OpenShift Enterprise

a Containerized Application Platform

@SamuelTerburg
OpenShift “Specialist” Solution Architect

March 2016

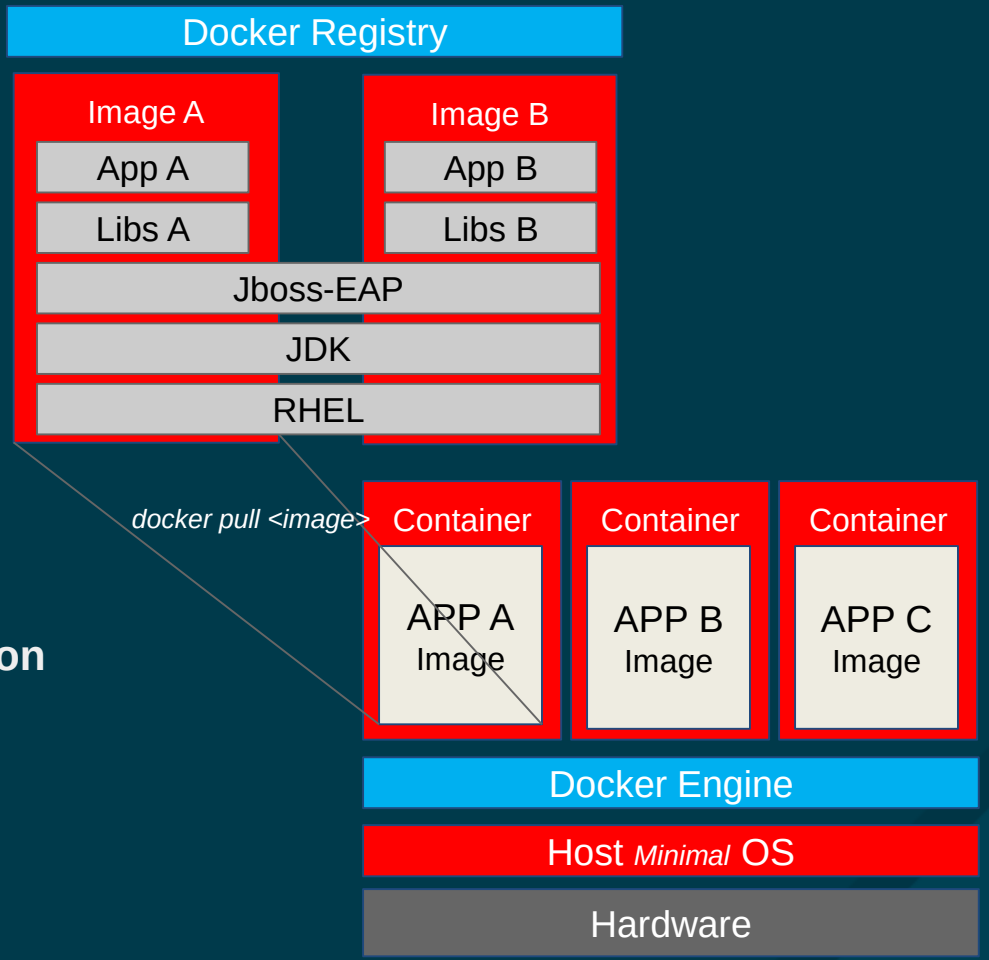
Agenda

- Docker
- Kubernetes added-value
- OpenShift added-value
- Demo
- Q & A

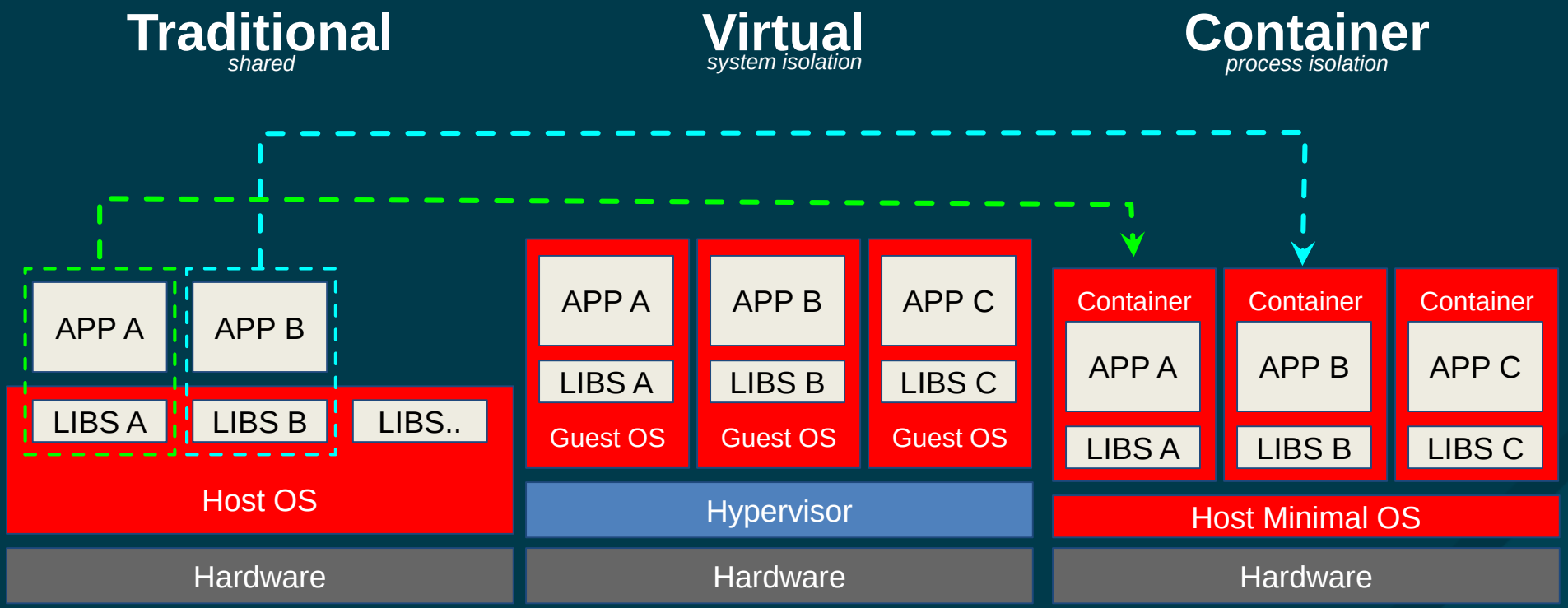

Container Technology
- Docker
redhat.

Images & Containers

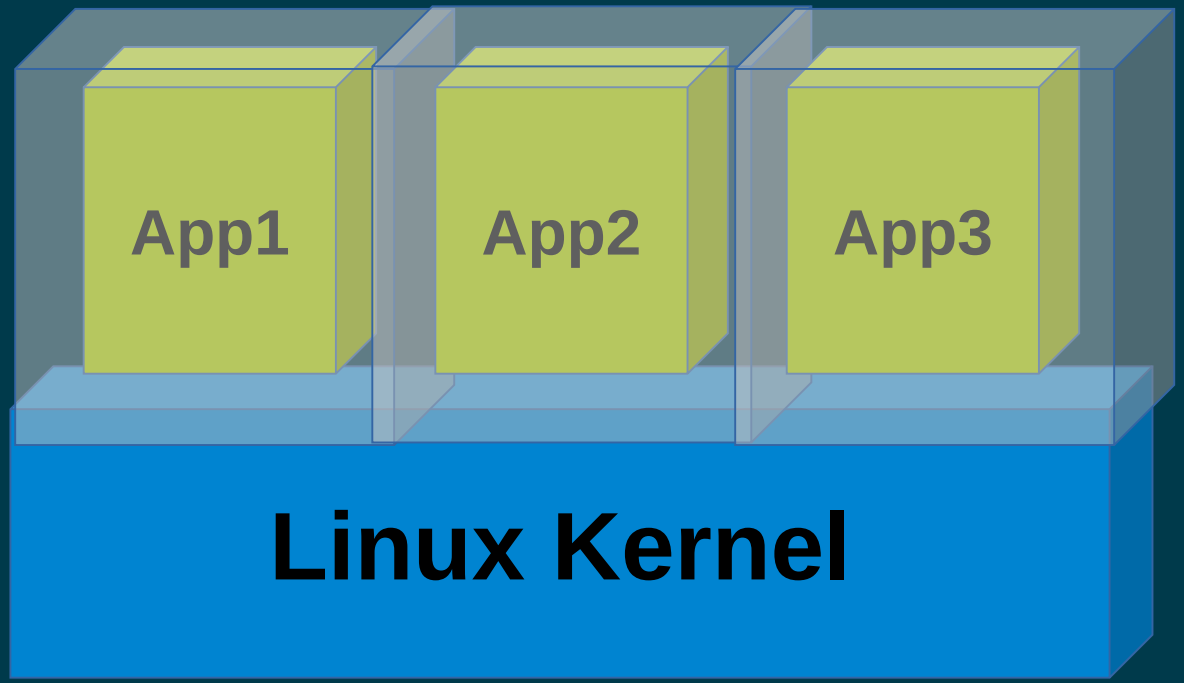
- **Docker “Image”**
  - **Unified Packaging format**
    - **Like “war” or “tar.gz”**
    - **For any type of Application**
    - **Portable**
- **Docker “Container”**
  - **Runtime**
  - **Isolation**



Evolution


Traditional
shared
Virtual
system isolation
Container
process isolation
APP A
APP B
LIBS A
LIBS B
LIBS..
Host OS
Hardware
APP A
APP B
APP C
LIBS A
LIBS B
LIBS C
Guest OS
Guest OS
Guest OS
Hypervisor
Hardware
Container
Container
Container
APP A
APP B
APP C
LIBS A
LIBS B
LIBS C
Host Minimal OS
Hardware

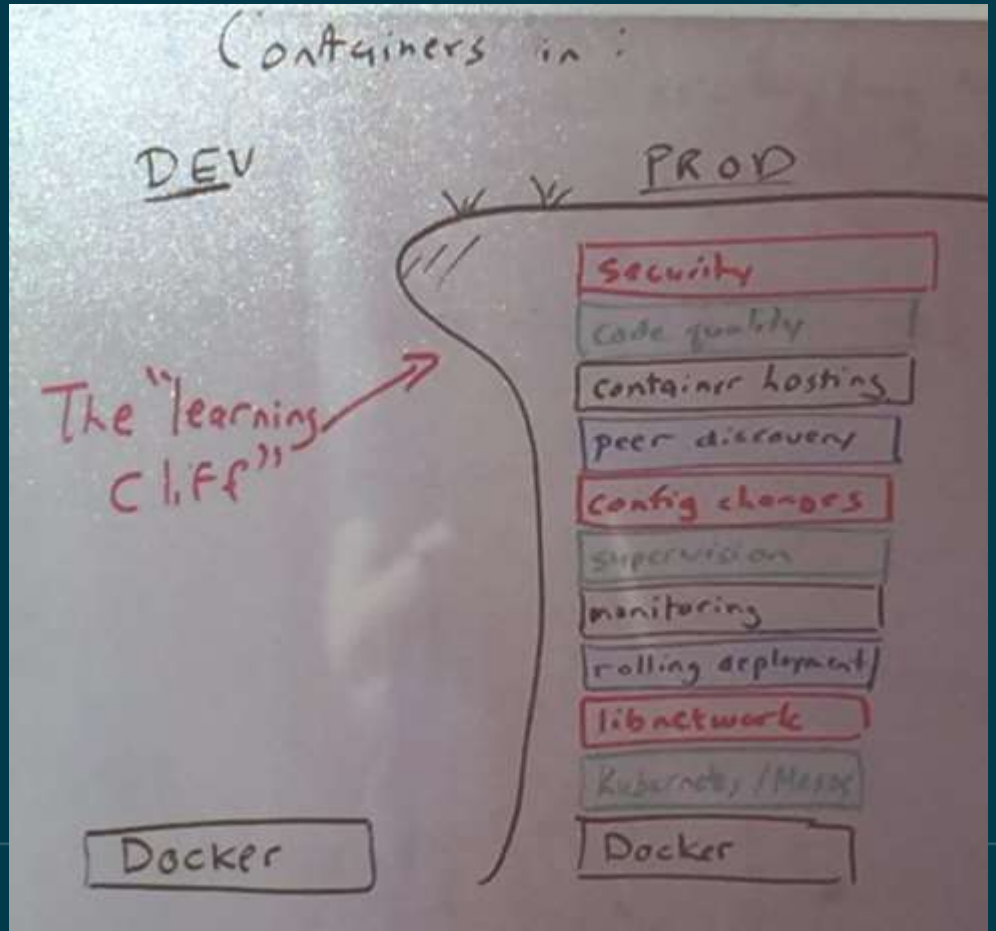
Isolation, not Virtualization



- Kernel Namespaces
  - Process
  - Network
  - IPC
  - Mount
  - User
- Resource Limits
  - Cgroups
- Security
  - SELinux


Container Orchestration
- Kubernetes
redhat.

