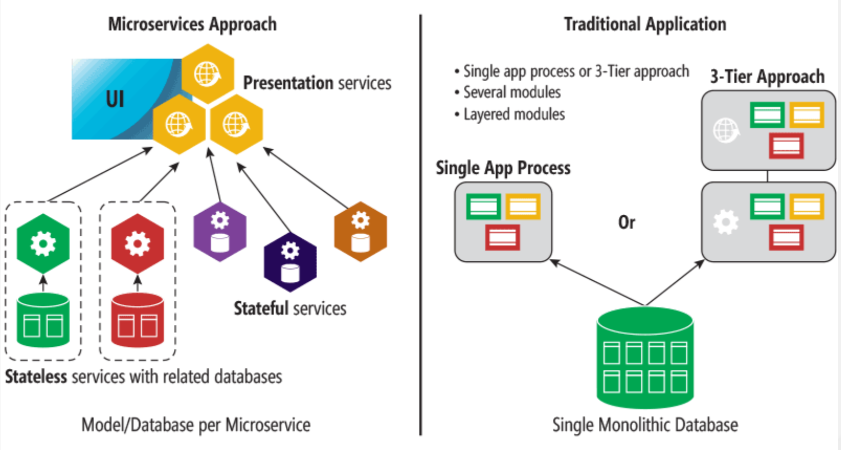


WHAT IS A MICROSERVICE?

Architecture style

Structures an application as a collection of loosely coupled services, which implement business

Each service is running in its own process and communicating with lightweight mechanisms like REST



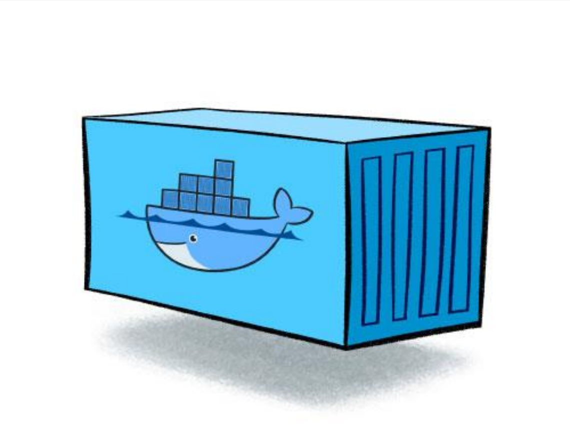
WHAT IS A CONTAINER?

Slice up the OS to run multiple apps on a single VM

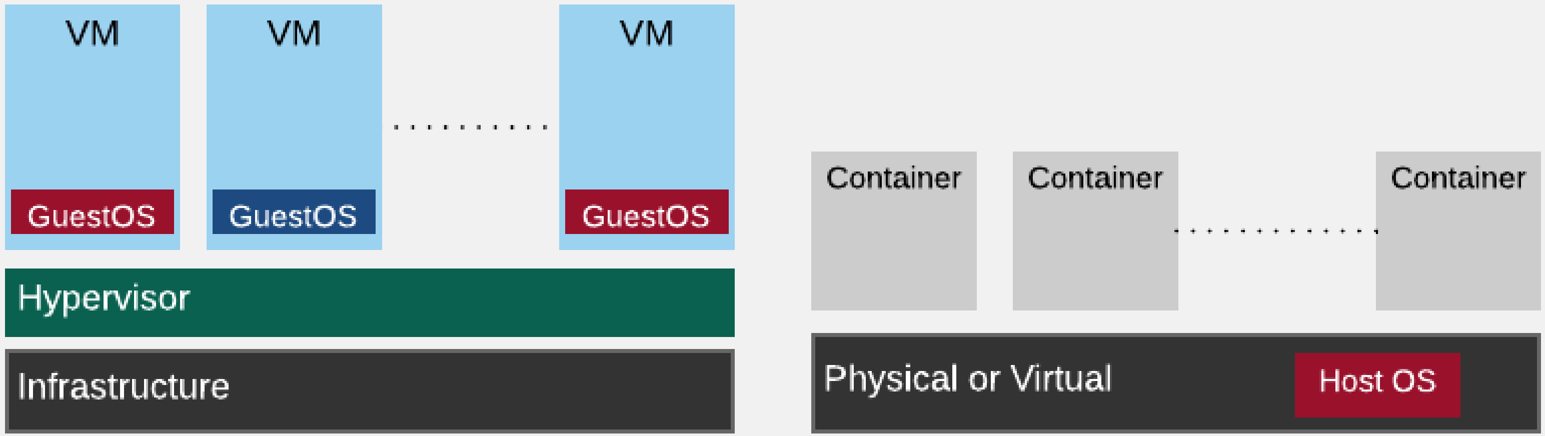
Every container has an isolated view

Shared kernel, very fast start-up, and repeatable execution

Cannot mix OS types on hosts and containers



UNDERSTANDING CONTAINERS



VM VM VM

GuestOS GuestOS GuestOS

Hypervisor

Infrastructure

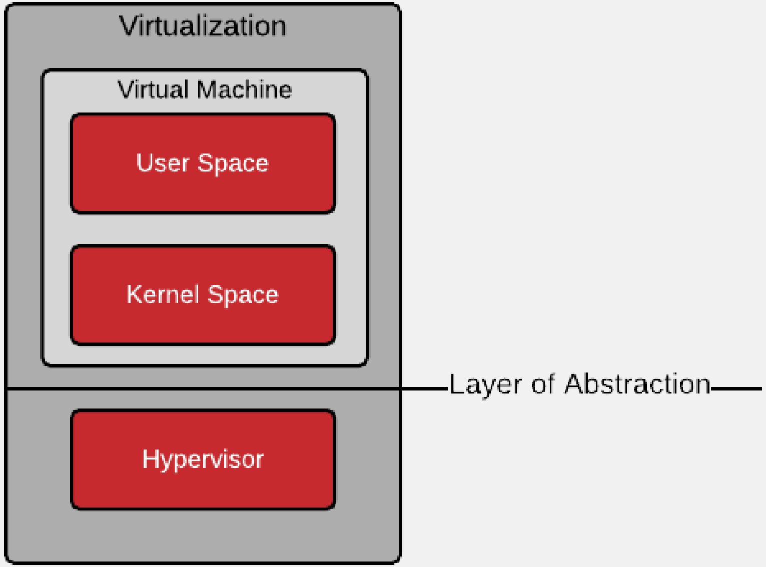
Container Container Container

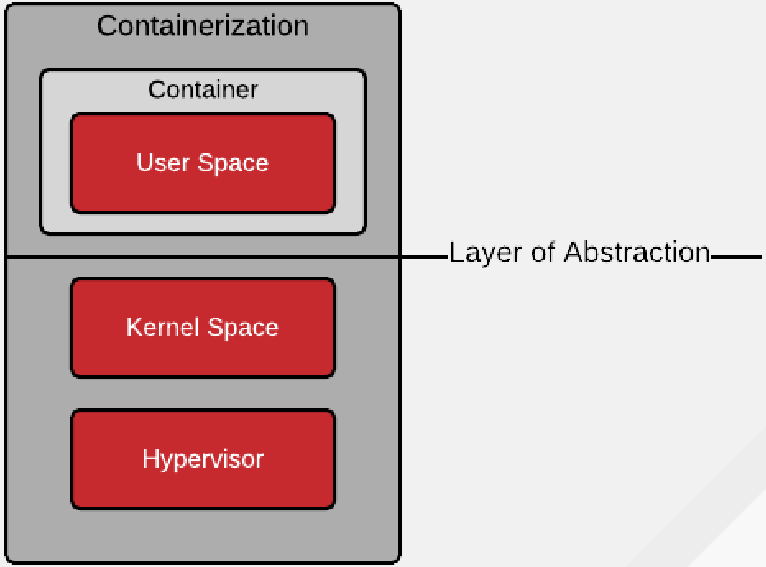
Physical or Virtual

Host OS

Virtual Machines vs Containers

VIRTUALIZATION VS CONTAINERIZATION





WHAT ABOUT DOCKER?

Containers have been around for many years

Linux kernel: cgroups, namespaces

Docker Inc. did not invent them

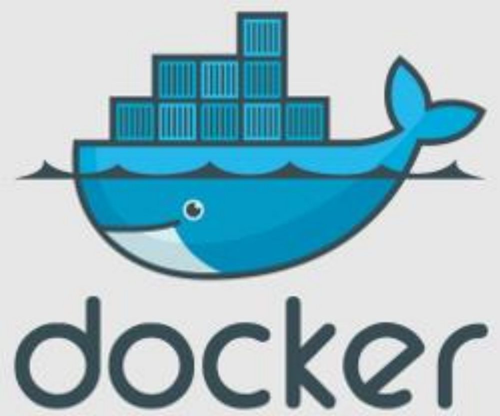
They created open source software to build and manage containers

Docker makes containers easy

Super easy. Fast learning curve

Docker is a container format and a set of tools

Docker CLI, Docker Engine, Docker Swarm, Docker Compose, Docker Machine



CONTAINERS ARE GREAT, RIGHT?

Containerized Apps
on Docker Hub

900K

Image Pulls
on Docker Hub

12B

Source: DockerCon 2017 Keynote

Container Adoption Rate in Production

35%

Source: RightScale 2017 State of the Cloud Report

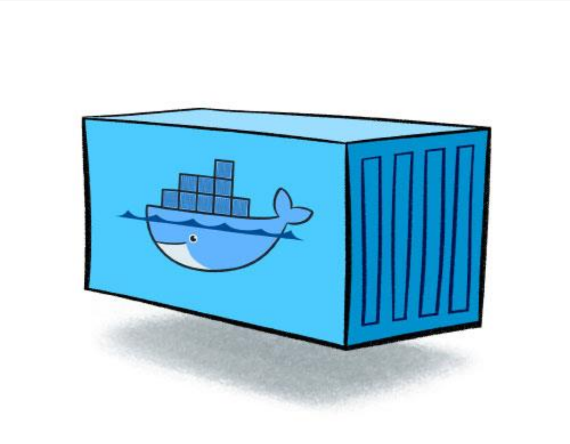
Why the Difference?

IT’S DIFFICULT!!!

CONTAINERS IN DEV

Docker makes it very easy to spin up containers on any OS (Windows, Mac, Linux) for development

It does not, however, help with the availability, scaling, and security requirements of production apps



WE NEED MORE THAN JUST CONTAINERS

Scheduling

Decide where to deploy containers

Lifecycle and health

Keep containers running despite failures

Discovery

Find other containers on the network

Monitoring

Visibility into running containers

Security

Control who can do what

Scaling

Scale containers up and down

Persistence

Survive data beyond container lifecycle

Aggregation

Compose apps from multiple containers

ORCHESTRATION

WHAT IS KUBERNETES?

Background

- "Kubernetes is an open-source system for automating deployment, scaling, and management of containerized applications”
- Schedules and runs application containers across a cluster of machines
- Kubernetes v1.0 released on July 21, 2015. Joe Beda, Brendan Burns, & Craig McLuckie

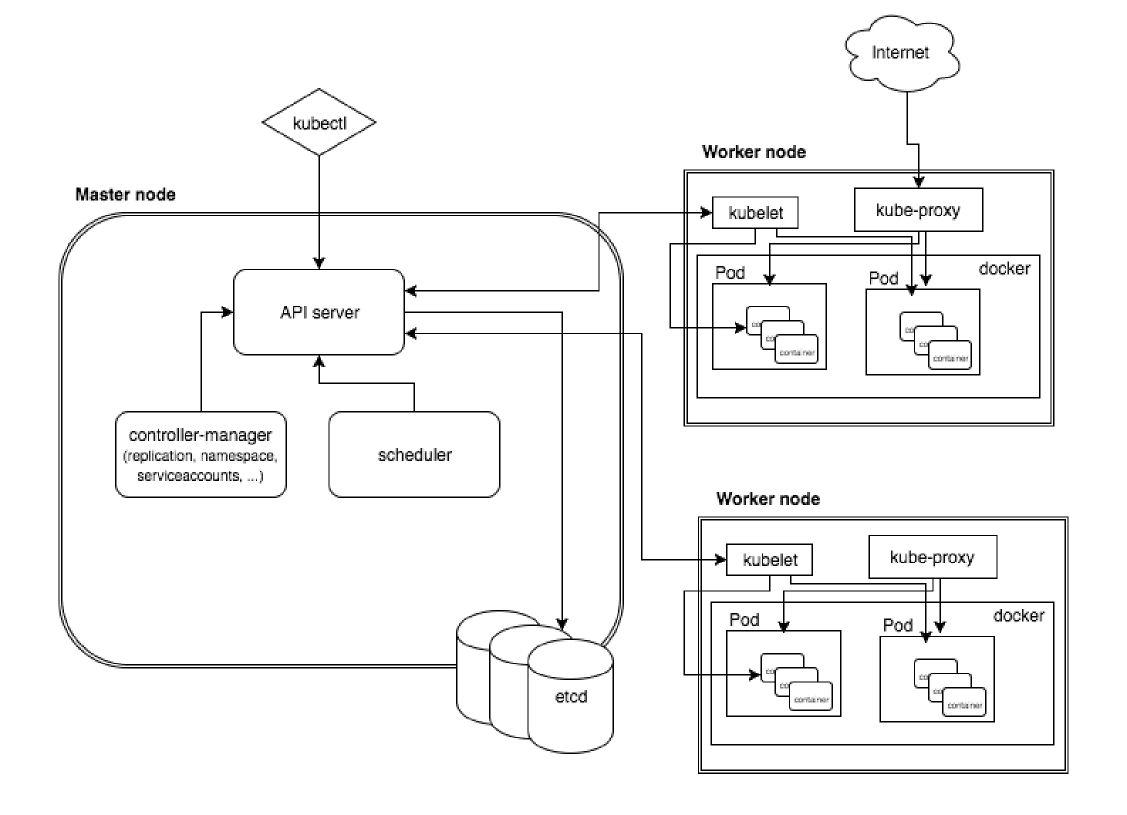
Key features

- Declarative infrastructure
- Self-healing
- Horizontal scaling
- Automated rollouts and rollbacks
- Service discovery and load balancing
- Storage orchestration
- Secret and configuration management

