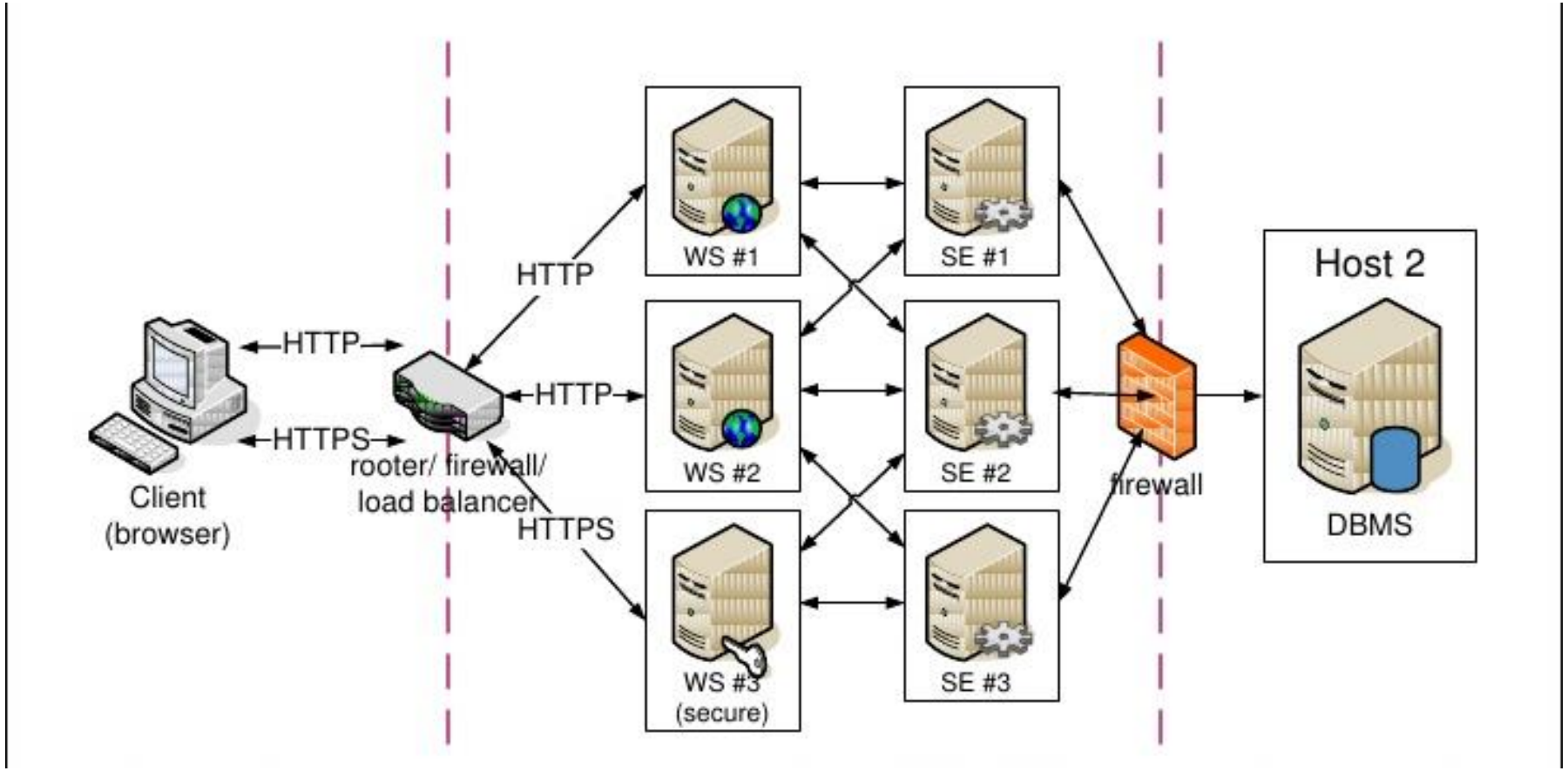
IBM MICROSERVICE I
SESSION 1

TEXAS SKILLS DEVELOPMENT FUND
TRAINING PARTNERSHIP

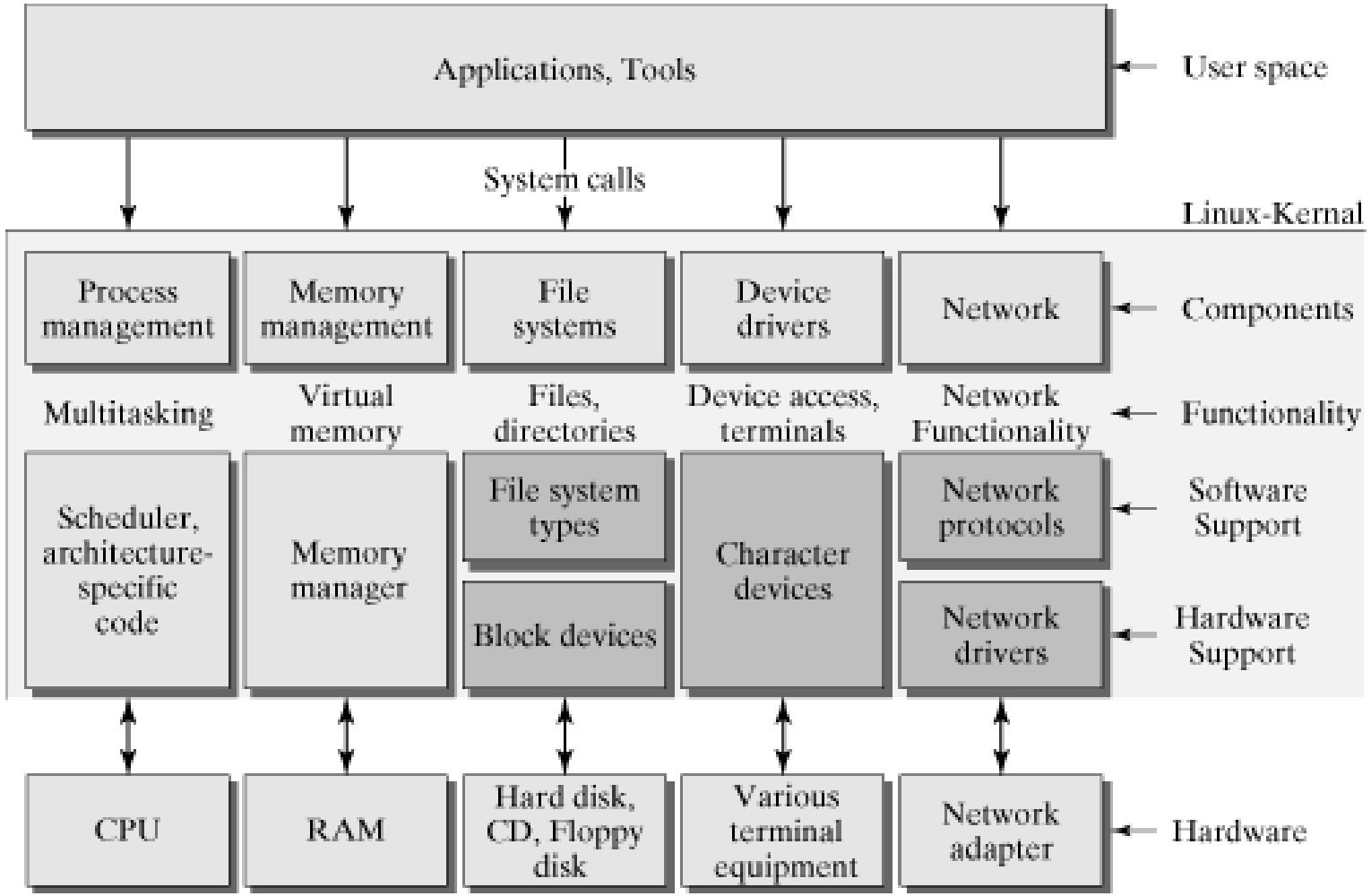
Course Overview

- The evolution of containers
  - VM, Containers, Docker, Kubernetes, OpenShift
- Container Security
- Hands-on Lab
- Istio & KNative

N-Tier Architecture



Linux (OS) Architecture



Containers vs. Virtual Machines

- VM
  - Abstraction –OS, memory, storage
  - Image – OS or Entire App Fabric (incl. target app)
  - Completely isolated host & VMs
- Require more resources to scale
- Containers vs. Virtual Machines
  - What's the Difference?

- Container
  - Occupy part of existing m/c
  - Share Kernel & lib files
  - Self contained images
  - Less Resources
  - Faster to scale
- Too many microservices can result in bloat

