
NIC
Future Edition

Docker 101

Docker 101

Damian Flynn

MVP Cloud & Datacenter + Cisco Champion

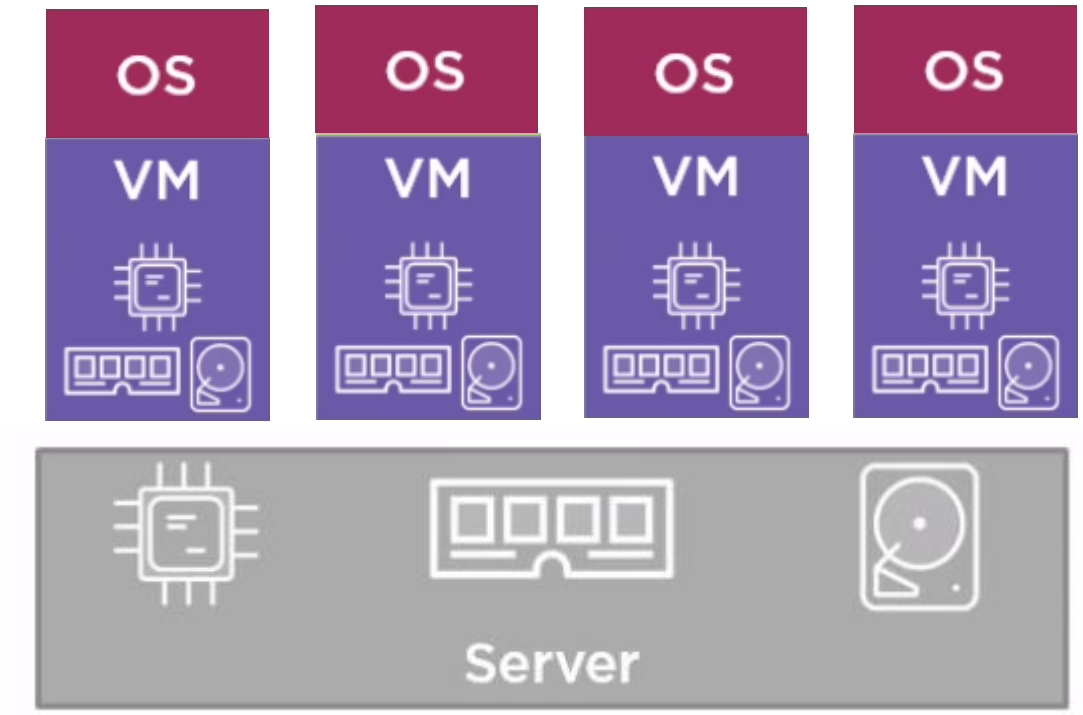
Lumagate

@Damian_Flynn

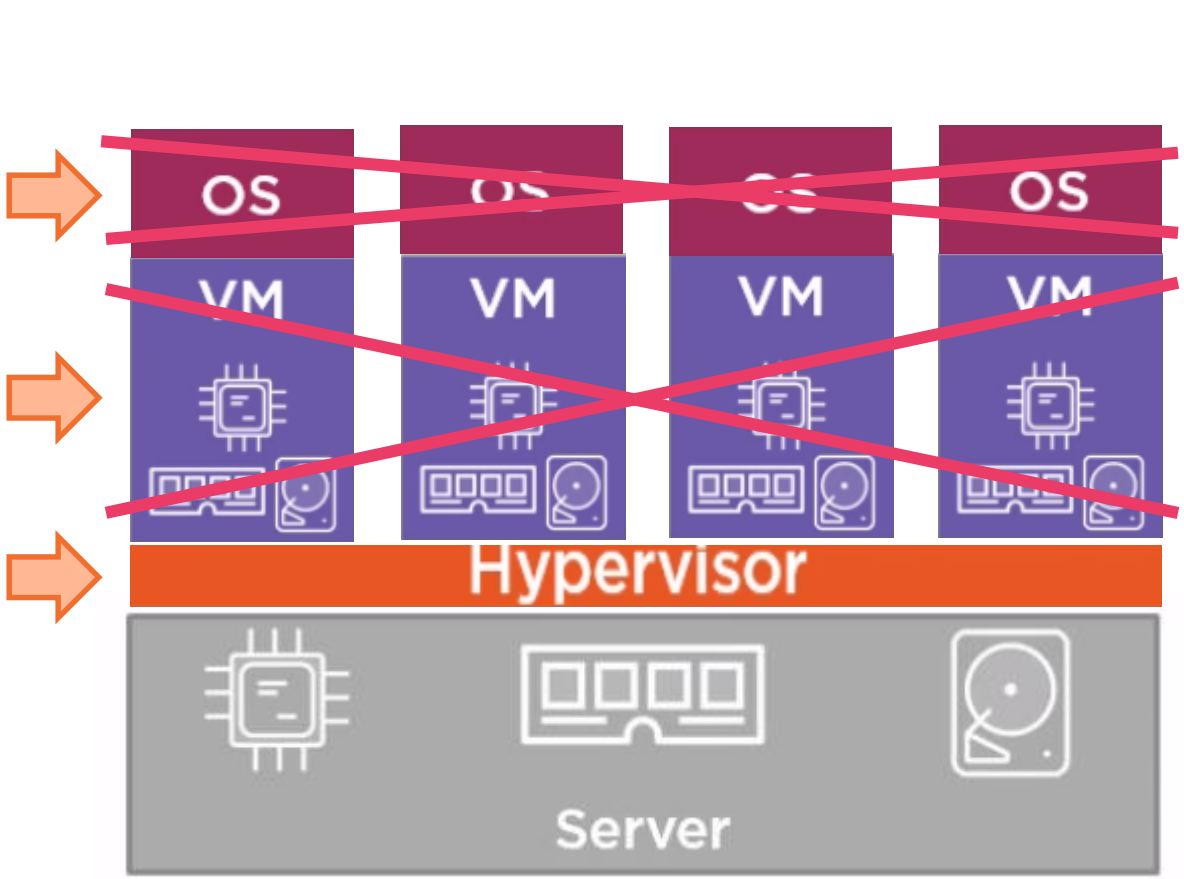
www.DamianFlynn.Com

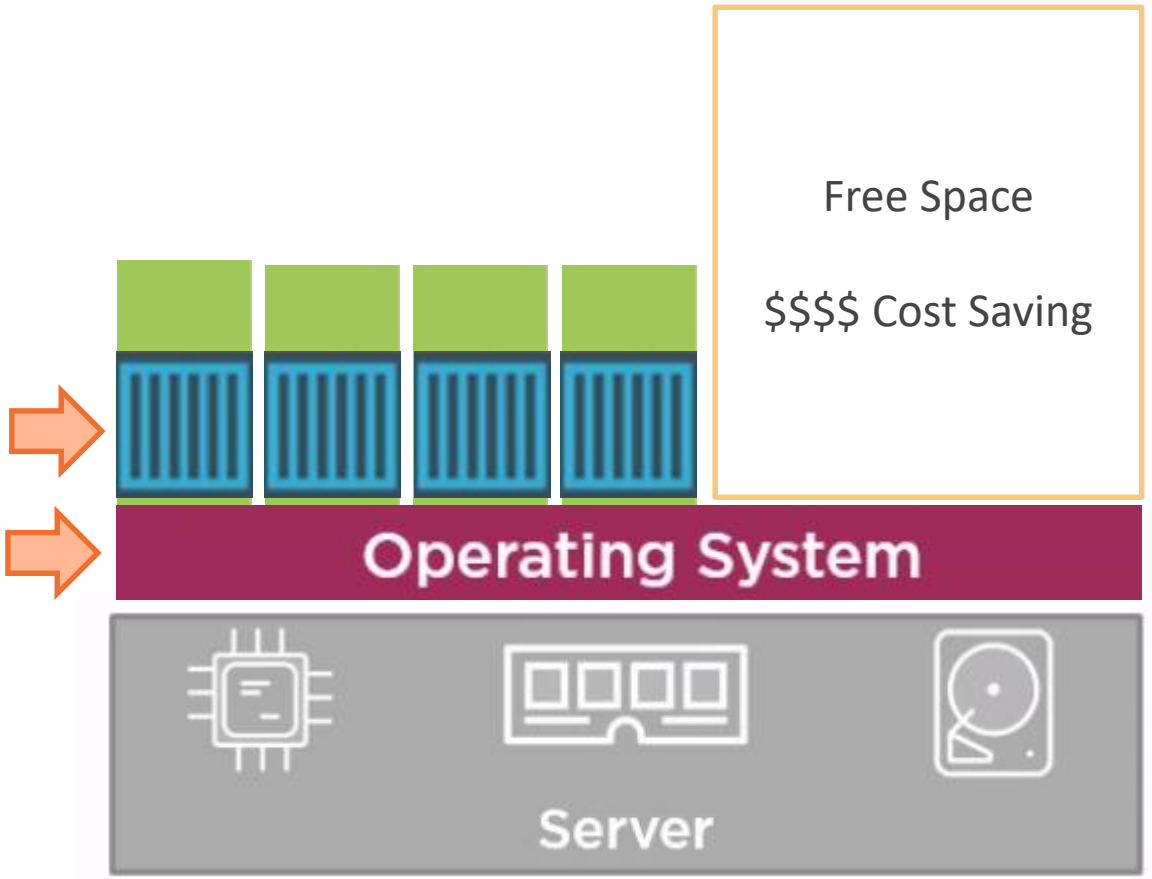
Hyper-Visor Challenge's

- Multiple Application Instances
- Instance Per Virtual Machine
  - Dedicated Host Resource Slice
  - Dedicated OS Instance
    - CPU
    - RAM
    - Disk
    - OS License
    - Administration

