



# **Red Hat Community Meeting - “TechLab”**

Docker, Kubernetes & OpenShift

Ingo Börnig, Solution Architect

Sebastian Faulhaber, Solution Architect

Roland Wolters, Solution Architect

2015-11-09

# AGENDA

Red Hat Community Meeting (“Tech Lab”) Düsseldorf

**15:45 -- 16:15**

Container und Container Demo

**16:15 -- 16:45**

Orchestrierung mit Kubernetes, Demo

**16:45 -- 17:30**

Developer Workflow mit OpenShift, Demo

**17:30 -- ???**

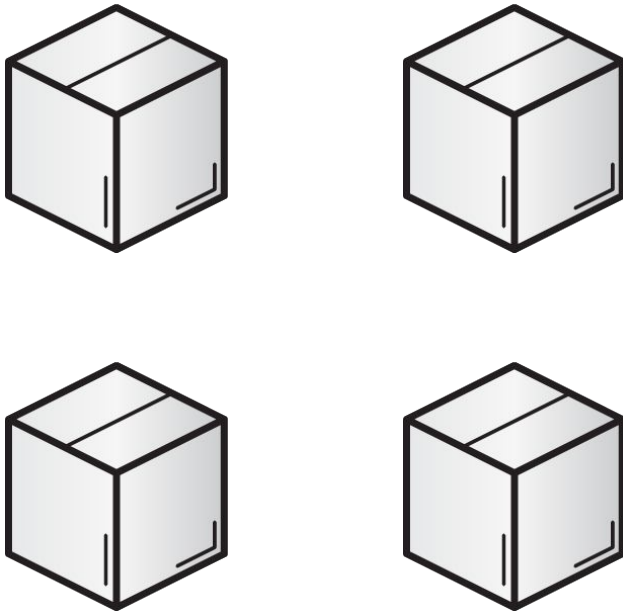
Pause, Snacks

# Problem: handle different goods

different goods  
+ different packaging  
= high transportation costs



# Solution: uniform packaging



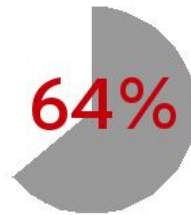
different goods  
+ same packaging  
= low transportation costs