Containers vs VMs: the eternal debate | Mirantis

Containers Transform

Monolith

N-Tier

Microservices

Applications

Datacenter

Hosted

Hybrid

Infrastructures

Waterfall

Agile

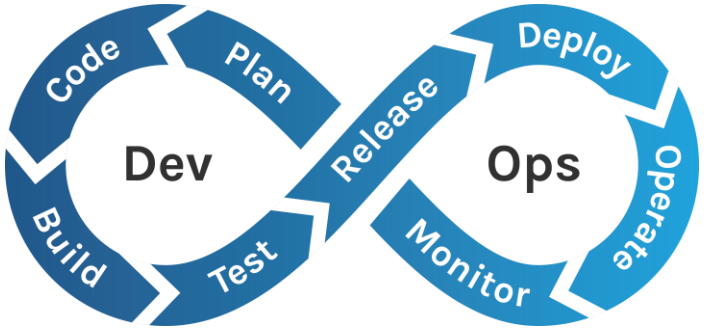
DevOps

Processes

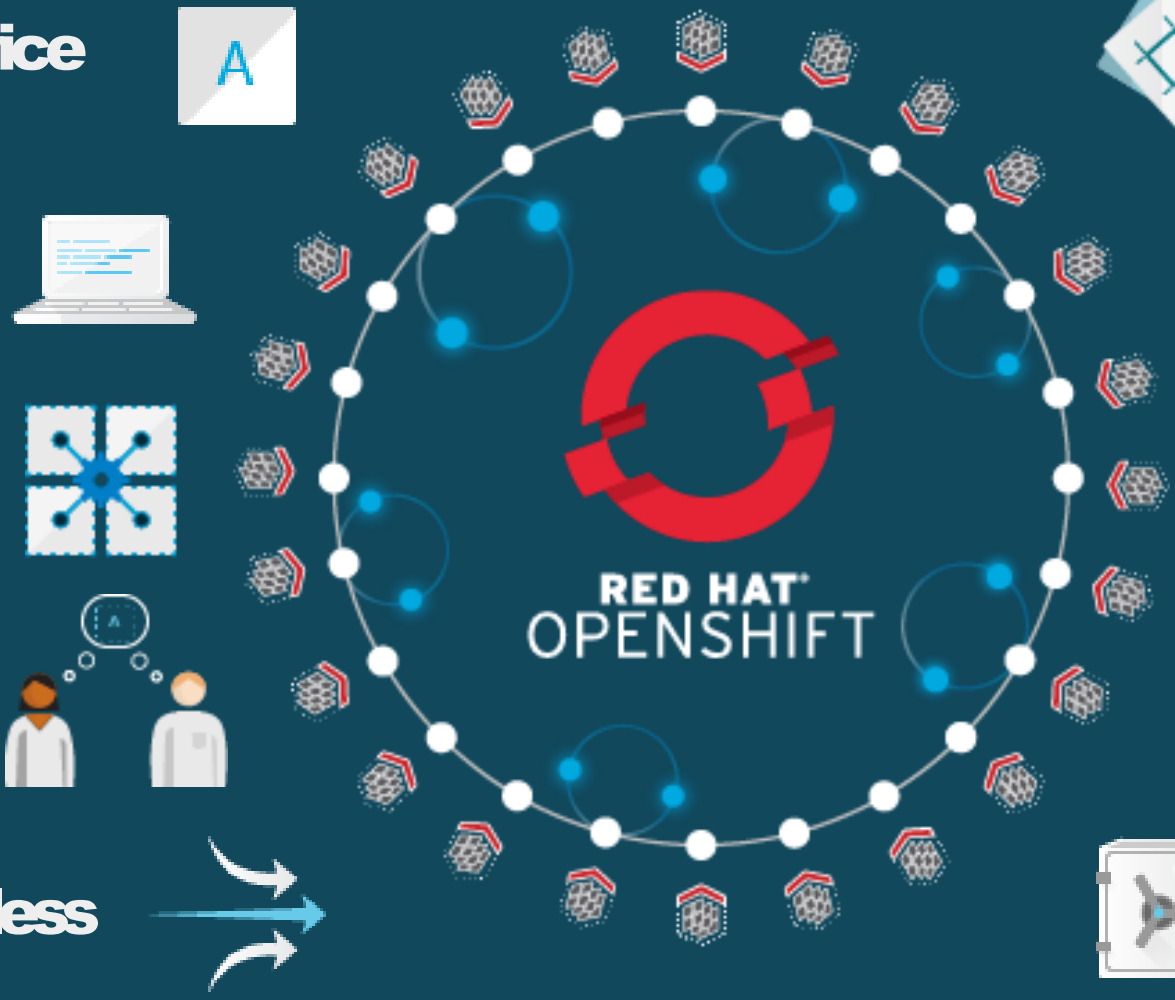
redhat

DevOps

- Developers
  - Package application and all of its dependencies
  - Deploy to any environment in seconds and enable CI/CD
  - Easily access and share containerized components
- Operations