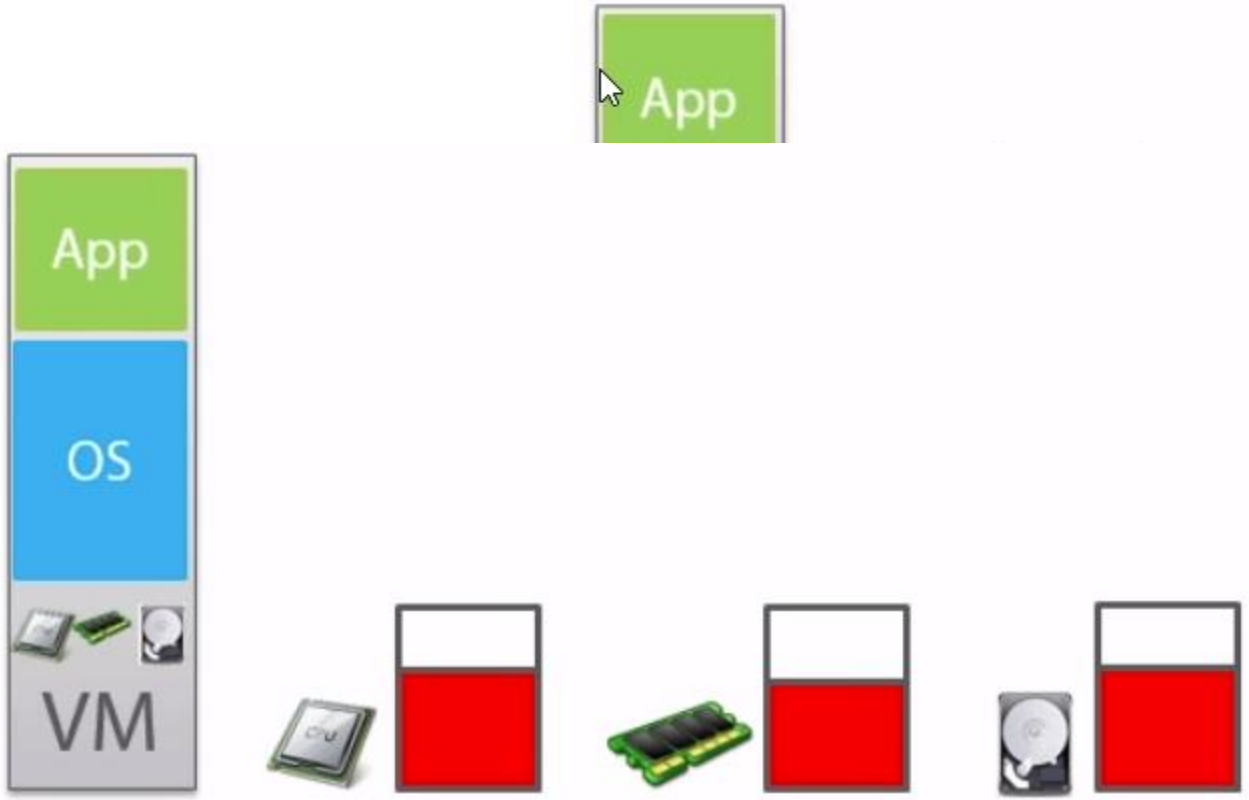


Containers


OS
OS
OS
OS
VM
VM
VM
VM
Hypervisor
Server


Free Space
$$$$ Cost Saving
Operating System
Server

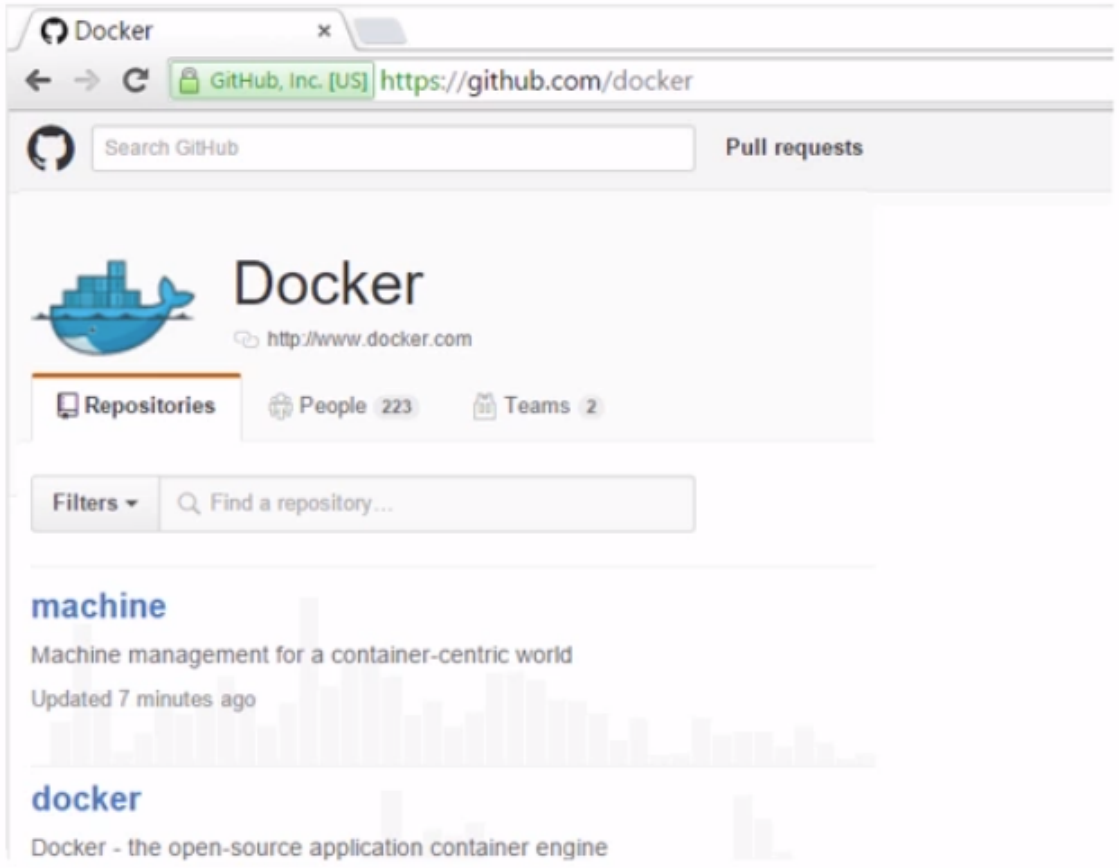