Kubernetes Resources

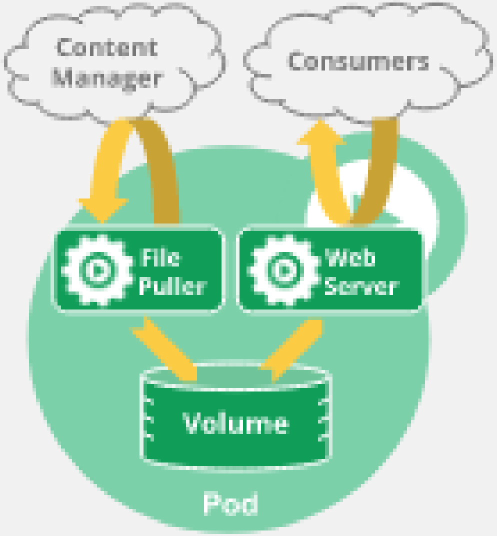
pod	deployment
service	replica set
ingress	daemon set, job
namespace	secret, config-map

Kubernetes Architecture



What is a Pod?

- Pod is the basic building block in kubernetes
- Pods are how containers are delivered
- Can be multiple containers (e.g. - side car)
- Encapsulates container(s), storage, network IP, and options on how to run

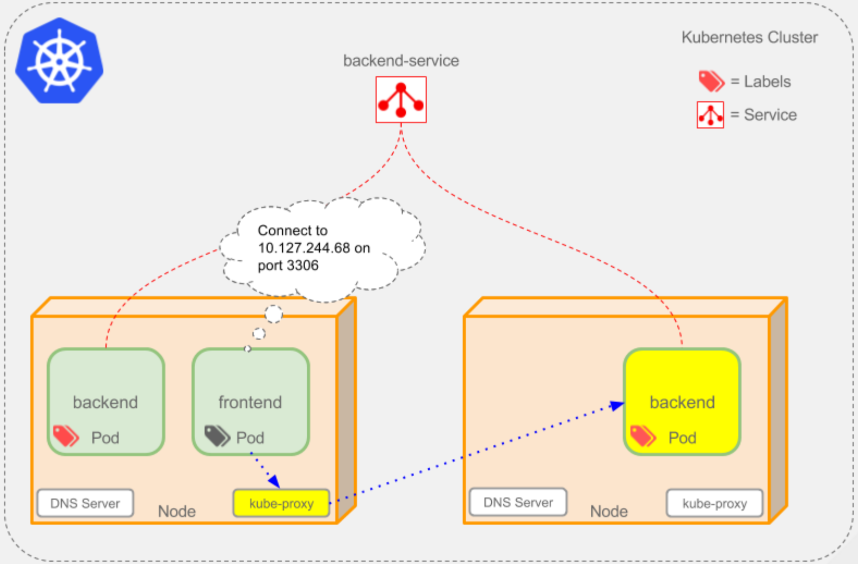


Manifest File: Pod

```yaml
apiVersion: v1
kind: Pod
metadata:
  name: redis-django
  labels:
    app: web
spec:
  containers:
    - name: key-value-store
      image: redis
      ports:
        - containerPort: 6379
    - name: frontend
      image: django
      ports:
        - containerPort: 8000
```

What is a Service?

- Defines a logical set of pods (your microservice)
- Essentially a virtual load balancer in front of pods



Manifest File: Service

```yaml
apiVersion: v1
kind: Service
metadata:
  name: my-service
spec:
  selector:
    app: web
  type: ClusterIP
  ports:
  - protocol: TCP
    port: 80
    targetPort: 9376
```

Service Types

ClusterIP:
Exposes the service on a cluster-internal IP. Choosing this value makes the service only reachable from within the cluster

NodePort:
Exposes the service on each Node’s IP at a static port (the NodePort)
Connect from outside the cluster by requesting <NodeIP>:<NodePort>

LoadBalancer:
Exposes the service externally using a cloud provider’s load balancer

Deployments

Provides declarative updates for Pods and Replica Sets

Deployment describes ”desired state”

Can:

- Create deployment to rollout ReplicaSet
- Declare new state for pods (e.g. – new imageTag)
- Rollback to earlier state
- Scale up/down
- Check rollout history
- Clean-up

Manifest File: Deployment

```
apiVersion: apps/v1beta1
kind: Deployment
metadata:
  name: ocpweb-deploy
spec:
  replicas: 5
  template:
    metadata:
      labels:
        app: ocpweb
    spec:
      containers:
      - name: ocpweb
        image: ded101/ocpweb
        ports:
        - containerPort: 8080
```

Namespaces

Allow for multiple virtual clusters backed by the same physical cluster

Logical separation

Namespace used in FQDN of Kubernetes services

E.g. - <service-name>.<namespace-name>.svc.cluster.local

Every Kubernetes resource type is scoped to a namespace (except for nodes, persistentVolumes, etc.)

Intended for environments with many users, teams, projects

Labels and Selectors

Not related to CSS

Labels are key/value pairs for any API object in Kubernetes

”Label selectors” == queries against labels to match objects

Use cases:

- Associating pods to a service
- Pinning workloads to specific nodes
- Selecting a subset of resources

Labs 1-5

Labs 1-5 Overview

Link to the labs:
http://labs.apps.ocp.cloudvillage.in/#/workshop/ocptigerteam/module/0-setting-up-client-tools

Labs

1. Setting up client tools

2. Create an App from a Docker image

3. Create an App using Docker build

4. Using Web Console

5. Creating an application using JBoss EAP builder image

OPENSHIFT OVERVIEW AND ARCHITECTURE REVIEW

Container application platform based on Docker and Kubernetes for building, distributing and running containers at scale

