Introduction to software containers

Application in scientific practice

Docker and Singularity

Toni Hermoso Pulido

Bioinformatics Unit

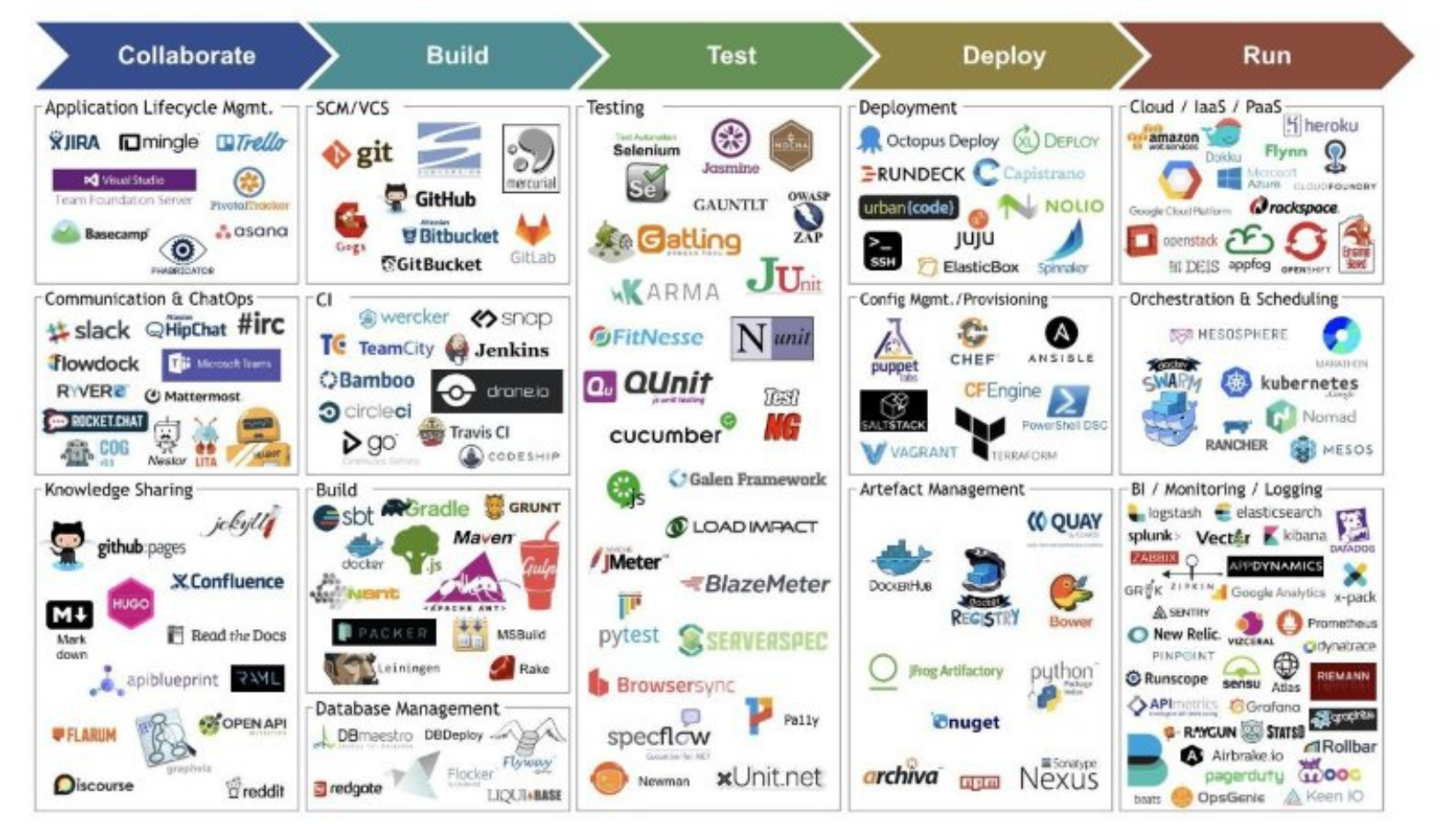
CRG, Barcelona

Link to online version

DevOps

- Software Engineering Culture
- Software Development + Software Operations
- Automate and monitor

DevOps



https://hostadvice.com/blog/devops-toolbox-jenkins-ansible-chef-puppet-vagrant-saltstack/