Containers



rs are far
/eight than
Virtual Machines

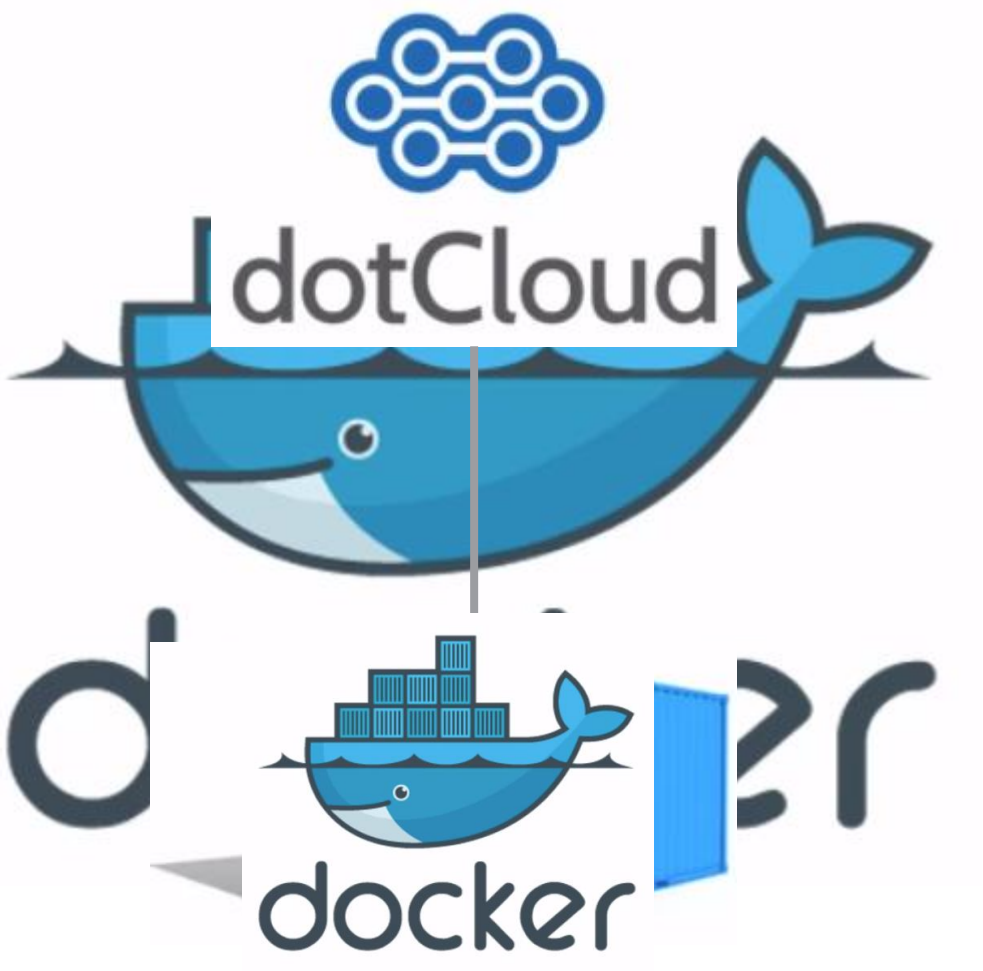
Docker

Docker



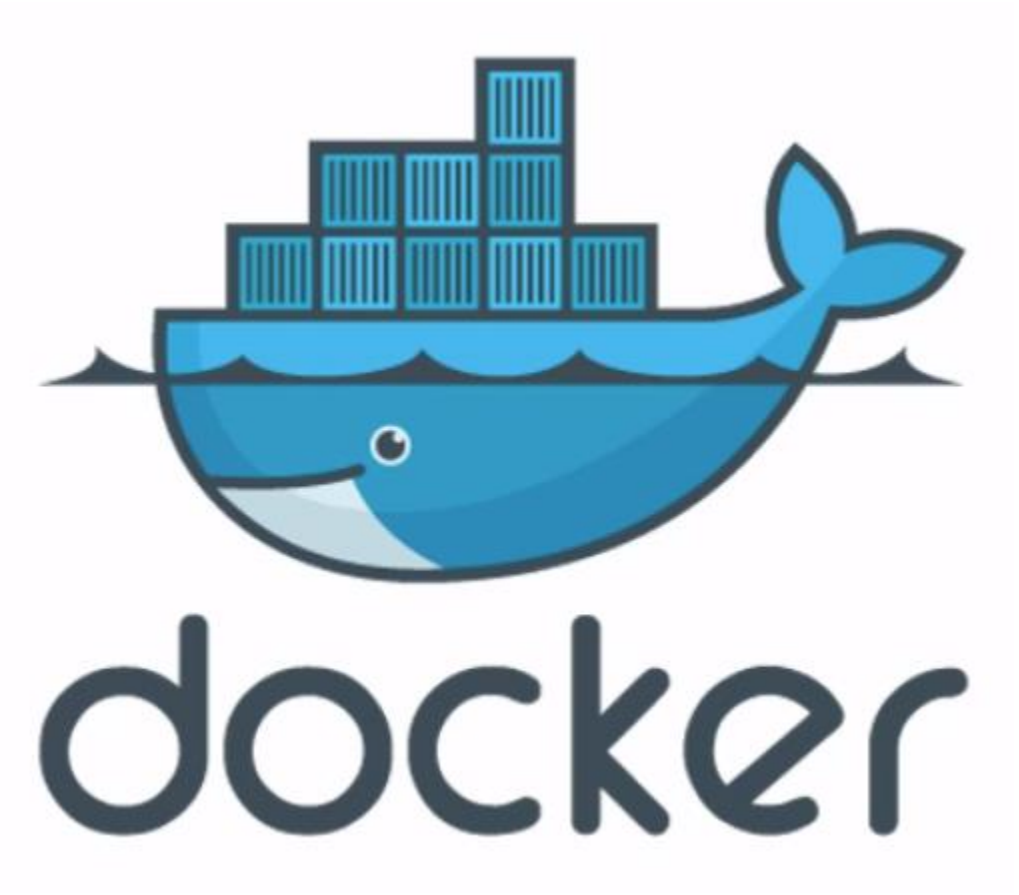
OPEN CONTAINER INITIATIVE

Docker Inc.



