CONTINUOUS DELIVERY IN DOCKERLAND

Entwicklertag Karlsruhe 2017

Nicolas Byl, codecentric AG

GETTING STARTED

Mission Statement

- You can follow the pipeline on your device.
- Install prequisites

Prequisites

https://github.com/nbyl/cd-workshop-demo

Organisational Stuff

- Ask questions anytime!
- Breaks?

I AM GROOT
THIS IS K8S!
memegenerator.net

CONTINUOUS DELIVERY

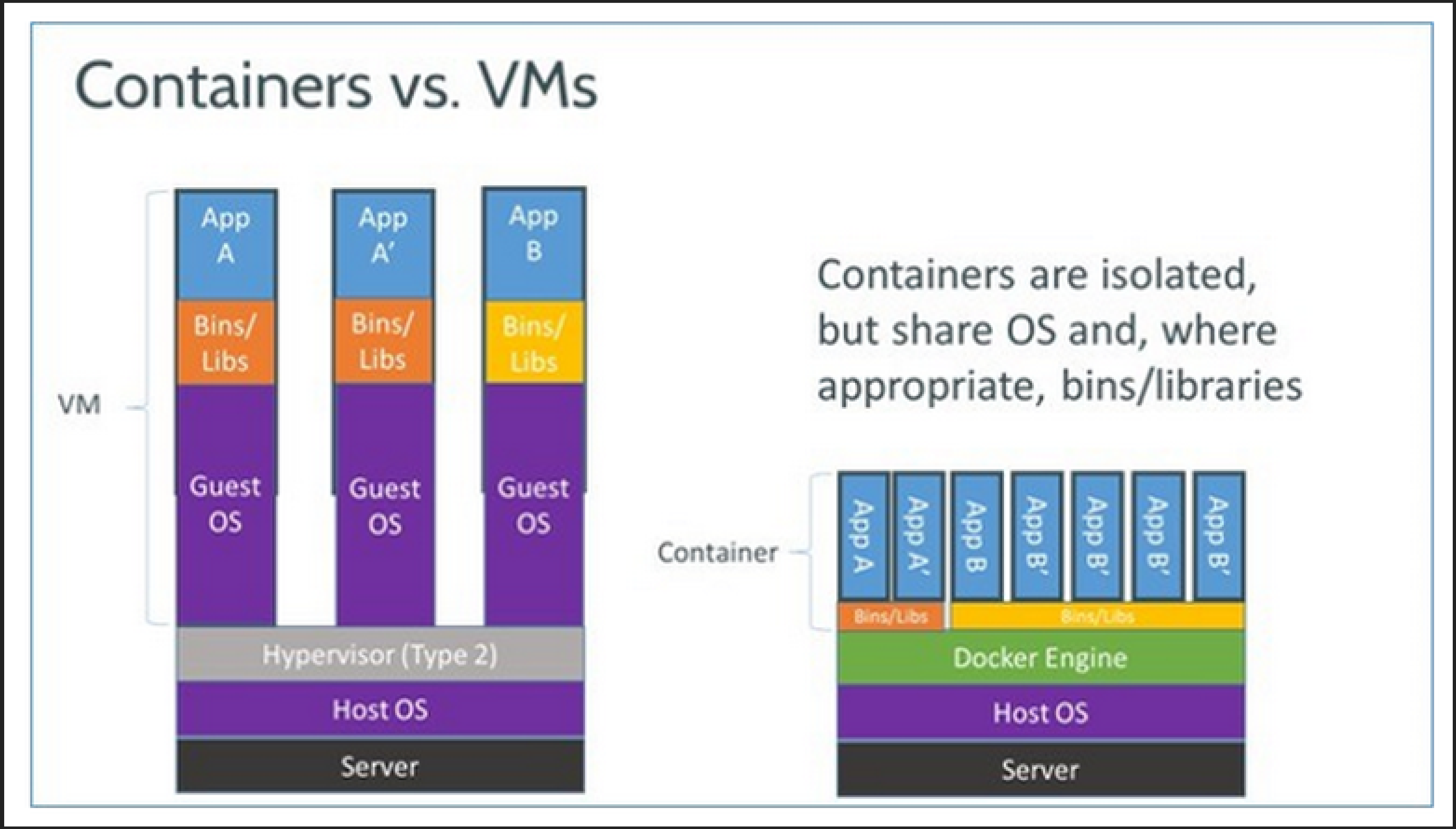
Key Concepts

- The software is releasable at at any time
- The whole process from source to production is automated
- Decouple technical rollout from feature rollout