Containers

Containers in New Jersey

Virtualisation

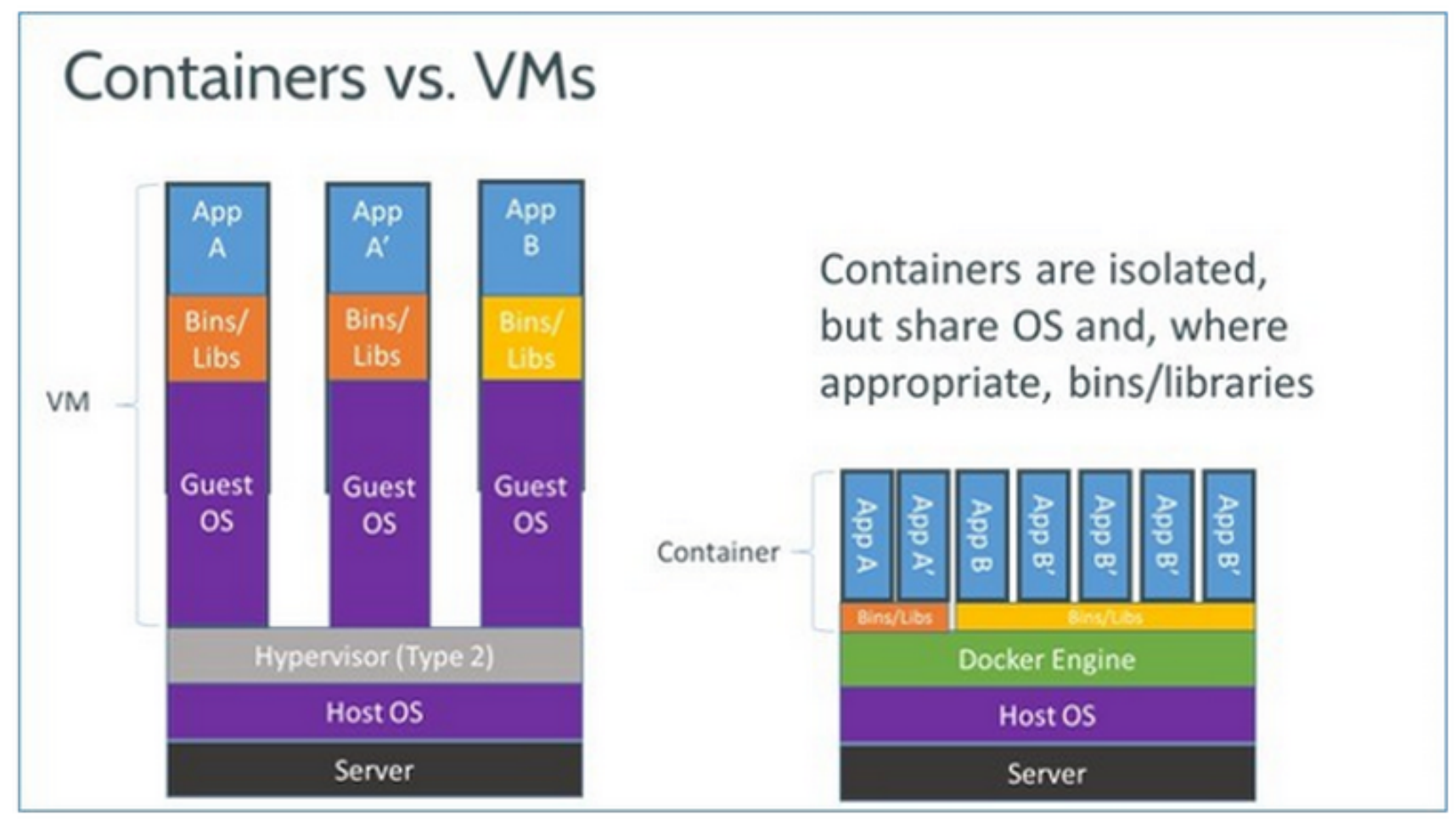
- Abstraction of physical hardware
- Depends on hypervisor (software)
- Do not confuse with hardware emulator
- Enable virtual machines
  - Every virtual machine with an Operating System

Containerisation

aka Lightweight virtualisation

- Abstraction of application layer
- Depends on host kernel (Operating System)
- Application and dependencies bundled all together

Virtual machines vs containers



http://www.zdnet.com/article/what-is-docker-and-why-is-it-so-darn-popular/

Virtualisation

Pros and Cons

- **PRO**: Very similar to a full OS
- **PRO**: With current solutions, high OS diversity
- **CON**: Need of more space and resources
- **CON**: Slower than containers
- **CON**: Not as good automating