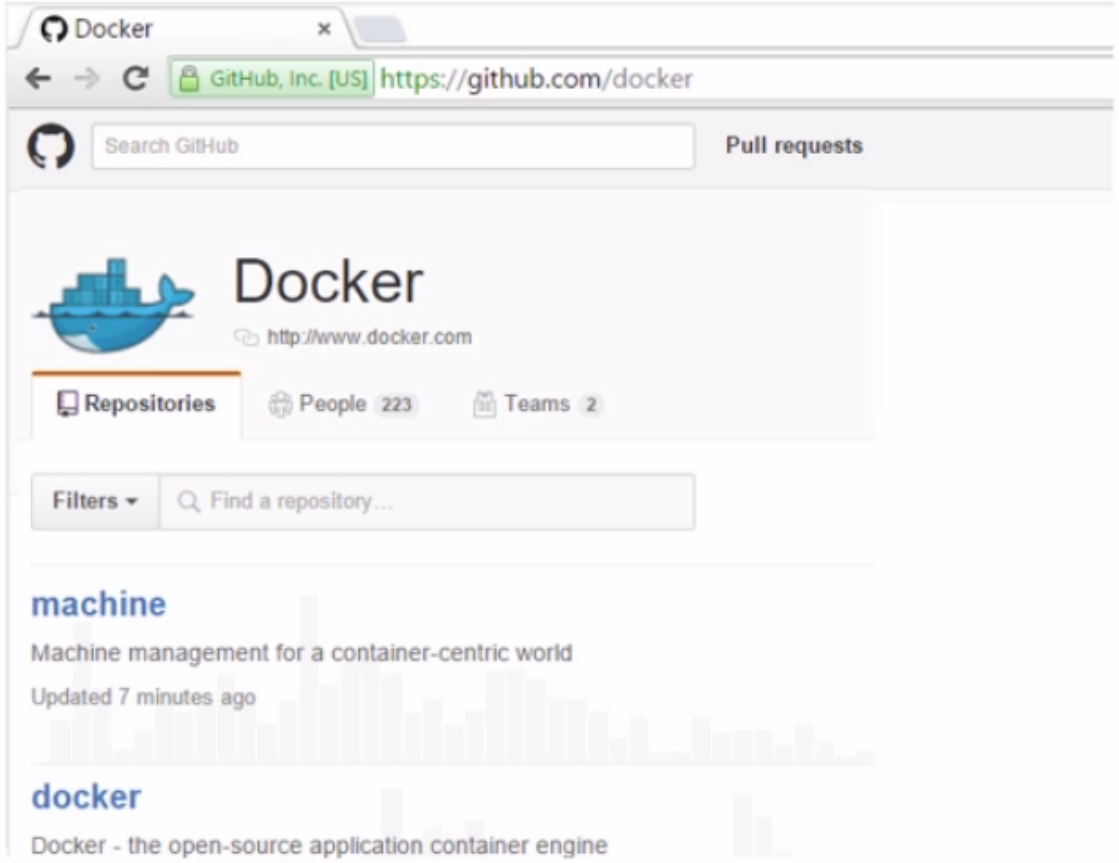
Docker Inc.

- Leading Technology company
- Billion Dollar Valuation
- Raised over $150M in venture Capital
- Approx. 200 Employees
- Acquired companies

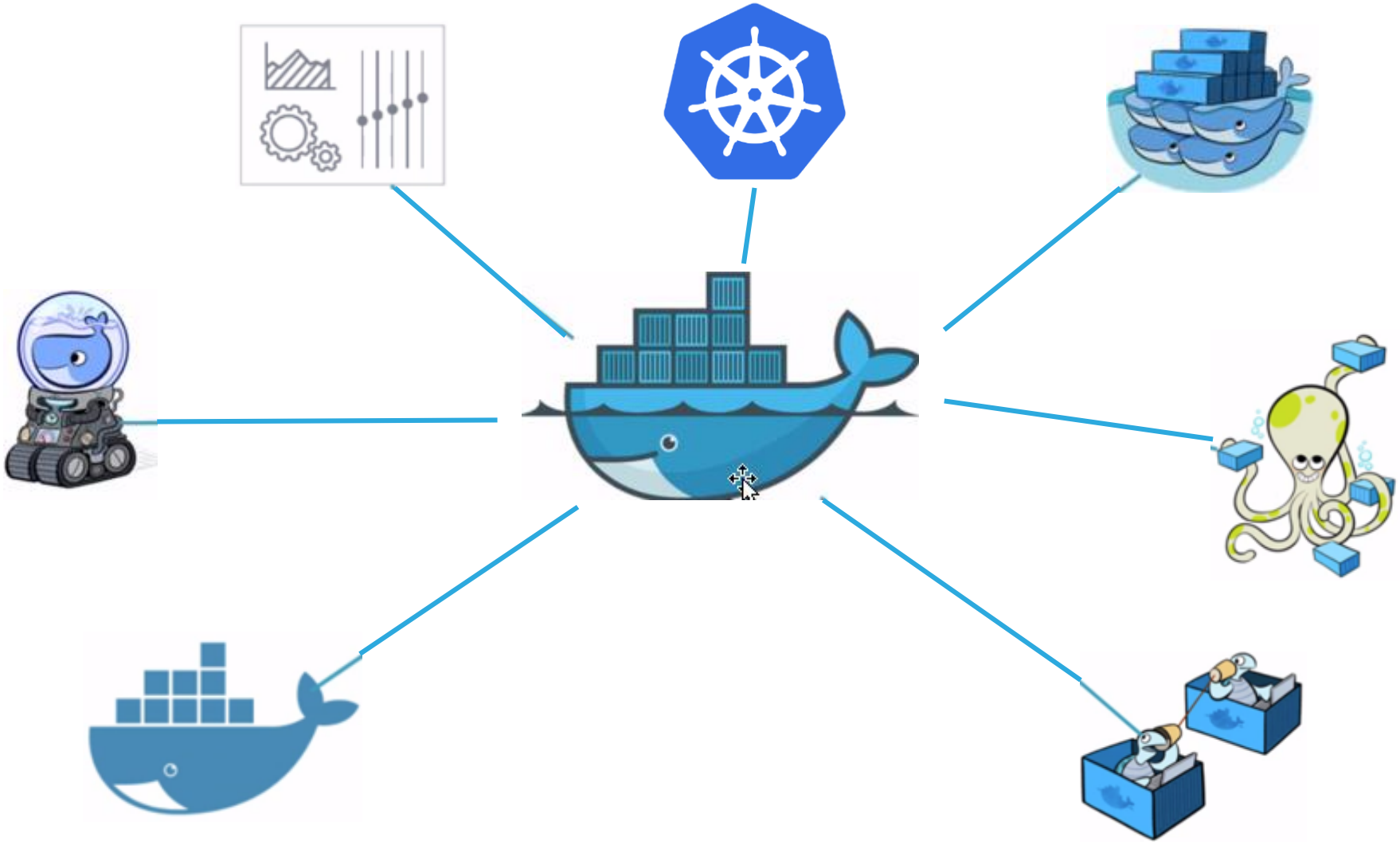


Docker Project

- Open Source
  - Apache License 2.0
- Focused on Tools & Process Efficiency Improvement
- Multiple tools