Continuous Delivery vs. Continuous Deployment

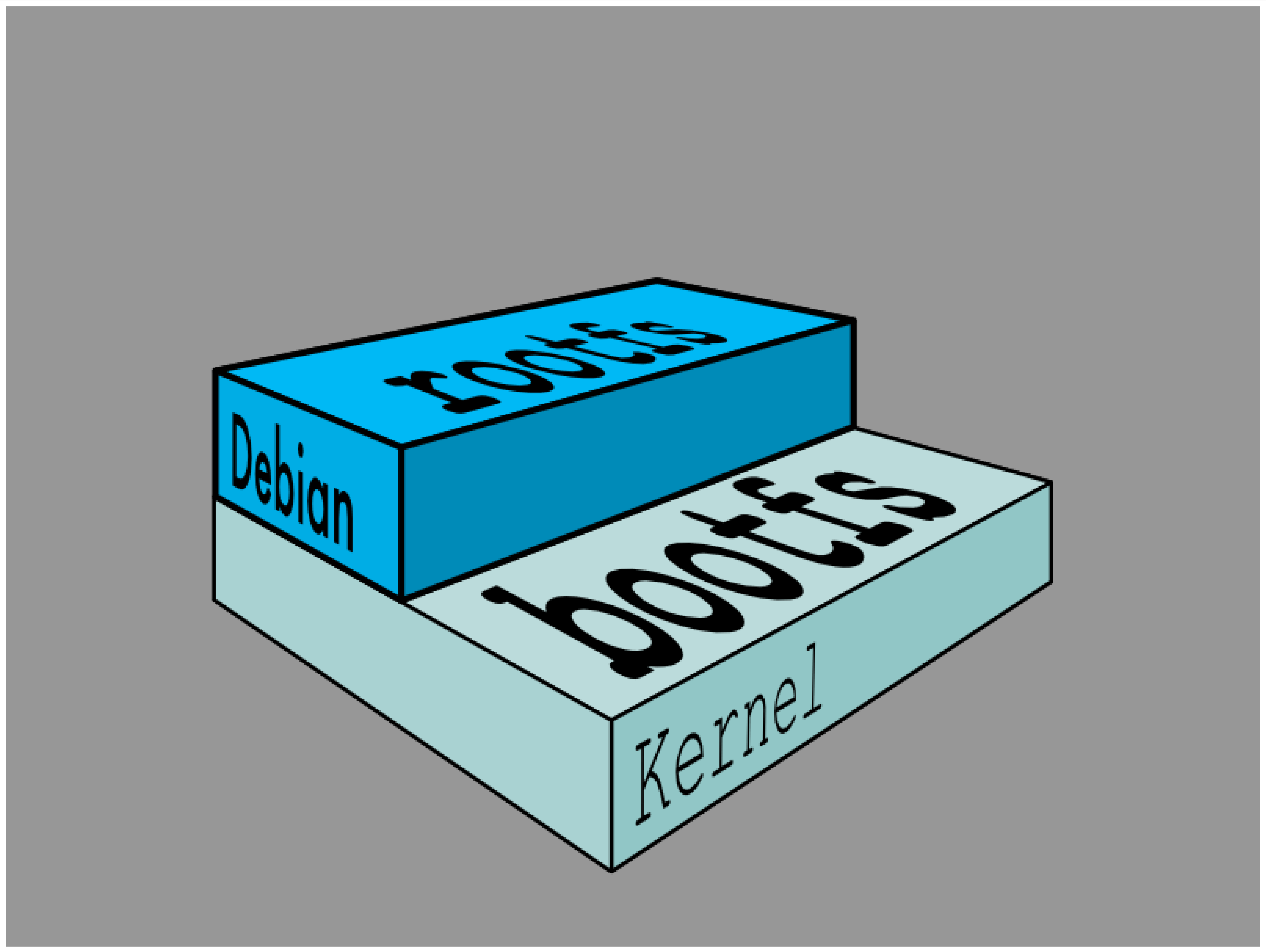
- Continuous Delivery: Software is **releasable** at any time
- Continuous Deployment: Software is **released** on every change