We need more than just packing and isolation



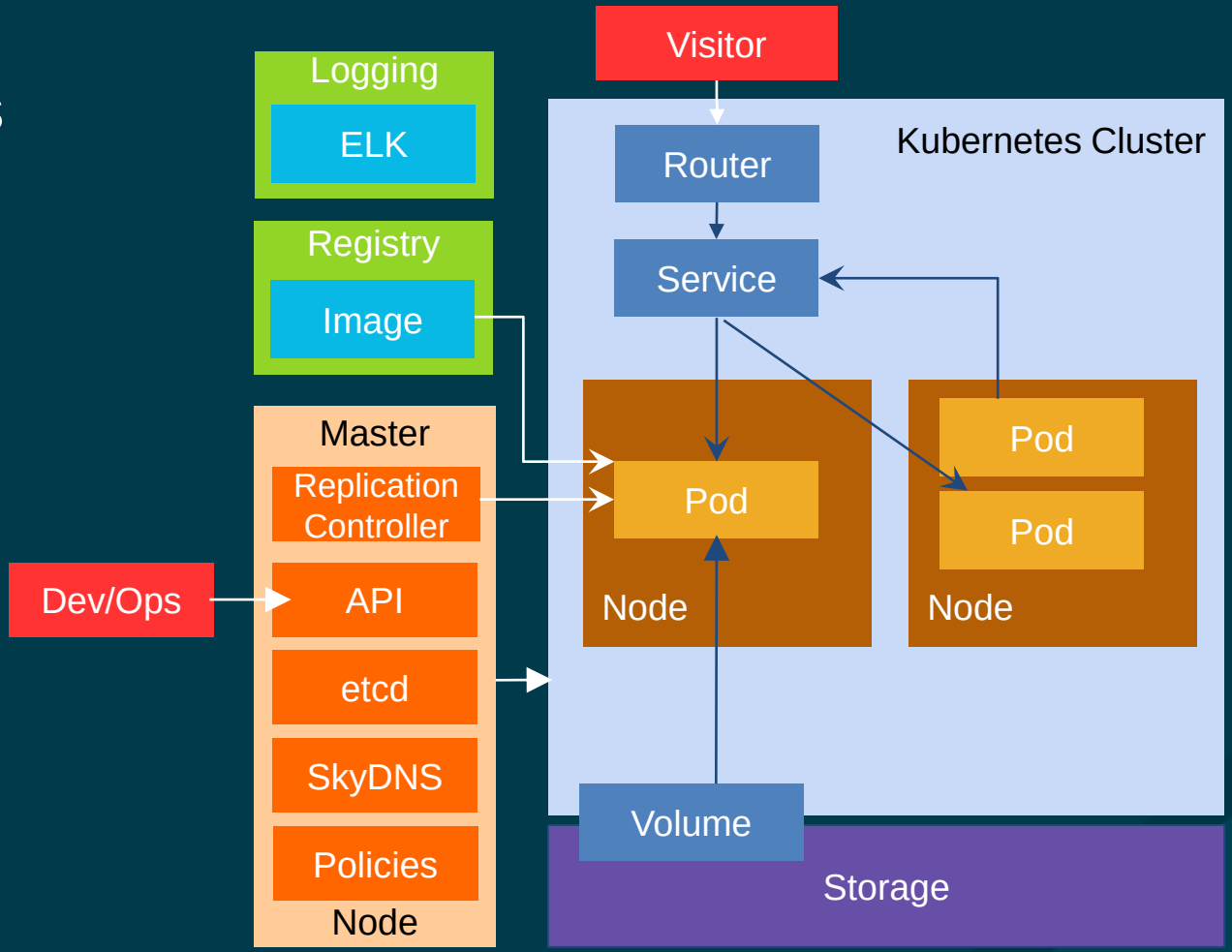
Kubernetes – Container Orchestration at Scale

Greek for “Helmsman”; also the root of the word “Governor” and “cybernetic”

- Container Cluster Manager
  - Inspired by the technology that runs Google
- Runs anywhere
  - Public cloud
  - Private cloud
  - Bare metal
- Strong ecosystem
  - Partners: Red Hat, VMware, CoreOS..
  - Community: clients, integration

Core Concepts

- Pod
- Labels & Selectors
- ReplicationController
- Service
- Persistent Volumes



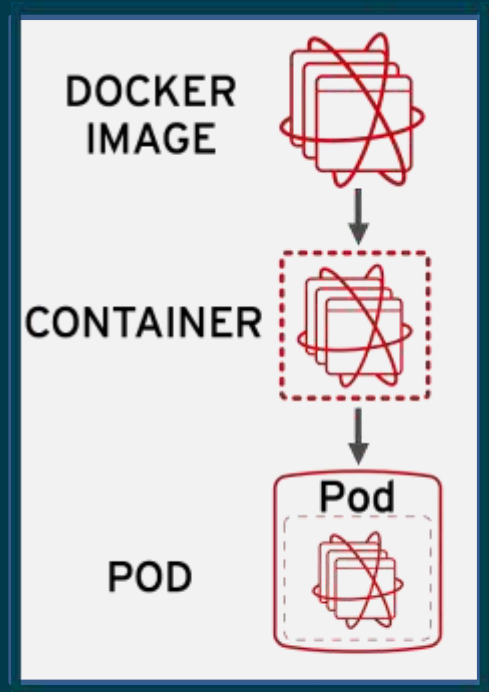
Pods

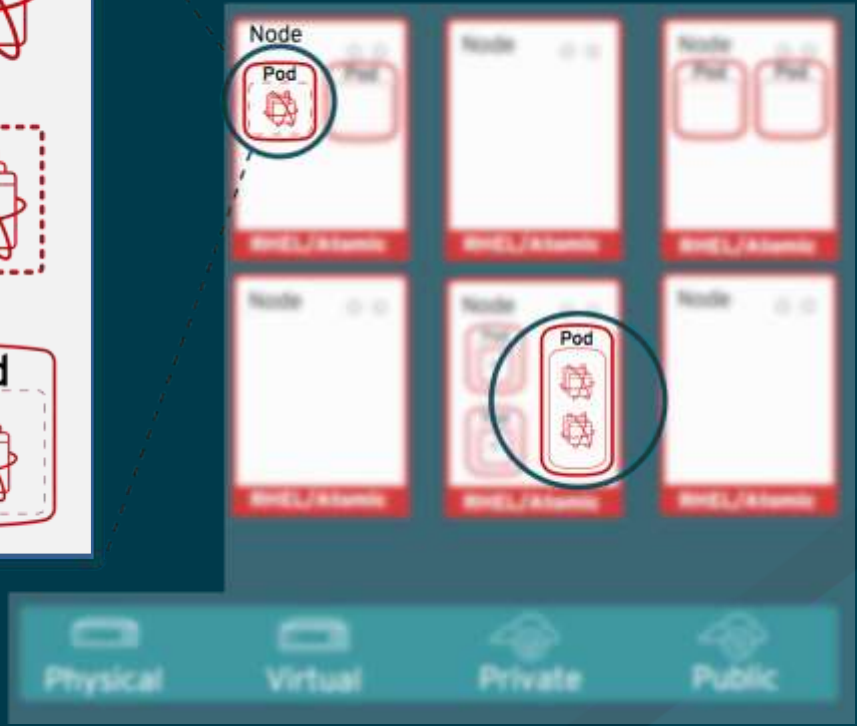
POD Definition:

- Group of Containers
- Related to each other
- Same namespace
- Emphemeral

Examples:

- Wordpress
- MySQL
- ~~Wordpress + MySQL~~
- ELK
- Nginx+Logstash
- Auth-Proxy+PHP
- App + data-load

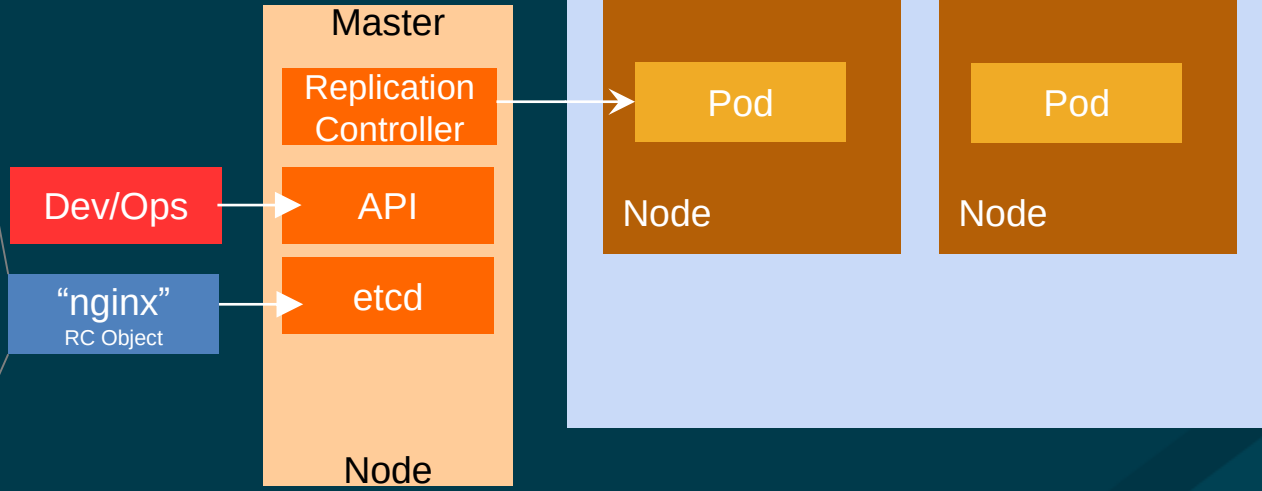




Replication Controller

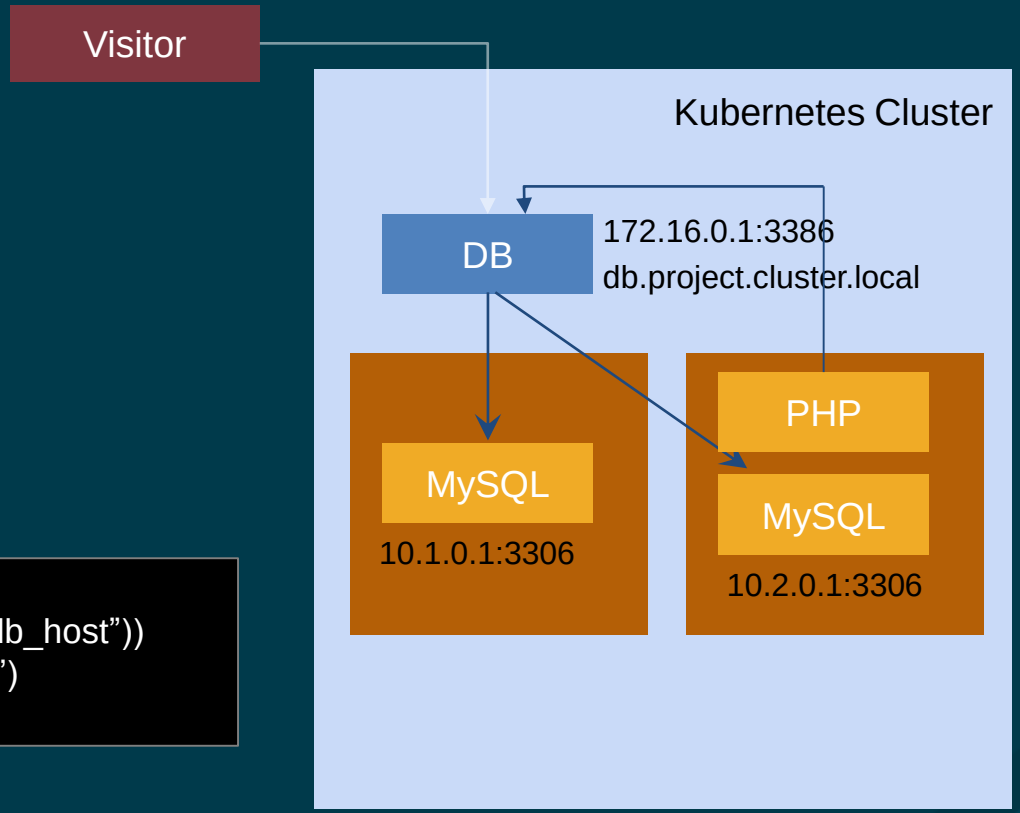
- Pod Scaling
- Pod Monitoring
- Rolling updates

```
kind: ReplicationController
metadata:
   name: nginx
spec:
   replicas: 2
   selector:
      app: nginx
template:
   metadata:
      name: nginx
      labels:
         app: nginx
   spec:
      containers:
       - name: nginx
         image: nginx:v2.2
         ports:
          - containerPort: 80
```




```
# kubectl create –f nginx-rc.yaml
```