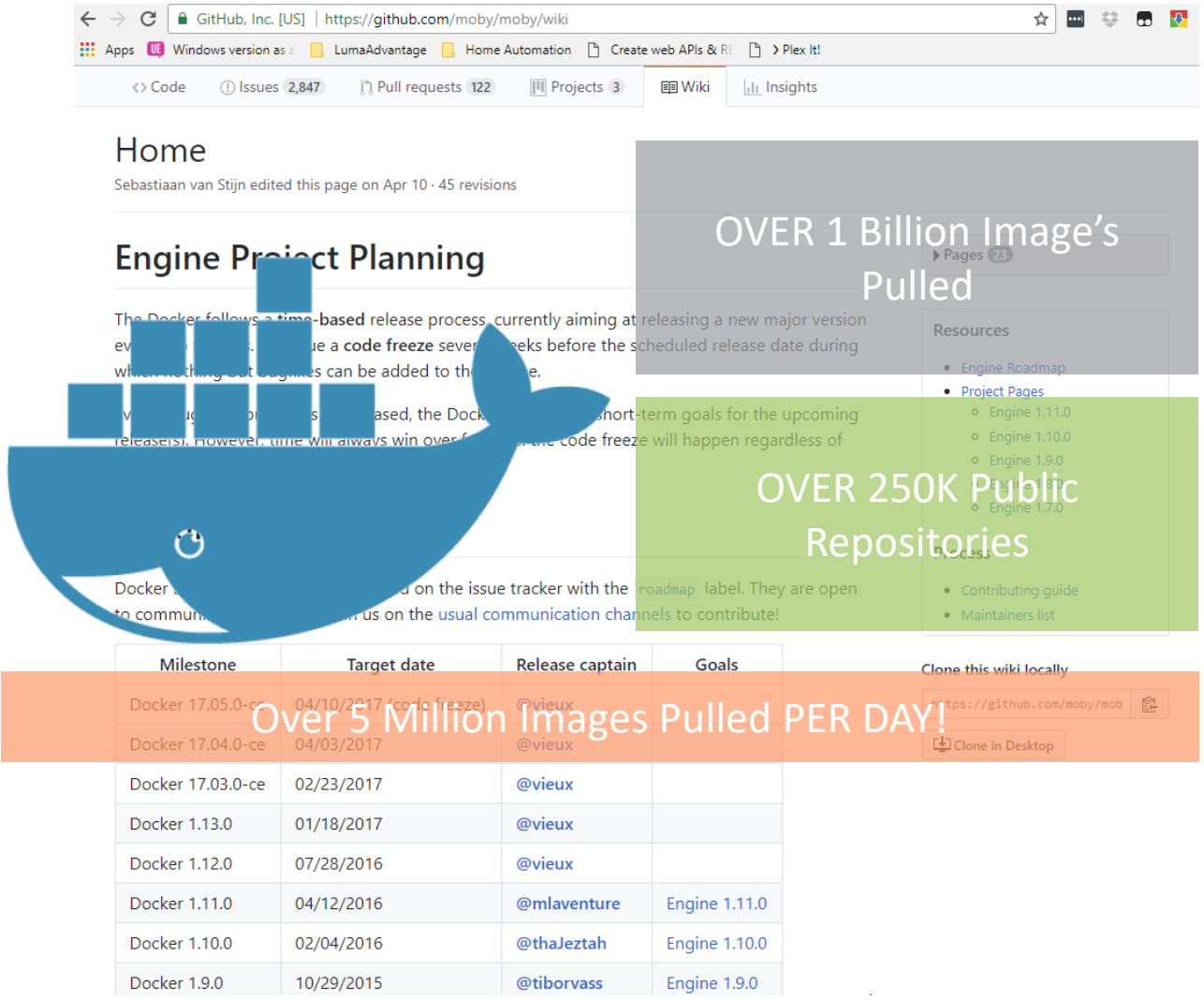


Docker Project

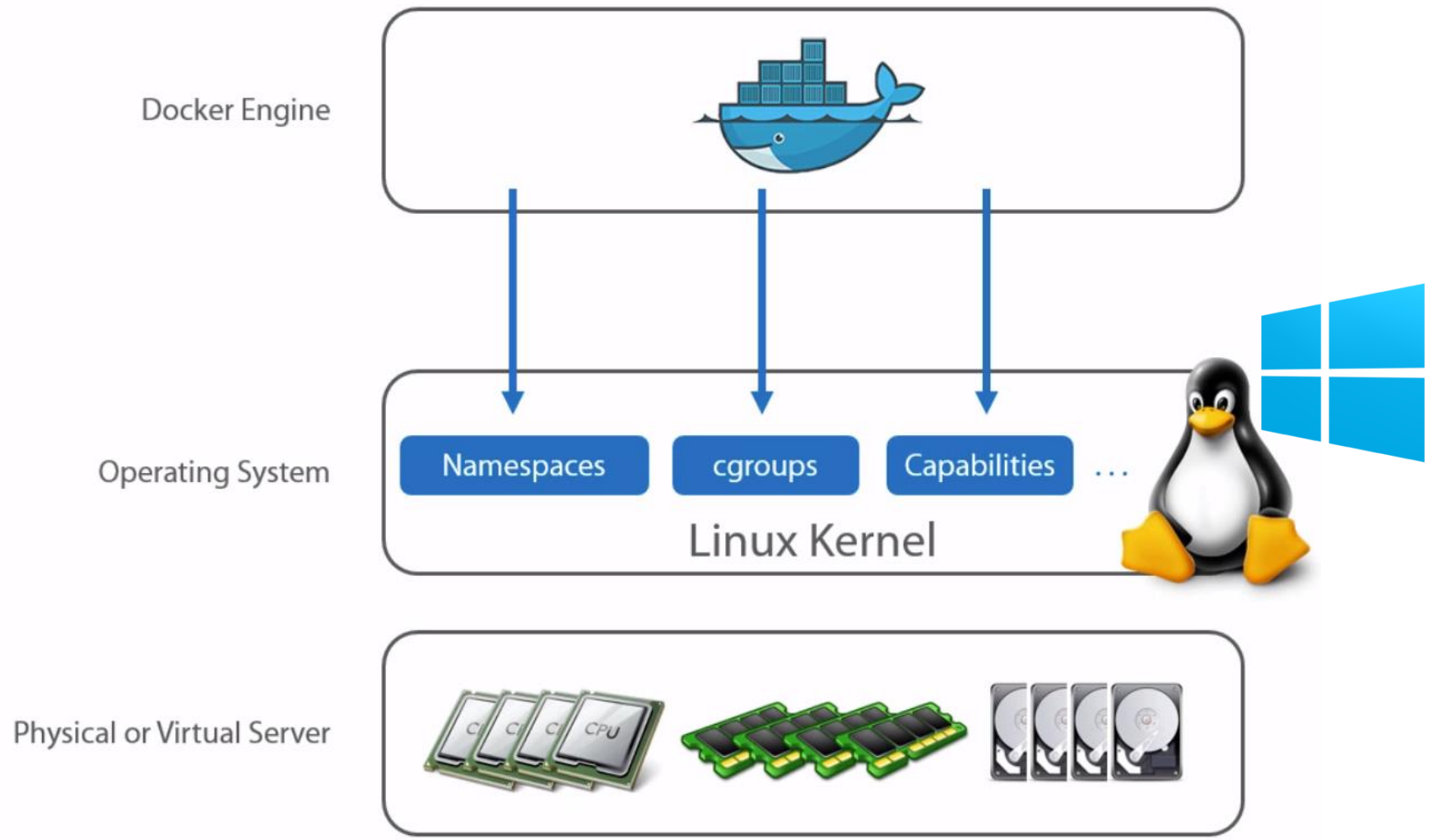


Docker Project

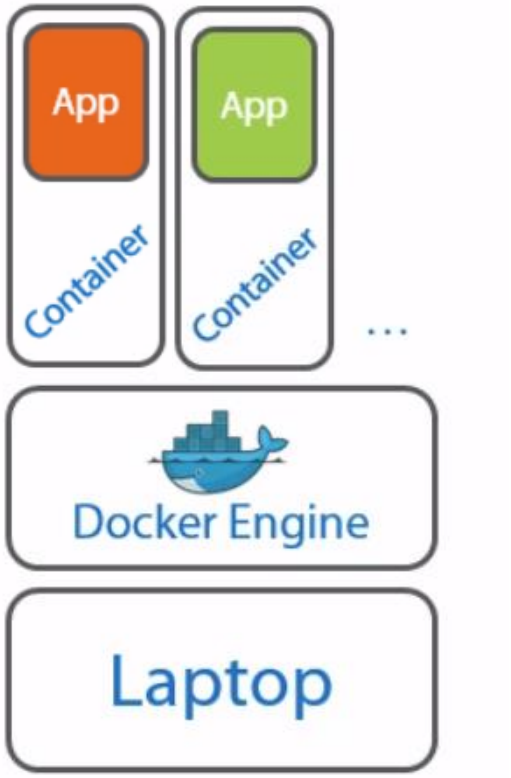
- Open Source
  - Apache License 2.0
- Focused on Tools & Process Efficiency Improvement
- Multiple tools
- Biggest Industry Players Involved



Docker Relationship



Docker Container Standardization



Open Container Initiative

Open Container Initiative

- Lightweight Governance Council
  - Standardize
    - Container Format
    - Container Runtime
- Vendor Neutral
- Platform Neutral
- Formed June 2015
- Linux Foundation

Summary

Docker Inc.