RED HAT® OPENSHIFT

RED HAT CONTAINER SOLUTIONS

HOST

RED HAT® ENTERPRISE LINUX® ATOMIC HOST

MANAGEMENT

RED HAT® CLOUDFORMS

PLATFORM

RED HAT® OPENSHIFT

STORAGE

RED HAT® STORAGE

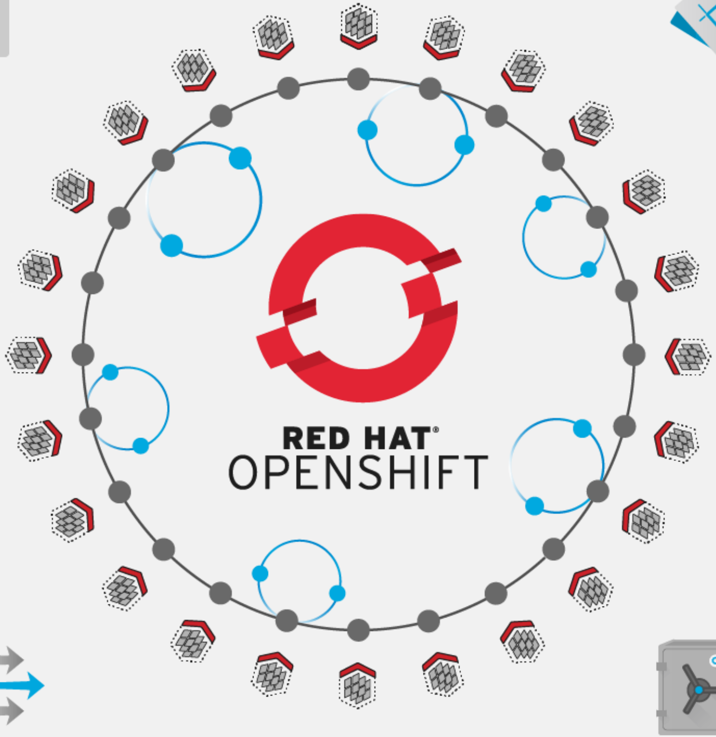
Self-Service

Multi-language

Automation

Collaboration

Multi-tenant



Standards-based

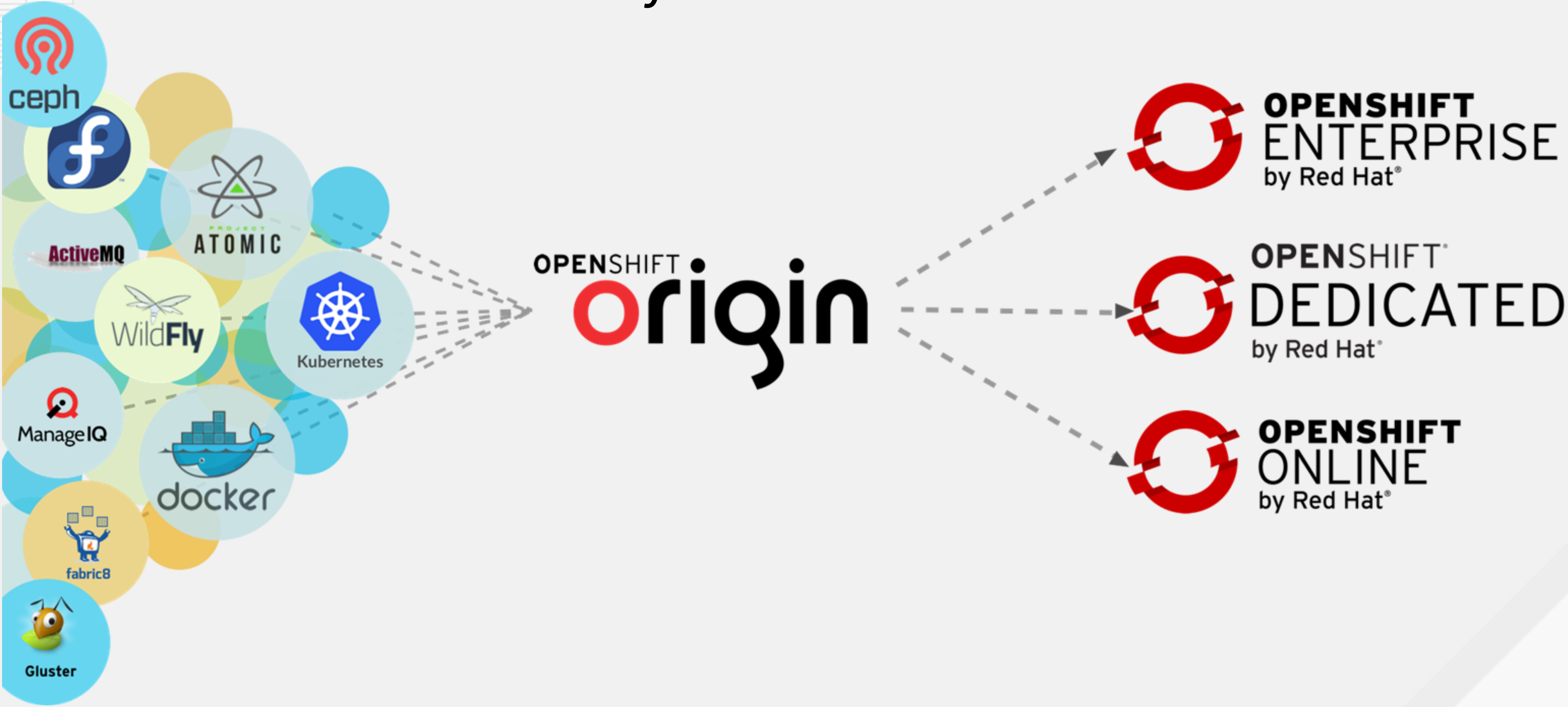
Web-scale

Open Source

Enterprise Grade

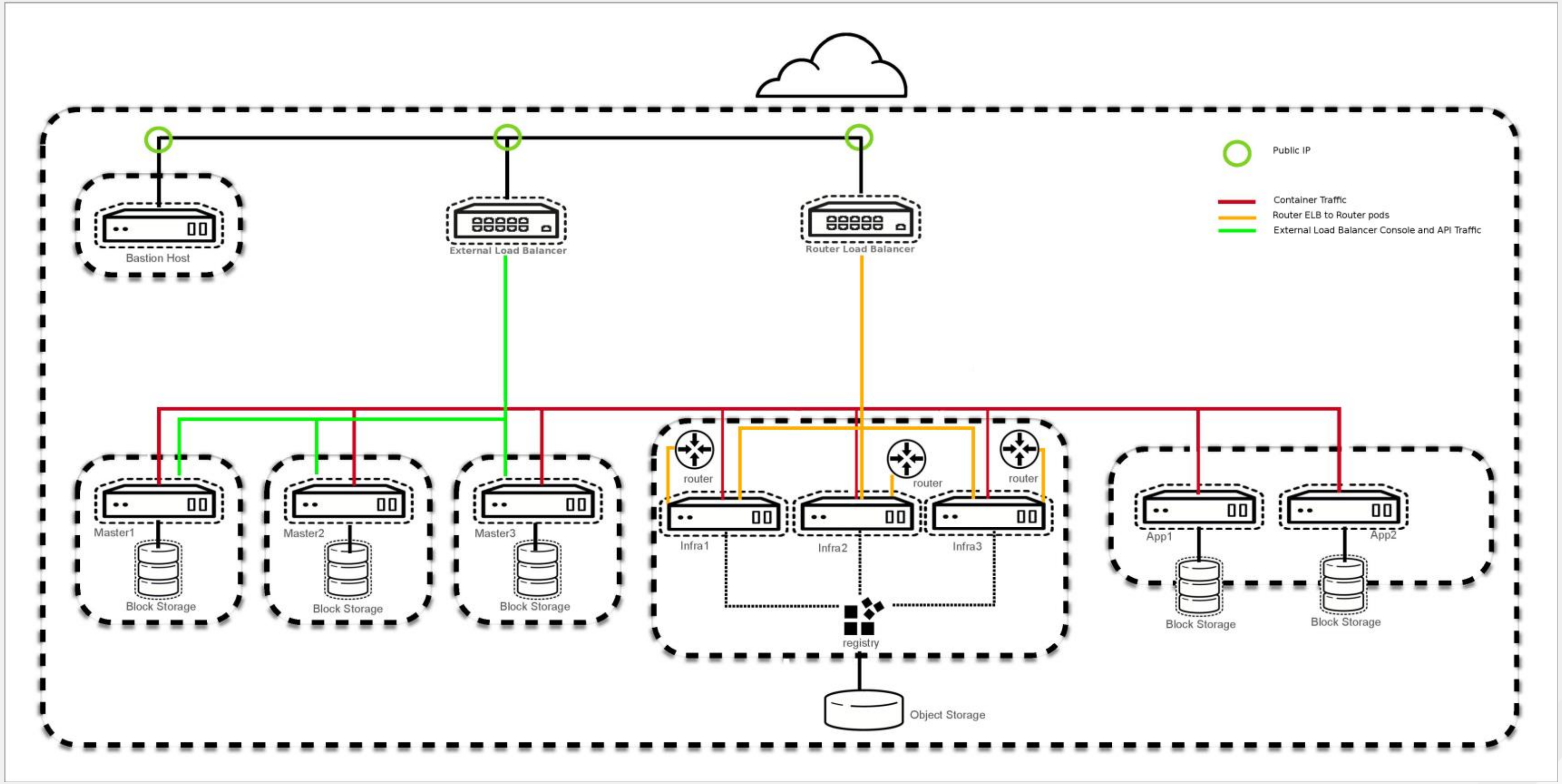
Secure

Community Powered Innovation


ceph
ActiveMQ
PROJECT ATOMIC
WildFly
Kubernetes
ManageIQ
docker
fabric8
Gluster
OPENSHIFT origin
OPENSHIFT ENTERPRISE by Red Hat
OPENSHIFT DEDICATED by Red Hat
OPENSHIFT ONLINE by Red Hat

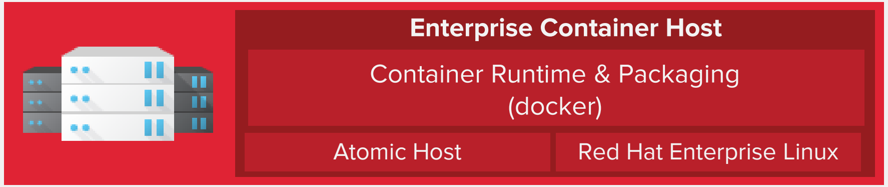
GENERAL DISTRIBUTION

Reference Architecture



OPENSHIFT CONTAINER PLATFORM

Infrastructure Automation & Cockpit



Enterprise Container Host

Container Runtime & Packaging (docker)

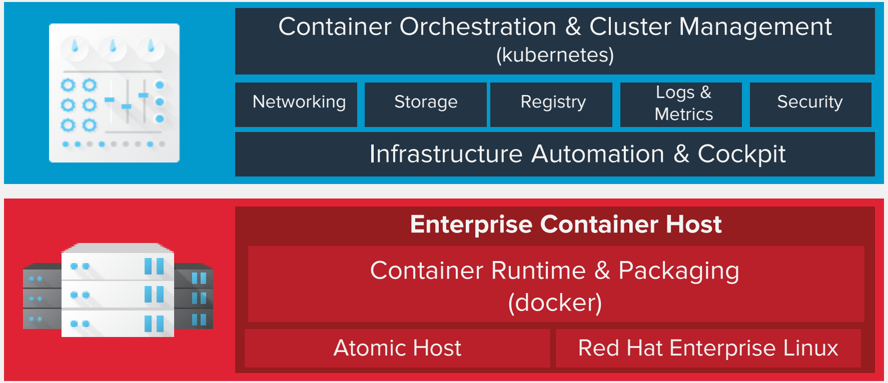
Atomic Host

Red Hat Enterprise Linux

Trusted by Fortune Global 500 companies

OPENSHIFT CONTAINER PLATFORM

OpenShift Application Lifecycle Management



Container Orchestration & Cluster Management
(kubernetes)

Networking | Storage | Registry | Logs & Metrics | Security

Infrastructure Automation & Cockpit

Enterprise Container Host

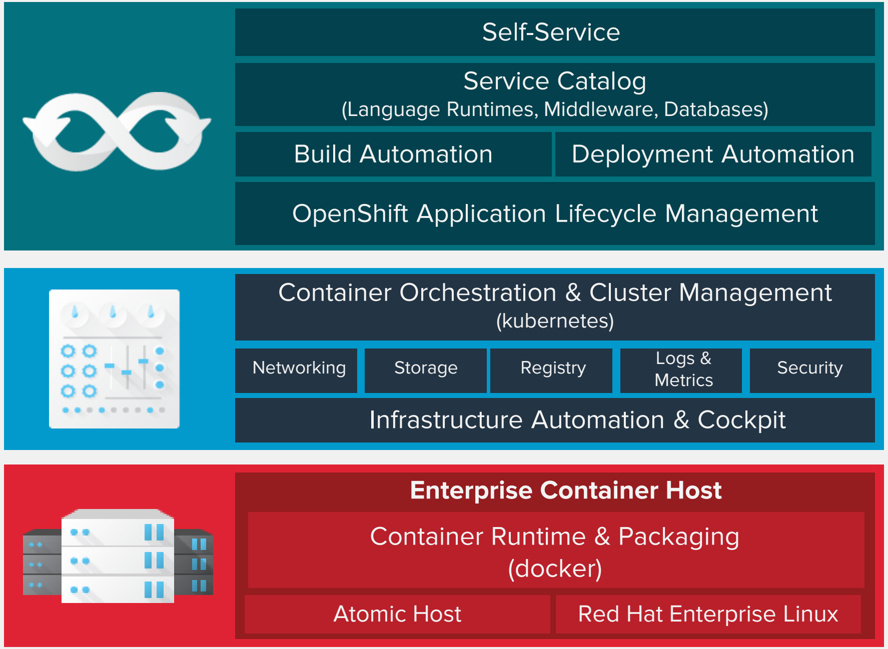
Container Runtime & Packaging
(docker)

Atomic Host | Red Hat Enterprise Linux

Enterprise Kubernetes++ container orchestration

Trusted by Fortune Global 500 companies

OPENSHIFT CONTAINER PLATFORM



Self-Service

Service Catalog
(Language Runtimes, Middleware, Databases)

Build Automation

Deployment Automation

OpenShift Application Lifecycle Management

Container Orchestration & Cluster Management
(kubernetes)

Networking

Storage

Registry

Logs & Metrics

Security

Infrastructure Automation & Cockpit

Enterprise Container Host

Container Runtime & Packaging
(docker)

Atomic Host

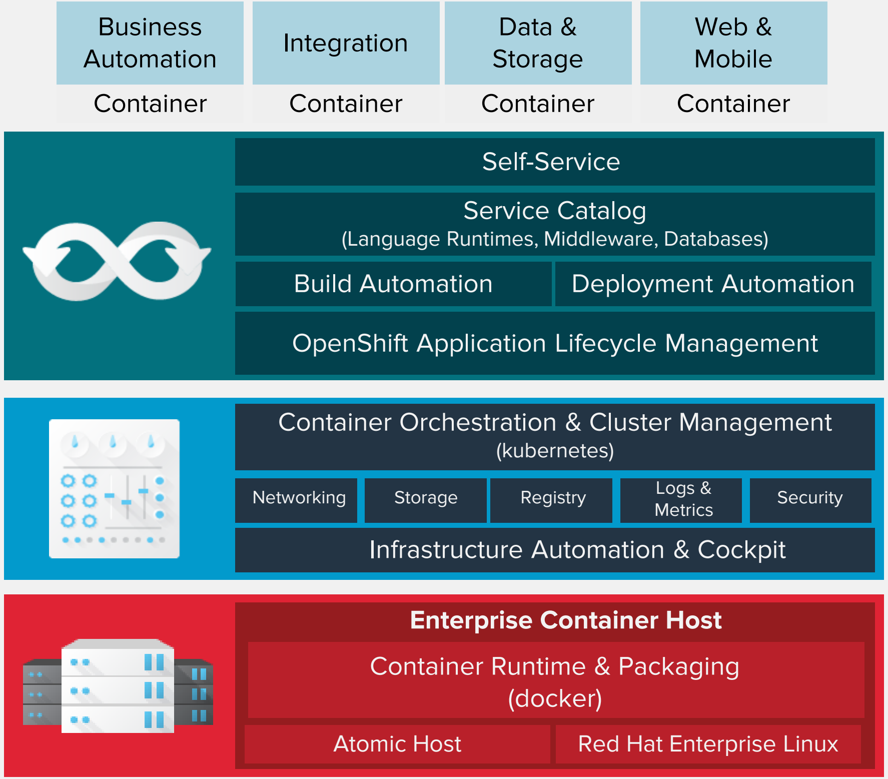
Red Hat Enterprise Linux

Developer Experience

Enterprise Kubernetes++ container orchestration

Trusted by Fortune Global 500 companies

OPENSHIFT CONTAINER PLATFORM



Business Automation
Container

Integration
Container

Data & Storage
Container

Web & Mobile
Container

Self-Service

Service Catalog
(Language Runtimes, Middleware, Databases)

Build Automation | Deployment Automation

OpenShift Application Lifecycle Management

Container Orchestration & Cluster Management
(kubernetes)

Networking | Storage | Registry | Logs & Metrics | Security

Infrastructure Automation & Cockpit

Enterprise Container Host

Container Runtime & Packaging
(docker)

