Pentaho and Docker Docker Docker!

Rogier Wessel

Outline

- Define
- Docker standalone
- Docker composition
- Docker orchestration!
- Demo that!

Define: docker

Define: docker

Open source **engine** to easily create **lightweight, portable, self-sufficient** containers from **any application**.

Container technologie is sort of like “light weight” virtualisation

- VM’s are ***huge***, containers are ***small***
- VM’s start ***slow***, containers at ***lightning speed***
- VM’s ***do not integrate*** very well, containers “***natively***” integrate services
- VM’s are **Pets**, containers are **Cattle**
- Containers adhere to the ***12 factor app methodology***

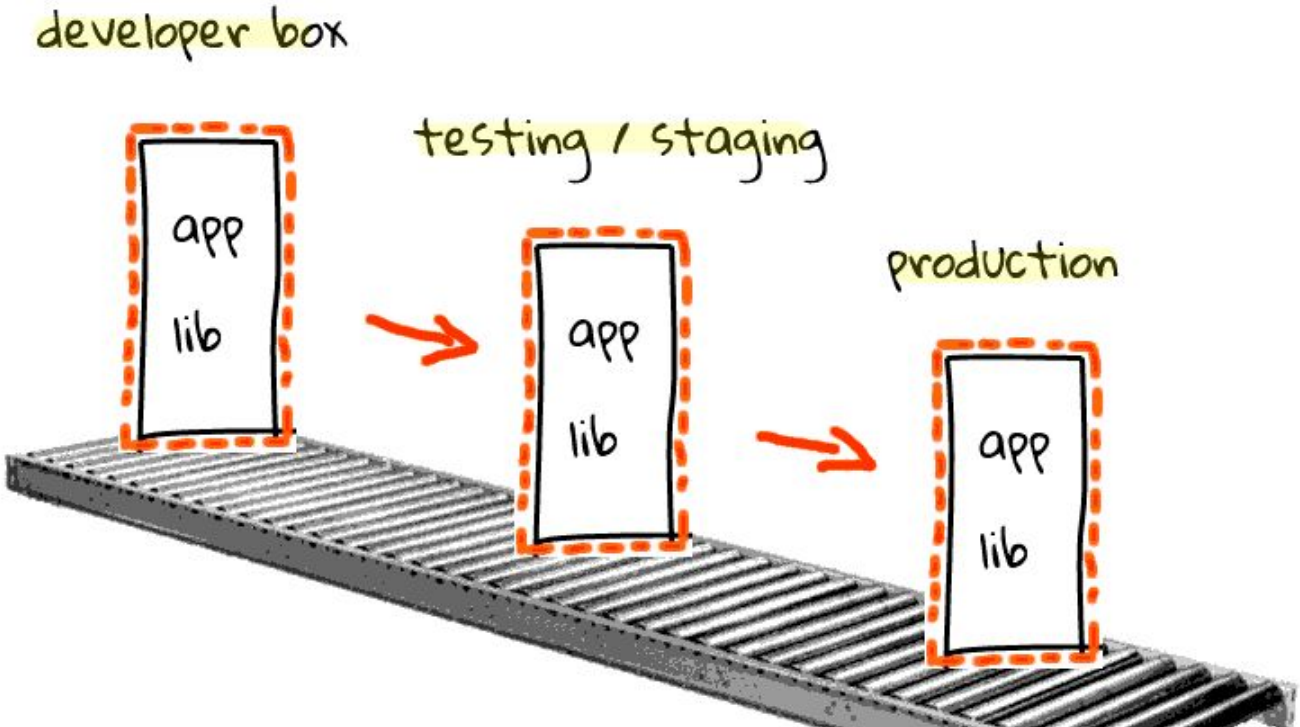
Define: 12 factor app

- **Codebase** One codebase tracked in revision control, many deploys
- **Dependencies** Explicitly declare and isolate dependencies
- **Config** Store config in the environment
- **Backing Services** Treat backing services as attached resources
- **Build, release, run** Strictly separate build and run stages
- **Processes** Execute the app as one or more stateless processes
- **Port binding** Export services via port binding
- **Concurrency** Scale out via the process model

Define: 12 factor app

- **Disposability** Maximize robustness with fast startup and graceful shutdown
- **Dev/prod parity** Keep development, staging, and production as similar as possible
- **Logs** Treat logs as event streams
- **Admin processes** Run admin/management tasks as one-off processes