  - Sandboxed application processes on a shared Linux OS kernel
  - Simpler, lighter, and denser than virtual machines
  - Portable across different environments



Critical features for both Dev and Ops



Self-Service

Multi-language

Automation

Collaboration

Seamless

Standards-based

Web-scale

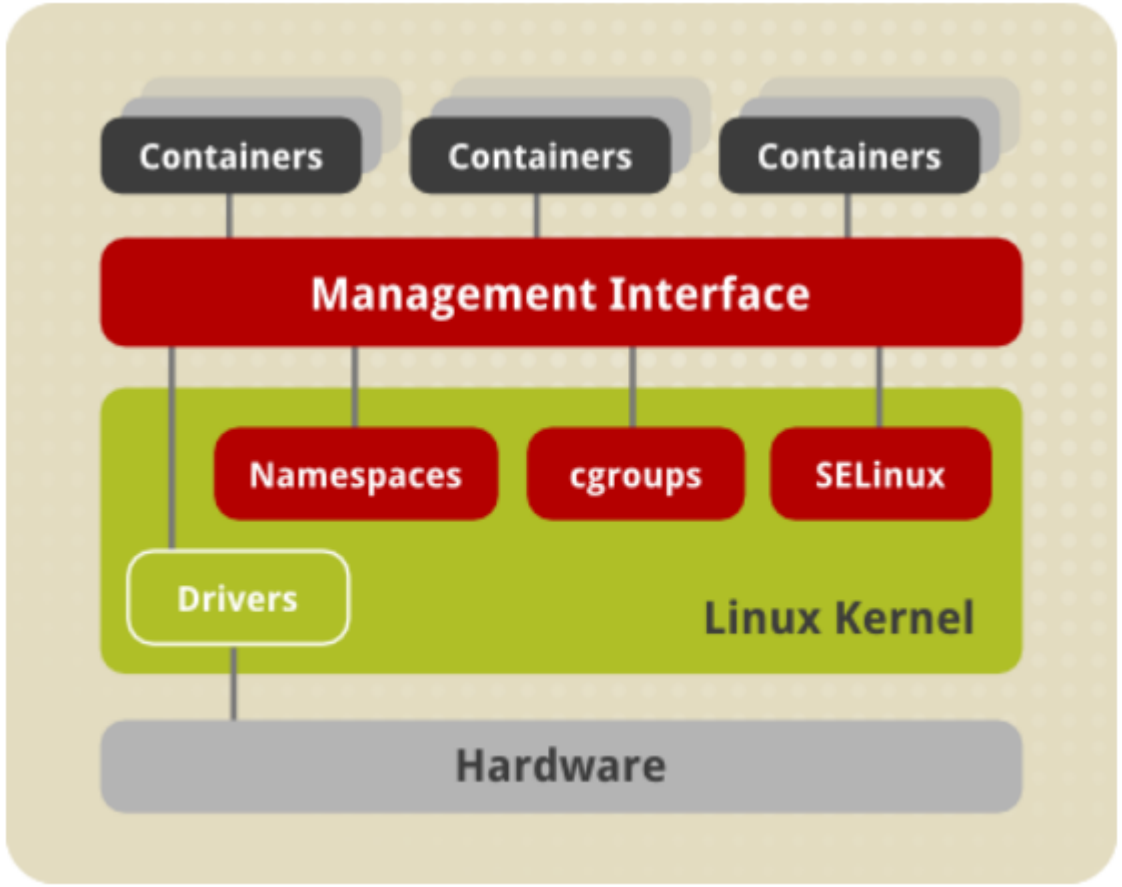
Open Source

Enterprise Grade

Secure

redhat

Container Architecture



Chapter 1. Introduction to Linux Containers Red Hat Enterprise Linux Atomic Host 7 | Red Hat Customer Portal