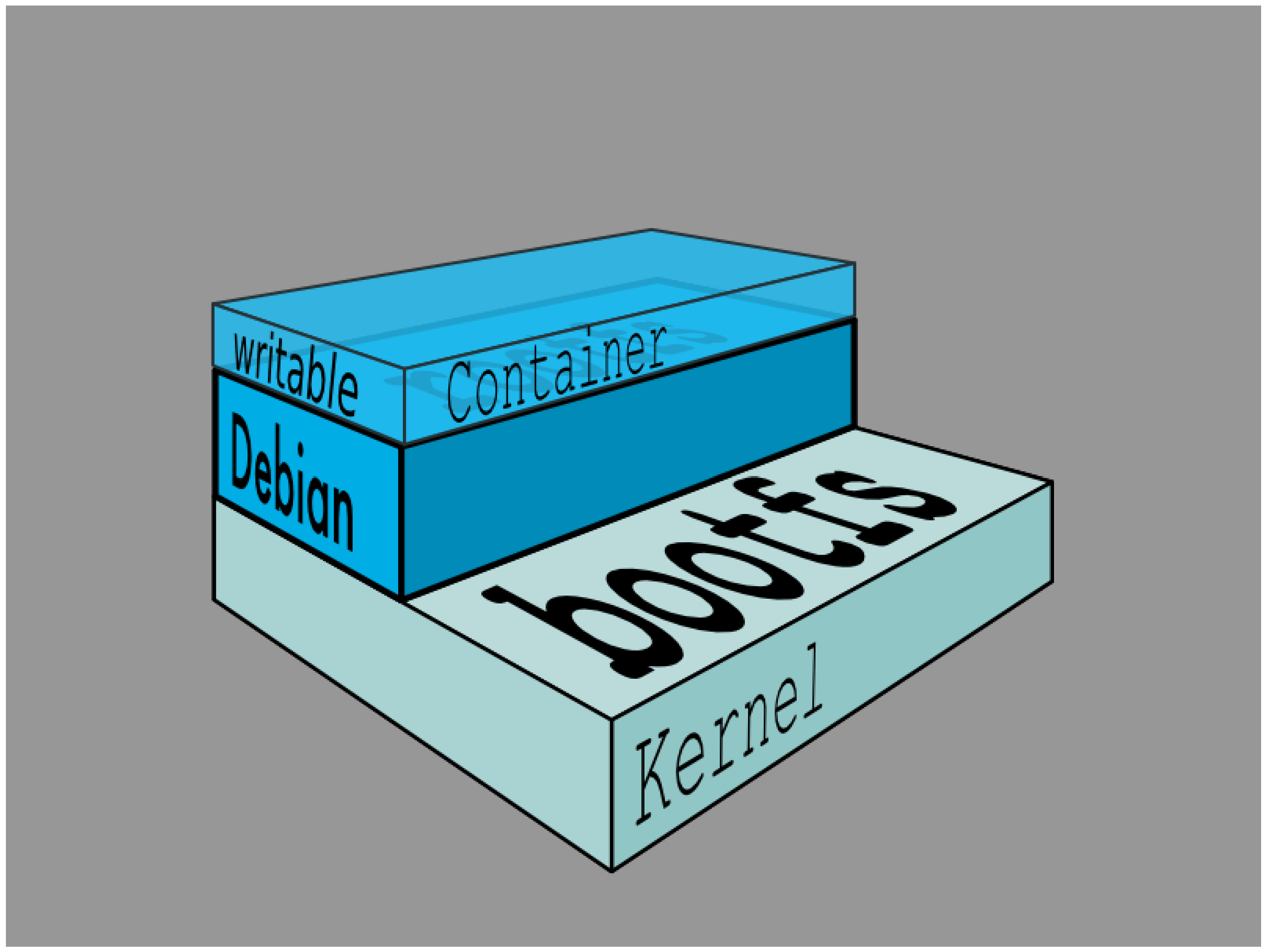
DOCKER

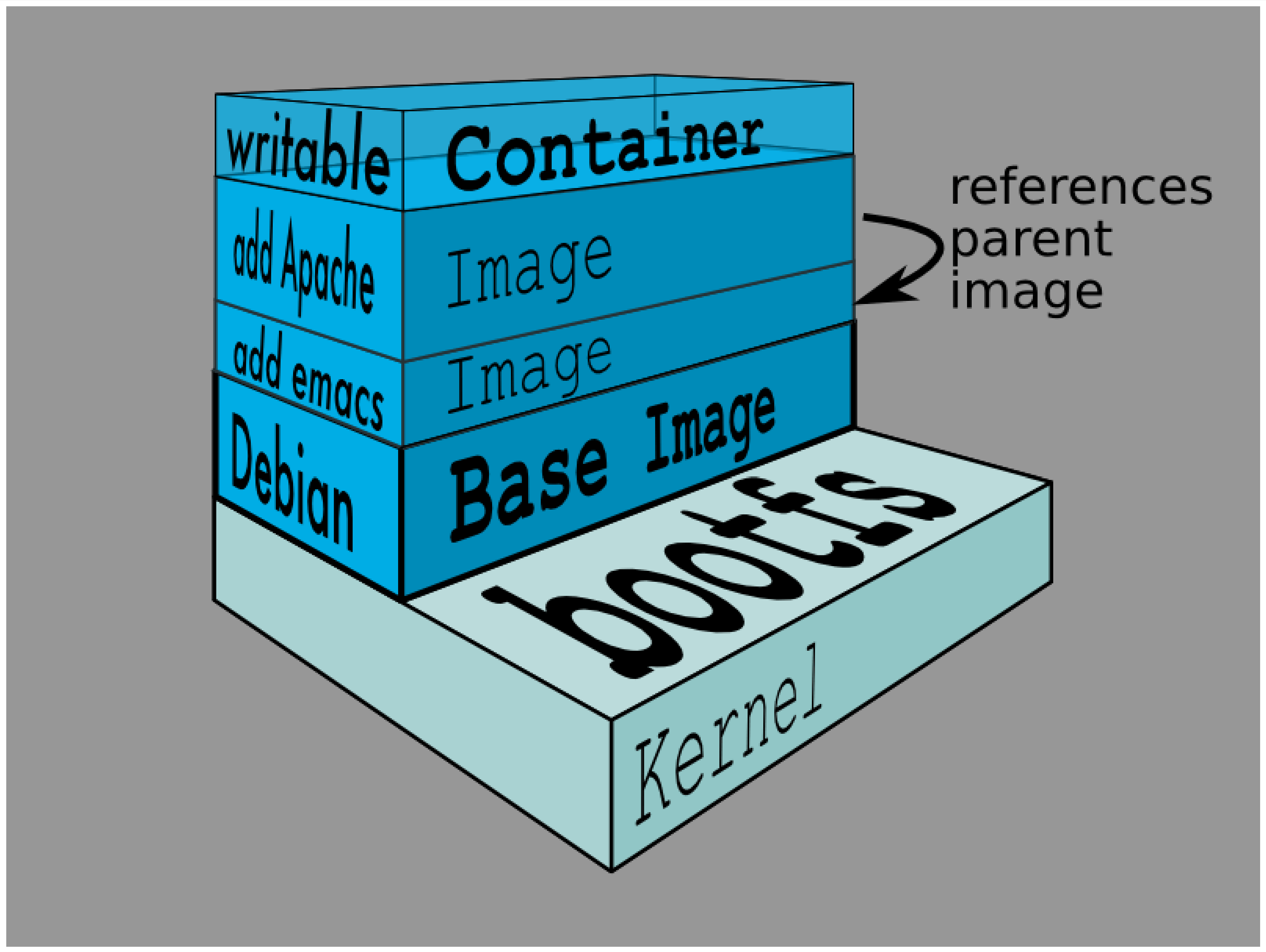


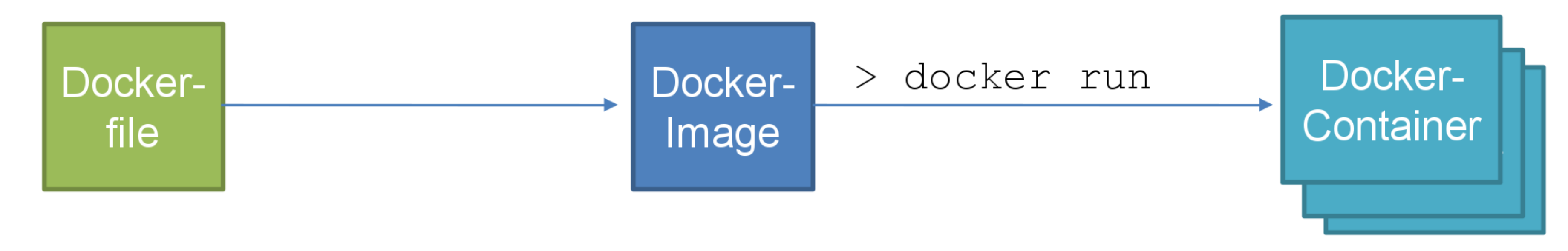
Containers vs. VMs

Containers are isolated, but share OS and, where appropriate, bins/libraries


rootfs
Debian
bootfs
Kernel


writable
Container
Debian
bootfs
Kernel


writable
Container
add Apache
Image
add emacs
Image
Debian
Base Image
references
parent
image
bootfs
Kernel