Containerisation

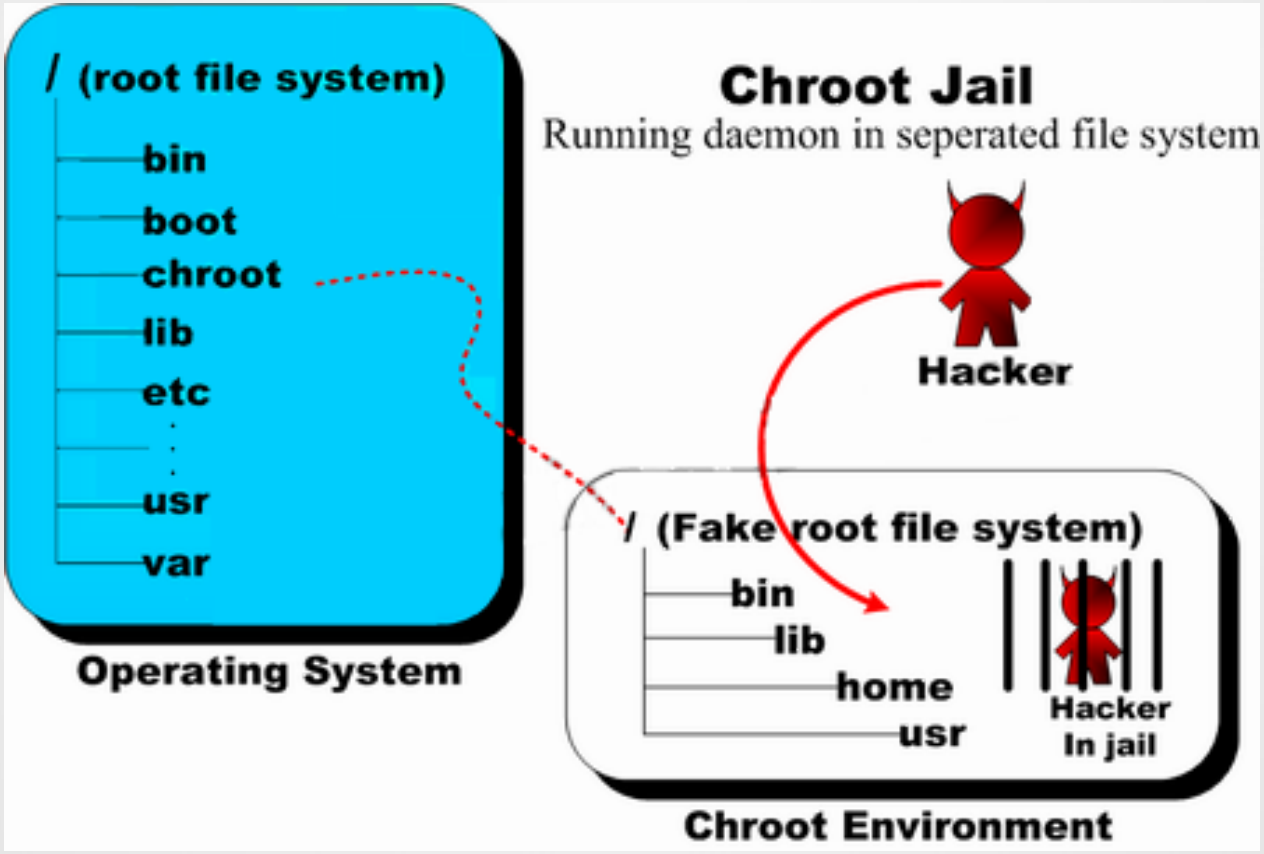
Pros and Cons

- **PRO**: Faster
- **PRO**: No need of full OS installation. Less space.
- **PRO**: Current solutions allow easier distribution of recipes. More portability
- **PRO**: Easier automation
- **CON**: Some cases might not be exactly the same as a full OS
- **CON**: With current solutions, still less OS diversity

History of containers

chroot

- chroot jail (BSD jail) - First concept 1979
- Notable use in SSH and FTP servers
- Honeypot, recovery of systems, etc.



https://sysopsio.wordpress.com/2016/09/09/jails-in-linux/

Additions in Linux kernel

- cgroups (control groups), before 'process containers'
  - isolates resource usage (CPU, memory, disk I/O, network, etc.) of a collection of processes
- Linux namespaces
  - one set of kernel resources restrict to one set of processes