# Container

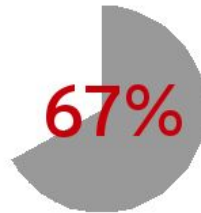


# Container are already there!

64 % already use containers  
or evaluate using them in  
the future



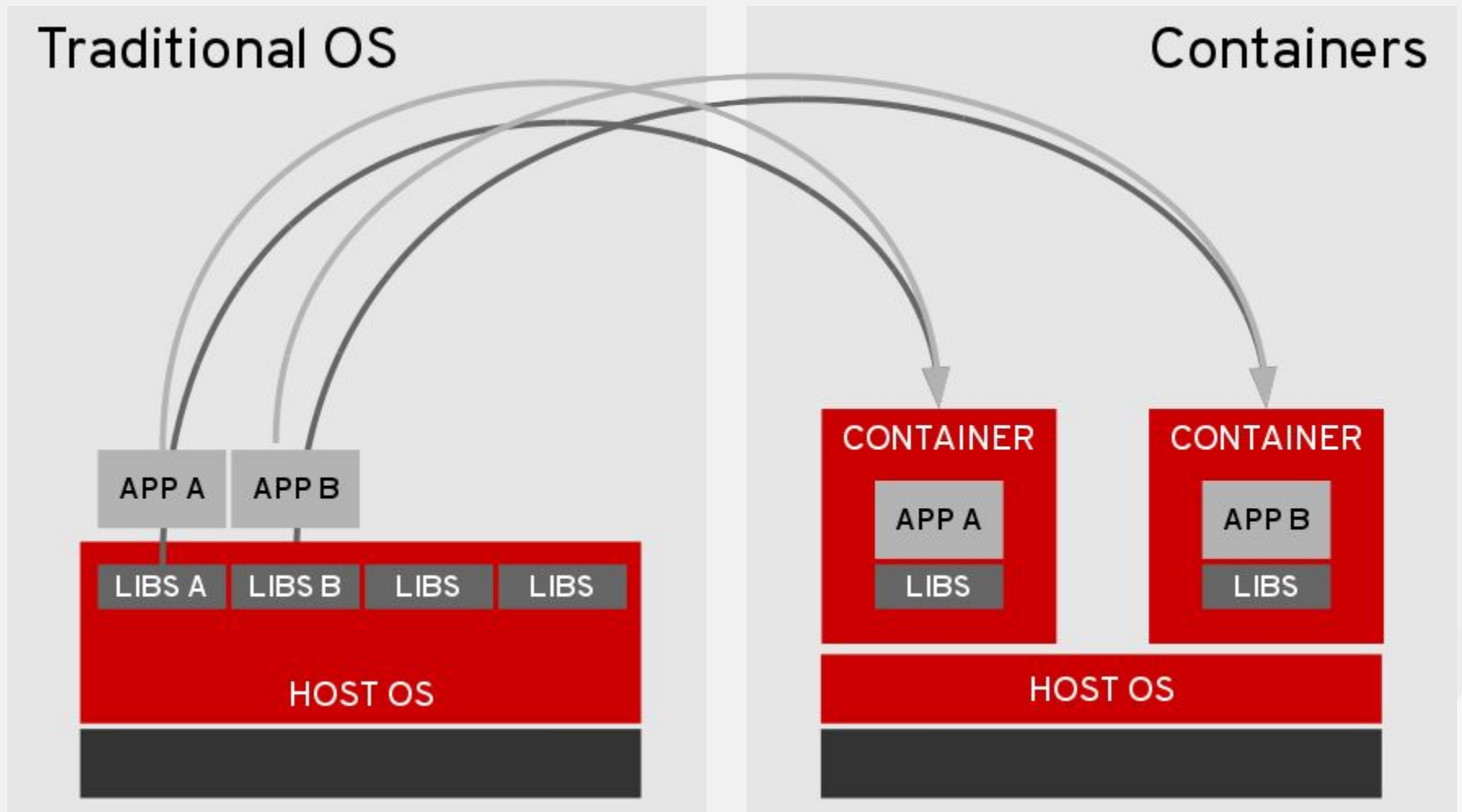
67 % plan productive use in  
the next two years