Open Container Initiative

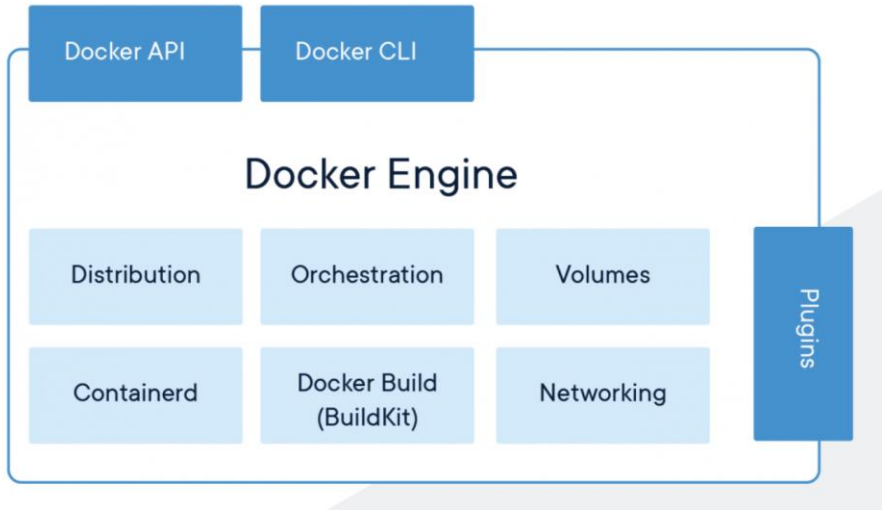
- Image Format Specification
- Interoperable tools to run containers
  - Building, transport, preparation

[image-spec/spec.md at master · opencontainers/image-spec · GitHub]

- Image Layout, Manifest
- Image Index
- Filesystem Layers
- Image Configuration
- Annotations
- Conversion
- Considerations
  - Extensibility
  - Canonicalization

Enterprise Needs

Container Runtime



Challenges

- Inter-services communication
  - > 100s
- Application Management over all associated containers
  - Container elasticity (scale up/down)
  - Service performance management
  - Release Train

Kubernetes – Features & Needs

Features

- Orchestrate containers across multiple hosts.
- Maximize resources needed to run enterprise apps.
- Control and automate application deployments and updates.
- Mount and add storage to run stateful apps.
- Scale containerized applications and their resources on the fly.
- Declaratively manage services, which guarantees the deployed applications are always running as designed
- Health-check and self-heal apps with auto-placement, auto-restart, auto-replication, and autoscaling.

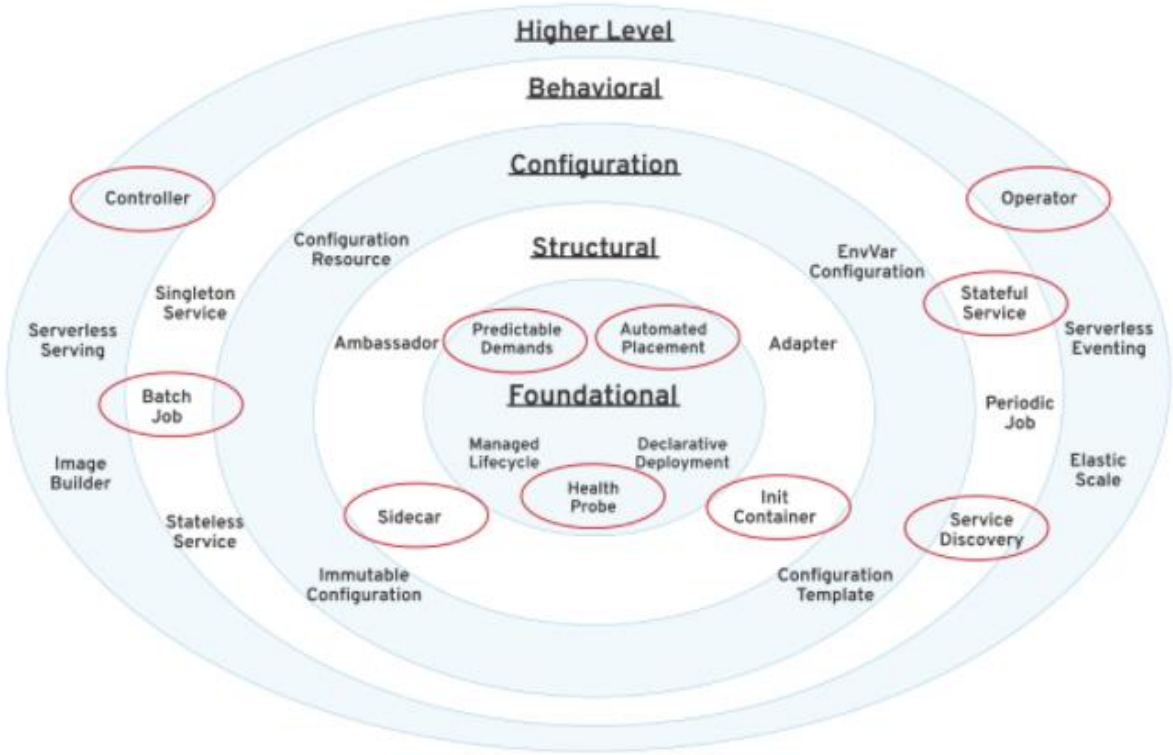
Needs

- Registry, e.g. Docker Registry.
- Networking – OpenvSwitch, intelligent edge routing.
- Telemetry - Kibana, Hawkular, and Elastic.
- Security - LDAP, SELinux, RBAC, OAUTH with multitenancy layers.
- Automation - Ansible playbooks for installation and cluster life cycle management.
- Services, through a rich catalog of popular app patterns

What is Kubernetes?