Atomic Host | Red Hat Enterprise Linux

Traditional, stateful, and cloud-native apps

Developer Experience

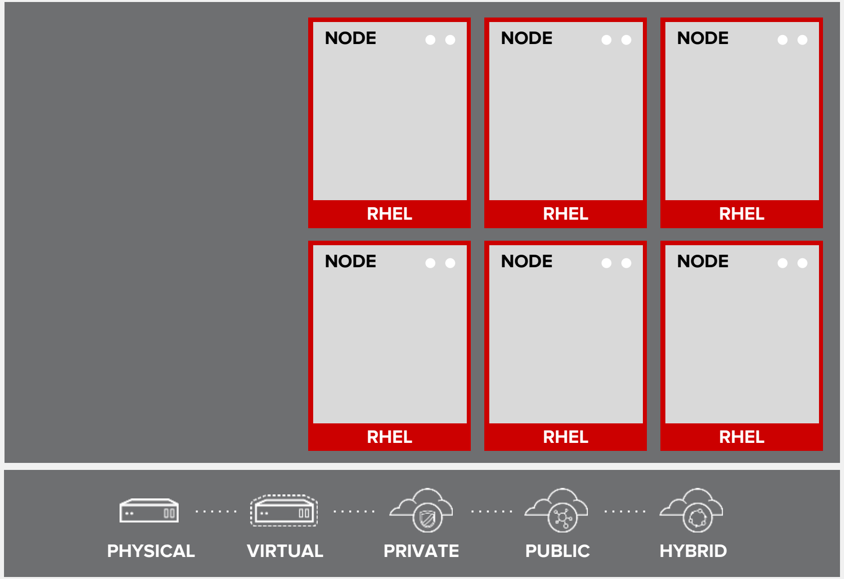
Enterprise Kubernetes++ container orchestration