Google starts over 2  
billion containers each  
week

Source: TechValidate survey of 383 global IT decision makers and professionals

# What are Linux Containers?

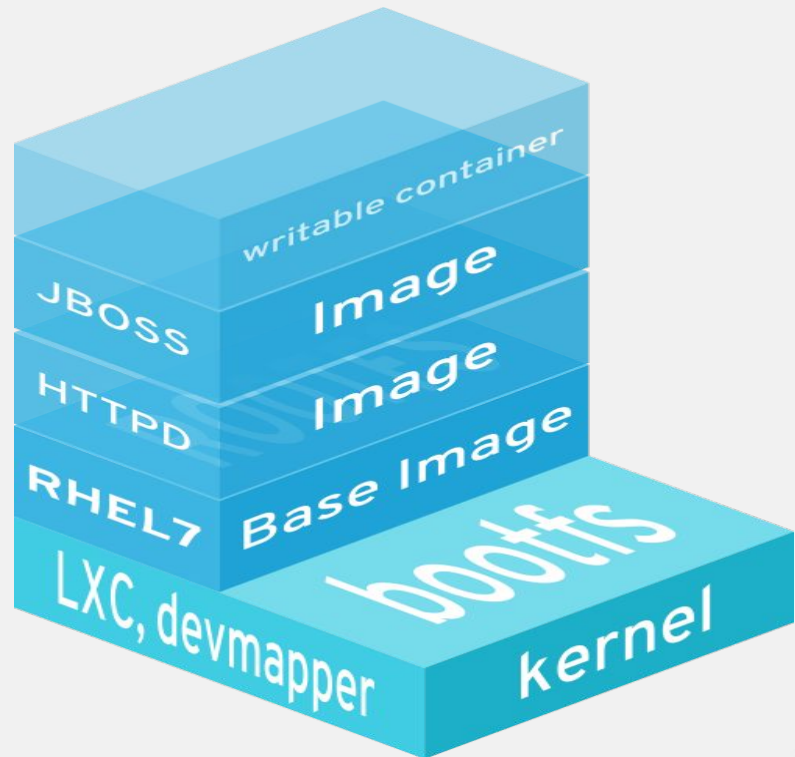




# Container, Docker-Style

aka “Open Container project”

- In the Docker world, a container is a running instance of an *image*
- Based on linux containers (namespaces, control groups)
- A file system layer cake
- Includes all of the components necessary to run a process





# Docker-Tools

- CLI-Tool: more like Git
- `pull` & `push`
- versions, tags
- `diff` of images and running containers possible