Service



Service Definition:

- Load-Balanced Virtual-IP *(layer 4)*
- Abstraction layer for your App
- Enables Service Discovery
  - DNS
  - ENV

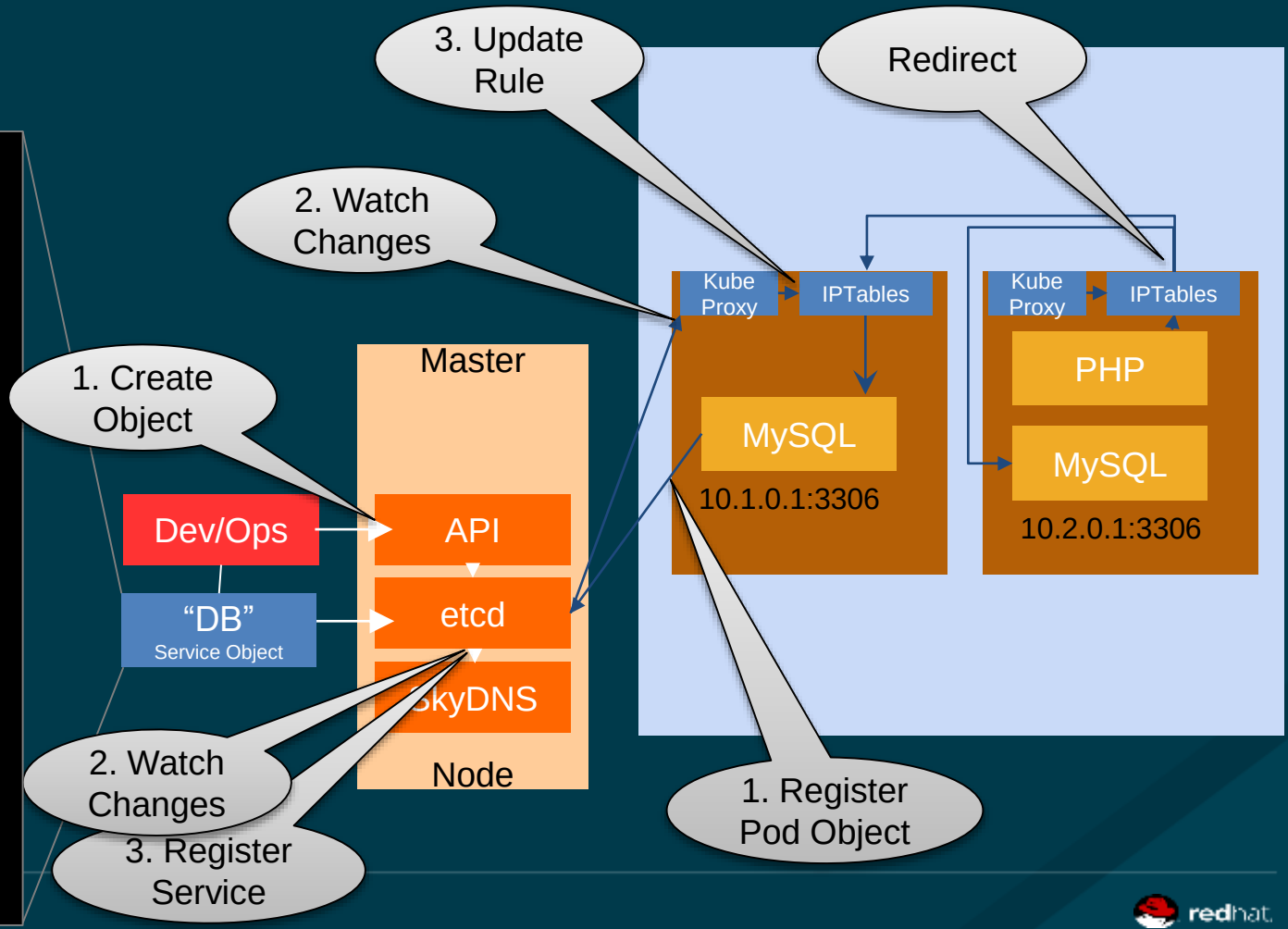
```
<?php
   mysql_connect(getenv(“db_host”))
   mysql_connect(“db:3306”)
?>
```

Examples:

- frontend
- database
- api

Service

```
- apiVersion: v1
  kind: Service
  metadata:
    labels:
      app: MySQL
      role: BE
      phase: DEV
    name: MySQL
  spec:
    ports:
    - name: mysql-data
      port: 3386
      protocol: TCP
      targetPort: 3306
    selector:
      app: MySQL
      role: BE
    sessionAffinity: None
    type: ClusterIP
```



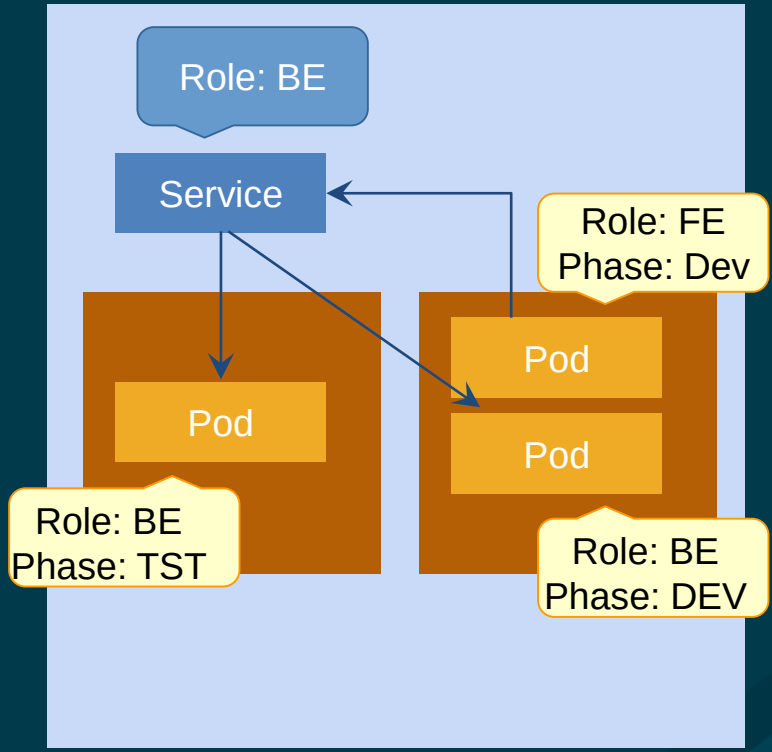
redhat.

Labels & Selectors

think SQL 'select ... where ...'

```
- apiVersion: v1
  kind: Service
  metadata:
    labels:
      app: MyApp
      role: BE
      phase: DEV
    name: MyApp
  spec:
    ports:
    - name: 80-tcp
      port: 80
      protocol: TCP
      targetPort: 8080
    selector:
      app: MyApp
      role: BE
    sessionAffinity: None
    type: ClusterIP
```

```
- apiVersion: v1
  kind: Pod
  metadata:
    labels:
      app: MyApp
      role: BE
      phase: DEV
    name: MyApp
```



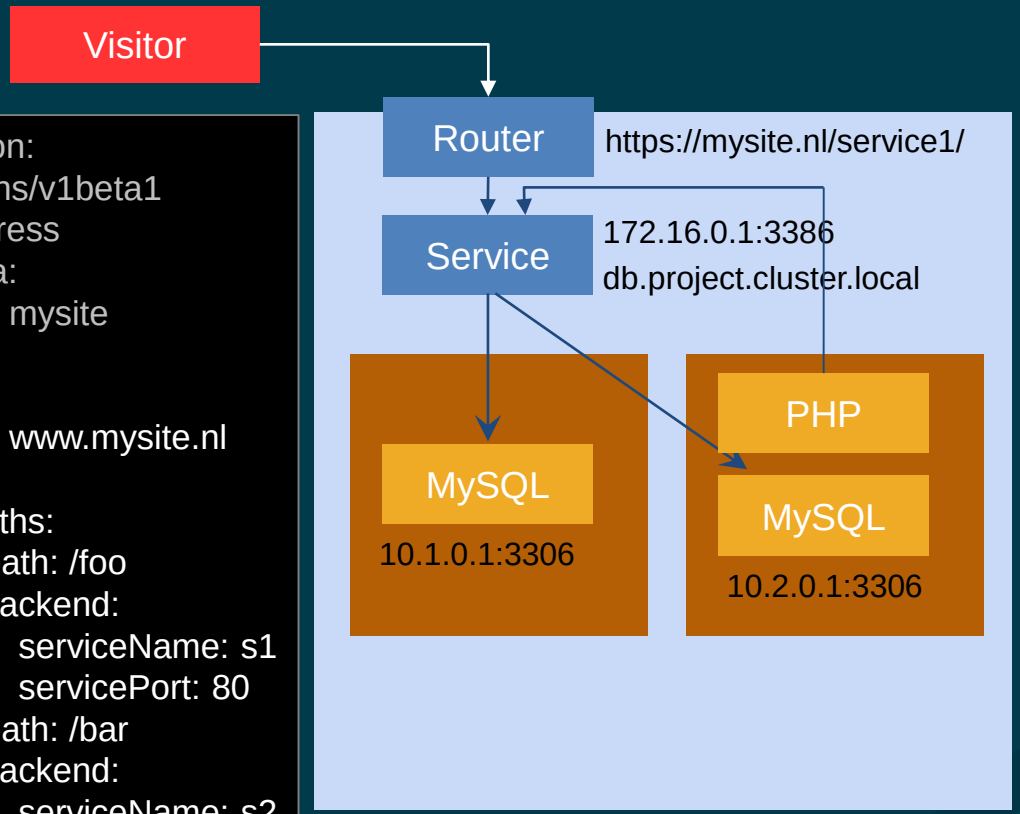
Ingress / Router

- Router Definition:
  - Layer 7 Load-Balancer / Reverse Proxy
  - SSL/TLS Termination
  - Name based Virtual Hosting
  - Context Path based Routing
  - Customizable (image)
    - HA-Proxy
    - F5 Big-IP

Examples:

- https://www.mysite.nl/myapp1/
- http://www.mysite.nl/myapp2

```
apiVersion:
extensions/v1beta1
kind: Ingress
metadata:
   name: mysite
spec:
   rules:
   - host: www.mysite.nl
     http:
        paths:
        - path: /foo
          backend:
             serviceName: s1
             servicePort: 80
        - path: /bar
          backend:
             serviceName: s2
             servicePort: 80
```



Persistent Storage

for Ops:

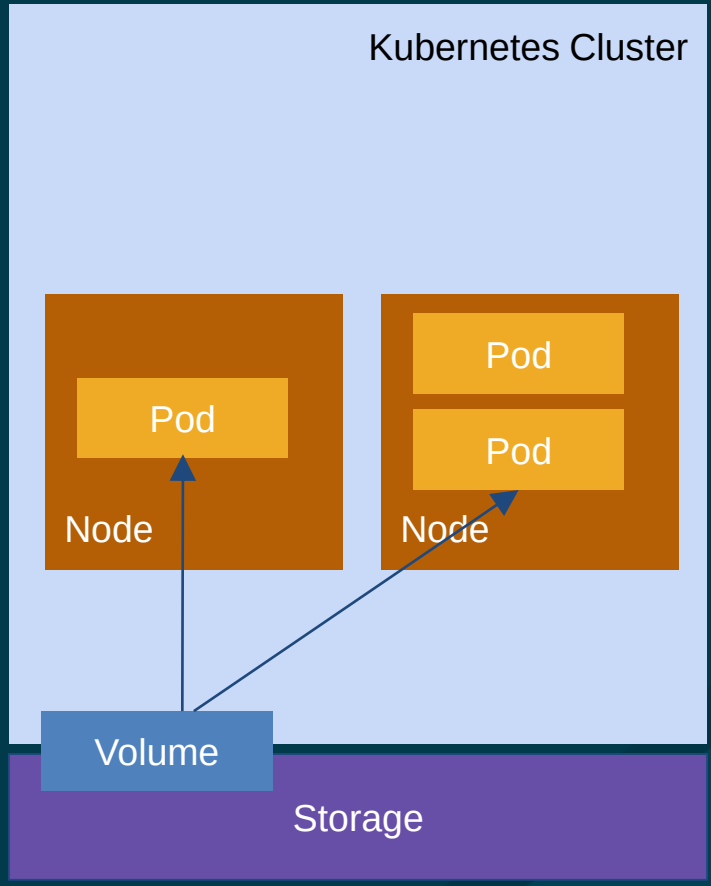
- Google
- AWS EBS
- OpenStack's Cinder
- Ceph
- GlusterFS
- NFS
- iSCSI
- FibreChannel
- EmptyDir

for Dev:

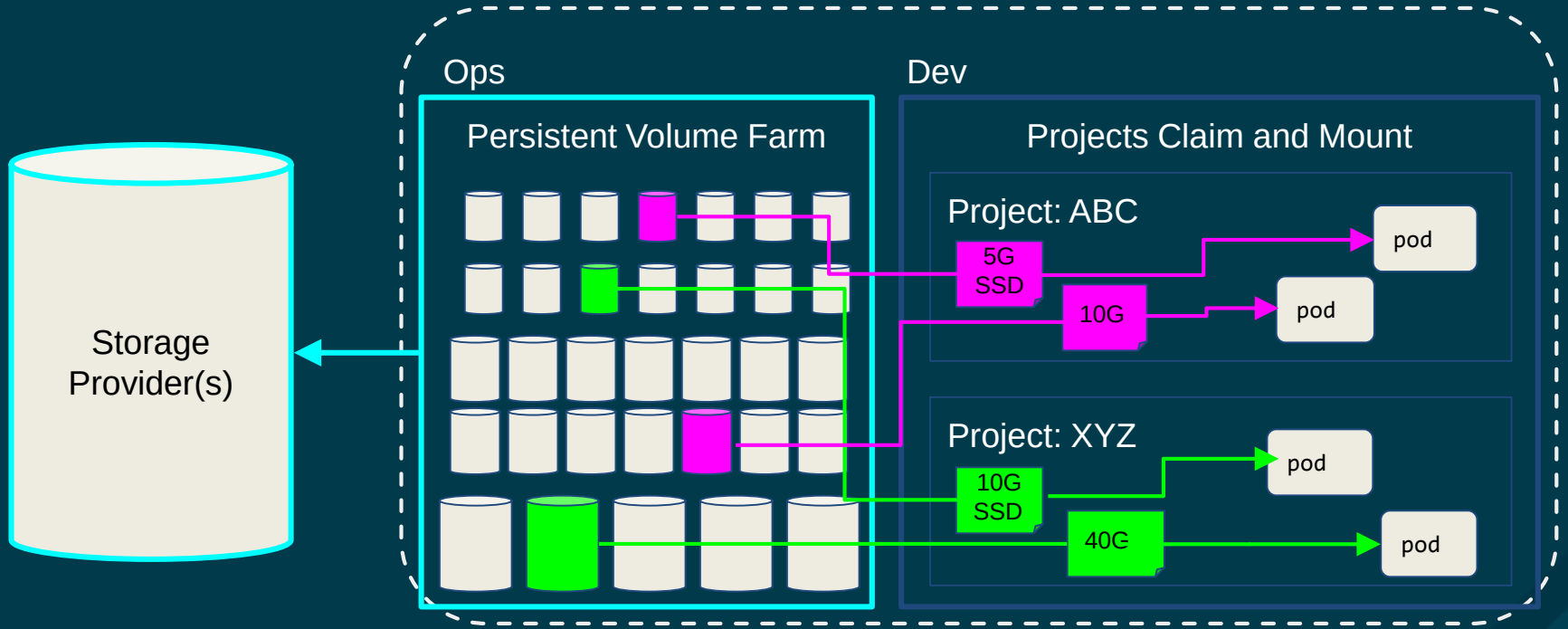
- “Claim”

```
kind: PersistentVolume
metadata:
    name: pv0003
spec:
    capacity:
        storage: 8Gi
    accessModes:
     - ReadWriteOnce
    nfs:
        path: /tmp
        server: 172.17.0.2
```

```
kind: PersistentVolumeClaim
metadata:
    name: myclaim
spec:
    accessModes:
     - ReadWriteOnce
    resources:
        requests:
            storage: 8Gi
```

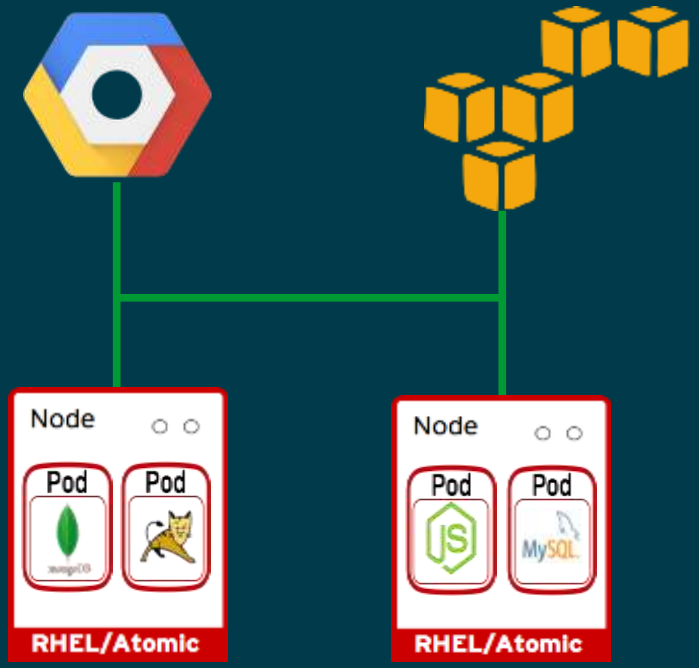


Persistent Volume Claim



Networking

- Each Host = 256 IPs
- Each POD = 1 IP



Programmable Infra:

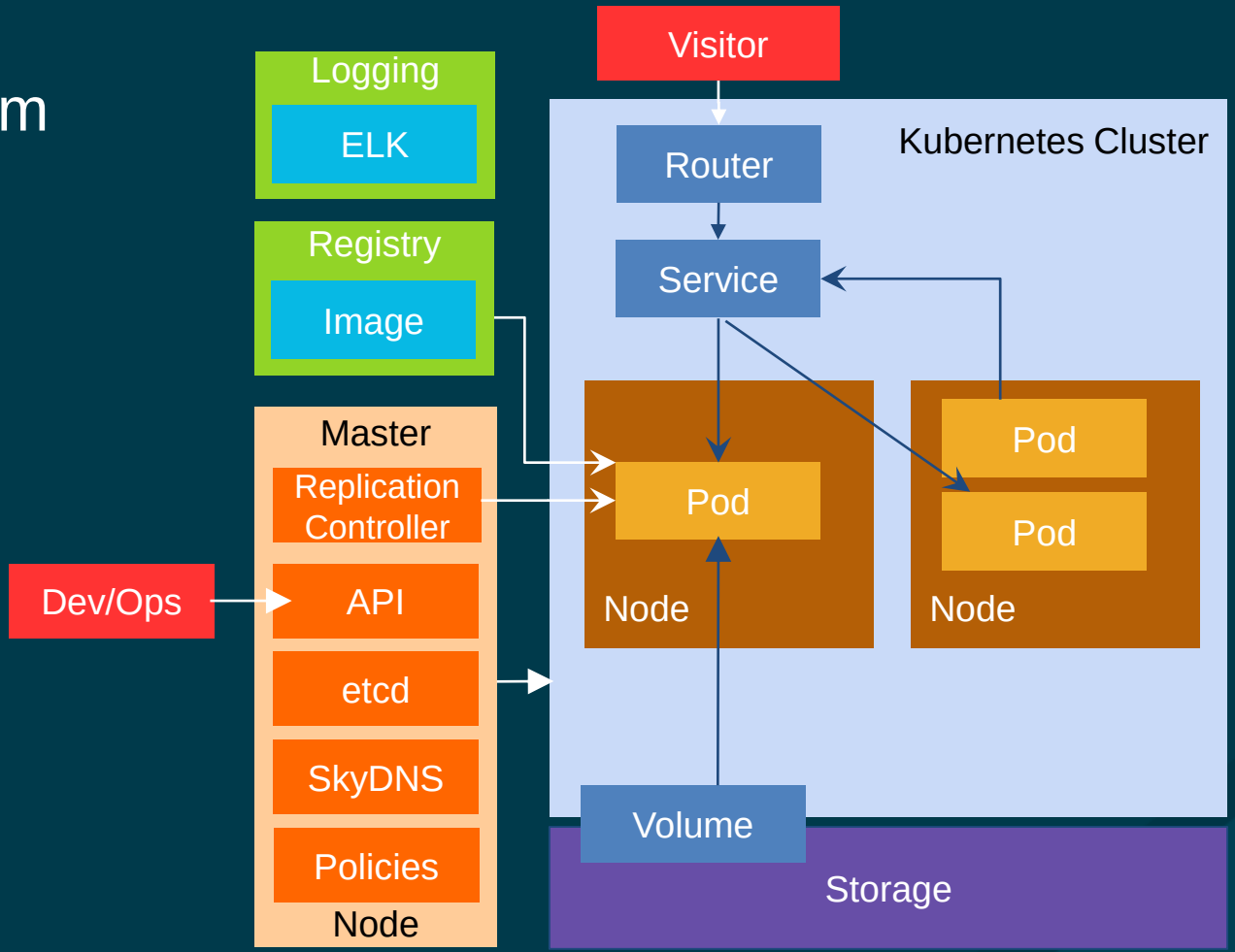
- GCE / **GKE**
- AWS
- OpenStack
- Nuage