Trusted by Fortune Global 500 companies

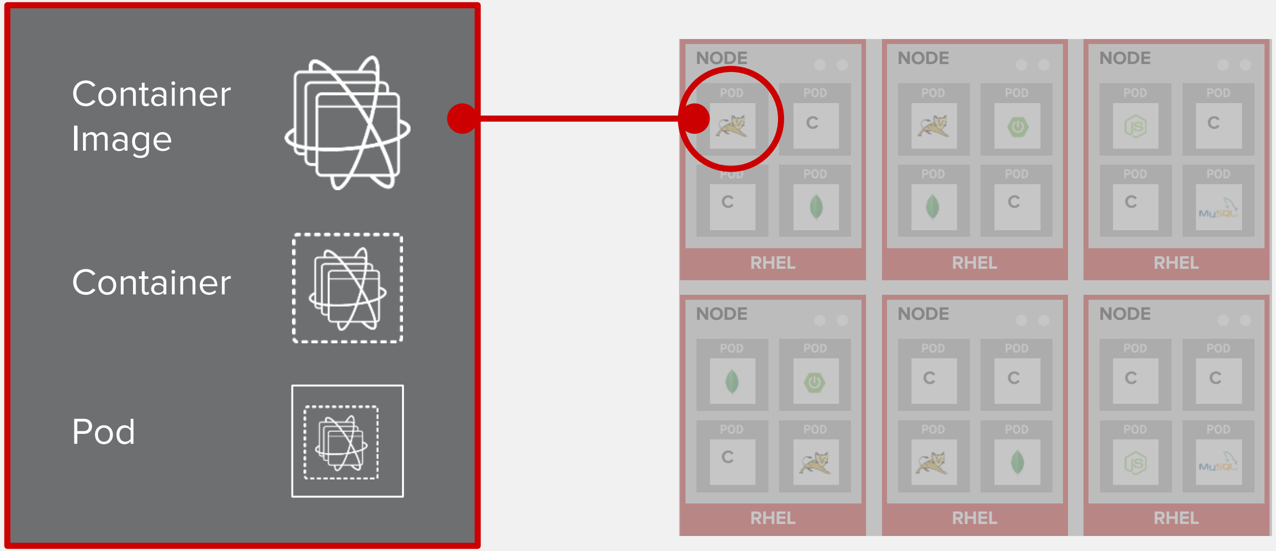
YOUR CHOICE OF INFRASTRUCTURE

PHYSICAL VIRTUAL PRIVATE PUBLIC HYBRID

NODES RHEL INSTANCES WHERE APPS RUN



APPS RUN IN CONTAINERS

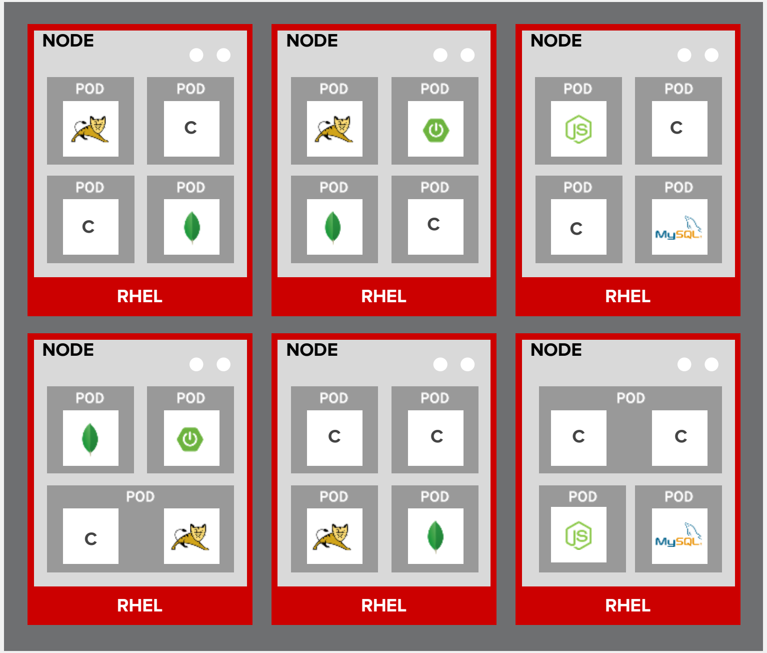


Container Image

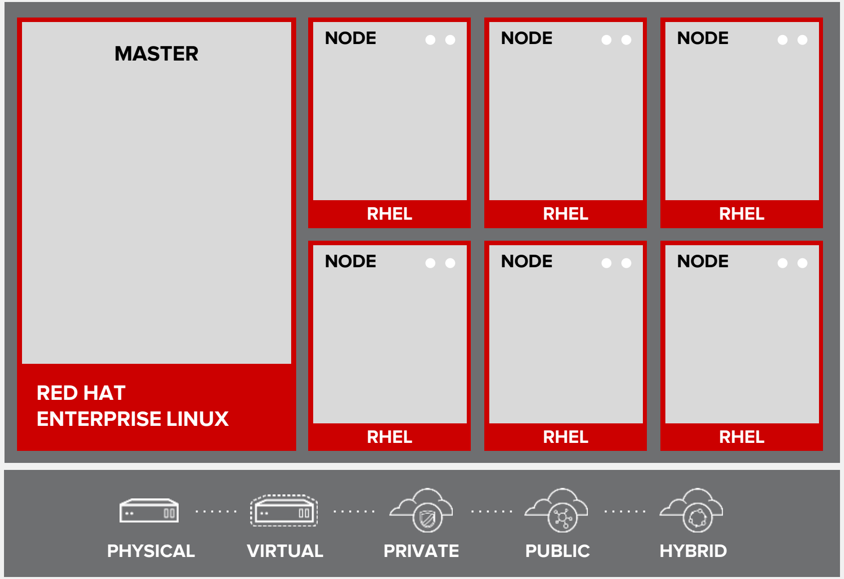
Container

Pod

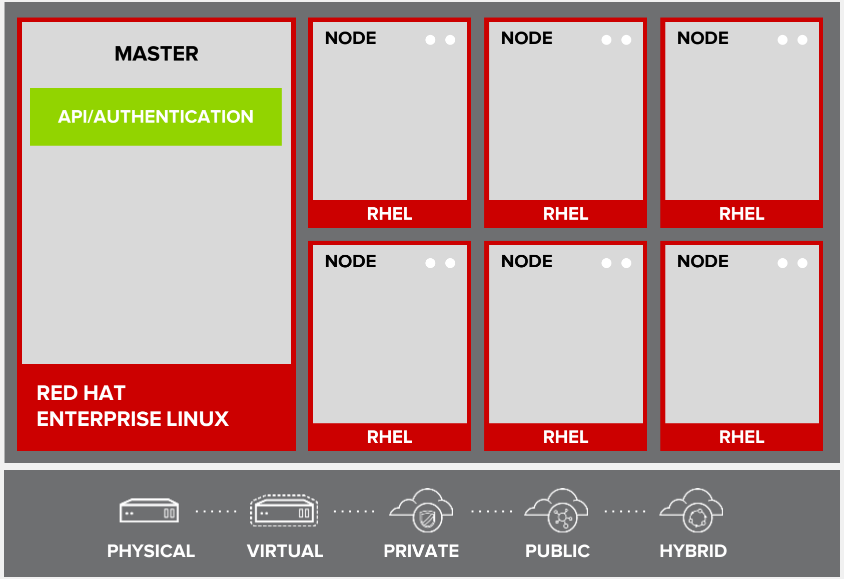
PODS ARE THE UNIT OF ORCHESTRATION



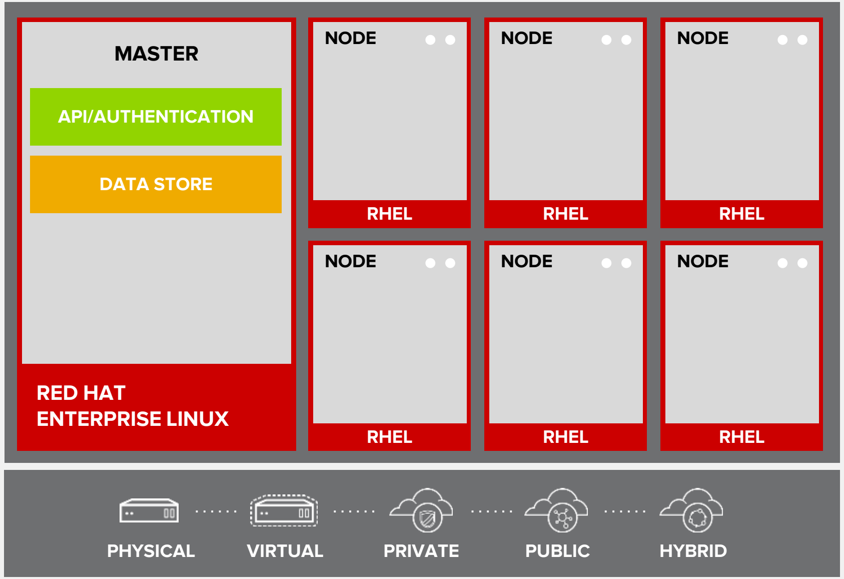
MASTERS ARE THE CONTROL PLANE



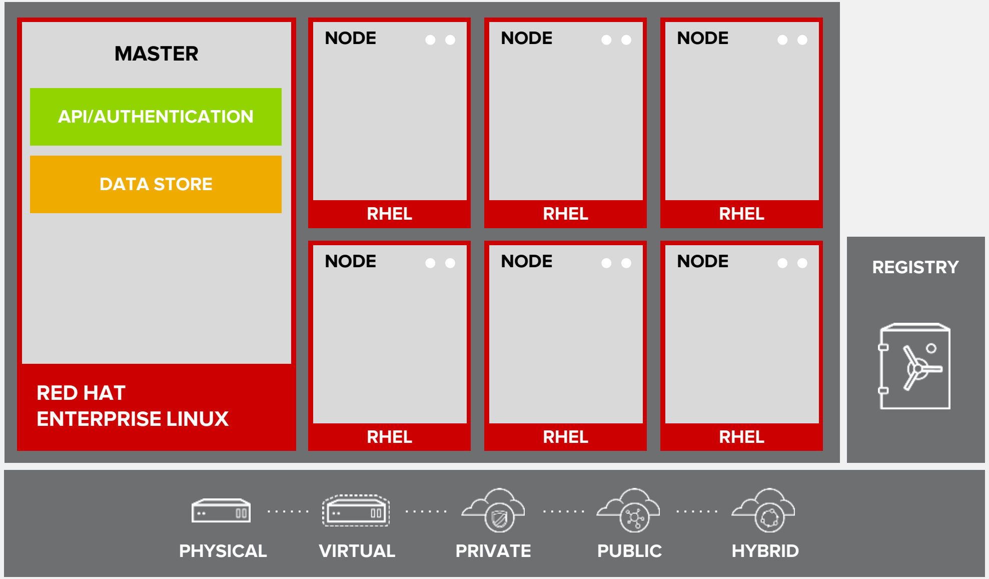
API AND AUTHENTICATION



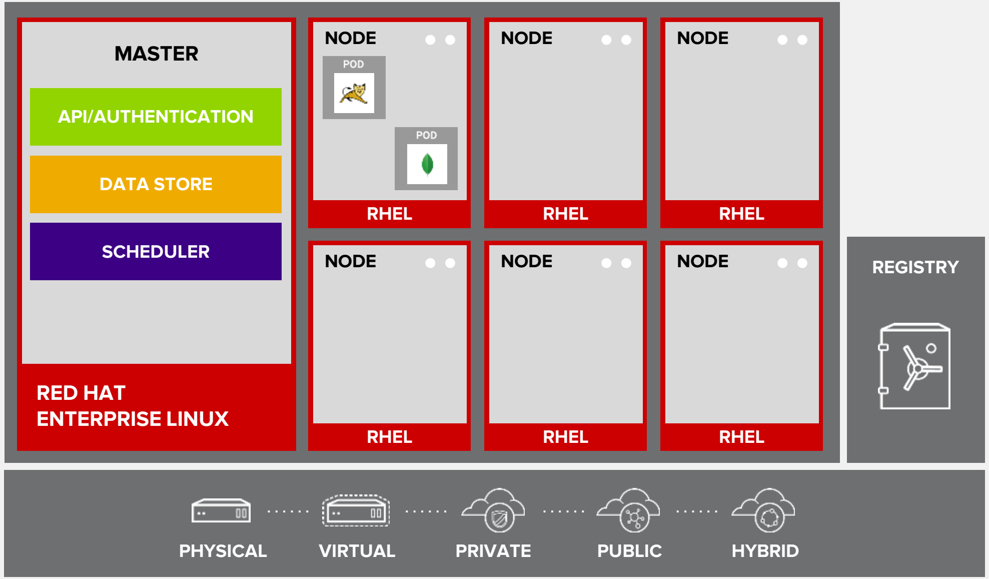
DESIRED AND CURRENT STATE



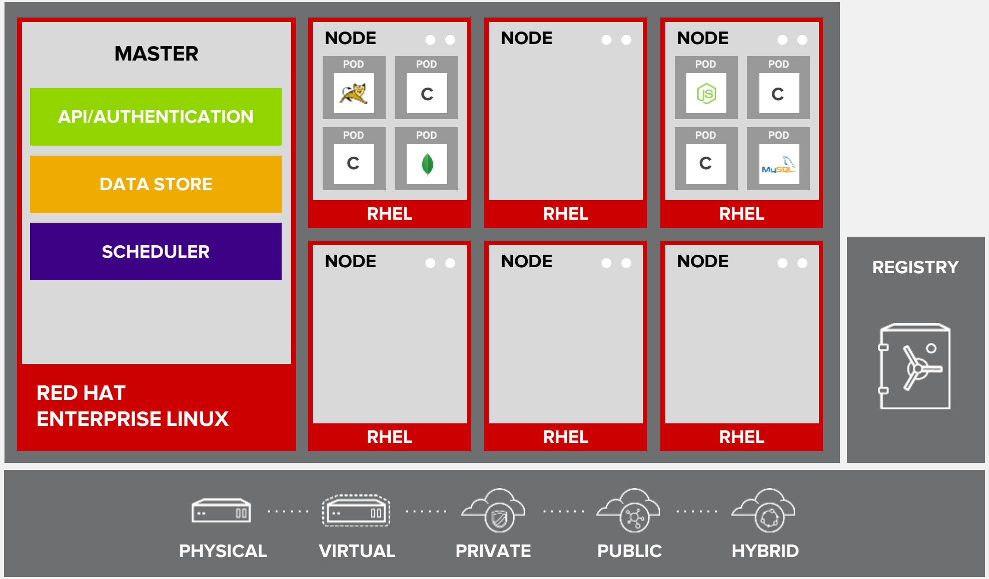
INTEGRATED CONTAINER REGISTRY



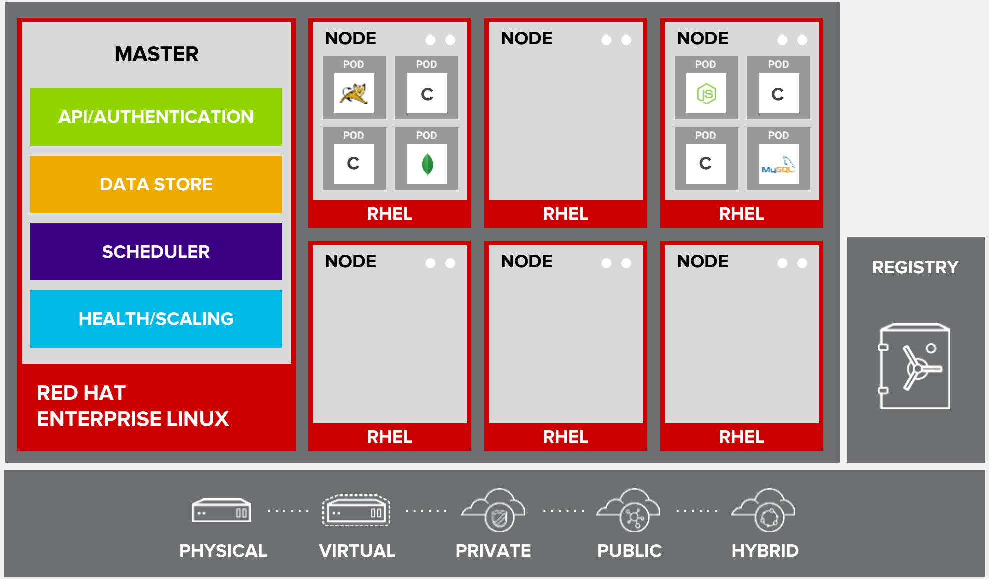
ORCHESTRATION AND SCHEDULING



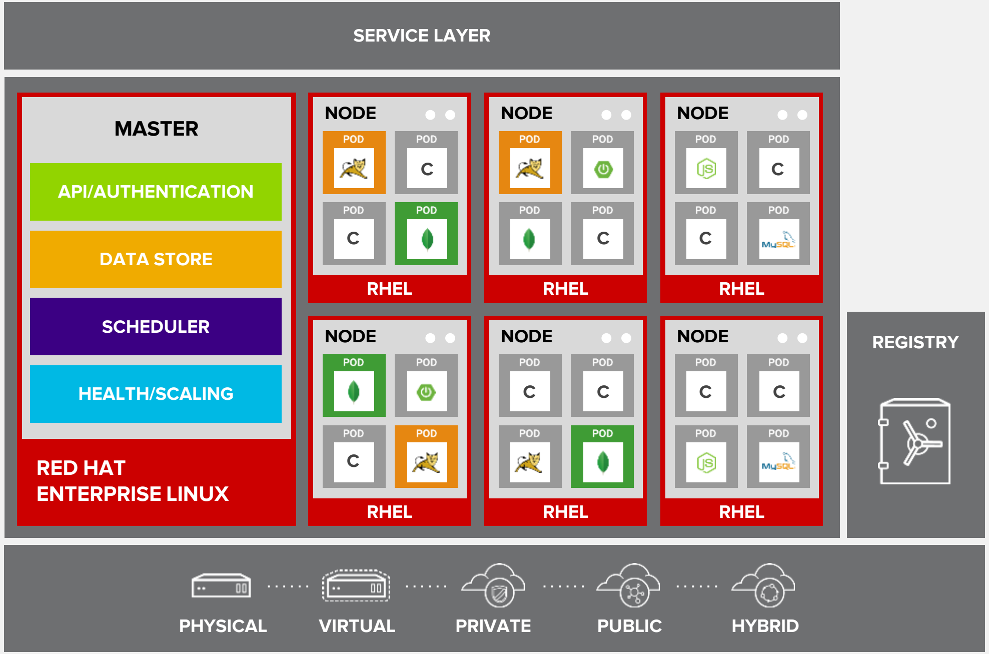
PLACEMENT BY POLICY



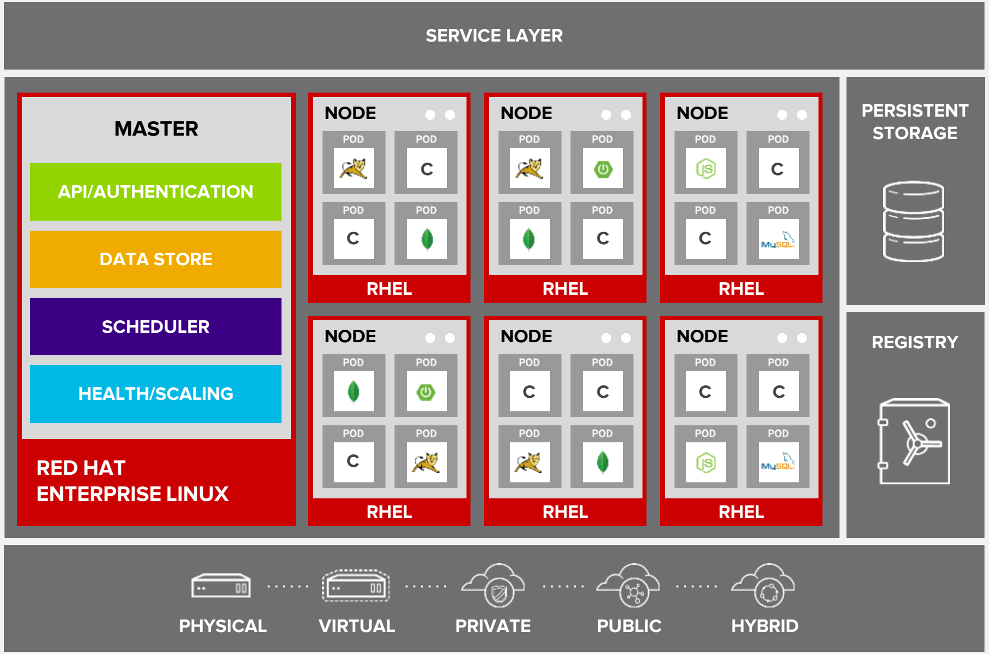
AUTOSCALING PODS



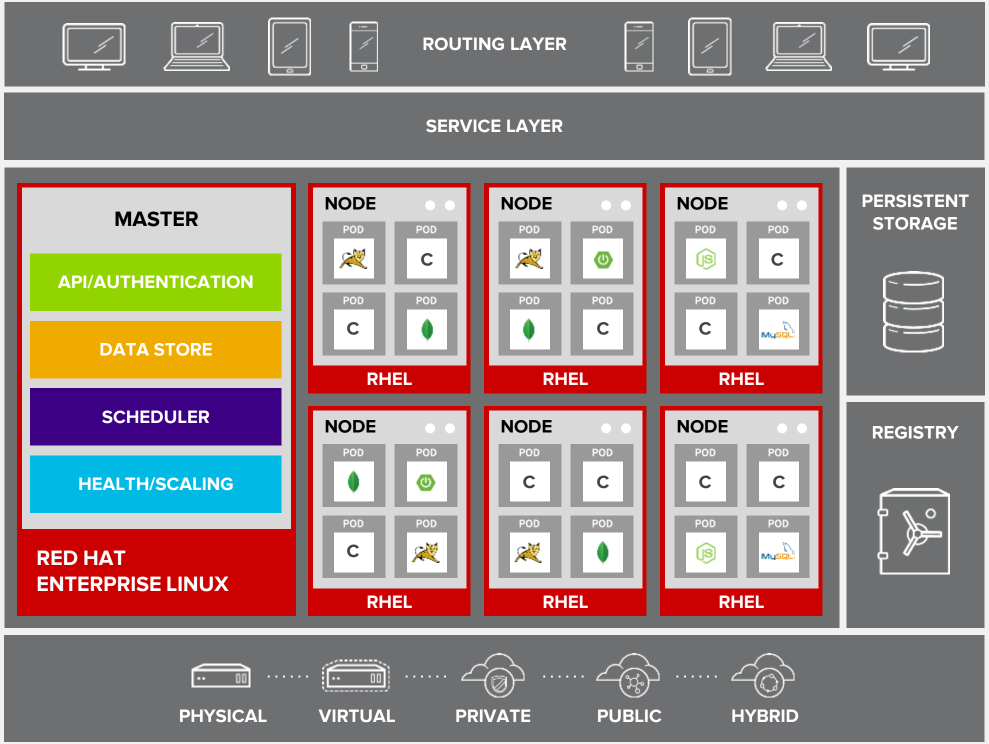
SERVICE DISCOVERY



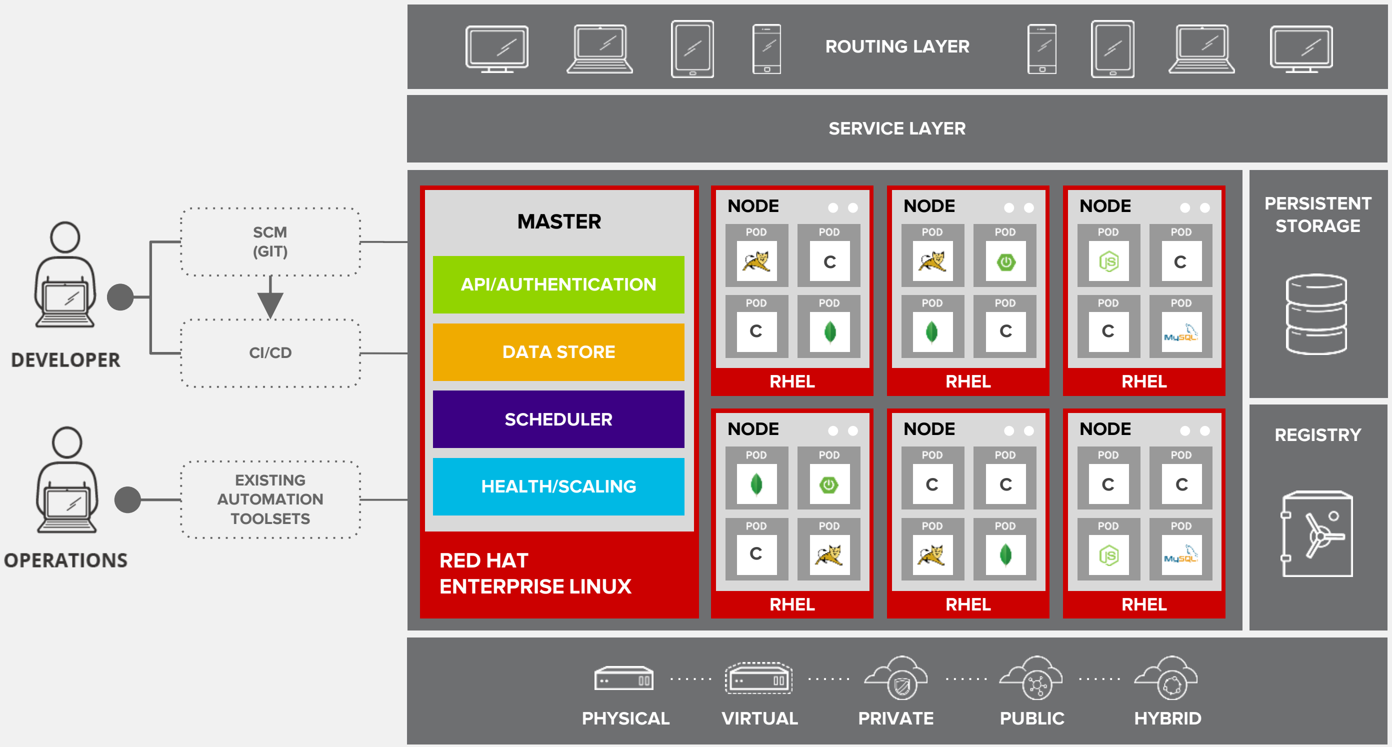
PERSISTENT DATA IN CONTAINERS



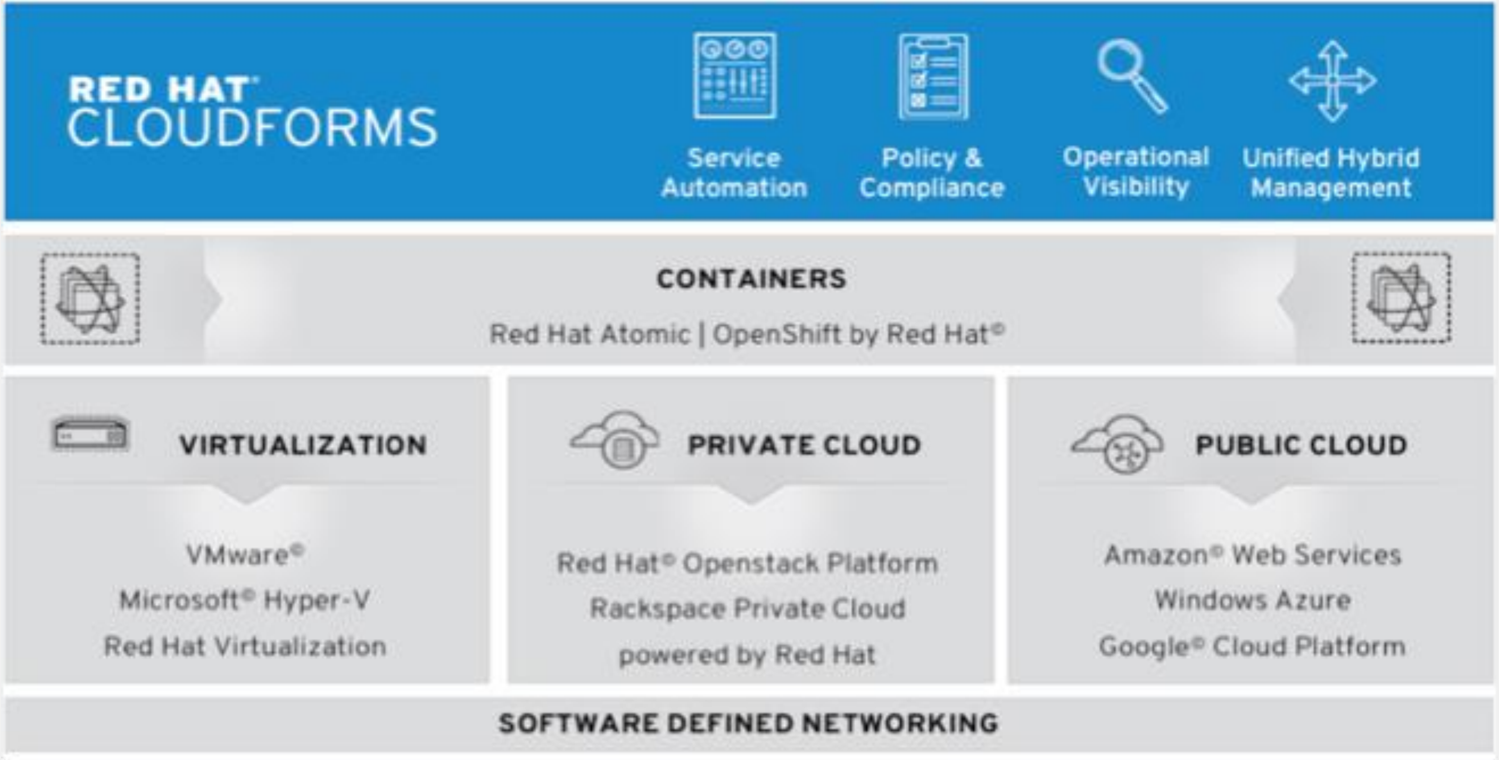
ROUTING AND LOAD-BALANCING



ACCESS VIA WEB, CLI, IDE AND API



OpenShift Management with Cloud Forms



