# DOCKER WORLD WALK THROUGH

## JAKUB VEVERKA (@JWERAK)

## ABOUT ME

- Appuri
- Container meetup

## ABOUT TALK

- Introduction to (Docker like) containers
- Development lifecycle with containers
- What are containers good/bad for

## ABOUT CONTAINERS

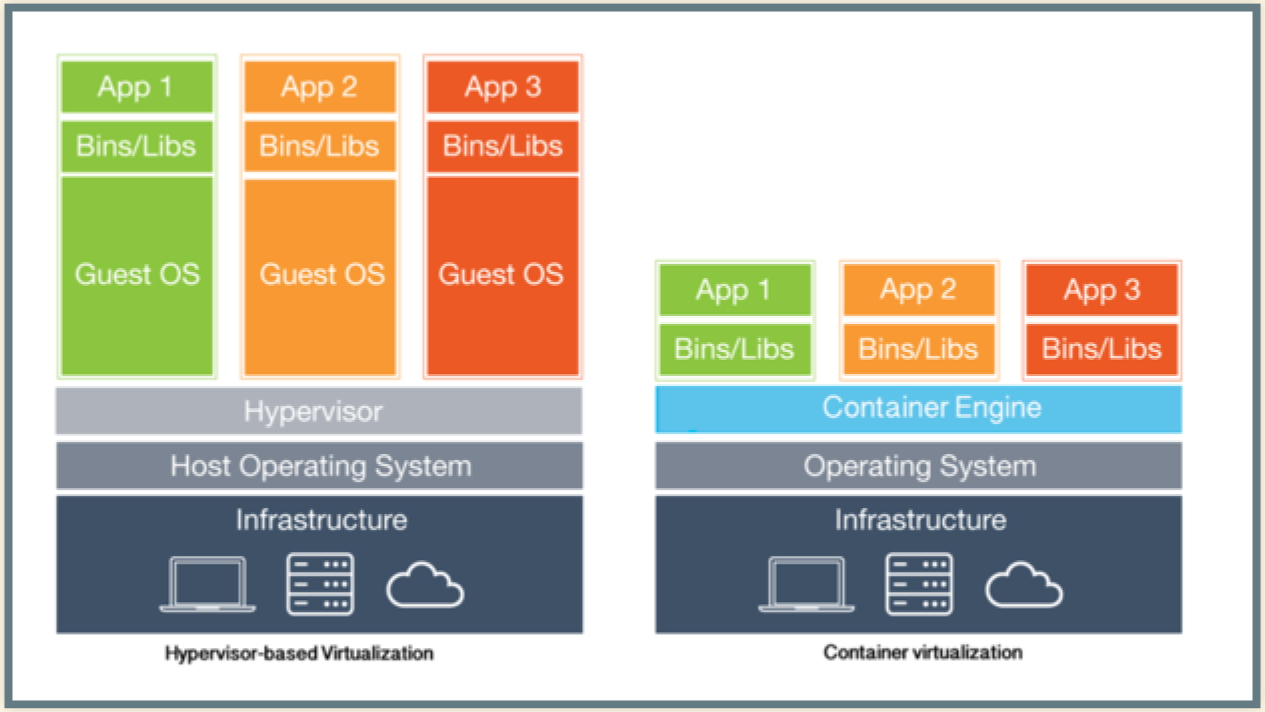
## EVOLUTION OF CONTAINERS

- chroot - 1979 (Unix)
- Jails - 2000 (FreeBSD)
- Linux-VServer - 2001 (Linux) (first namespace separation)
- Seccomp - 2005 (Linux)
- cgroups - 2006 (Linux)
- Linux Namespaces + LXC - 2008 (Linux)
- Docker opensourced - 2013 (Linux)

## WHAT ARE CONTAINERS

- package up an application
- contain applications
- consistent among environments
- kernel features
  - Namespaces
  - cgroups
  - Selinux, capabilities, …
- Why containers and not only Docker?