DEMO

# What docker does:

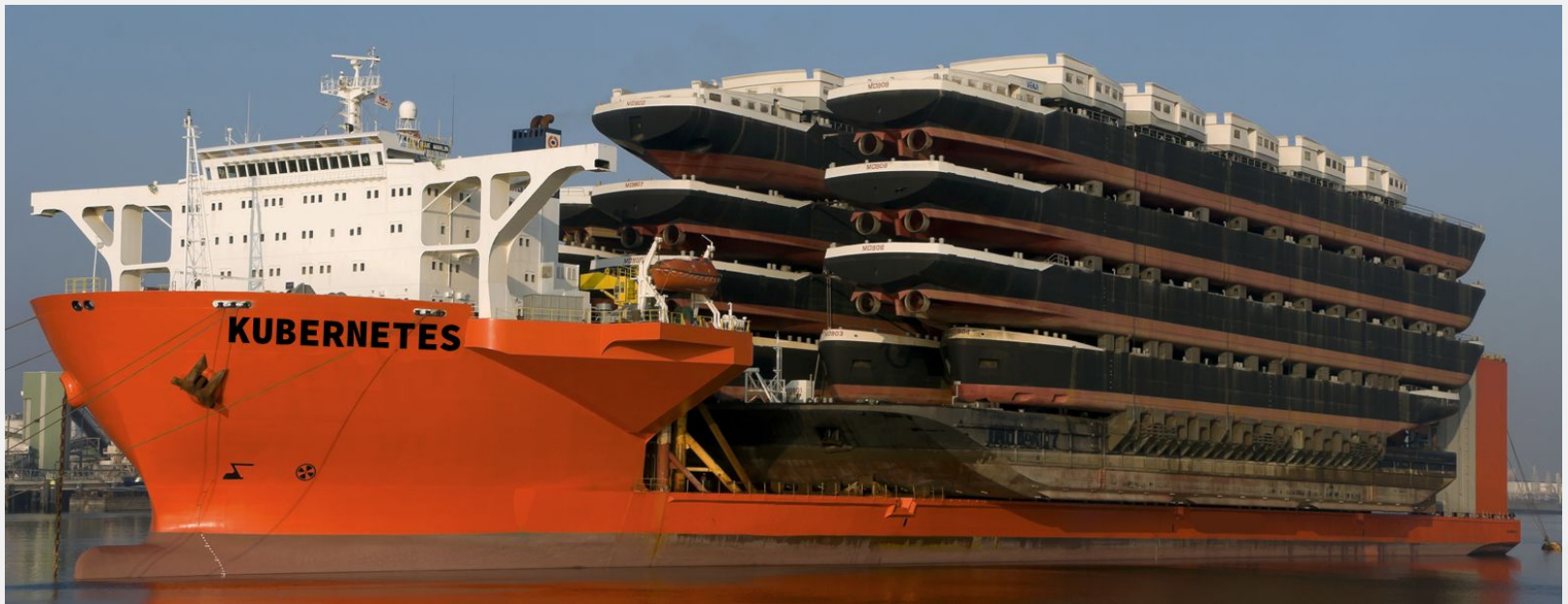
- Portability
- Workflow
- Easy
- Speed

# Docker doesn't:

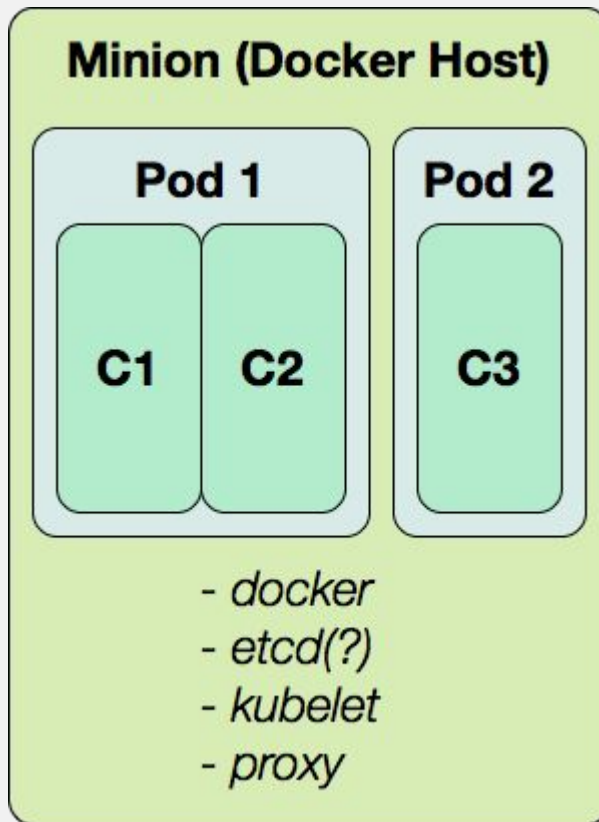
- See beyond a single host
- Provision related containers as a unit
- Have capacity for handling mass configuration & deployment

# Component #2

Kubernetes



# Kubernetes Terminology



**Pod:** One or more interrelated Docker containers

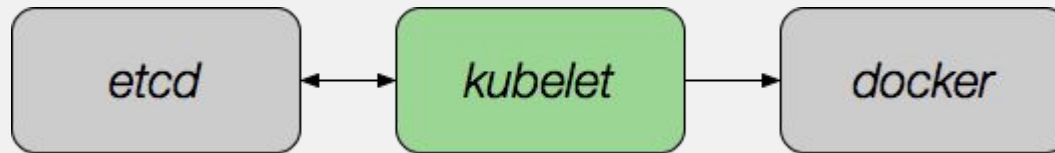
**Service:** A configuration unit for the kube-proxy