RED HAT® CLOUDFORMS

Service Automation

Policy & Compliance

Operational Visibility

Unified Hybrid Management

CONTAINERS

Red Hat Atomic | OpenShift by Red Hat®

VIRTUALIZATION

VMware®

Microsoft® Hyper-V

Red Hat Virtualization

PRIVATE CLOUD

Red Hat® Openstack Platform

Rackspace Private Cloud powered by Red Hat

PUBLIC CLOUD

Amazon® Web Services

Windows Azure

Google® Cloud Platform

SOFTWARE DEFINED NETWORKING

Cloud Forms

Key Product Features

AGENTLESS

EASY DEPLOYMENT

ANSIBLE AUTOMATION

SIMPLE, POWERFUL, AGENTLESS

MULTI-TENANCY AND RBAC

SEGMENT USER ACCESS,
FINE GRAINED ACCESS CONTROL

CONTINUOUS DISCOVERY

BROWN-FIELD MANAGEMENT,
INTEROPERATES WITH OTHER MGMT

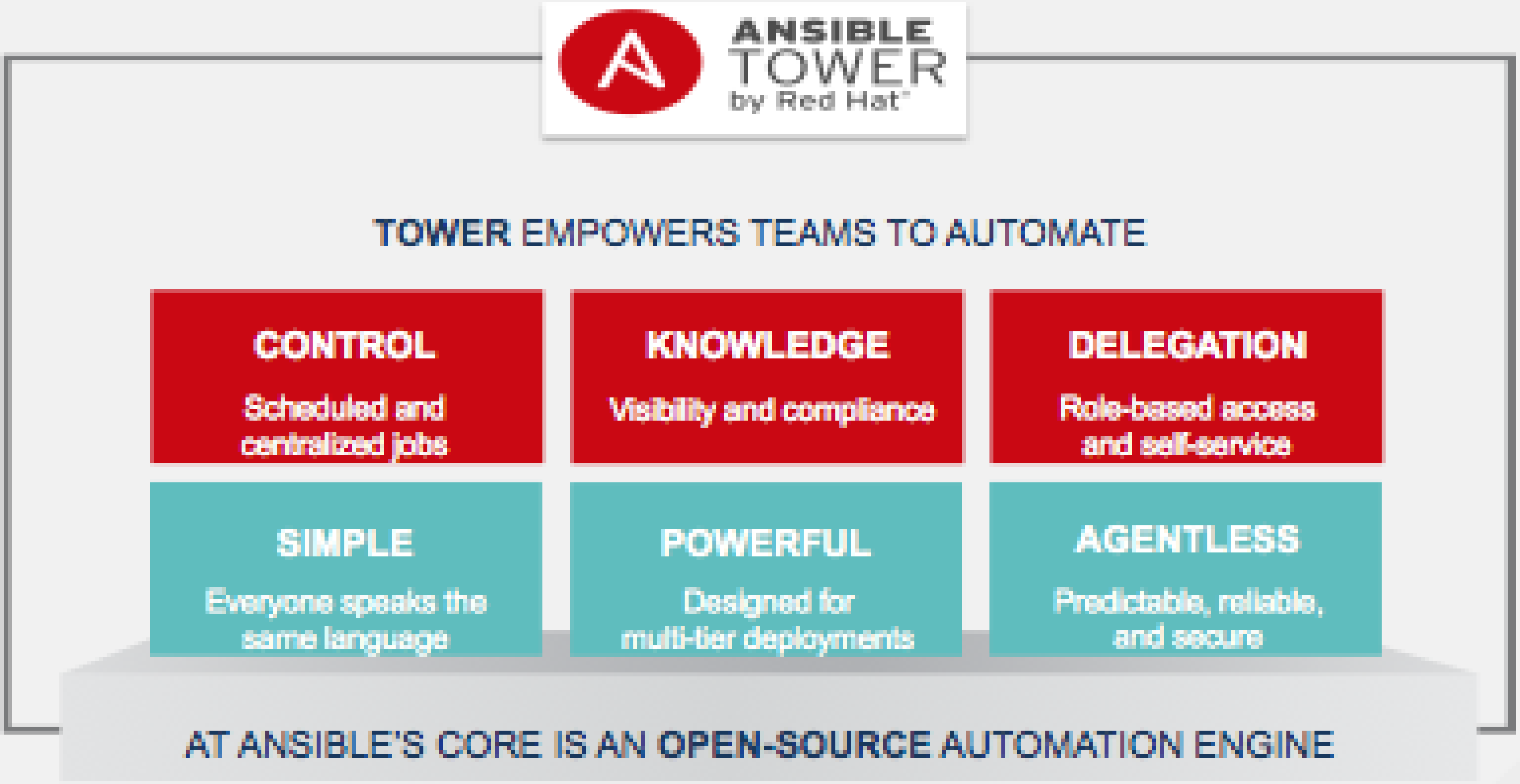
VIRTUAL APPLIANCE

EASY INSTALL, EASY MAINTENANCE

FEDERATED GLOBAL DEPLOYMENTS

HIGHLY SCALABLE, HIGHLY AVAILABLE
MULTI-REGION DEPLOYMENTS

ANSIBLE TOWER



ANSIBLE TOWER by Red Hat

TOWER EMPOWERS TEAMS TO AUTOMATE

CONTROL
Scheduled and centralized jobs

KNOWLEDGE
Visibility and compliance

DELEGATION
Role-based access and self-service

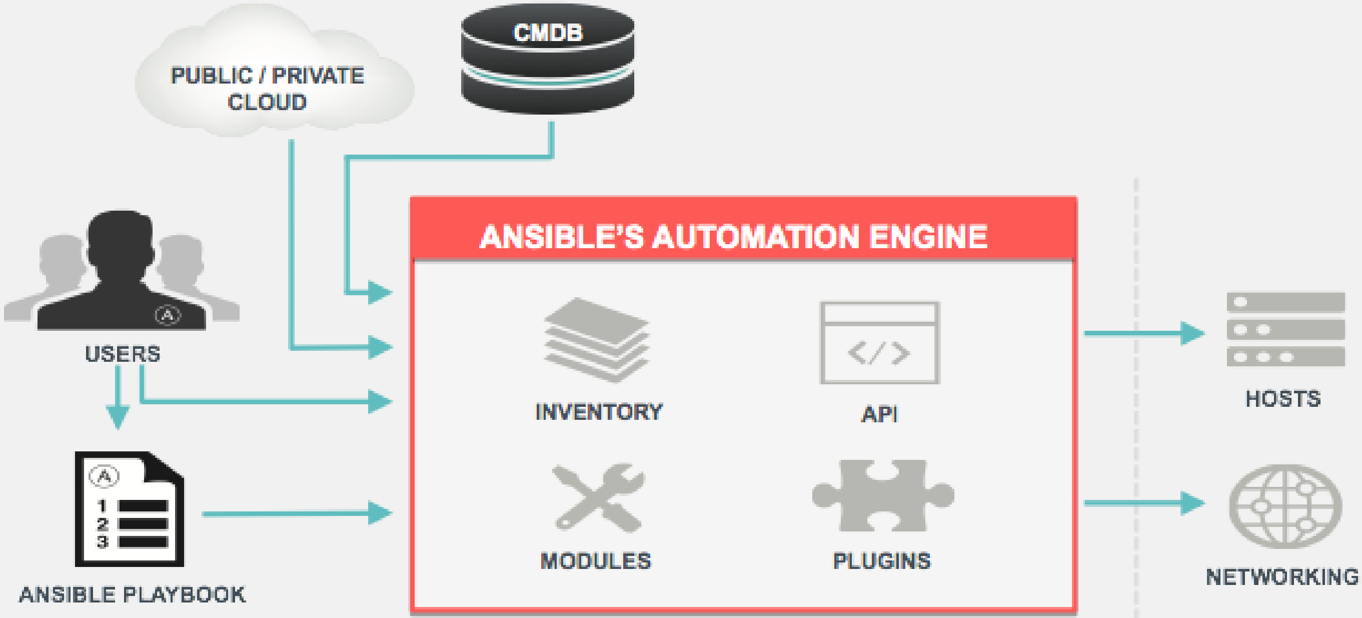
SIMPLE
Everyone speaks the same language