Docker-file

Docker-Image

```
> docker run
```

Docker-Container

Dockerfile

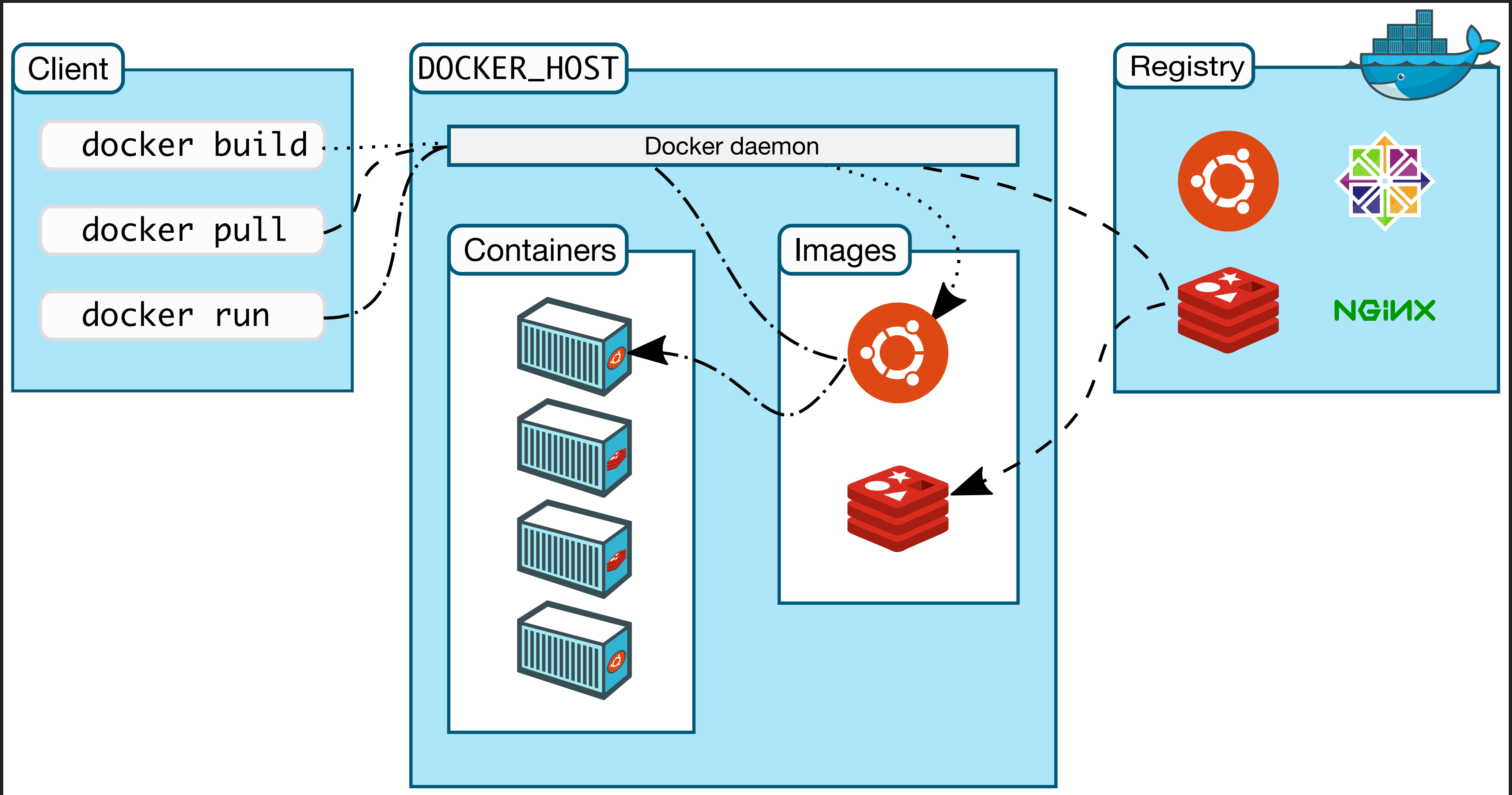
```
FROM java:8

MAINTAINER Marcel Birkner <marcel.birkner@codecentric.de>

ADD target/edmp-sample-app*.jar app.jar

RUN bash -c 'touch /app.jar'

ENTRYPOINT ["java","-jar","/app.jar"]}
```