Silicon Valley Startup

Venture Capital Backed

Growing trough Acquisitions

Docker Project

More Efficient
than Physical

Wasted Resources
&
Duplicate OPEX

pen Container Initiative

Greater Efficiencies

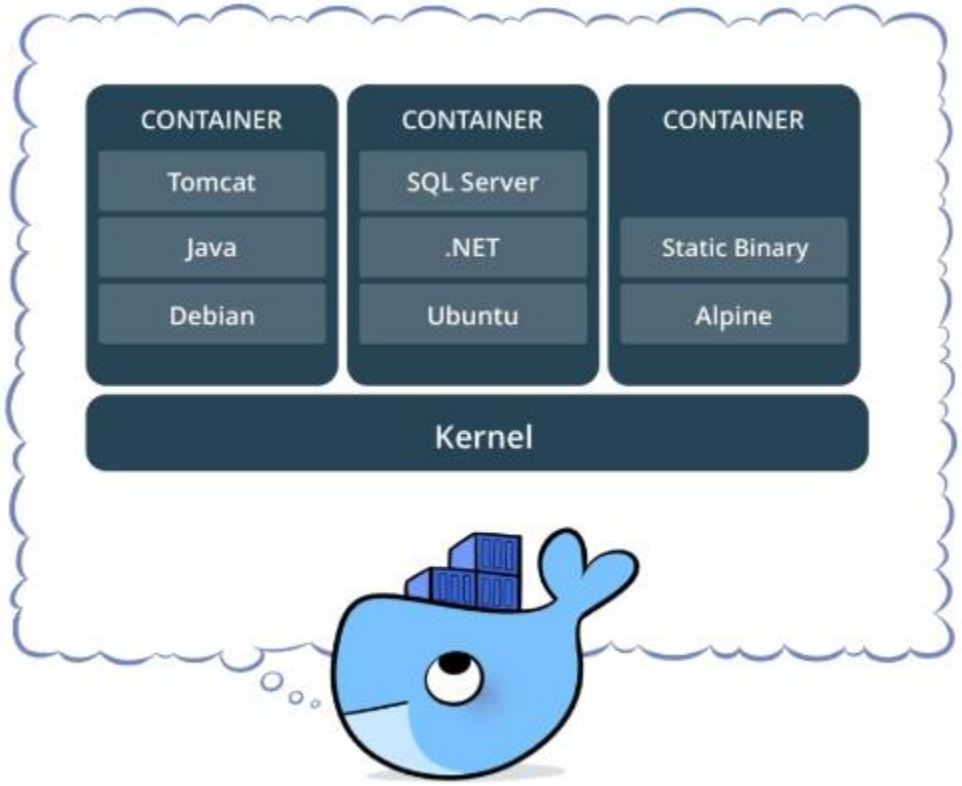
Less mature than Hyper-Visors

Maturing Eco-System

Docker Containers are NOT Virtual Machines

- Easy connection to ma
- Fundamentally
  - Different Architecture
  - Different Benfit’s

What is a container



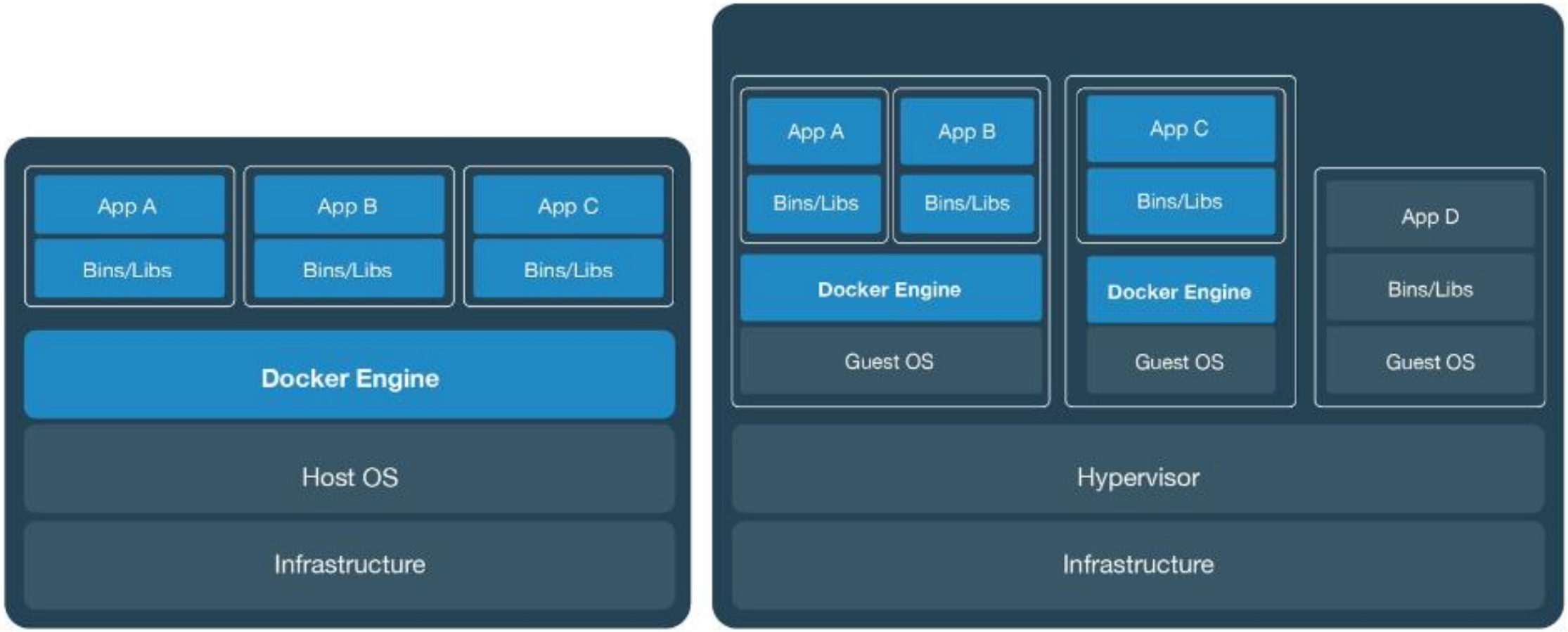
- Standardized Packaging for software and dependencies
- Isolated apps from each other
- Works for all major Linux distributions
- Containers native to Windows Server 2016