Additions in Linux kernel

Linux Containers

Docker 1.10 and later

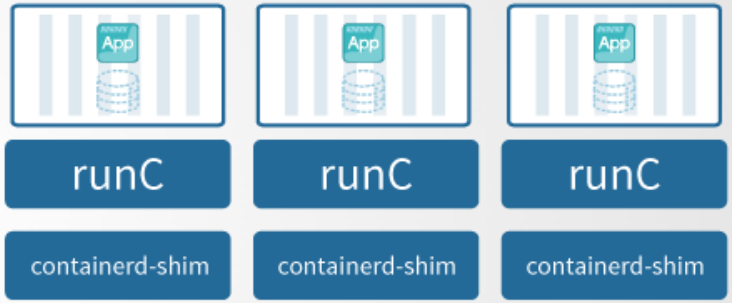
liblxc

namespaces

cgroups

SELinux/AppArmor

Linux kernel



containerd

Docker Engine

namespaces

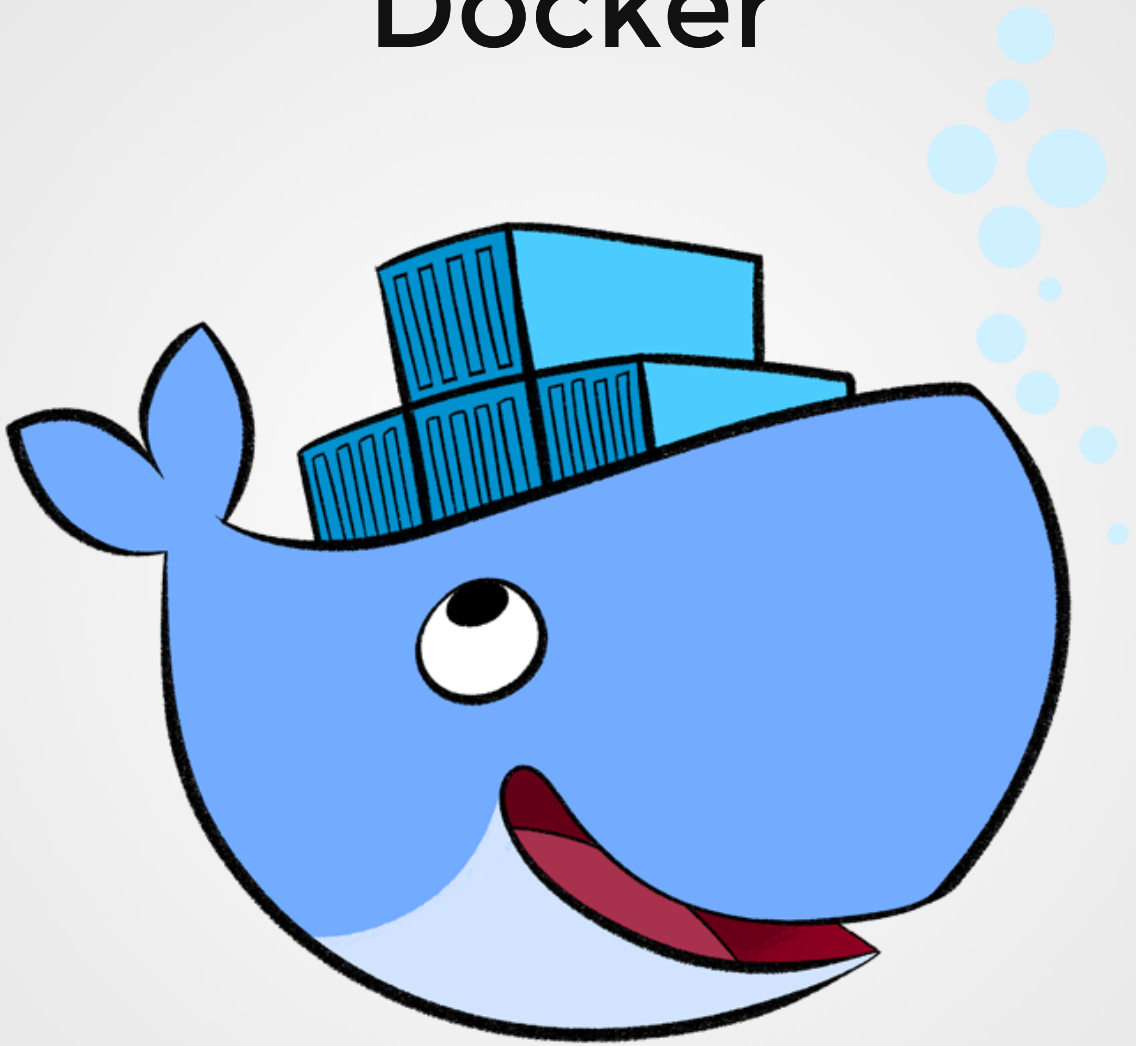
cgroups

SELinux/AppArmor

Linux kernel

https://robinsystems.com/blog/containers-deep-dive-lxc-vs-docker-comparison/

Docker



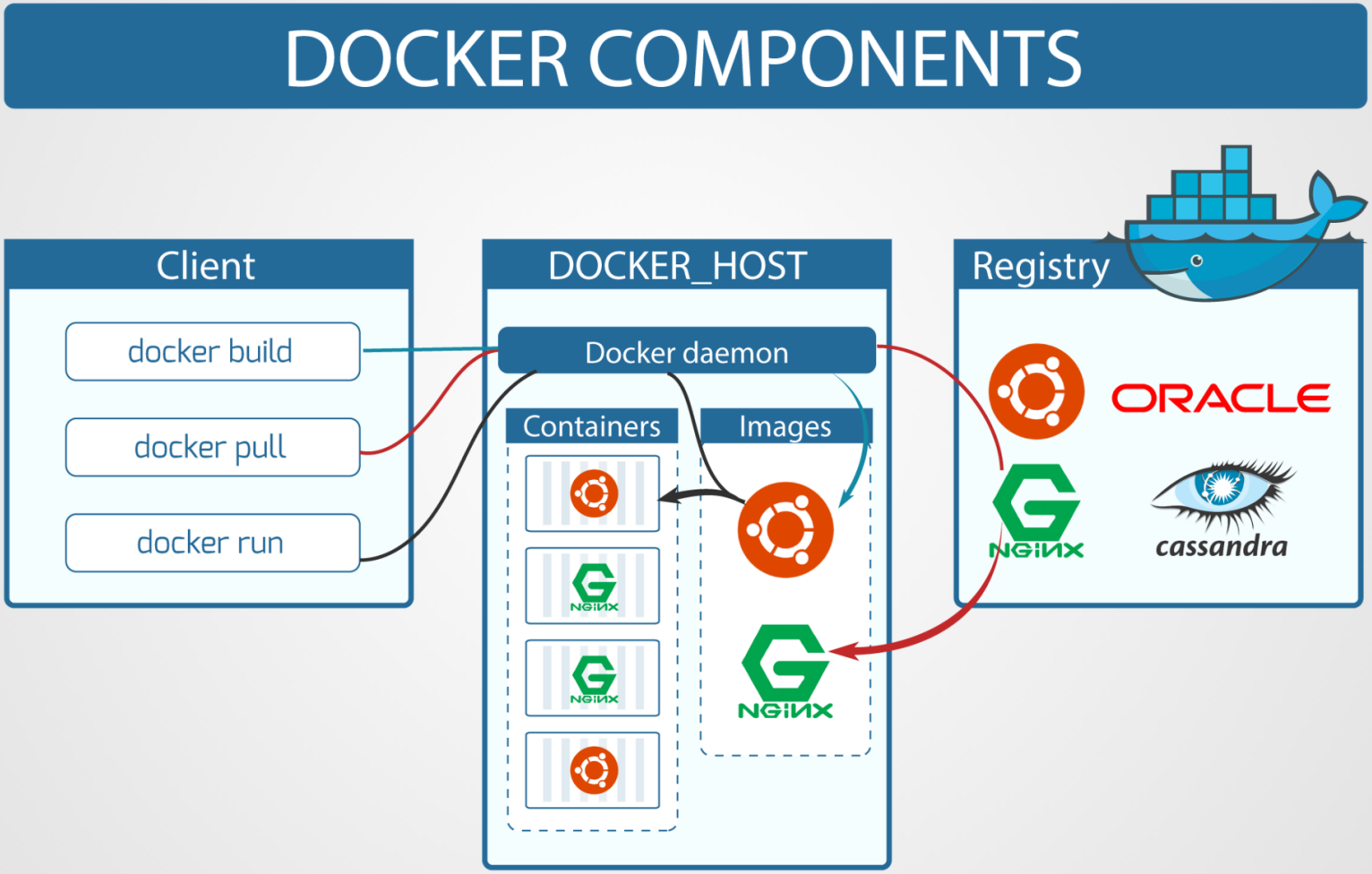
Docker

- Platform for developing, shipping, and running applications
- Infrastructure as application/code
- Established Open Container Initiative

As a software:

- Docker Community Edition
- Docker Enterprise Edition

Docker architecture



Docker image

- Read-only templates.
- Containers are run from them
- Images are not run
- Images have several layers

Docker image - Building

- Can be built from existing base images
  - E.g. : Ubuntu, Apache
- Any modification from a base image is a new layer
- Base images can be generated with tools such as Debootstrap
  - Reminds of creation of a bootable (USB) Image