Client

docker build

docker pull

docker run

DOCKER_HOST

Docker daemon

Containers

Images

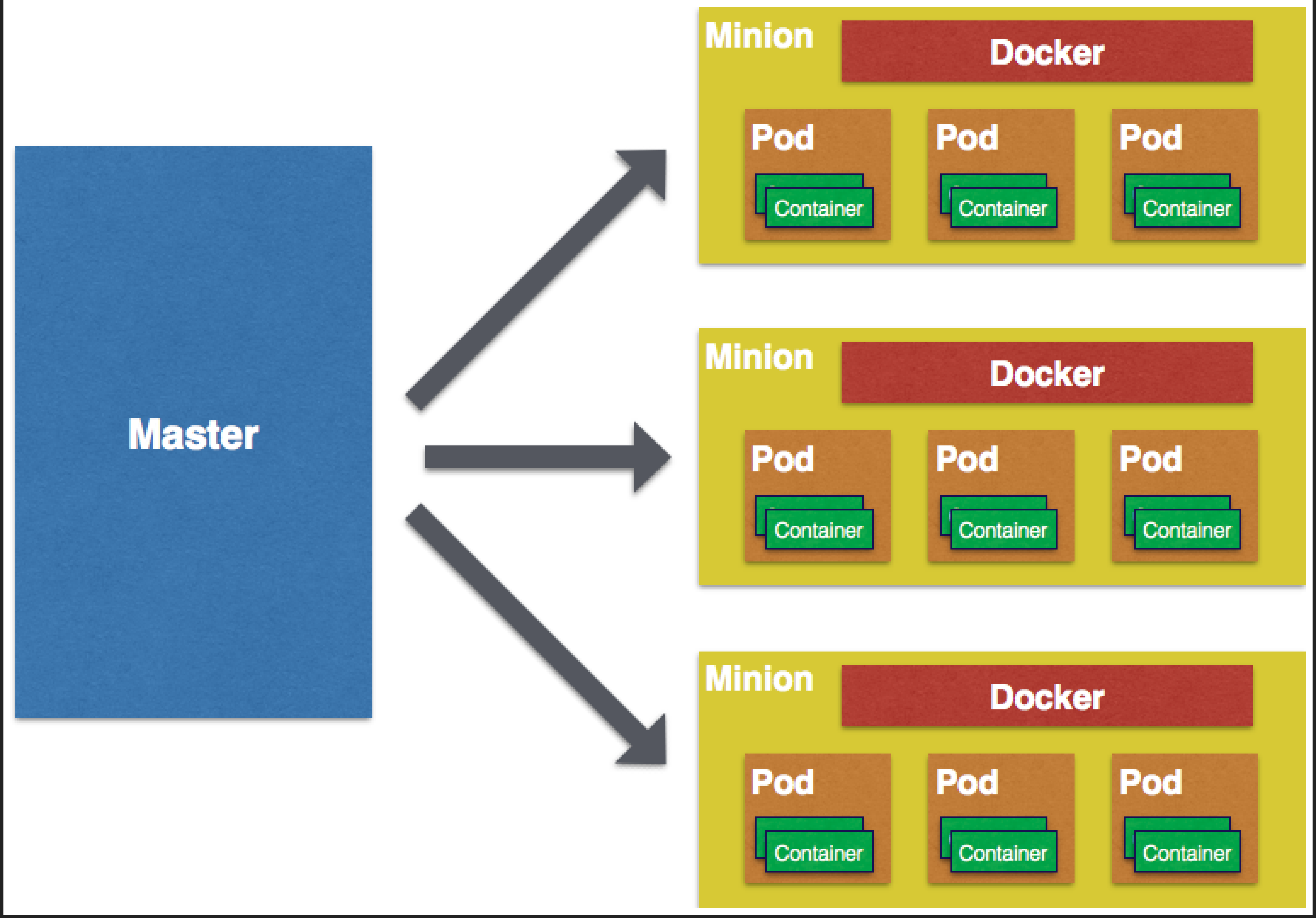
Registry

KUBERNETES

"Kubernetes is an open-source platform for automating deployment, scaling, and operations of application containers across clusters of hosts, providing container-centric infrastructure."

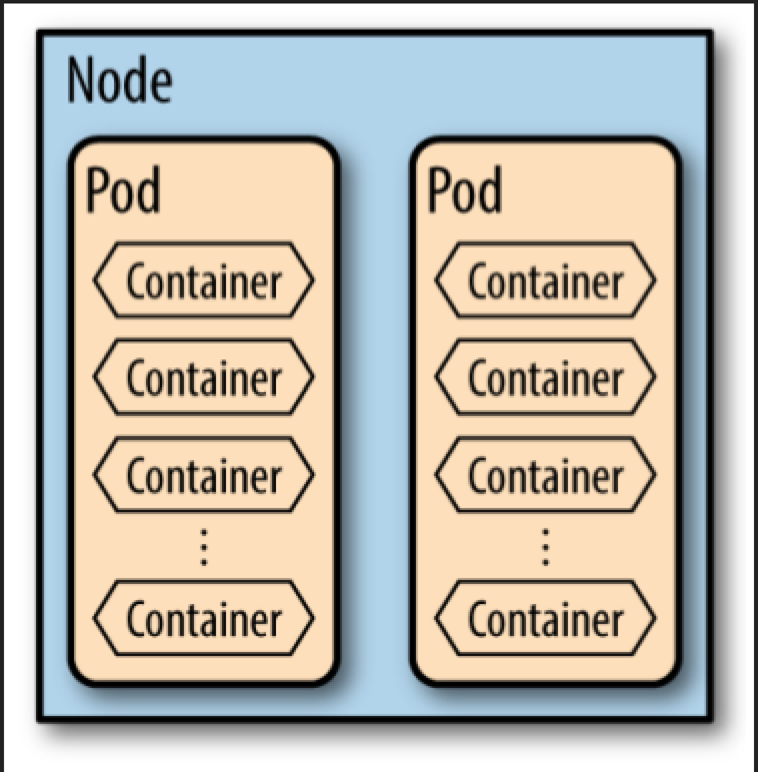
kubernetes by Google

- **portable**: public, private, hybrid, multi-cloud
- **extensible**: modular, pluggable, hookable, composable
- **self-healing**: auto-placement, auto-restart, auto-replication, auto-scaling