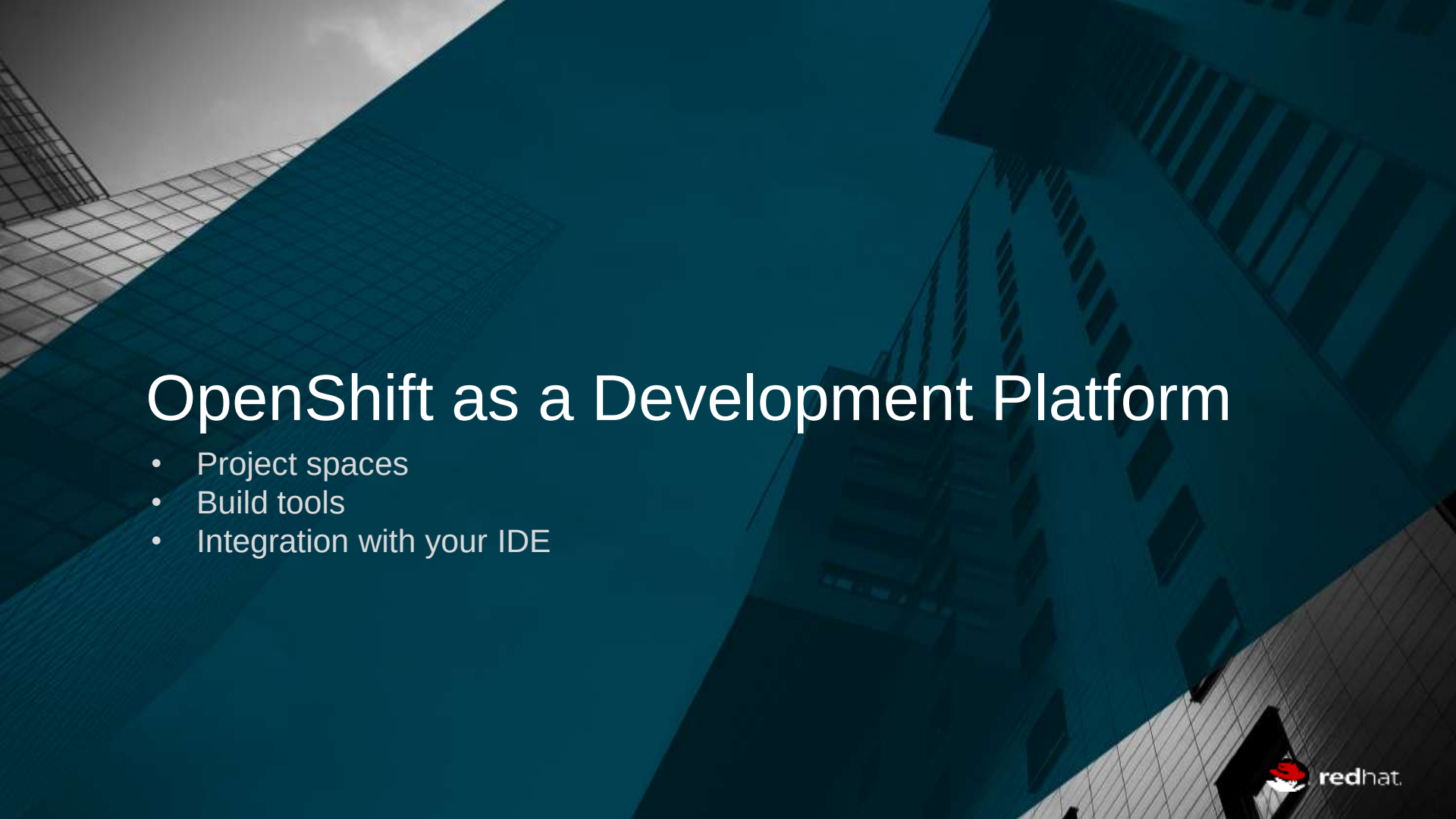
Overlay Networks:

- Flannel
- Weave
- **OpenShift-SDN**
- Open vSwitch

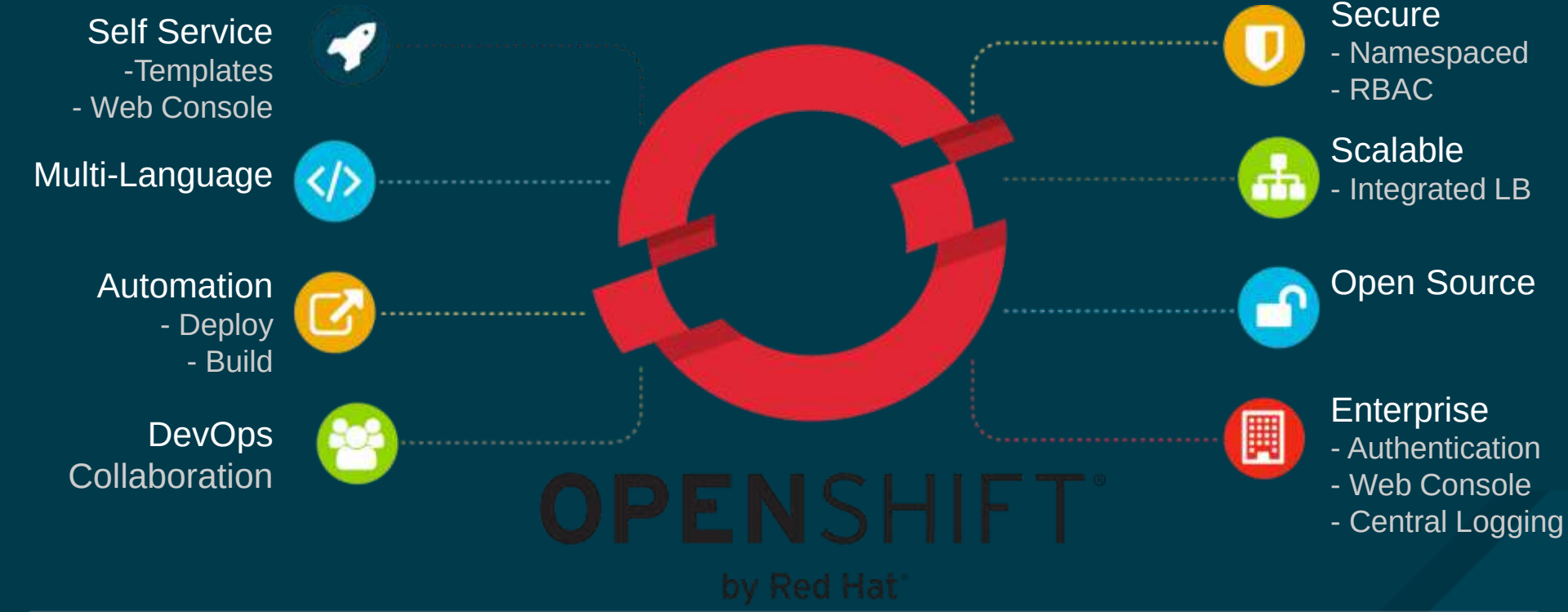
Hosting Platform

- Scheduling
- Lifecycle and health
- Discovery
- Monitoring
- Auth{n,z}
- Scaling

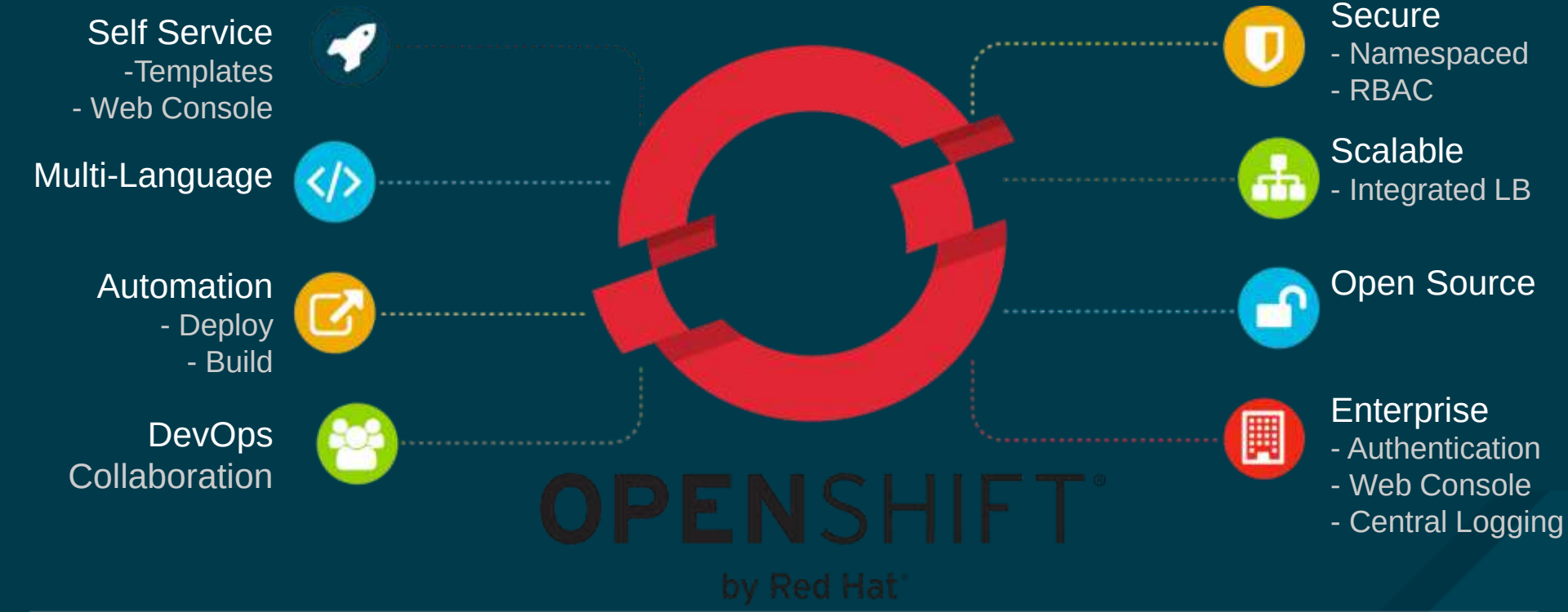



OpenShift as a Development Platform
• Project spaces
• Build tools
• Integration with your IDE
redhat.

We need more than just Orchestration !



OpenShift is Red Hat's Container Application Platform (PaaS)



Kubernetes Embedded

https://master:8443/api = Kubernetes API
/oapi = OpenShift API
/console = OpenShift WebConsole

OpenShift:

- 1 Binary for Master
- 1 Binary for Node
- 1 Binary for Client

- Docker-image
- Vagrant-image

Kubernetes:

- ApiServer, Controller, Scheduler, Etcd
- KubeProxy, Kubelet
- Kubectl

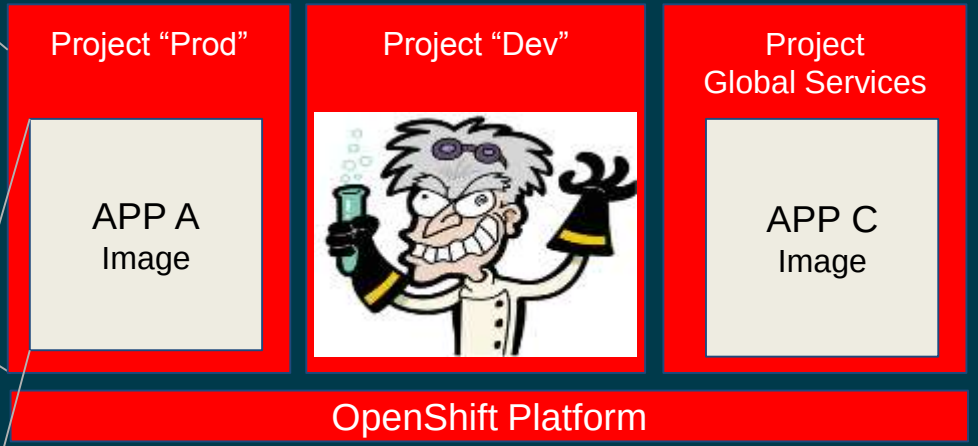
Project Namespaces

Project

- Sandboxed Environment
- Network VXLan
- Authorization Policies
- Resource Quotas
- *Ops in Control, Dev Freedom*