**Label:** Used with pods to specify identifying metadata

**Master:** Runs cluster-level control plane services

**Minion/Node:** A docker host running the kubelet and the proxy service

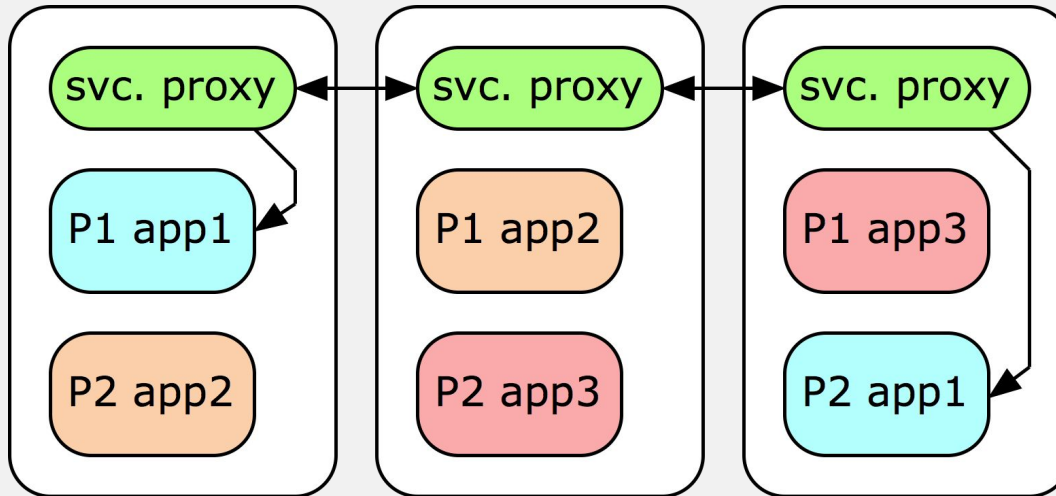
# Minion Daemon: kubelet



- Pod management
- Take instructions from cluster master



# Minion Daemon: kube-proxy



- The proxy service maps a common port on *every minion* to relevant pods *across the entire cluster*
- Relevant pods are chosen by comparing a *label* on the proxy definition to labels on the running pods

# Cluster Management

## Cluster Master

- *apiserver + nginx*
- *k8s-scheduler*
- *controller-manager*
- *etcd(?)*
- *kubecfg.sh*

**Kubernetes API:** RESTful API for Kubernetes

**Scheduler:** Choose minions for pods

**Controller Manager:** Monitoring service for deployed pods

**kubecfg/oc:** CLI for working with a Kubernetes cluster

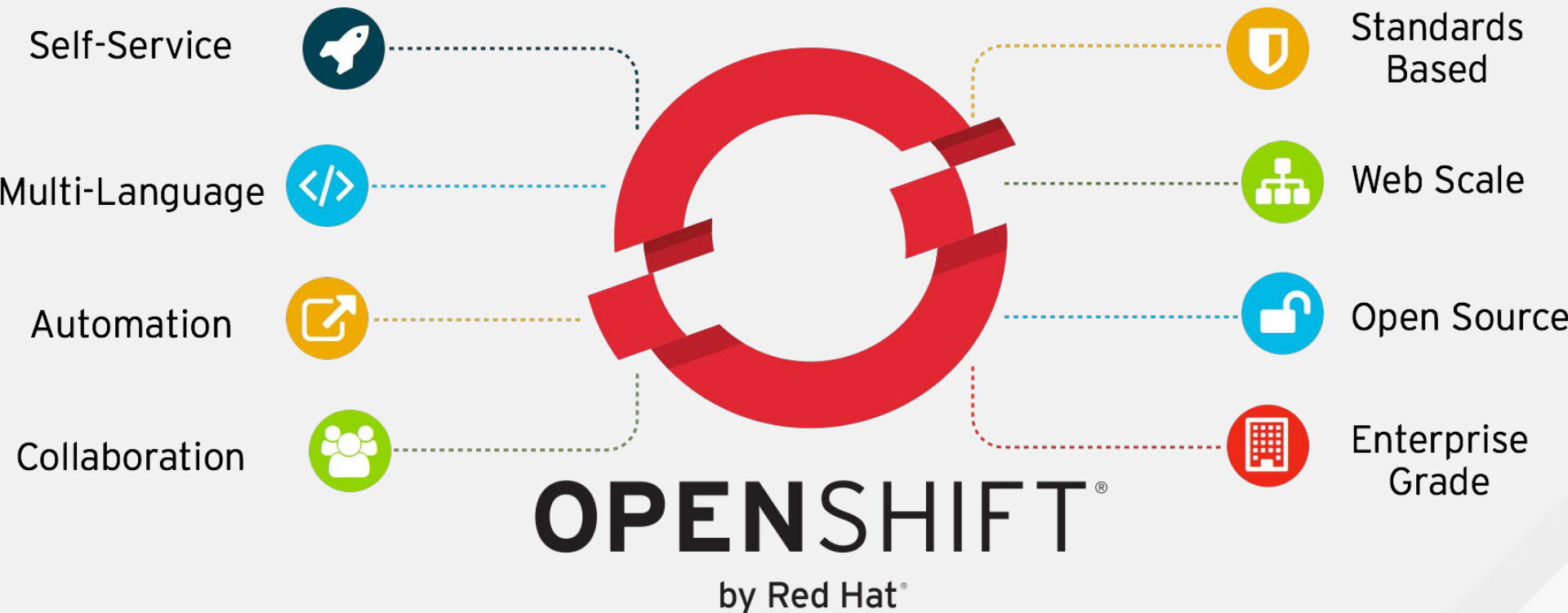


DEMO

# Kubernetes doesn't:

- Have a concept of a complete *application*
- Have capacity for building and deploying Docker images from source code
- Have a focus on a user or admin *experience*