Standardized Packaging

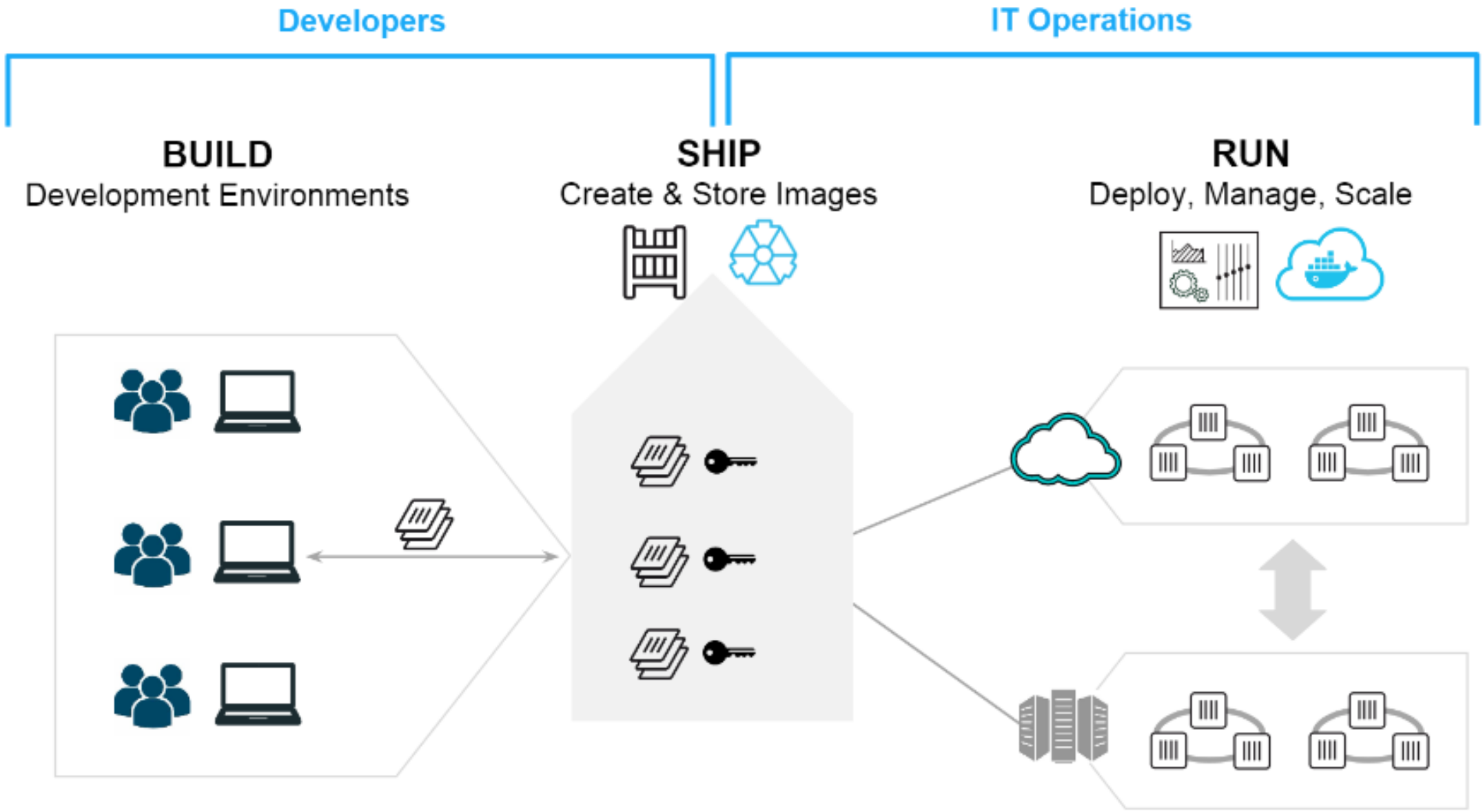
Beware of Smugglers

- Containers CAN contain malicious code
- Ensure your TRUST the content of the container

They are NOT Mutually Exclusive!



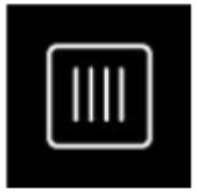
Using Docker



Vocabulary

Docker Image

- The basis of a docker container, represents a full application



Docker Container

- The standard unit in which the application service resides and executes

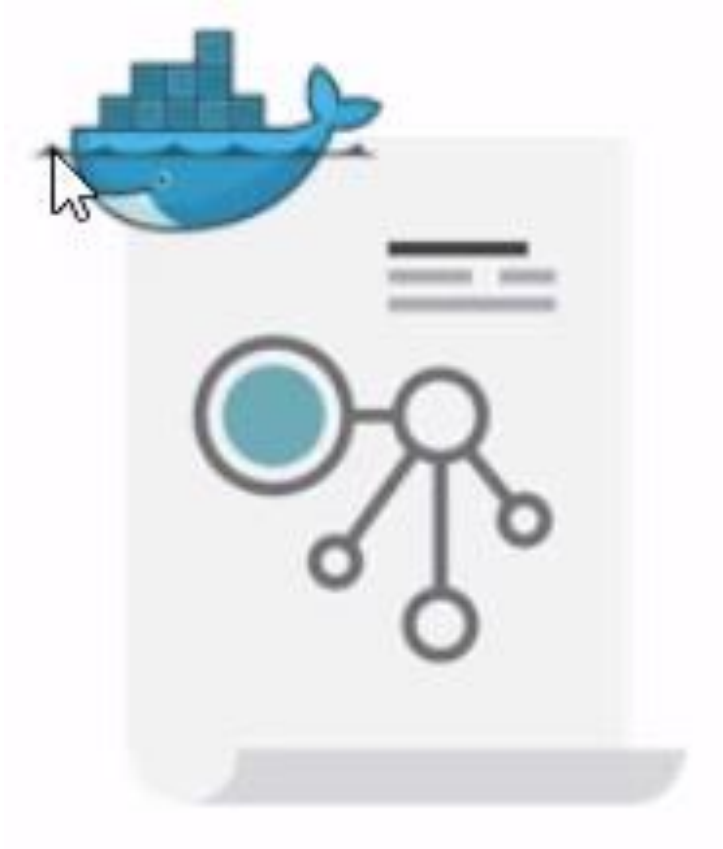
Docker Engine

- Creates, ships and runs Docker containers deployable on a physical or virtual, hosted locally, in a datacenter or cloud service provider

Registry Service

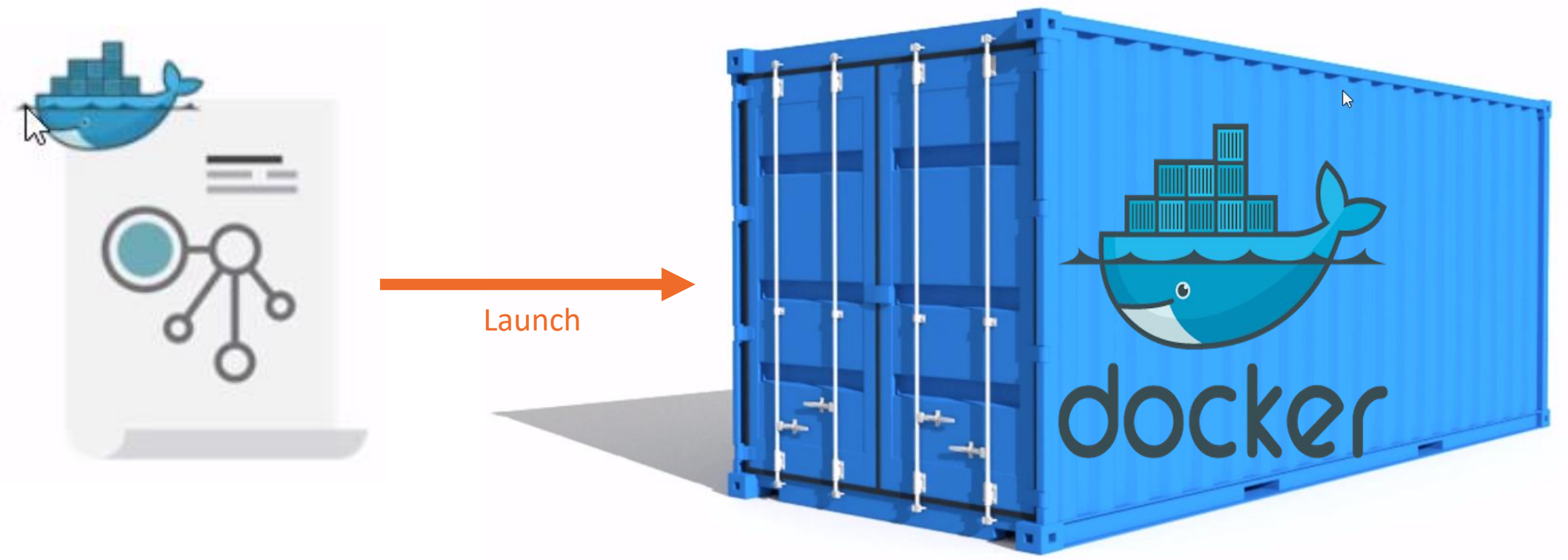
- Cloud or server based storage and distribution service for your