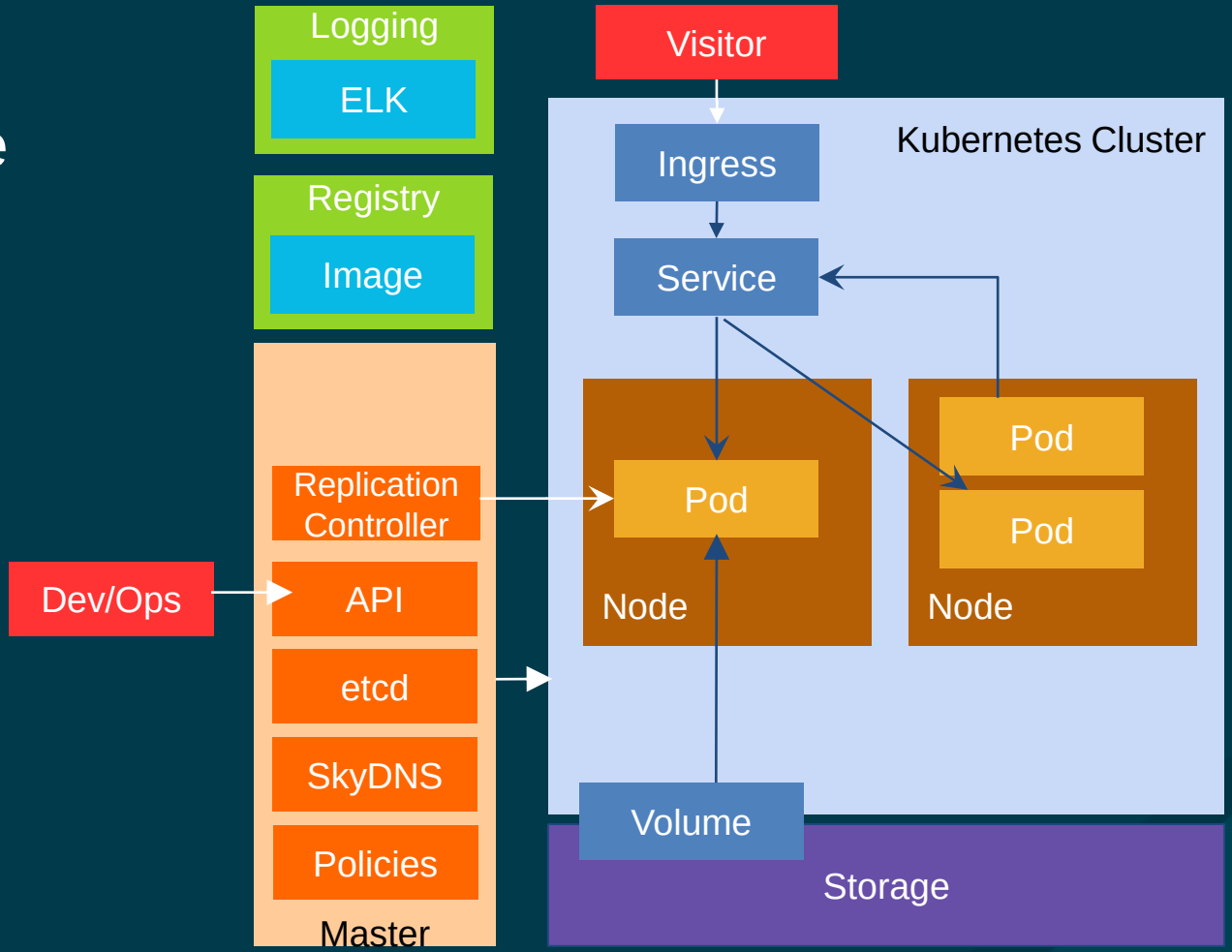
App

- Images run in Containers
- Grouped together as a Service
- Defined as Template




```
oc new-project  Project-Dev
oc policy add-role-to-user admin scientist1
oc new-app
  --source=https://gitlab/MyJavaApp
  --docker-image=jboss-eap
```

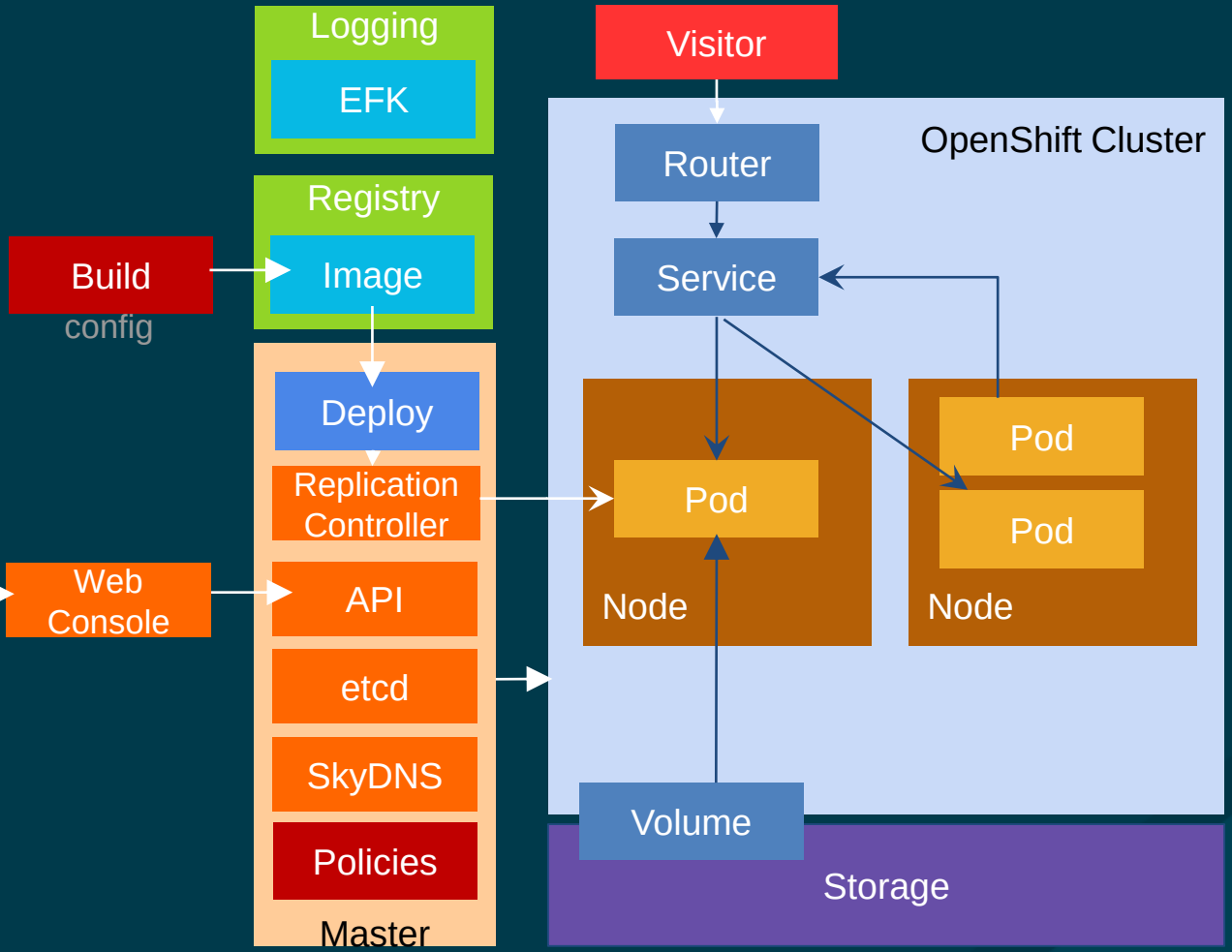
Kubernetes Hosting Architecture



OpenShift PaaS Architecture

- Added “Build”
- Added “Deployment”
- s/ELK/EFK/g
- s/Ingress/Router/g
- Added Policies + tools
- Added WebConsole

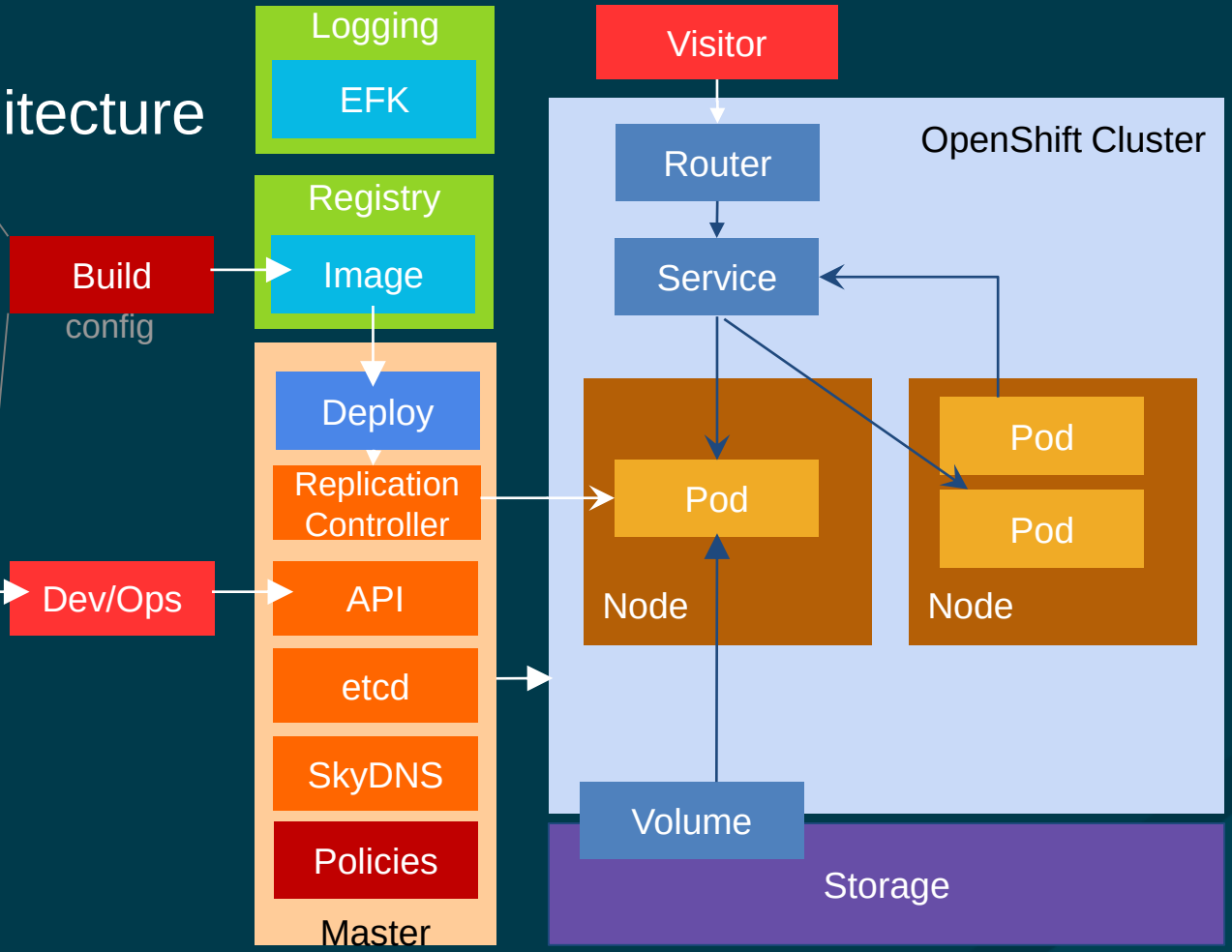
- OpenShift-SDN isolation



OpenShift
Build & Deploy Architecture

```
kind: "BuildConfig“
metadata:
    name: “myApp-build“
spec:
    source:
        type: "Git“
        git:
            uri: "git://gitlab/project/hello.git“
            dockerfile: “jboss-eap-6“
    strategy:
        type: "Source“
        sourceStrategy:
            from:
                kind: "Image“
                name: “jboss-eap-6:latest“
    output:
        to:
            kind: “Image“
             name: “myApp:latest“
 triggers:
     - type: "GitHub“
       github:
           secret: "secret101“
     - type: "ImageChange“
```

```
# oc start-build myApp-build
```



redhat.