Kubernetes Service Patterns

- Foundational
- Health Probe
- Predictable Demands
- Automated Placement
- Structural
  - Init Container & Sidecar
- Behavioral
  - Batch Job, Stateful Discovery
- Higher-level
  - Controller, Operator

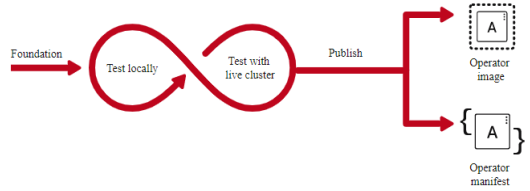


[Top 10 must-know Kubernetes design patterns | Red Hat Developer]

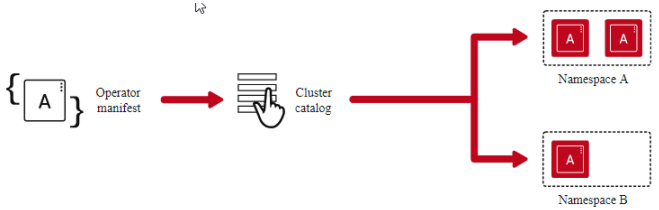
Kubernetes + OpenShift

- Developer
  - Built-in CI/CD workflow platform
  - Command-line interfaces
    - Node, Java, Ruby, Perl, PHP, Python
    - Odo – create, push, watch
- Operator
  - Leverage existing infrastructure
    - Physical, virtual
    - On-premise & Cloud
    - Hybrid & Multi-Cloud

- OpenShift Operator Framework
  - Community & Certified



  - Operator SDK
  - Operator package manager

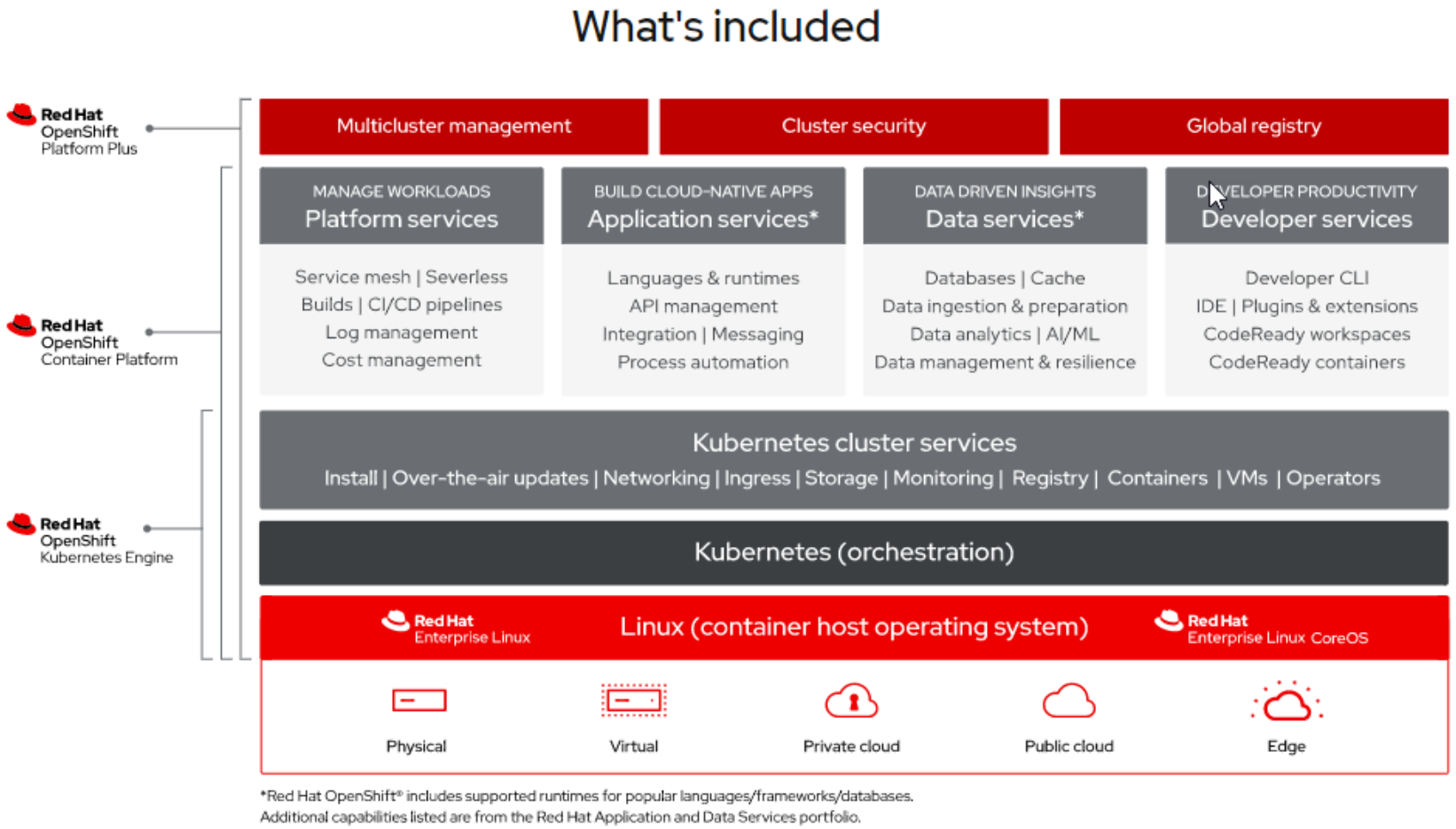


  - Operator metering

[Red Hat OpenShift, the open hybrid cloud platform built on Kubernetes]