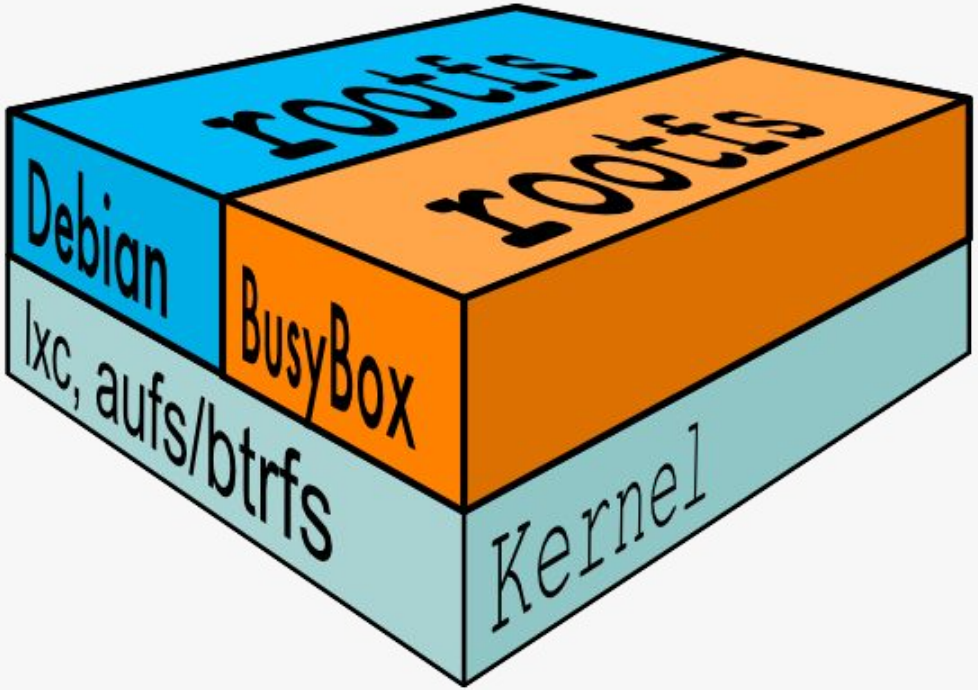
Define: docker

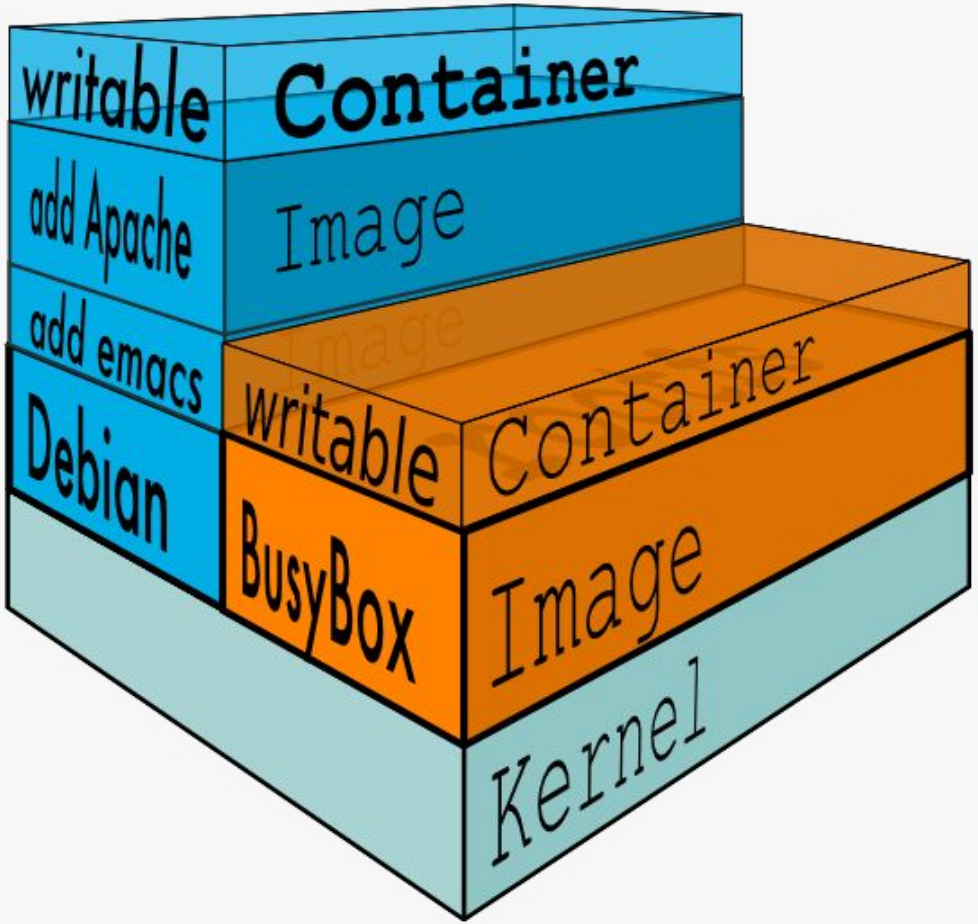


Docker standalone

- FROM
- MAINTAINER
- ADD / COPY
- RUN
- ENV
- EXPOSE
- ENTRYPOINT
- USER

```
FROM reelmetrics/java:7

MAINTAINER Rogier Wessel <rwessel@reelmetrics.com>

# Set correct environment variables.
ENV HOME /root
ENV PENTAHO_HOME /opt/pentaho/biserver-ee
ENV TOMCAT_HOME $PENTAHO_HOME/tomcat
ENV DB_TYPE postgresql
ENV PENTAHO_VERSION 5.3.0.0-213
#ENV PENTAHO_VERSION 5.4.0.0-128

# Use baseimage-docker's init system.
CMD ["/sbin/my_init"]

# ============================== Start Image Customization ==================
ADD biserver-ee-${PENTAHO_VERSION}.zip /biserver-ee.zip

RUN apt-get update && \
# apt-get upgrade -f -y && \
  apt-get install -f -y curl git zip pwgen && \
  useradd -m pentaho && mkdir /opt/pentaho && \
  unzip -q /biserver-ee.zip -d /opt/pentaho/ && \
  rm -rf ${PENTAHO_HOME}/promptuser.sh && \
  rm -rf /biserver-ee.zip && \
# Disable daemon mode for Tomcat
  sed -i -e 's/\(exec ".*"\) start/\1 run/' ${TOMCAT_HOME}/bin/startup.sh
```


rootfs
rootfs
Debian
BusyBox
lxc, aufs/btrfs
Kernel


writable
Container
add Apache
Image
add emacs
Debian
writable
BusyBox
Container
Image
Kernel

Docker composition

- Compose multiple containers
- Inject env specific variables
- Link containers
- Persist storage via docker-volumes
- Expose services to the host
- Perfect for single host deployments

```
postgres:
  image: registrydev.rm:5000/reelmetrics/postgres:latest
#  build: ../postgres
  volumes:
    - "/data/dev/postgres/:/var/lib/postgresql/data/:rw"
    - "/etc/localtime:/etc/localtime:ro"
  environment:
     - POSTGRES_USER
     - POSTGRES_PASSWORD
  expose:
    - "5432"
  ports:
    - "5432:5432"

infobright:
  image: registrydev.rm:5000/reelmetrics/infobright:latest
#  build: ../infobright
  volumes:
    - "/data/dev/infobright/load/:/load/"
    - "/data/dev/infobright/db/:/mnt/mysql_data/"
    - "/etc/localtime:/etc/localtime:ro"
  expose:
    - "5029"
  ports:
    - "5029:5029"
  environment:
     - MYSQL_ROOT_PASSWORD
```

Docker orchestration!