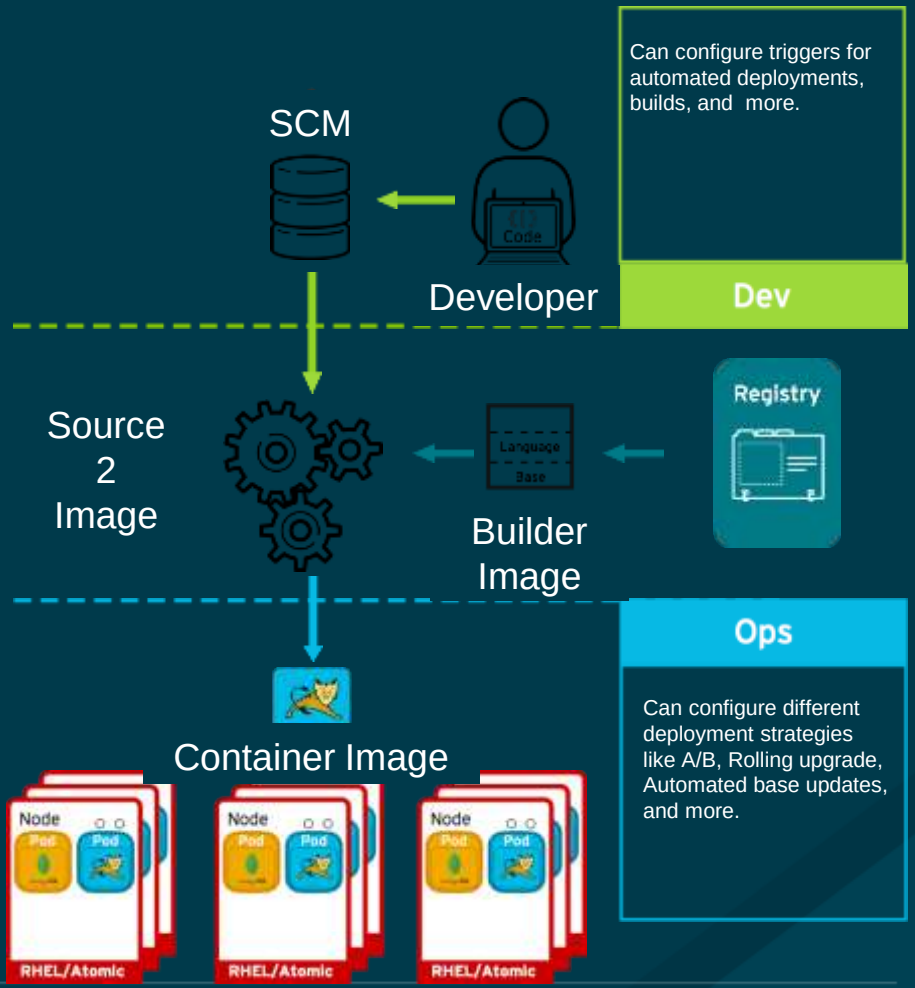
Build & Deploy an Image

Builder Images

- Jboss-EAP
- PHP
- Python
- Ruby
- Jenkins
- Customer
  - C++ / Go
  - S2I (bash) scripts

Triggers

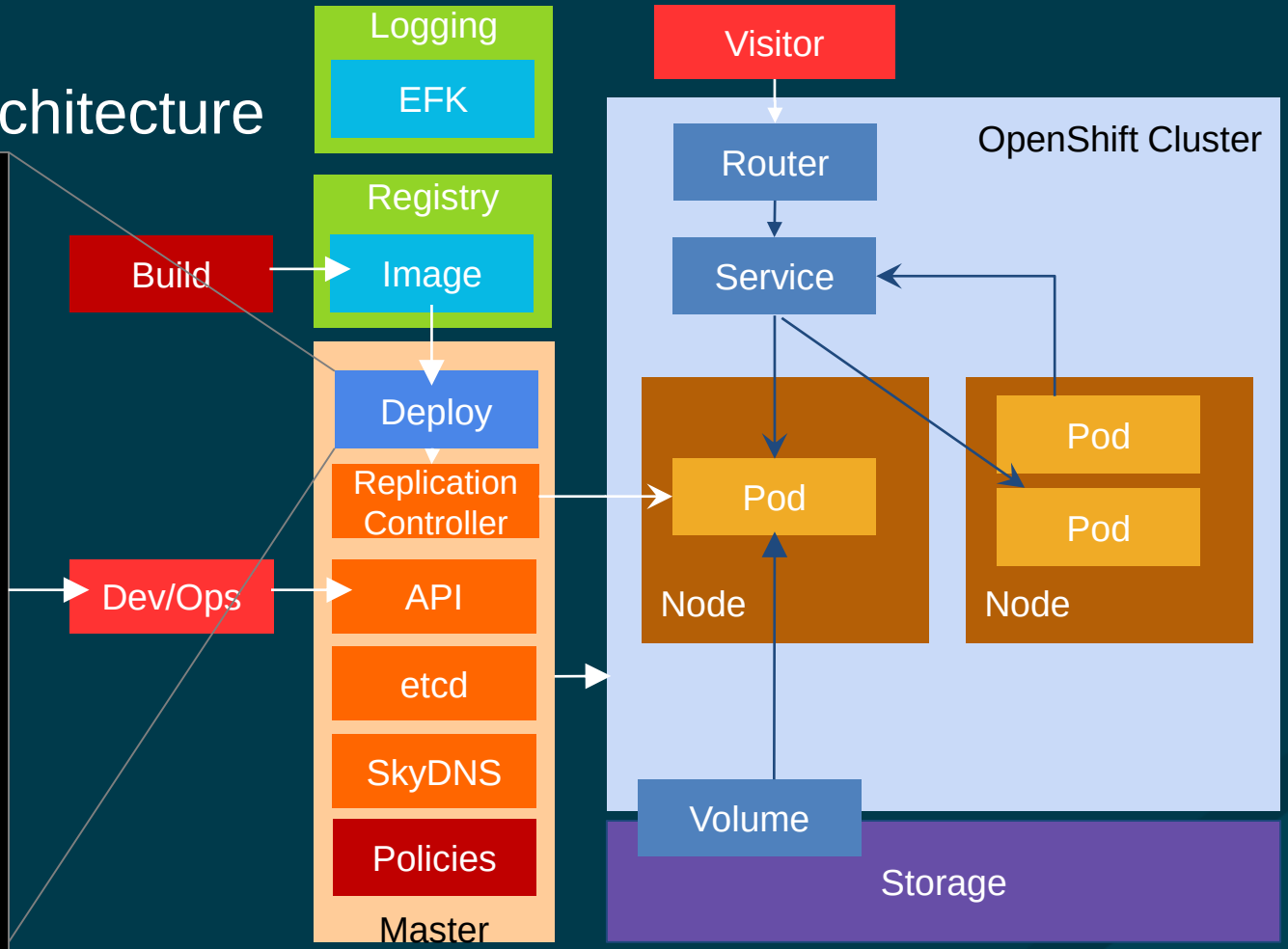
- Image Change (tagging)
- Code Change (webhook)
- Config Change



OpenShift
Build & Deploy Architecture

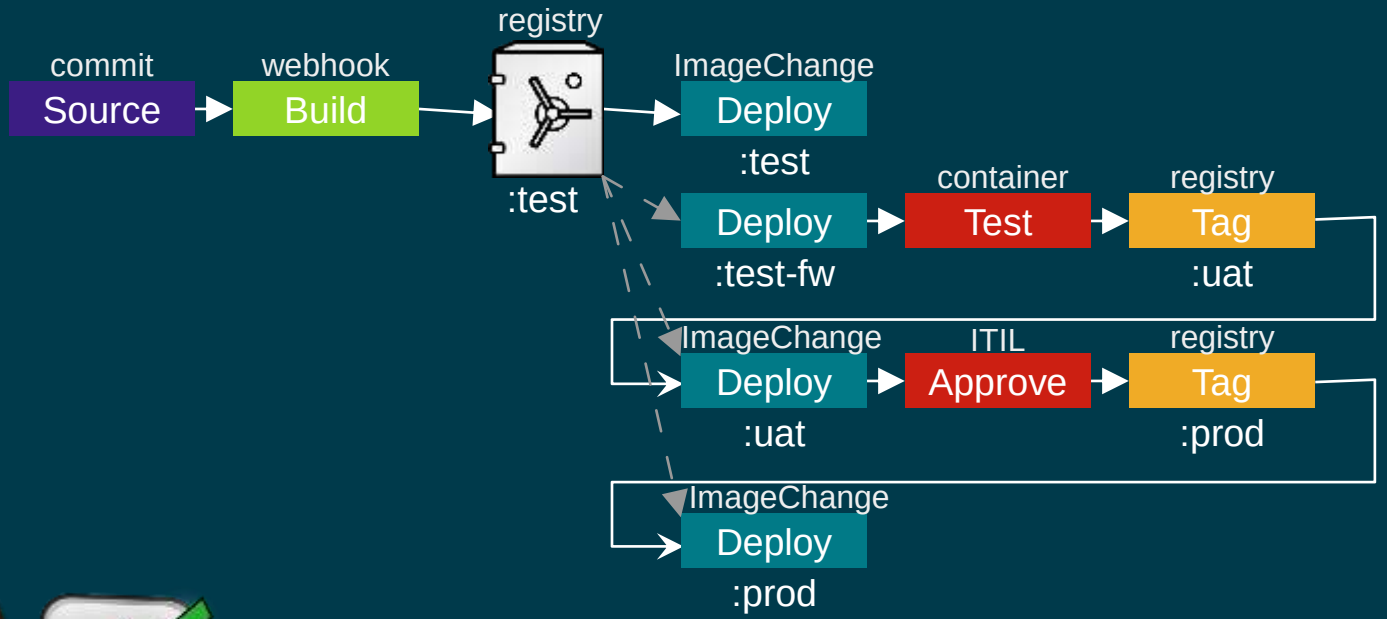
```
kind: “DeploymentConfig“
metadata:
    name: “myApp“
spec:
    replicas: 2
    selector:
        app: nginx
    template:
        metadata:
            name: nginx
            labels:
                app: nginx
        spec:
            containers:
             - name: nginx
               image: nginx:latest
               ports:
                - containerPort: 80
    triggers:
    - type: "ImageChange“
      from:
          kind: “Image”
          name: “nginx:latest
```

```
# oc deploy myApp --latest
```



redhat.

Continuous Integration Pipeline example



Template

```
apiVersion: v1
kind: List
Items:
  - apiVersion: v1
    kind: Pod
    …
  - apiVersion: v1
    kind: Service
    …
```

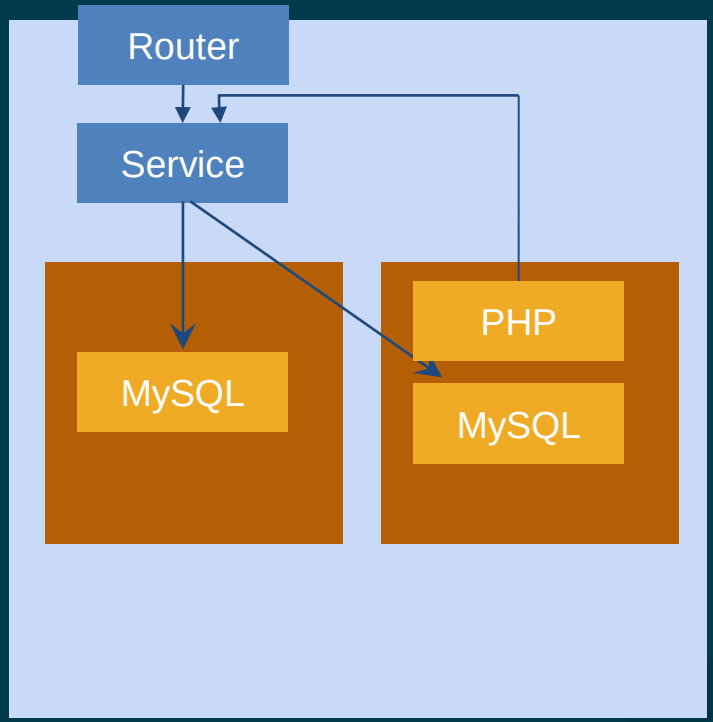
Kubernetes

```
curl –s https://get.helm.sh | bash

helm update
helm search redis
helm install   redis
```

Deis Helm

```
apiVersion: v1
kind: Template
metadata:
  name: redis-template
  annotations:
    description: "Description“
    iconClass: "icon-redis“
    tags: "database,nosql“
objects:
  - apiVersion: v1
    kind: Pod
    …
parameters:
  - description: Password
    from: '[A-Z0-9]{8}‘
    generate: expression
    name: REDIS_PASSWORD
labels:
  redis: master
```

OpenShift




Demo
redhat.