Docker image - Instructions

- Recipe file:
  - `Dockerfile`
- Instructions
  - *Every instruction generates an image layer*
  - FROM: use a base image (notice tag)
  - ADD, COPY: add files to image filesystem
  - RUN: execute command in image
  - ENV, ARG: Run and build environment variables
  - CMD, ENTRYPOINT: Command to execute when generated container starts

Docker image - Instructions

- Instructions
  - USER: User to run a following process (fallback: *root*)
  - WORKDIR: Location where following processes will be executed
  - VOLUME: Definition of mounting point (with host)
  - EXPOSE: Ports to be accessible from the container

Reference

Docker container

- Generated from an image (template)
- Image: read-only
- Container: read-write
- Can be converted into image
  - `docker commit`
- 1 imatge -> n diverse containers
  - Diversity:
    - Volumes / Mounting points
      - Different data or configs
    - Different exposed ports

Run container

```
$ docker run biocorecrg/c4lwg-2018 /bin/echo "Hello world!"
```

Run container as daemon

Run daemon

```
$ docker run -d --name mycontainer biocorecrg/c4lwg-2018 /bin/sh -c "while true; do echo hello world; sleep 1; done"
```

List runninng containers

```
$ docker ps
```

Log and info of container

```
$ docker log mycontainer
$ docker inspect mycontainer
```

Actions on containers

```
$ docker stop mycontainer
$ docker start mycontainer
$ docker restart mycontainer
```

More running on container

Execute on a running container

```
$ docker exec mycontainer /bin/echo "Bye, moon!"
```

Run a container interactively (from the beginning). Stops when exiting

```
$ docker run -ti mycontainer2 /bin/bash
```

Execute on a running container interactively

```
$ docker exec -ti mycontainer /bin/bash
```

Other Docker commands

remove container, remove image

```
$ docker rm mycontainer
$ docker rmi myimage
```

list all containers (even stopped ones)

```
$ docker ps -a
```

Docker registry and Docker hub

- Images are stored locally
- They can also be shared in a registry
- Main Public one: Docker hub

Examples:

https://hub.docker.com/u/biocorecrg/

https://github.com/CRG-CNAG/docker-debian-perlbrew

Biocontainers

http://biocontainers.pro

Repository of bioinformatics containers

https://dx.doi.org/10.1093/bioinformatics/btx192

Docker registry commands

download image

```
$ docker pull biocorecrg/debian-perlbrew:stretch
```

upload image

```
$ docker push myusername/mysexyimage
```

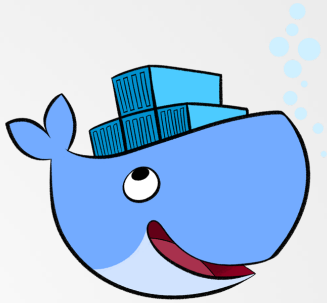
Singularity

containers for HPC

https://singularity.lbl.gov/

https://doi.org/10.1371/JOURNAL.PONE.0177459

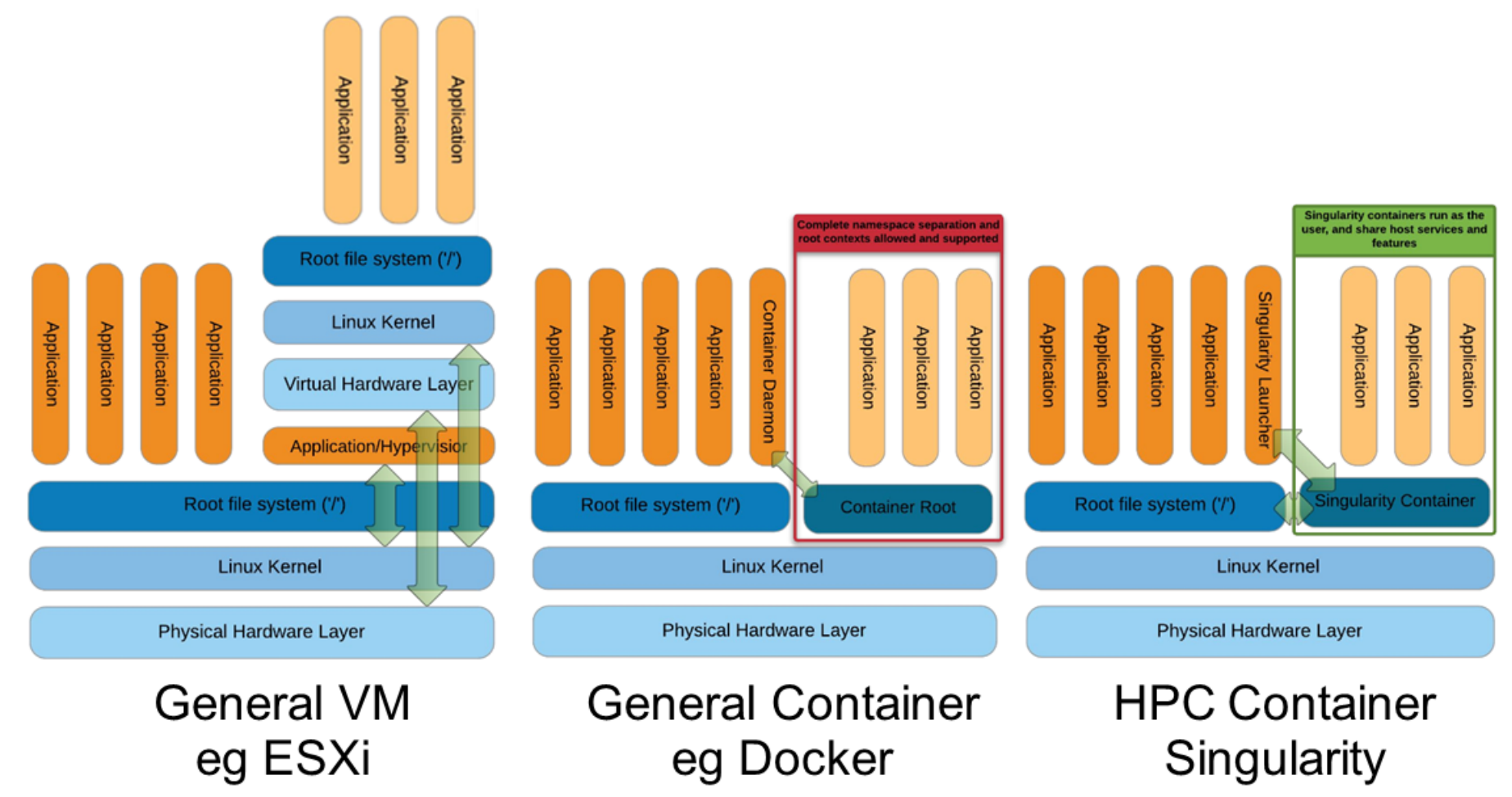
Singularity vs Docker



Summarising

- Docker -> Microservices
- Singularity -> HPC

Singularity architecture

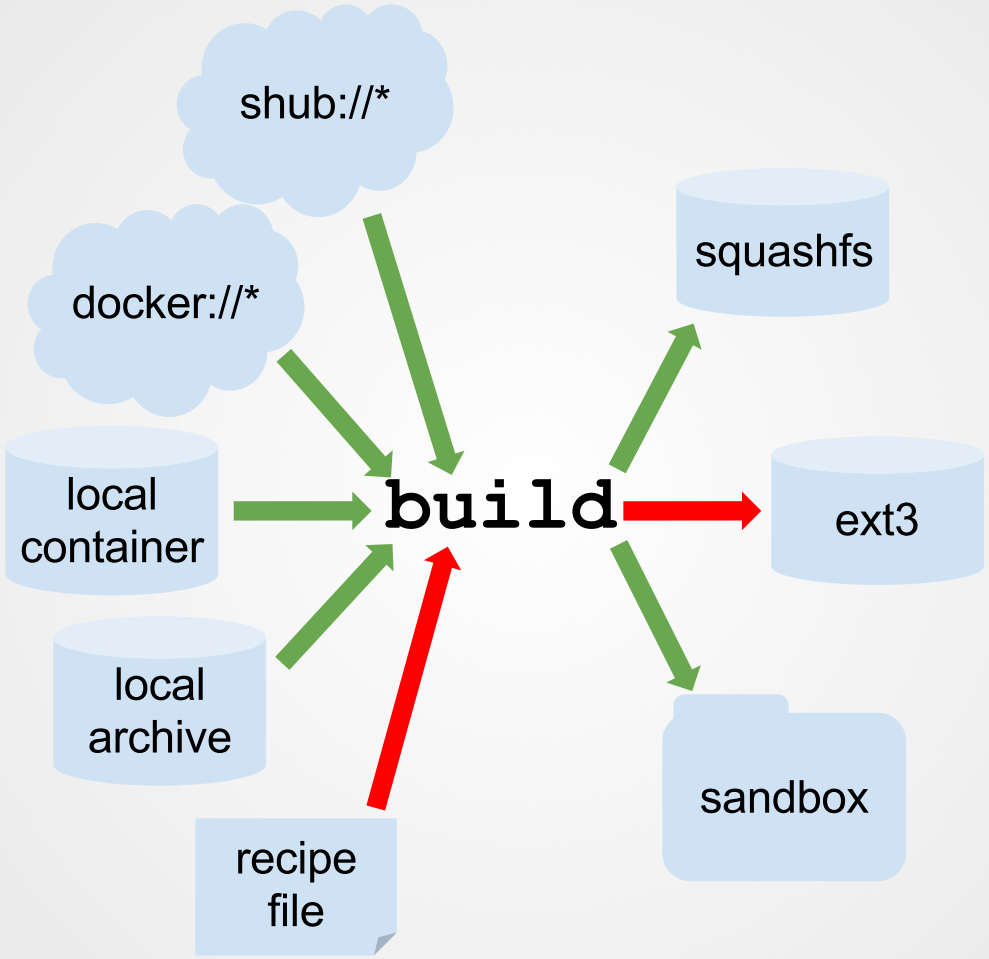


http://geekyap.blogspot.com.es/2016/11/docker-vs-singularity-vs-shifter-in-hpc.html

Singularity - Strenghts

- No dependency of a daemon
- Can be run as a simple user
- Image/container is a file (or directory)
  - More easily portable
- Two type of images
  - Read-only (production)
  - Writable (development)

Singularity - build



As of 2.4.x version

http://singularity.lbl.gov/docs-build-container

For local Docker images: https://github.com/singularityware/docker2singularity

Singularity recipes

```
BootStrap: docker
From: biocorecrg/c4lwg-2018

%runscript
    echo "Welcome to C4LWG-2018 Singularity Image"

%post
    echo "C4LWG-2018 image built"
```

http://singularity.lbl.gov/docs-recipes

Singularity recipes

```
BootStrap: debootstrap
OSVersion: xenial
MirrorURL:  http://fr.archive.ubuntu.com/ubuntu/
Include: build-essential curl python python-dev openjdk-8-jdk bzip2 zip unzip

%runscript
    echo "Welcome to Ubuntu C4LWG-2018 Singularity Image"

%post

    FASTQC_VERSION=0.11.5
    MULTIQC_VERSION=1.5
    BOWTIE_VERSION=1.2.1.1

    cd /usr/local; curl -k -L https://www.bioinformatics.babraham.ac.uk/projects/fastqc/fastqc_v${FASTQC_VERSION}.zip > f
    cd /usr/local; unzip fastqc.zip; rm fastqc.zip; chmod 775 FastQC/fastqc; ln -s /usr/local/FastQC/fastqc /usr/local/bi