Mesos

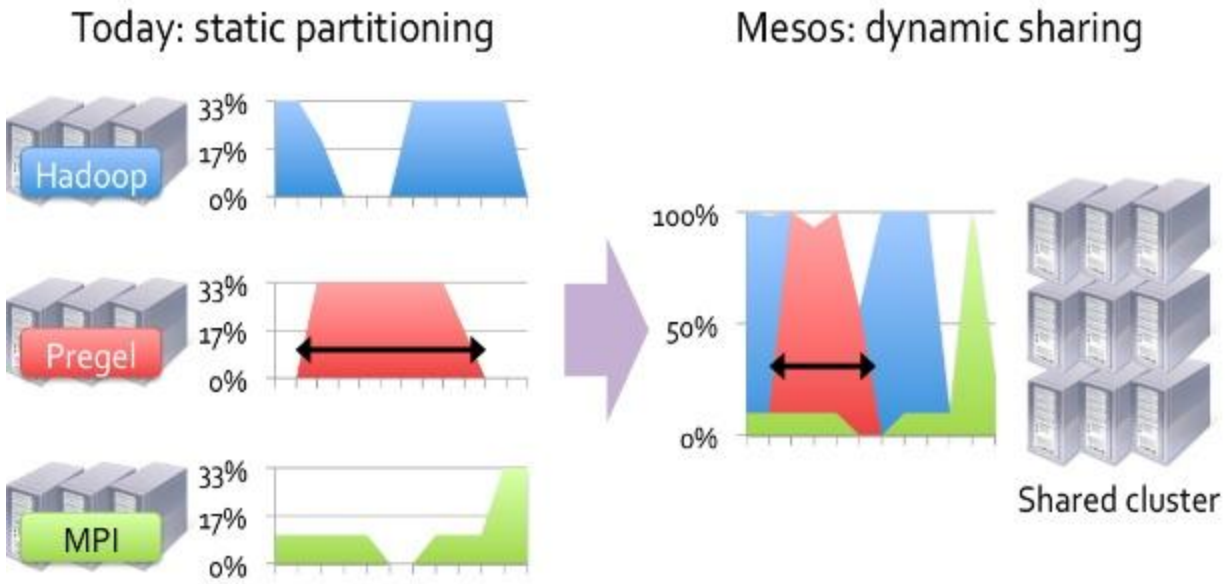
Mesos

Program against your datacenter like it's a single pool of resources

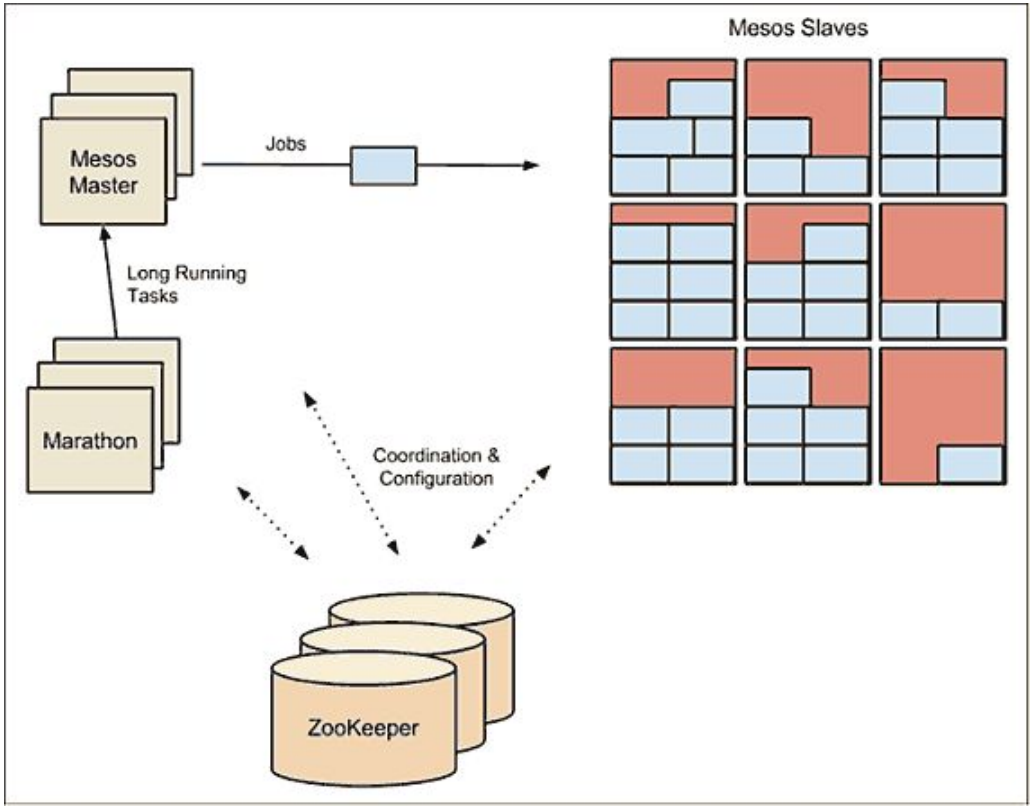
Apache Mesos **abstracts CPU, memory, storage, and other compute resources** away from machines (physical or virtual), enabling **fault-tolerant** and **elastic** distributed systems to easily be built and run effectively.

A **distributed systems kernel**. Mesos is built using the **same principles as the Linux kernel, only at a different level of abstraction**. The Mesos kernel runs on every machine and provides applications (e.g., Hadoop, Spark, Kafka, Elastic Search) with **API's for resource management** and **scheduling across entire datacenter and cloud environments**.

Optimise utilization



Mesos architecture



Demo that!

The end