Setup

```
yum install docker-engine

docker run openshift/origin

curl –s https://get.helm.sh | bash
helm update
```

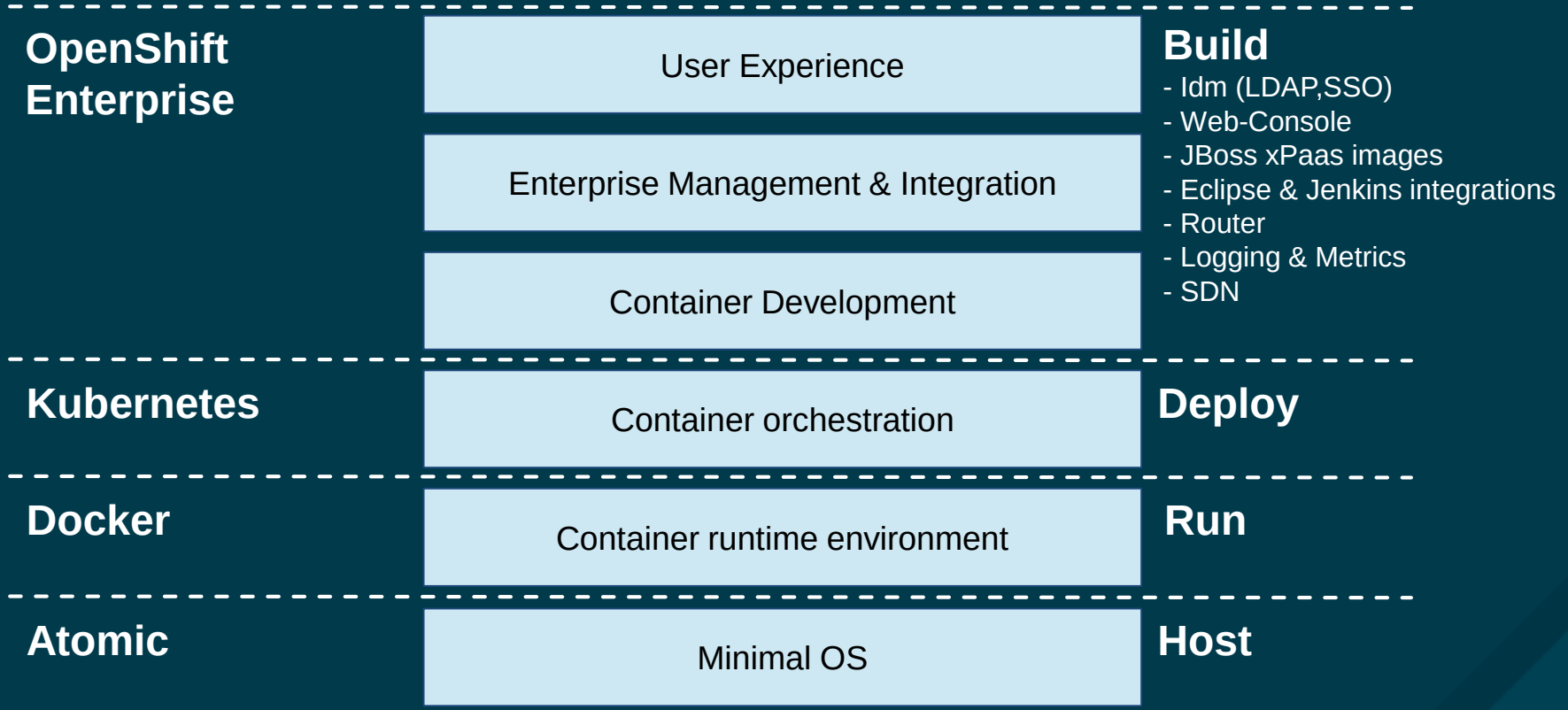
Setup

```
yum install docker-engine

docker run -d --name "ose" --privileged --net=host --pid=host \
        -v /:/rootfs:ro \
        -v /var/run:/var/run:rw \
        -v /sys:/sys:ro \
        -v /var/lib/docker:/var/lib/docker:rw \
        -v /var/lib/origin/openshift.local.volumes:/var/lib/origin/openshift.local.volumes:z \
        -v /var/lib/origin/openshift.local.config:/var/lib/origin/openshift.local.config:z \
        -v /var/lib/origin/openshift.local.etcd:/var/lib/origin/openshift.local.etcd:z \
        openshift3/ose start \
         --master="https://${OSE_MASTER_IP}:8443" \
         --etcd-dir="/var/lib/origin/openshift.local.etcd" \
         --hostname=`hostname` \
         --cors-allowed-origins=.*

curl –s https://get.helm.sh | bash
helm update
```

Setup Client

```
docker run –entrypoint=cat openshift/origin /usr/bin/oc >/usr/local/bin/oc

ln –s /var/lib/origin/openshift.local.config/admin.kubectl ~/.kubectl
```

OpenShift's Added Value



Our JBoss Middleware xPaas Service Catalog

Application Container Services

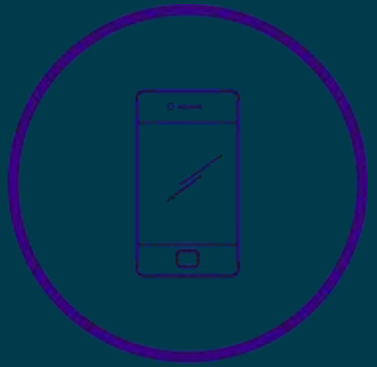
- JBoss EAP
- JBoss Web Server / Tomcat
- JBoss Developer Studio

Integration Services

- Fuse
- A-MQ
- Data Virtualization

Business Process Services

- Business Process Management *
- Business Rules Management System

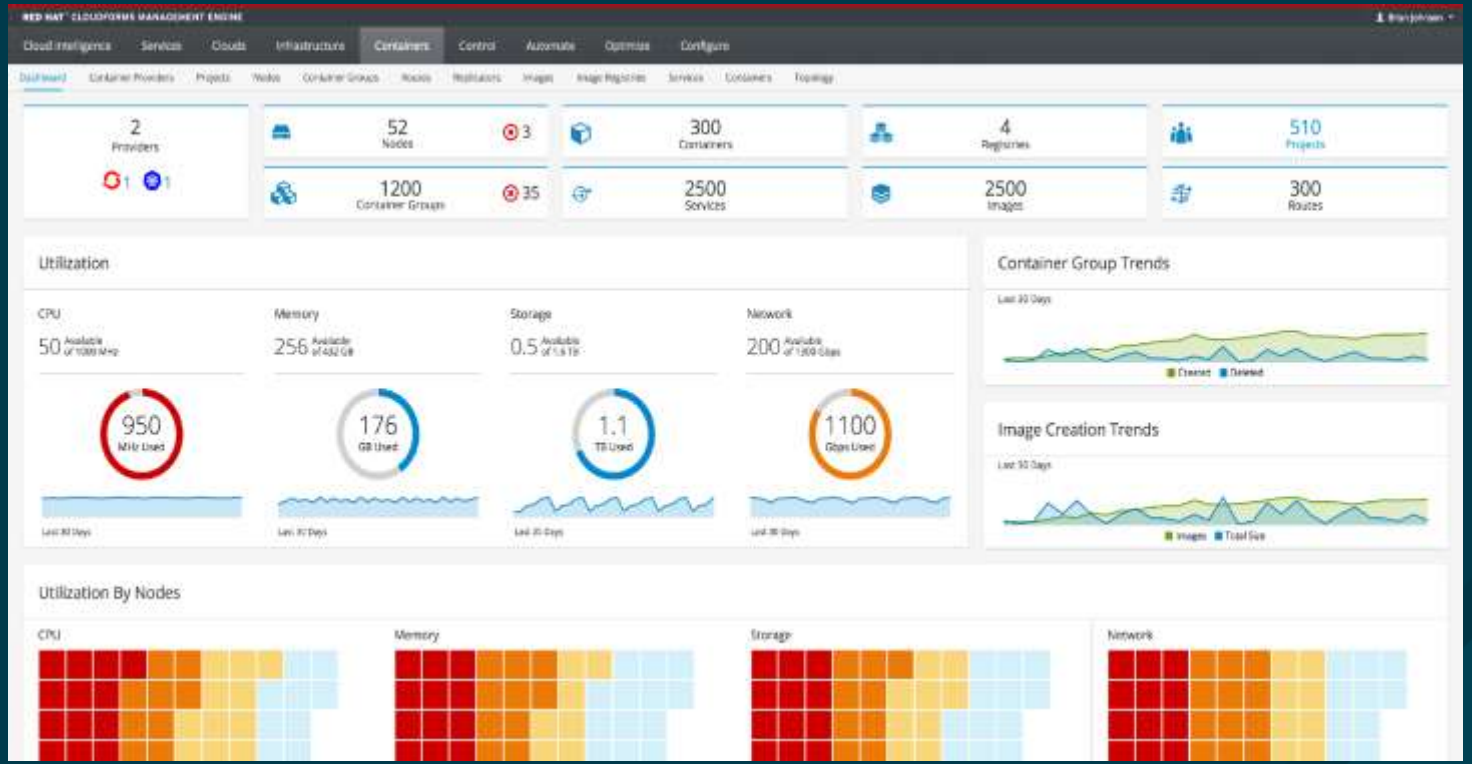


Mobile Services

- Red Hat Mobile / FeedHenry *

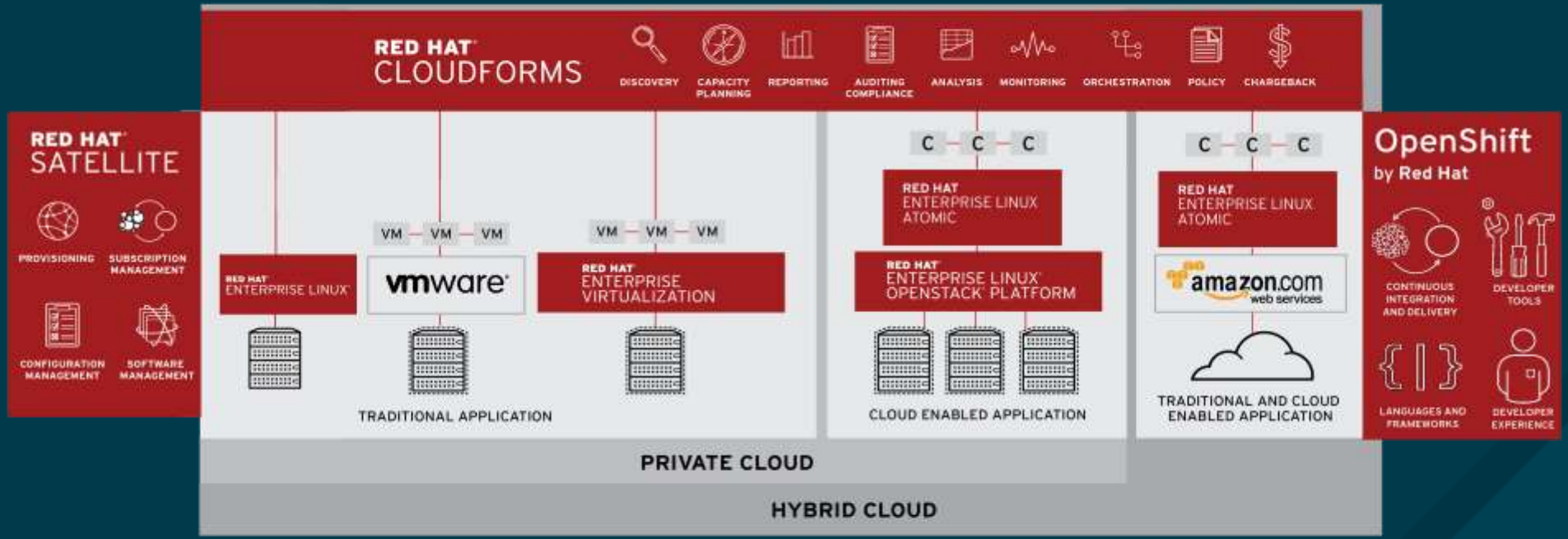
* Coming Soon

CloudForms Management



RED HAT CLOUD SUITE FOR APPLICATIONS

Cloud Management – Alternative Virtualization – OpenStack – Containers – Development



Questions?


plus.google.com/+RedHat


facebook.com/redhatinc


nl.linkedin.com/in/samuelterburg


twitter.com/SamuelTerburg


youtube.com/user/RedHatVideos