# OpenShift – The runtime platform for containers





# OpenShift 3



**DEVOPS TOOLS & USER EXPERIENCE**

**LANGUAGE RUNTIMES, MIDDLEWARE,  
DATABASES AND OTHER SERVICES**

**CONTAINER ORCHESTRATION & MANAGEMENT**

**CONTAINER API**

**CONTAINER HOST**

- Standard containers API
- Web-scale container orchestration & management
- Container-optimized OS
- Largest selection of supported application runtimes & services
- Robust tools and UX for Development & Operations
- Industry standard, web scale distributed application platform

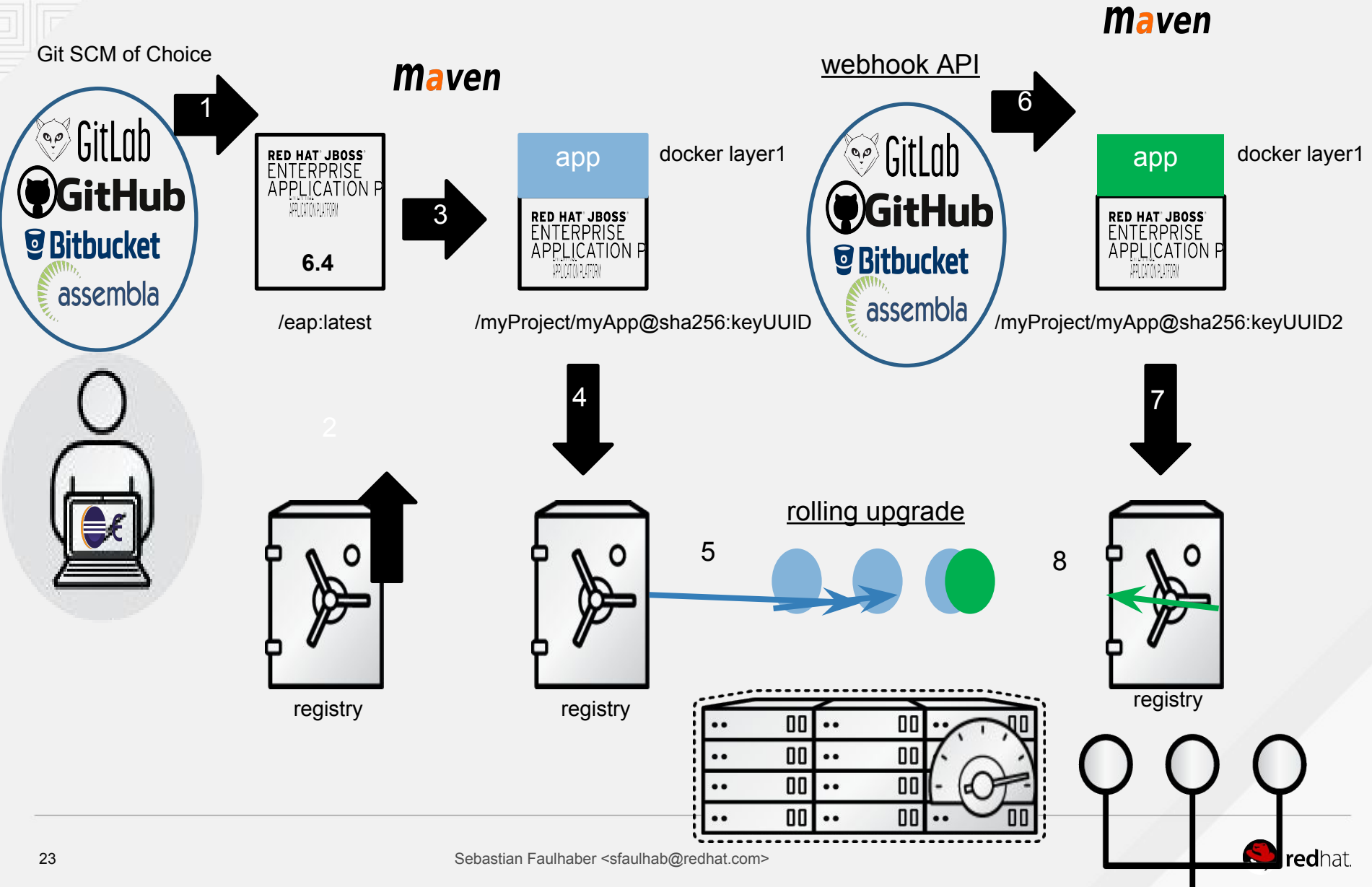
# OpenShift Application Services



- From Red Hat
- From ISV Partners
- From the Community



# SOURCE 2 IMAGE (S2I) / DEVOPS EXPERIENCE



# DOCKER BUILDS

## Integrated Docker Builds

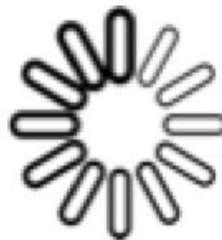
**Developer**



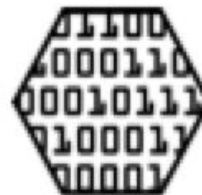
**Dockerfile**



**Build**



**Image**



**Deploy**



# SELF SERVICE FOR DEVELOPERS

Check out the video at:

<https://youtu.be/2yClzJJFBVw>

OPENSIFT ORIGIN

Documentation test

Projects: sample Filter by labels: Label key Add Create...

Overview Project sample Details

Browse

Settings

SERVICE: database routing traffic on 172.30.110.5 - port 5434 -> 3306 (TCP)

DEPLOYMENT: database, #1 created an hour ago, triggered by a config change

POD TEMPLATE

ruby-helloworld-database

Image: openshift/mysql-55-centos7:latest

Ports: 3306 (TCP)

PODS

Running 172.17.0.25

SERVICE: FRONTEND routing traffic on 172.30.107.208 - port 5432 -> 8080 (TCP)

DEPLOYMENT: frontend, #3 created 4 minutes ago, triggered by a new image for origin-ruby-sample:latest

POD TEMPLATE

ruby-helloworld

Image: sample/origin-ruby-sample (origin-rub)

Build: ruby-sample-build (ruby-sample-build 2-1432919588)

Source: Git

Ports: 8080 (TCP)