[Operators on Red Hat OpenShift]

OpenShift Features



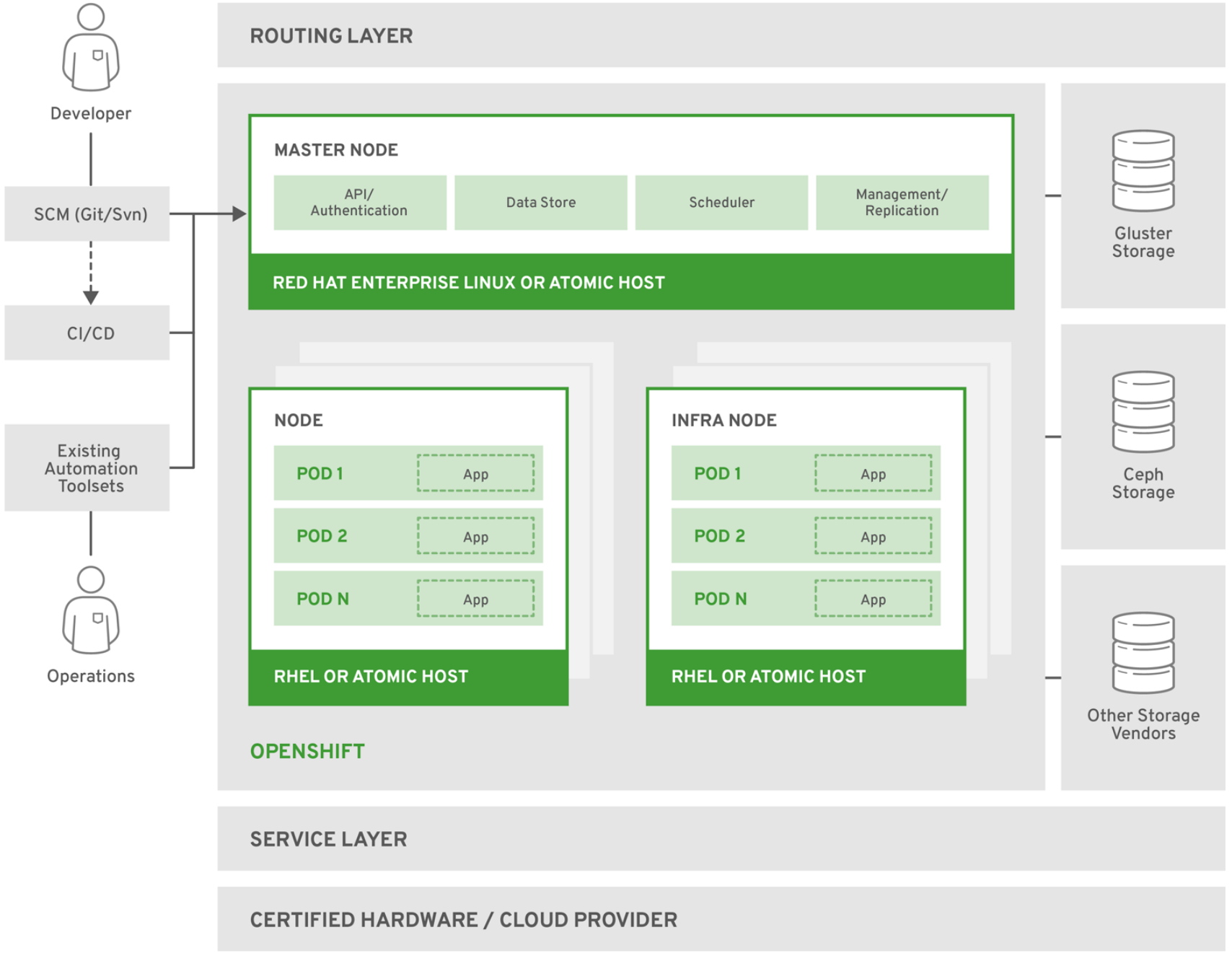
[Container Platform Features - Red Hat OpenShift]

Red Hat OpenShift Container Platform

- Set of modular components and services
  - Red Hat CoreOS and Kubernetes.
- PaaS capabilities
  - Remote management, increased security, monitoring and auditing, application life-cycle management, and self-service interfaces for developers.
- An OpenShift cluster = Kubernetes cluster + OpenShift management tools
  - CLI, web console for productive workflows and easy to execute common tasks

Term	Definition
Infra Node	Node server - infrastructure services l(monitoring, logging, or external routing)
Console	A web Ul to interact with cluster resources.
Project	OpenShift's extension of Kubernetes' namespaces Allows UAC definition for resources