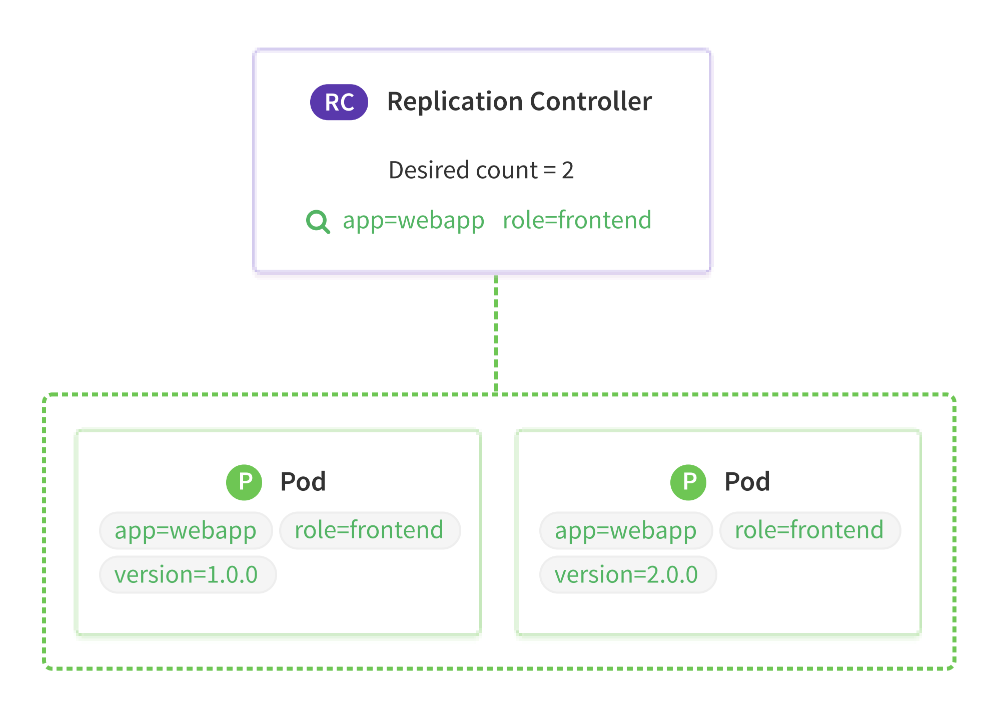

Master
Minion
Docker
Pod
Container
Pod
Container
Pod
Container
Minion
Docker
Pod
Container
Pod
Container
Pod
Container
Minion
Docker
Pod
Container
Pod
Container
Pod
Container

Pods & Co.

Pods


Node
Pod
Container
Container
Container
⋮
Container
Pod
Container
Container
Container
⋮
Container

Replication Controller



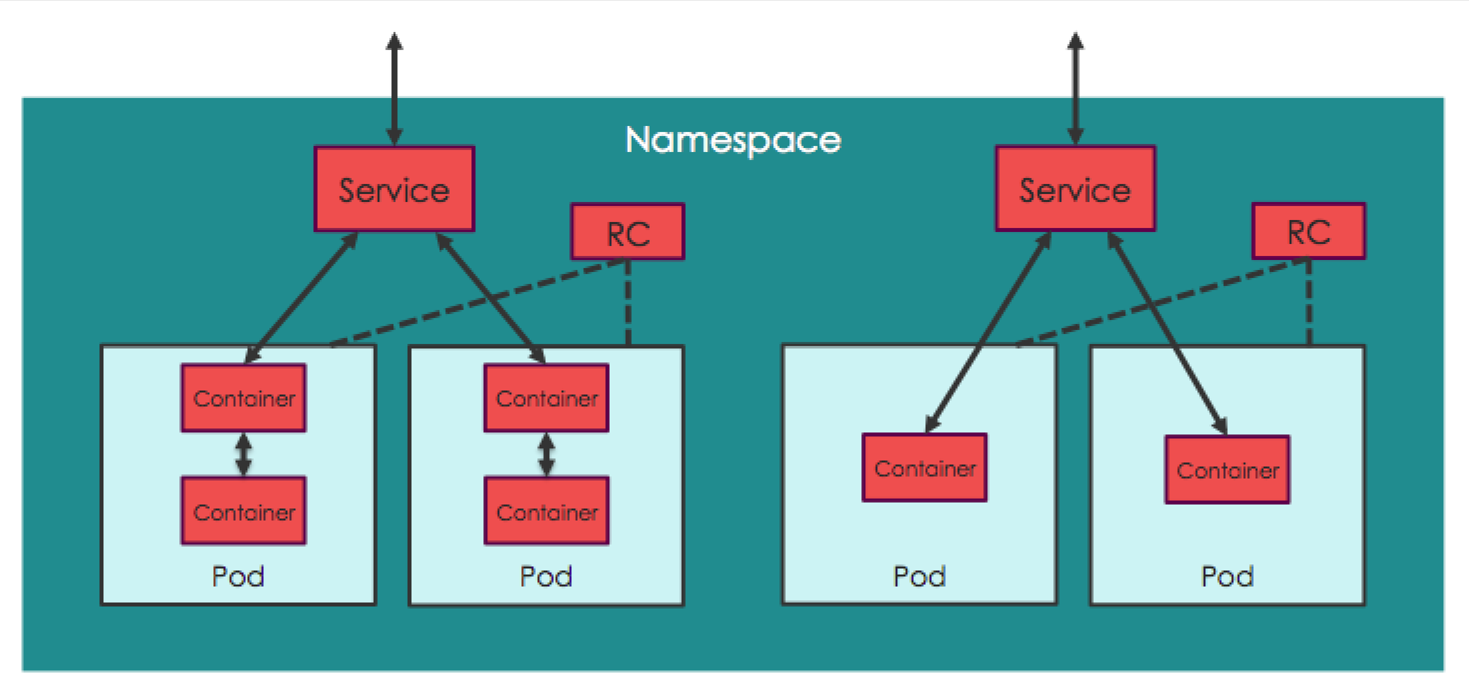
Deployment

- combination of pod & replication controller
- edited as a unit

```
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  name: frontend
spec:
  replicas: 3
  template:
    metadata:
      labels:
        app: guestbook
        tier: frontend
    spec:
      containers:
      - name: php-redis
        image: gcr.io/google-samples/gb-frontend:v4
        resources:
```

Services

- endpoint for
  - a set of pods
  - an external endpoint
- can be resolved using
  - DNS
  - environment variables


Namespace
Service
RC
Container
Container
Pod
Container
Container
Pod
Service
RC
Container
Pod
Container
Pod

SHOWCASE PROJECT

confy

- microservice to manage conference talks and speakers
- technology:
  - REST-API
  - Gradle
  - Spring Boot (Web, JPA)
  - UI using ng-admin

https://github.com/nbyl/confy

Target Platform

- kubernetes
- PostgreSQL

Testing Requirements

- automatic testing using the target database
- user acceptance test before exposing new versions

Build Pipeline

- build
- publish docker container
- integration test
- user acceptance test
- production

Assumptions

- any manual interaction will be done in reasonable period
- our cluster will (for now) contain of only one host

LAB 1: SETUP

Recently on this Program…

Have you installed the prerequisites?