POWERFUL
Designed for multi-tier deployments

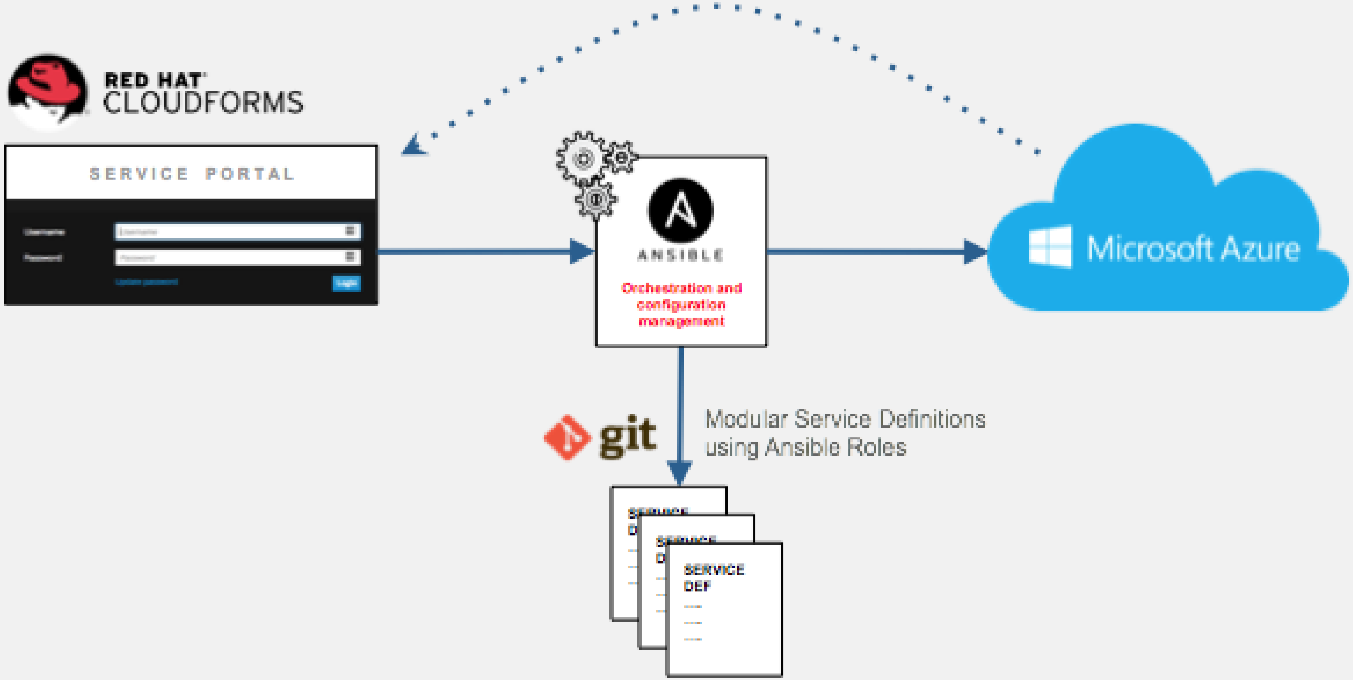
AGENTLESS
Predictable, reliable, and secure

AT ANSIBLE'S CORE IS AN **OPEN-SOURCE** AUTOMATION ENGINE

HOW ANSIBLE WORKS



Red Hat Management on Azure



Let’s take a break!

Labs 6-10

Labs 6-10 Overview

Link to the labs:
http://labs.apps.ocp.cloudvillage.in/#/workshop/ocptigerteam/module/0-setting-up-client-tools

Labs

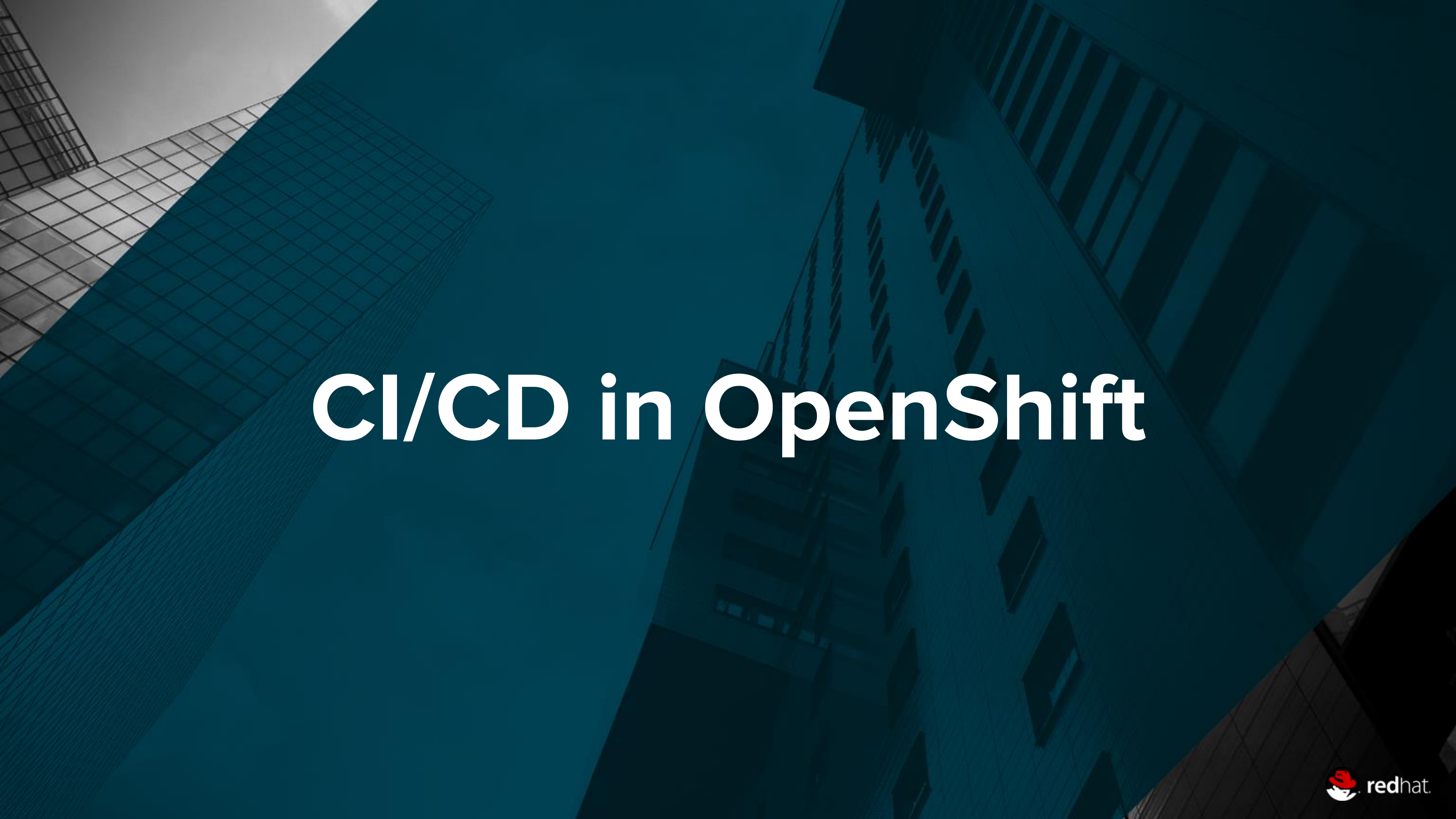
6. Using Templates

7. Scale up and Scale down and Idle the application instances

8. Binary Deployment of a war file

9. Using SSL in your application

10. Blue-Green Deployments



CI/CD in OpenShift

CI/CD in OpenShift

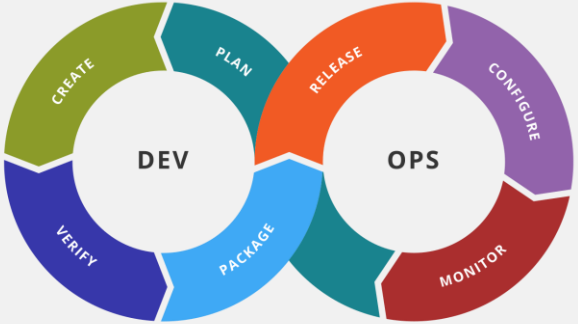
What is CI/CD?

CI – Continuous Integration

- Development Methodology
- Daily integrations of developers code, verified by automated builds

CD – Continuous Deployment

- Continuous, successful, and repeatable methodology to deploying code
- Big key: everything is automated (build, testing, and deployment



CI/CD in OpenShift

Jenkins

- Leading open source automation server

- Helps to set up a continuous integration or continuous delivery environment for almost any combination of languages and source code repositories using pipelines, as well as automating other routine development tasks

- Built into OpenShift and integrated into “pipelines”

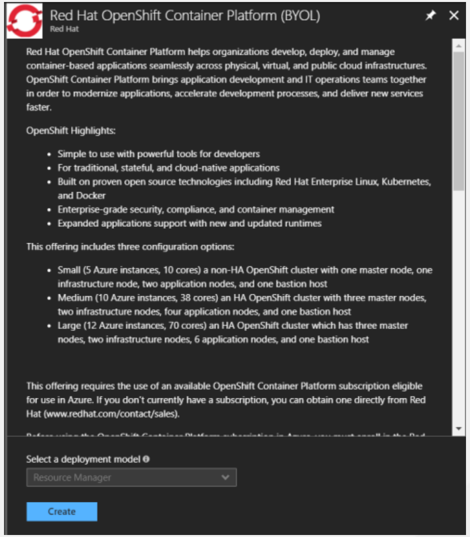
OPENSHIFT ON AZURE QUICK START

Getting Started on Azure

Red Hat OpenShift Container Platform (BYOL)

This offering includes three configuration options:

- Small (5 Azure instances, 10 cores) a non-HA OpenShift cluster with one master node, one infrastructure node, two application nodes, and one bastion host
- Medium (10 Azure instances, 38 cores) an HA OpenShift cluster with three master nodes, two infrastructure nodes, four application nodes, and one bastion host
- Large (12 Azure instances, 70 cores) an HA OpenShift cluster which has three master nodes, two infrastructure nodes, 6 application nodes, and one bastion host



Red Hat OpenShift Container Platform (BYOL)

Red Hat

Red Hat OpenShift Container Platform helps organizations develop, deploy, and manage container-based applications seamlessly across physical, virtual, and public cloud infrastructures. OpenShift Container Platform brings application development and IT operations teams together in order to modernize applications, accelerate development processes, and deliver new services faster.

OpenShift Highlights:

- Simple to use with powerful tools for developers
- For traditional, stateful, and cloud-native applications
- Built on proven open source technologies including Red Hat Enterprise Linux, Kubernetes, and Docker
- Enterprise-grade security, compliance, and container management
- Expanded applications support with new and updated runtimes

This offering includes three configuration options:

- Small (5 Azure instances, 10 cores) a non-HA OpenShift cluster with one master node, one infrastructure node, two application nodes, and one bastion host
- Medium (10 Azure instances, 38 cores) an HA OpenShift cluster with three master nodes, two infrastructure nodes, four application nodes, and one bastion host
- Large (12 Azure instances, 70 cores) an HA OpenShift cluster which has three master nodes, two infrastructure nodes, 6 application nodes, and one bastion host

This offering requires the use of an available OpenShift Container Platform subscription eligible for use in Azure. If you don't currently have a subscription, you can obtain one directly from Red Hat (www.redhat.com/contact/sales).