OpenShift & K8s Architecture

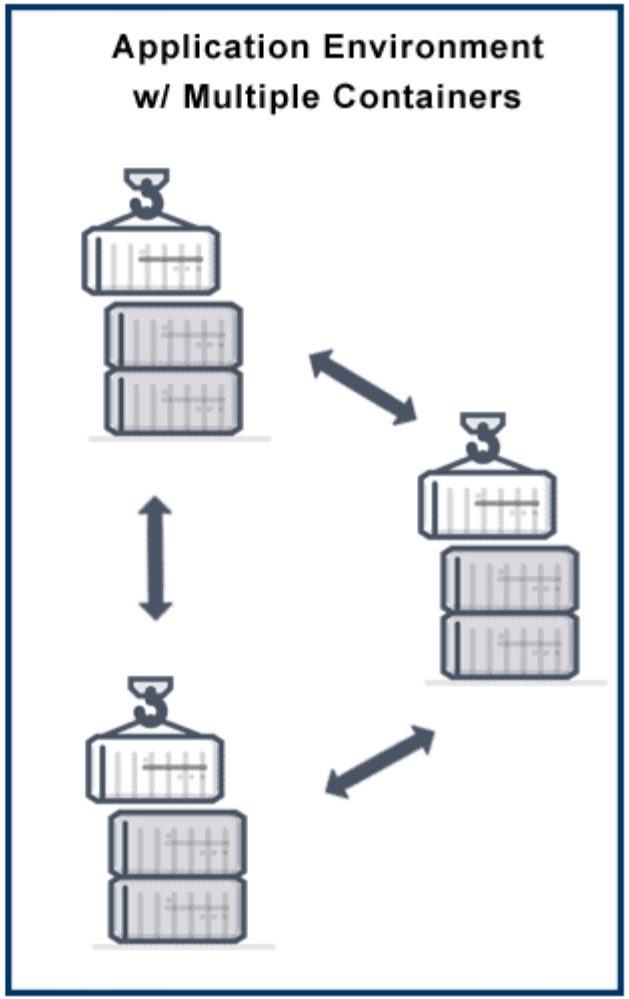


[Overview | Architecture | OpenShift Container Platform 3.5]

Container Orchestration Software
(Docker, Openshift & Kubernetes)

Automate:

- Configuration
- Provisioning
- Availability
- Scaling
- Security
- Resource allocation
- Load balancing
- Health monitoring

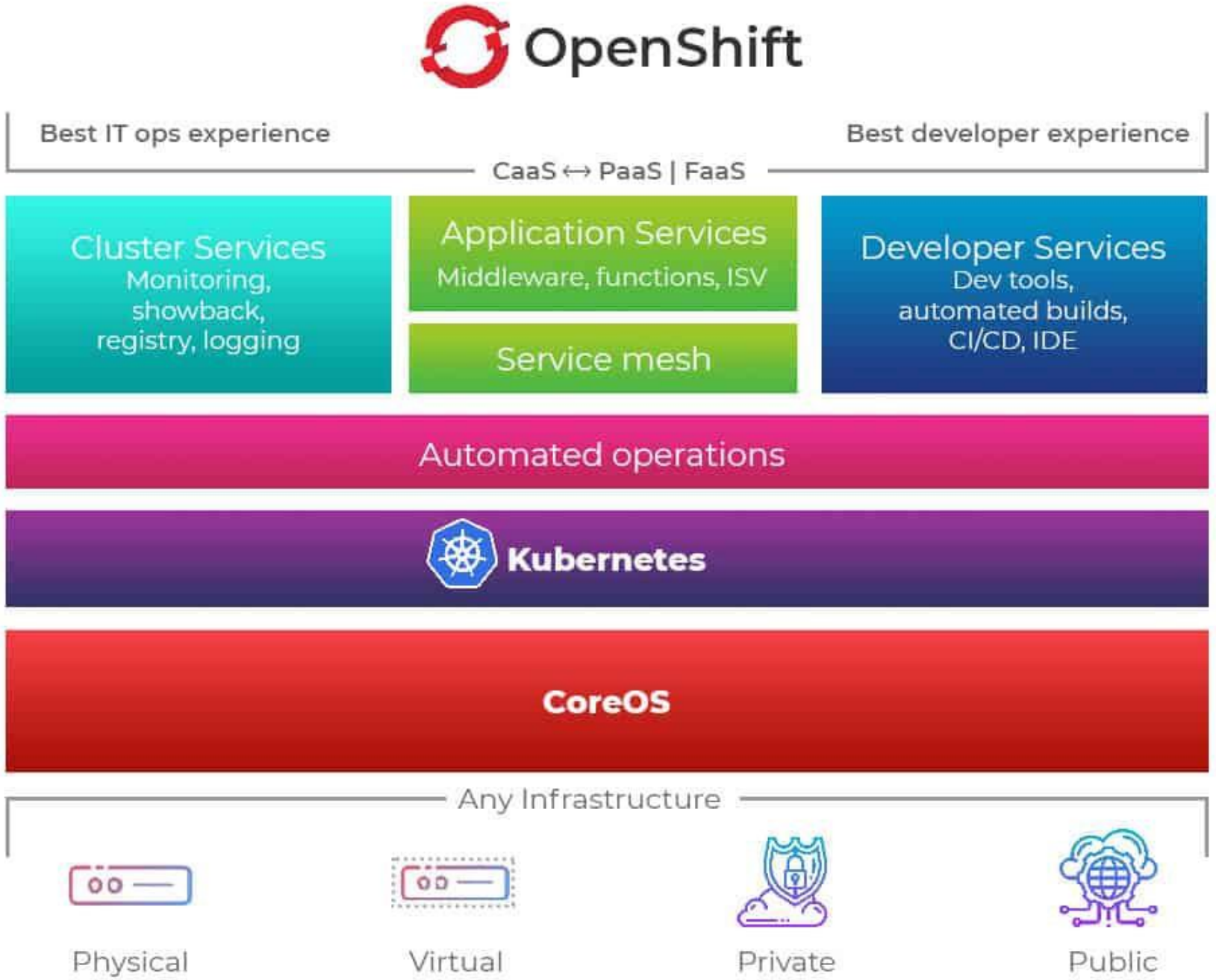


Kubernetes & OpenShift

Kubernetes Explained

What is OpenShift?