Create a cluster

```
minikube start --memory 4096
kubectl apply -f minikube/storageclass.yml
```

Install Helm

```
helm init
```

Install Jenkins

```
helm install stable/jenkins --set Agent.Memory=1024Mi --name=cd
minikube service cd-jenkins
```

LAB 2: CONTINUOUS INTEGRATION

Objectives

- build an artefact of the software
- run all unit tests and in-tree integration tests

Jenkins Kubernetes Plugin

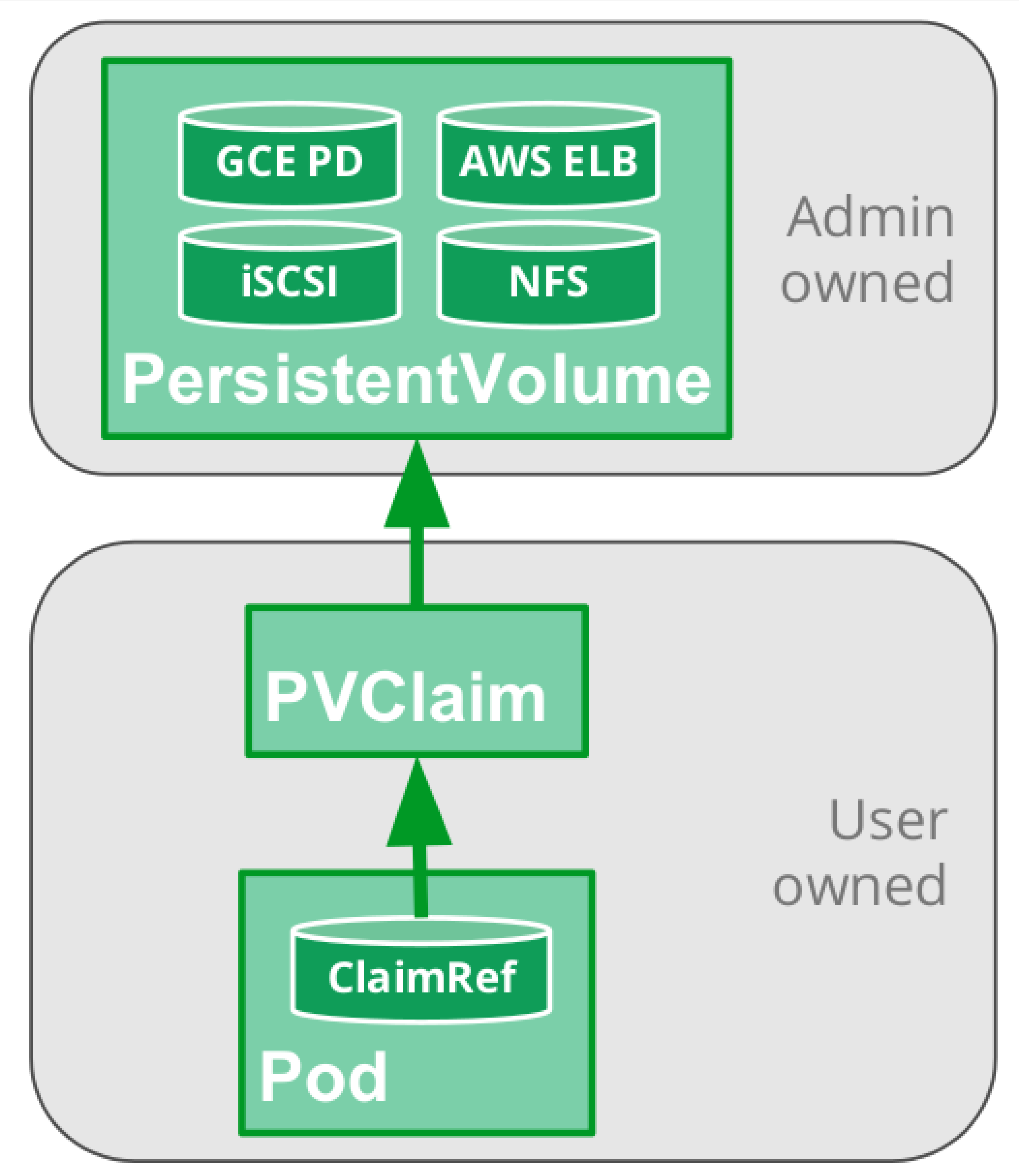
- creates a new pod for every job
- use the pod as a temporary build slave

Let's Go

- create a new pipeline job
- use https://github.com/nbyl/cd-workshop-demo.git as SCM source for your Jenkinsfile

caveat

- build cache is gone after every build


GCE PD
AWS ELB
iSCSI
NFS
PersistentVolume
Admin owned
PVClaim
ClaimRef
Pod
User owned

persistent volumes

- create manually
- use storageclass with auto-provisioner

```
kind: StorageClass
apiVersion: storage.k8s.io/v1beta1
metadata:
  name: generic
provisioner: kubernetes.io/host-path
```

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
name: gradle-cache
annotations:

volume.beta.kubernetes.io/storage-class: "generic"
spec:
accessModes:

- ReadWriteOnce
resources:

requests:
```

LAB 3: BUILDING A DOCKER CONTAINER

Objectives

- build a docker container containing the application
- make the container available for deployment

Build Pod Revisited

- using build pods is scalable and reproducible
- the host docker daemon is not reachable

Docker-in-Docker Builds

- mount the docker sockets inside the container
  - /run/docker.sock
  - /var/run/docker.sock
- image will be built in host docker
- **Warning**: possible security problem

LAB 4: INTEGRATION TESTING

Objectives

- create a deployment description for the application
- deployment the application in a configuration analog to production
- run an integration test verifying the basic functionality

Helm Chart

```
helm create helm/confy
[edit]
helm upgrade --install dev-confy helm-confy
```

Configure the Application

```
env:
- name: SPRING_DATASOURCE_URL
  value: {{ .Values.database.url }}
- name: SPRING_DATASOURCE_DRIVER
  value: {{ .Values.database.driver }}
- name: SPRING_DATASOURCE_USERNAME
  value: {{ .Values.database.username }}
- name: SPRING_DATASOURCE_PASSWORD
  value: {{ .Values.database.password }}
```

Simulate roduction

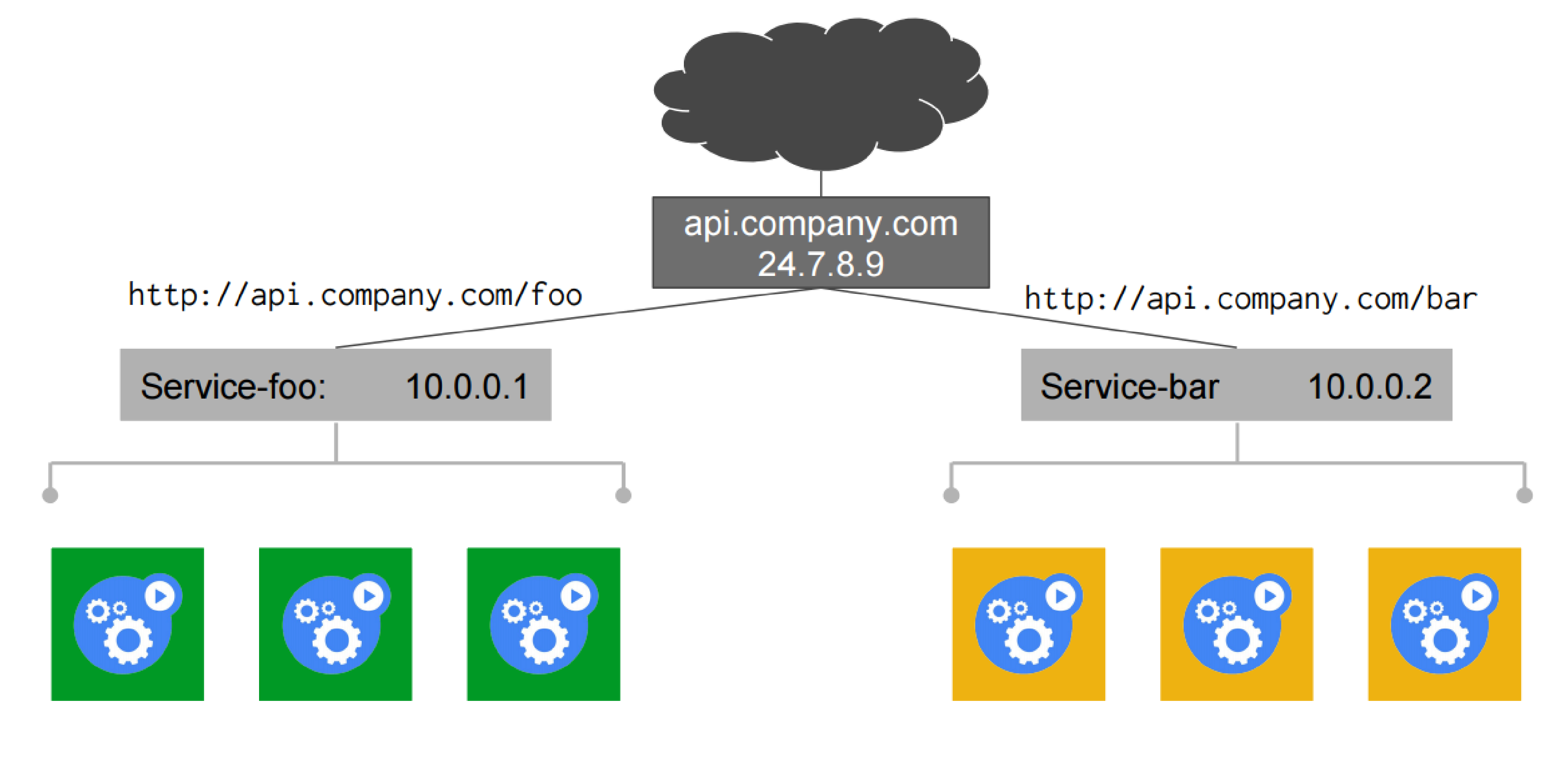
- use the same database server as in production
- treat system as blackbox

LAB 5: USER ACCEPTANCE TESTING

Objectives

- deploy the application accessible for users
- allow manual testing
- continue to production after manual confirmation

Ingress



api.company.com
24.7.8.9
http://api.company.com/foo
http://api.company.com/bar
Service-foo: 10.0.0.1
Service-bar 10.0.0.2

Ingress Controller

- Read Ingress state route accordingly
- Implementations:
  - NGINX
  - traefik
  - F5
  - ...

LAB 5: PRODUCTION DEPLOYMENT

Objectives

- make application available for end users (nothing new to learn)

WRAP UP

Links

- https://kubernetes.io
- https://www.cncf.io/
- https://www.openshift.com/promotions/kubernetes.html
- https://fabric8.io
- https://github.com/ramitsurana/awesome-kubernetes

The End

@NicolasByl

codecentric