PODS

Running 172.17.0.46

Running 172.17.0.47

ReplicationController

Name: database-1

Namespace: sample

Created: May 29, 2015 11:57:55 AM

Replicas: 1

Selector

deployment: database-1

deploymentconfig: database

name: database

Pod Template

Restart policy: Always

DNS policy: ClusterFirst

Containers

Name: ruby-helloworld-database

Image: openshift/mysql-55-centos7:latest

Ports: 3306/TCP

Env vars: MYSQL\_DATABASE=root, MYSQL\_PASSWORD=mpjCfdse, MYSQL\_USER=userjOS

Volumes: none

Labels

template: application-template-stibuild

Annotations

kubecti.kubernetes.io/... 1

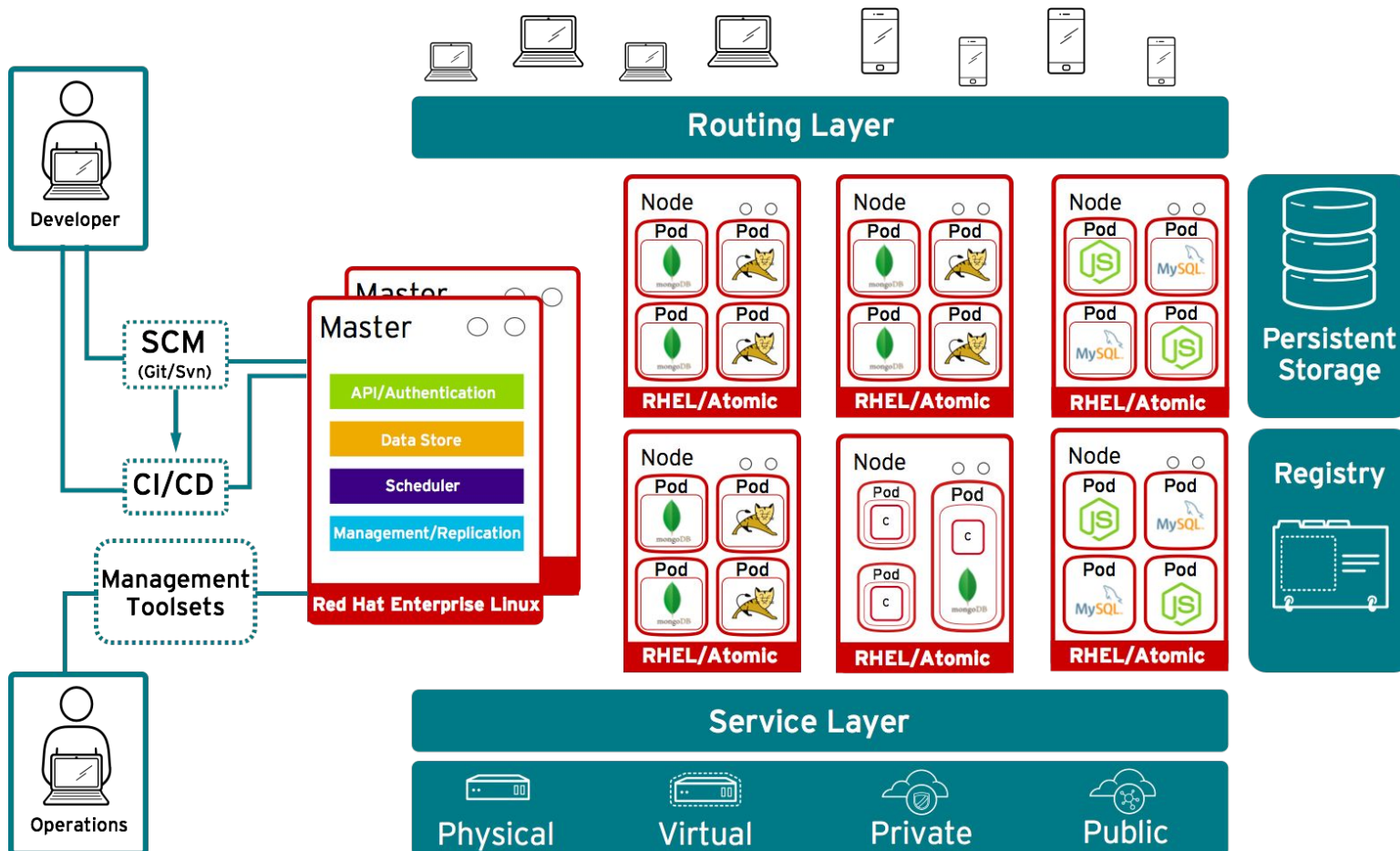
openshift.io/deployer-... database-1-deploy

openshift.io/deployme... 1

openshift.io/deployme... database

openshift.io/deployme... Complete

openshift.io/encoded-... ("kind": "DeploymentConfig", "apiVersion": "v1beta3", "metadata": {"name": "database", "namespace": "sample", "selfLink": "/osapi/...



DEMO





# THANK YOU



[plus.google.com/+RedHat](https://plus.google.com/+RedHat)



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



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



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



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