Kubernetes and OpenShift: What's the Difference?



How Red Hat OpenShift can support your hybrid multicloud environment - Servers & Storage (ibm.com)

OpenShift Architecture

- Base - Red Hat CoreOS
  - Immutable operating system for container execution
- CRI-O - Kubernetes Container Runtime Interface to enable using OCI compatible runtimes
  - CRI-O can use any container runtime that satisfies CRI
  - rune (used by the Docker service), libpod (used by Podman) or rkt (from CoreOS).
- *Resources*
  - *D*escribe multi-container applications connection
- *Etcd* - distributed key-value store
  - Configuration and state information in cluster
- *Custom Resource Definitions (CRDs)*
  - resource types stored in Etcd
  - Form the state and configuration of all resources managed by OpenShift

OpenShift Architecture

- PaaS infrastructure functions - networking and authorization
  - RHOCP uses the basic container infrastructure from Kubernetes and the underlying container runtime for most internal functions
  - Most RHOCP internal services run as containers orchestrated by Kubernetes
- Runtimes and x-PaaS - base container images for developers
  - Preconfigured with a particular runtime, e.g., JBoss EAP and ActiveMQ.
  - *Red Hat OpenShift Application Runtimes (RHOAR)* - runtimes optimized for cloud native applications in OpenShift.
  - Red Hat JBoss EAP, OpenJDK, Thorntail, Eclipse Vert.x, Spring Boot, and Node.js.
- DevOps tools and user experience
  - RHOCP provides web UI and CLI management tools
  - OpenShift web UI and CLI tools - REST APIs

OpenShift, K8s Resource Types

- OpenShift
  - Deployment config
  - Build config
  - Routes
- Kubernetes
  - Pods
  - Services
  - Replication Controllers
  - Persistent Volumes
  - PV Claims
  - ConfigMaps
  - Secrets

Networking

- Each container in a Kubernetes cluster has an IP address assigned from an internal network
  - Accessible only from the node running the container
  - Container's ephemeral nature, IP addresses are constantly assigned and released.
- Kubernetes *software-defined network (SDN)*
  - Internal container networks from multiple nodes
  - Allows containers from any pod, inside any host, to access pods from other hosts
  - Access to the SDN only works from inside the same Kubernetes cluster
- Containers inside Kubernetes pods should not connect to each other's dynamic IP address directly
- Services resolves this problem by linking more stable IP addresses from the SDN to the pods. If pods are restarted, replicated, or rescheduled to different nodes, services are updated, providing scalability and fault tolerance.
- External access to containers – NodePort attribute; not scalable
- *OpenShift Route* resources – external access scalable and simpler
  - Route defines external-facing DNS names and ports for a service
  - A router (ingress controller) forwards HTTP and TLS requests to the service addresses inside the Kubernetes SDN
  - The only requirement is that the desired DNS names are mapped to the IP addresses of the RHOCP router nodes.


Accenture

ViaVarejo

Runtastic


Walls.io


Proshore

all

TOKIGAMES TECH

Ericsson

DBS C2E

The Walt Disney ...


ConnectedMobility


Nuxeo


DevSecOps


DevOps


Samsviran

STS

Infrastructure


Interhyp AG

TECLO-INOMERA

system

Hazeorid


Node backend


Melio Consulting

BI X Digital Lab

Vungle

Shpock

Cloud9 IDE

Streamdata.io

dotgroup

Prattle

Huia

API Evangelist


datahub


Ukuli Data

CXDA


Red Hat Developer ...

CSAG

Thybit


budikom.net


Intelinvest

Mobile Enterprise

Codecraft Solutions

LooWID

Digistarters

Obsdeck Software

OpenShift Business Value

636%
five-year ROI

10 months
to payback

20% higher
DevOps and development team productivity

Almost 3x
more new features

29% faster
application development life cycles

$21.62M
in higher annual revenue

22% fewer
VMs required

21% more efficient
IT infrastructure teams

https://www.redhat.com/rhdc/managed-files/cl-idc-business-value-openshift-analyst-material-f28051-202104-en_0.pdf

OpenShift Labs

- Red Hat OpenShift on IBM Cloud
- Develop in a preconfigured OpenShift environment available for four hours at no charge
- Getting started with Red Hat OpenShift

Quay

- Quay Documentation
- Can use GitHub credentials

Podman

- Podman
- What is Podman?
  - Simply put: **alias docker=podman**
  - What is podman?
- podman/README.md at master · containers/podman (github.com)
  - Podman commands
    - Search, pull, images
    - registry_name/user_name/image_name:tag
  - Run command, Inspect (configuration)
    - Interactive terminal and detach modes, environment variables
- Red Hat Universal Base Image (container)

Podman

podman

- Podman is a daemonless, open source, Linux native tool
  - Find, run, build, share and deploy applications using Open Containers Initiative (OCI) Containers and Container Images
- Podman provides a command line interface (CLI) similar to Docker
  - Alias Docker to Podman (alias docker=podman)
  - OCI compliant Container Runtime
  - Interface with the operating system and create the running container
  - Indistinguishable from any other common container engine
- Podman manages the entire container ecosystem
  - Pods, containers, container images, and container volumes using the libpod library

What is Podman? — Podman documentation

Red Hat Container Catalog

- Over 5000 images
- Red Hat Ecosystem Catalog