    cd /usr/local; curl --fail --silent --show-error --location --remote-name https://github.com/BenLangmead/bowtie/relea
    cd /usr/local; unzip -d /usr/local bowtie-${BOWTIE_VERSION}-linux-x86_64.zip
    cd /usr/local; rm bowtie-${BOWTIE_VERSION}-linux-x86_64.zip
    cd /usr/local/bin; ln -s ../bowtie-${BOWTIE_VERSION}/bowtie* .

    curl --fail --silent --show-error --location --remote-name  https://bootstrap.pypa.io/get-pip.py
    python get-pip.py

    pip install numpy matplotlib
    pip install -I multiqc==${MULTIQC_VERSION}

    echo "C4LWG-2018 image built"

%labels
    Maintainer Biocorecrg
Version 0.1.0
```

http://singularity.lbl.gov/build-debootstrap

Singularity - Weaknesses

- At the time of writing only good support in Linux
  - *Not a big deal in HPC environments, though*
- For some uses you need root account (or sudo)
- Still young project compared to other solutions

Singularity - build

Build read-only image from Docker (Squash FS)

```
$ singularity build c4lwg-2018.simg docker://biocorecrg/c4lwg-2018
```

Build writable image from Docker (ext3 FS)

```
$ sudo singularity build --writable c4lwg-2018.img docker://biocorecrg/c4lwg-2018
```

Build from recipe

```
$ sudo singularity build c4lwg-2018.xenial.simg Singularity.xenial
```

Singularity - run

Execute a command

```
$ singularity exec c4lwg-2018.simg /bin/echo 'Hello world'
```

Execute a command (with clean environment)

```
$ singularity exec -e c4lwg-2018.simg /bin/echo 'Hello world'
```

Execute a shell

```
$ singularity shell c4lwg-2018.simg
```

Execute defined runscript (parameters can be used)

```
$ singularity run c4lwg-2018.simg
```

Further reading

- The impact of Docker containers on the performance of genomic pipelines.
- Performance Evaluation of Container-based Virtualization for High Performance Computing Environments

Sum-up:
containers in science

- Mantainability
- Portability
- Reproducibility

Play!

https://github.com/biocorecrg/C4LWG-2018