# CONTAINERS VS VMS



## DIFFERENCES

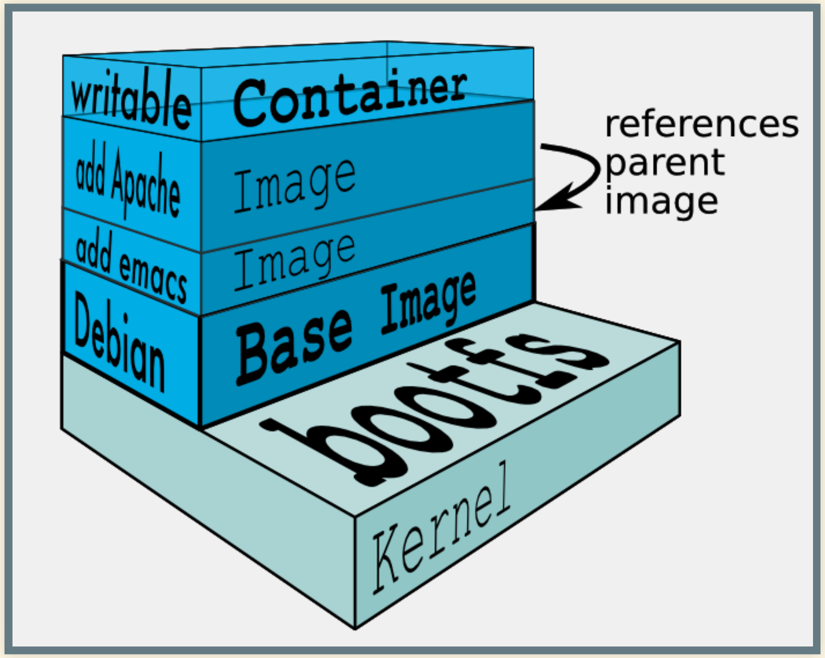
- Resource utilization
- Startup time
- deployment time
  - Containerimages

# HOW TO CREATE CONTAINER IMAGES

## Dockerfile -> image -> container

```
FROM debian
RUN apt-get install emacs
RUN apt-get install apache2
ADD app /app
CMD ["/usr/sbin/apache2", "-DFOREGROUND"]
```

## Docker image layers



## HOW TO DISTRIBUTE CONTAINER IMAGES

- registry
- artifactory
- http server
- ...

## DEVELOPMENT LIFECYCLE WITH APPS IN CONTAINERS

## CODING PHASE

- writing app
- sharing with colleagues
- booting to custom project
- but we had the same with vms, didn't we?

## CI/TEST PHASE

- Containerized CI slaves/minions
    - resource utilization
    - simple setup of custom build environments
- Containerized artefacts
    - Unified test and production libraries

## DEPLOYMENT/PRODUCTION

- Containers are best fit to dynamic environments
- Deploying containers manually is error prone and doesn't scale
- Containers on their own are not suitable for production

## DEPLOYMENT/PRODUCTION

- Beyond Build, Ship, Run…
  - ensure requested apps are running
  - apps are healthy
  - apps are accessible
  - apps can talk to each other
  - apps are started in desired environment
  - …
- Kubernetes on LinuxDays
  - Kubernetes in Production talk in this room ;)
  - Kubernetes Workshop (room 111, 16:00)

## CONTAINERS AND SECURITY

- Do containers enhance security on its own?
  - chroot by default
  - resource limitation (rogue process can't steal from other) (cgroups)
- Selinux, AppArmor, etc
  - adds privilege separation layer between processes
- kernel capabilities
- Seccomp

## SUMMARY

## WHAT ARE CONTAINERS GOOD FOR

- simple packaging (including dependencies), only kernel is shared between containers
- applying limits
- stateless applications/microservices
- 12 factor apps

## WHAT ARE CONTAINERS BAD FOR (STILL)

- legacy applications
- stateful applications
- traditional SQL databases

## QUESTIONS?

## SOURCES

- http://rhelblog.redhat.com/2015/08/28/the-history-of-containers/
- http://www.infoworld.com/article/3072929/linux/containers-101-linux-containers-and-docker-explained.html