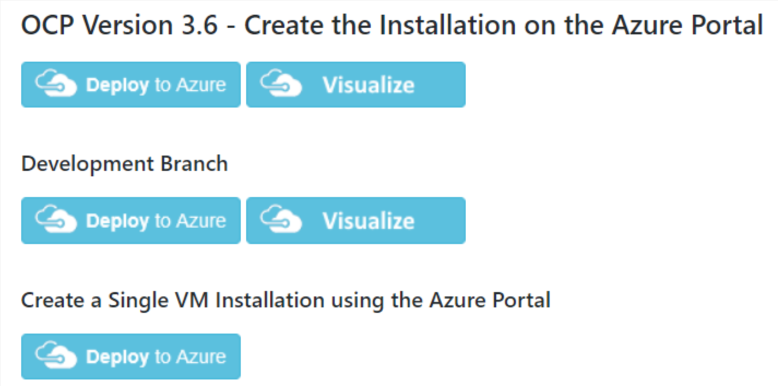
Select a deployment model: Resource Manager

Getting Started on Azure

ARM Templates

https://github.com/openshift/openshift-ansible-contrib/tree/master/reference-architecture/azure-ansible

- Cant deploy full architecture or single instance VM
- Single click deploy or customize templates to fit your need



OCP Version 3.6 - Create the Installation on the Azure Portal

Development Branch

Create a Single VM Installation using the Azure Portal

Labs 11-15

Labs 11-15 Overview

Link to the labs: http://labs.apps.ocp.cloudvillage.in/#/workshop/ocptigerteam/module/0-setting-up-client-tools

Labs

11. SCM Web Hooks

12. Rollback Applications

13. Code Promotion across Environments

14. Installing the JBoss Developer Studio

15. Deploy a SpringBoot Application

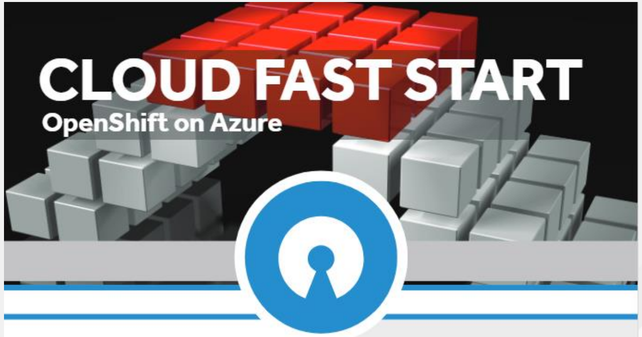
NEXT STEPS

Getting Started

Cardinal OpenShift Fast Start

Cardinal Solutions and Red Hat will guide you through a short-term, structured engagement that will:

- Educate you on OpenShift's capabilities and the value of using containers for development
- Help determine suitable use cases based on your current applications
- Deploy a working OpenShift cluster hosting your chosen workloads



redhat.

THANK YOU

plus.google.com/+RedHat

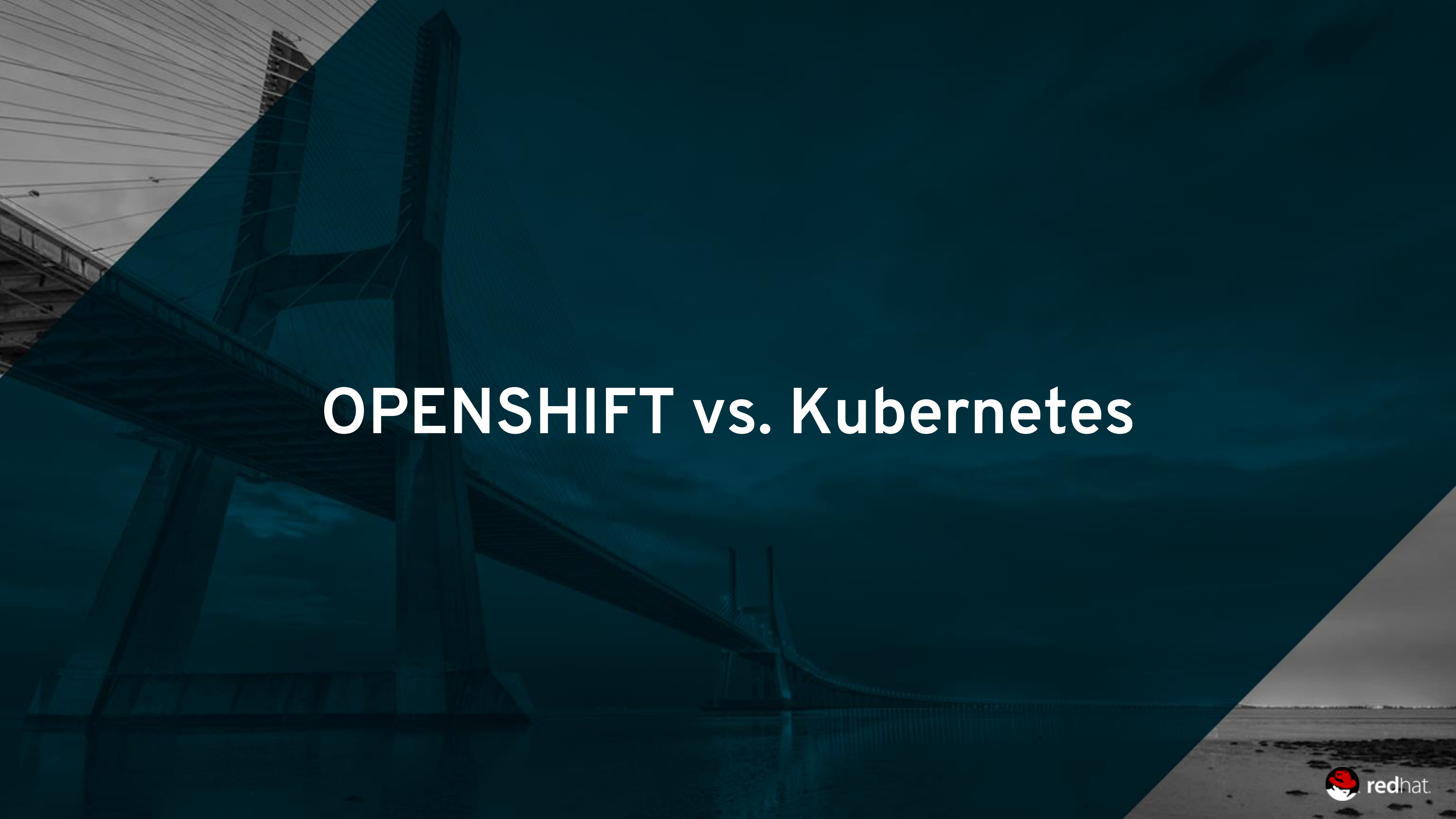
facebook.com/redhatinc

linkedin.com/company/red-hat

twitter.com/RedHatNews

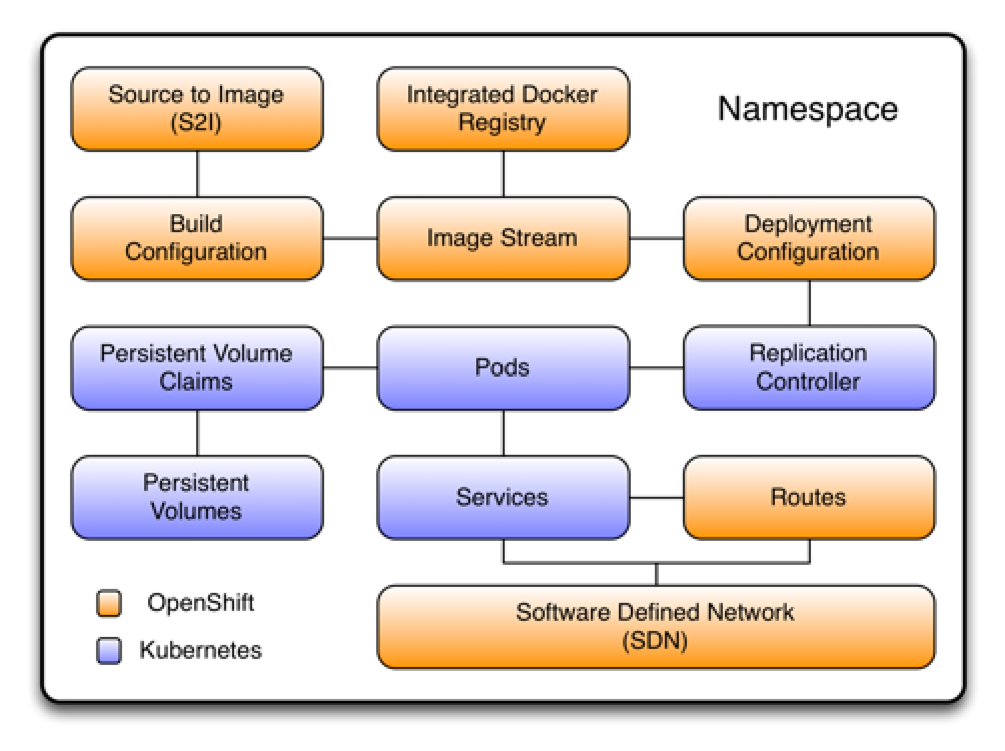
youtube.com/user/RedHatVideos

APPENDIX



OPENSHIFT vs. Kubernetes

Component Comparison



Source to Image (S2I)

Integrated Docker Registry

Namespace

Build Configuration

Image Stream

Deployment Configuration

Persistent Volume Claims

Pods

Replication Controller

Persistent Volumes

Services

Routes

Software Defined Network (SDN)

🟧 OpenShift

🟦 Kubernetes

Routers

A route exposes a service at a host name, like www.example.com, so that external clients can reach it by name.

Default Router in Openshift is an actual HAProxy container providing reverse proxy capabilities:

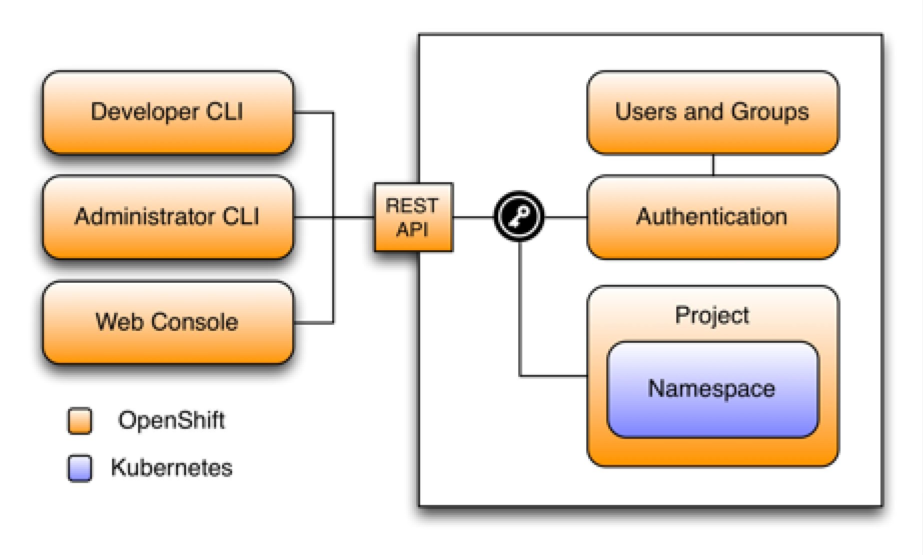
- SSL termination
- Sticky sessions
- Whitelist/blacklist

Projects

Effectively wraps a namespace, with access to the namespace being controlled via the project.

Access is controlled through an authentication and authorization model based on users and groups.

Provide the walls between namespaces, ensuring that users, or applications, can only see and access what they are allowed to.



Builds

Build system provides extensible support for build strategies that are based on selectable types specified in the build API

Docker build

- Invokes the docker build command using a repository with a Dockerfile and all required artifacts in it to produce a runnable image

Source-to-Image (S2I) build

- Framework that makes it easy to write images that take application source code as an input and produce a new image that runs the assembled application as output.

Custom build

OPENSHIFT LAB

WHAT NEXT?

Builds

Build system provides extensible support for build strategies that are based on selectable types specified in the build API

Docker build

- Invokes the docker build command using a repository with a Dockerfile and all required artifacts in it to produce a runnable image

Source-to-Image (S2I) build

- Framework that makes it easy to write images that take application source code as an input and produce a new image that runs the assembled application as output.

Custom build