Docker Images

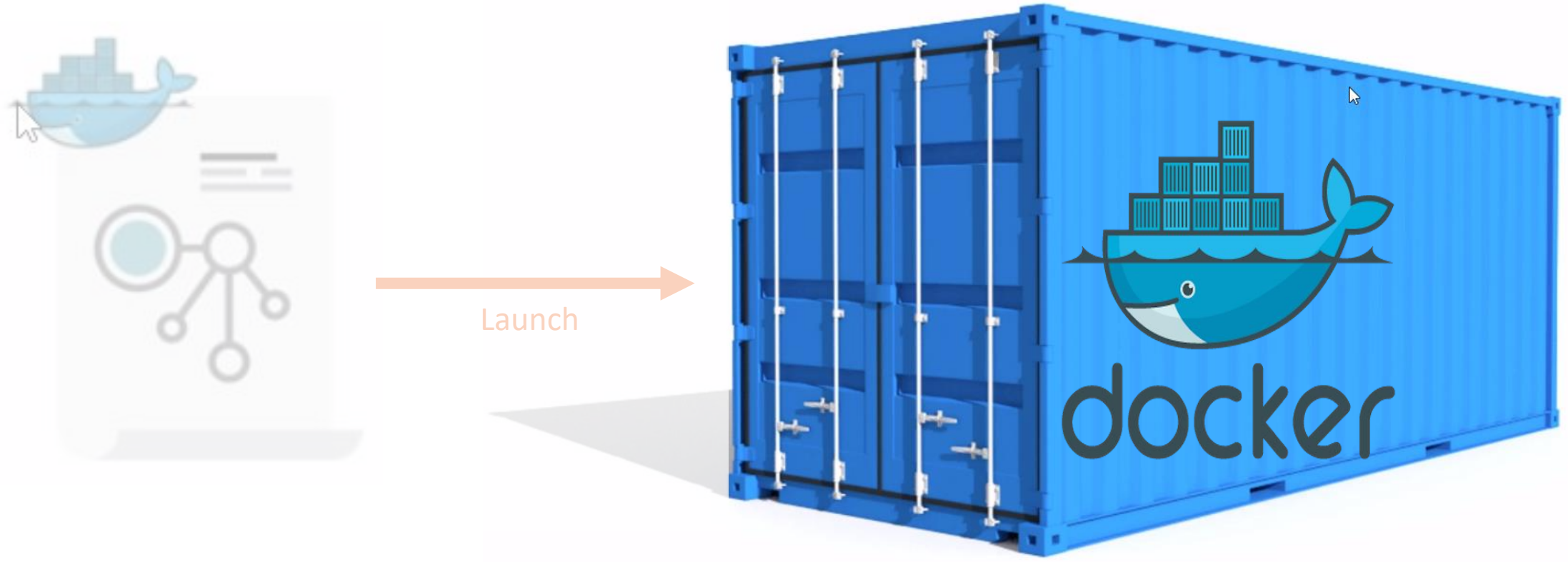


Docker Images

Docker Images



Docker Container’s



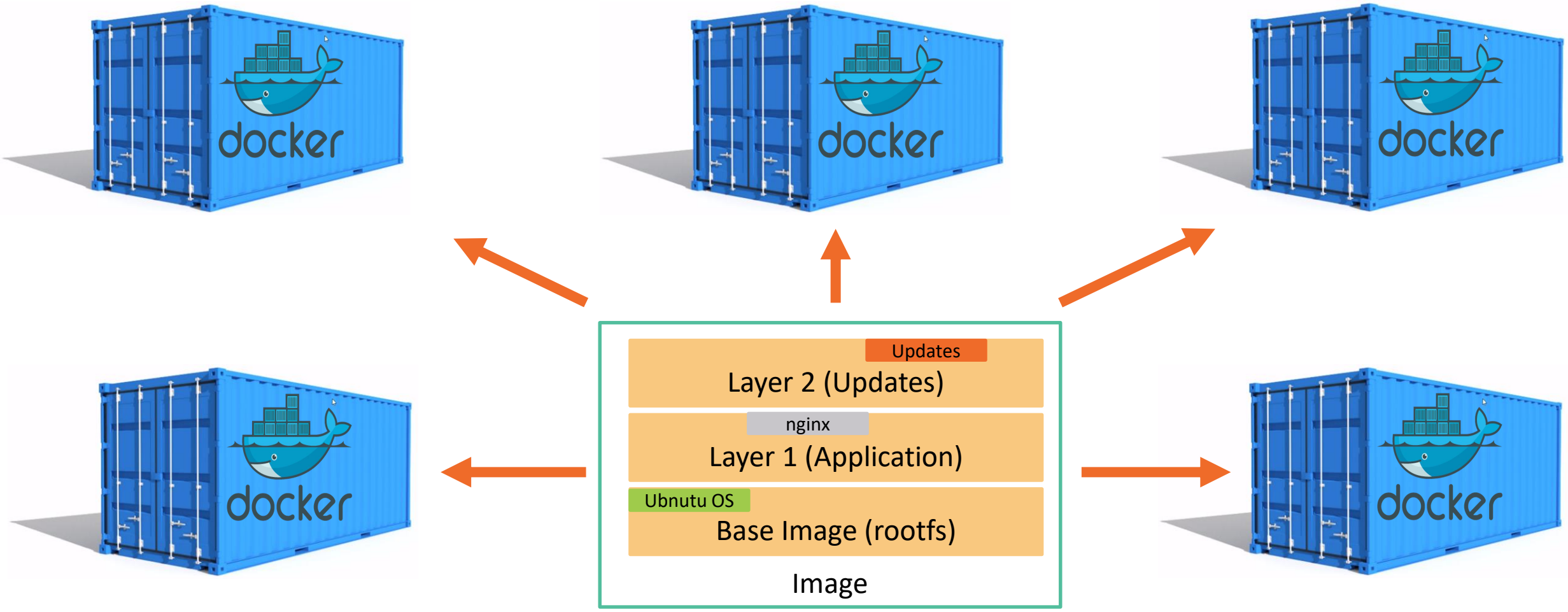
Image

Build Time

Container

Run Time

Images



Single Image deployed as Multiple Containers

Docker 101 – Hand On

- Check your images

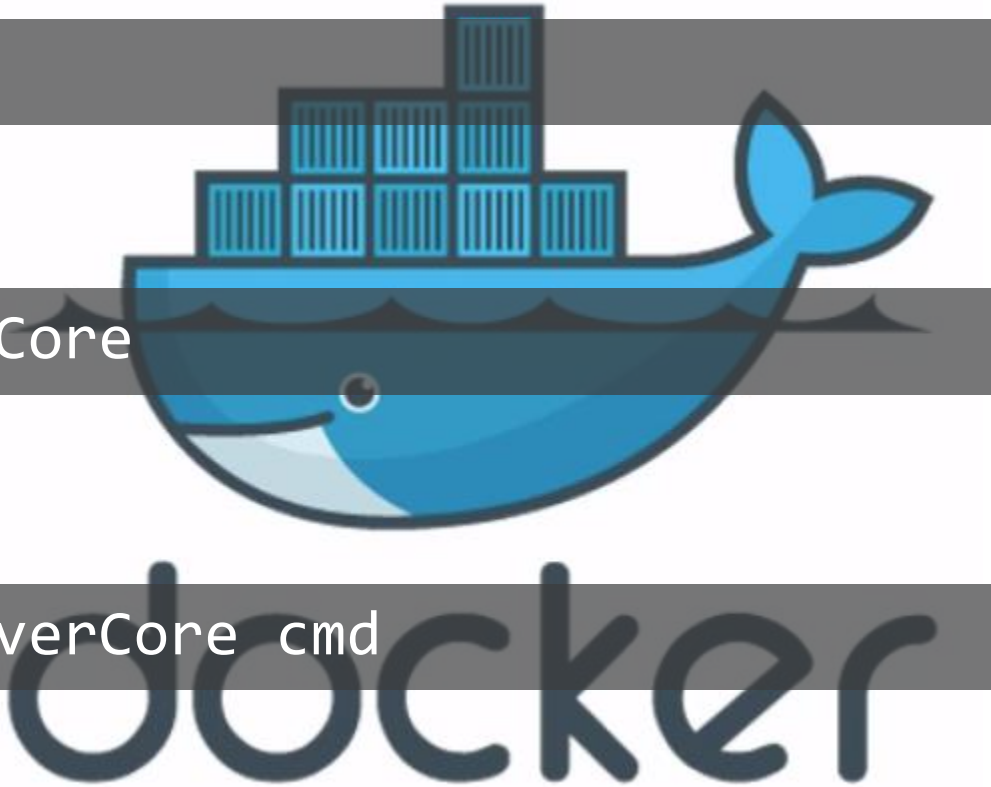
```
docker images
```

- Pull Sample Image

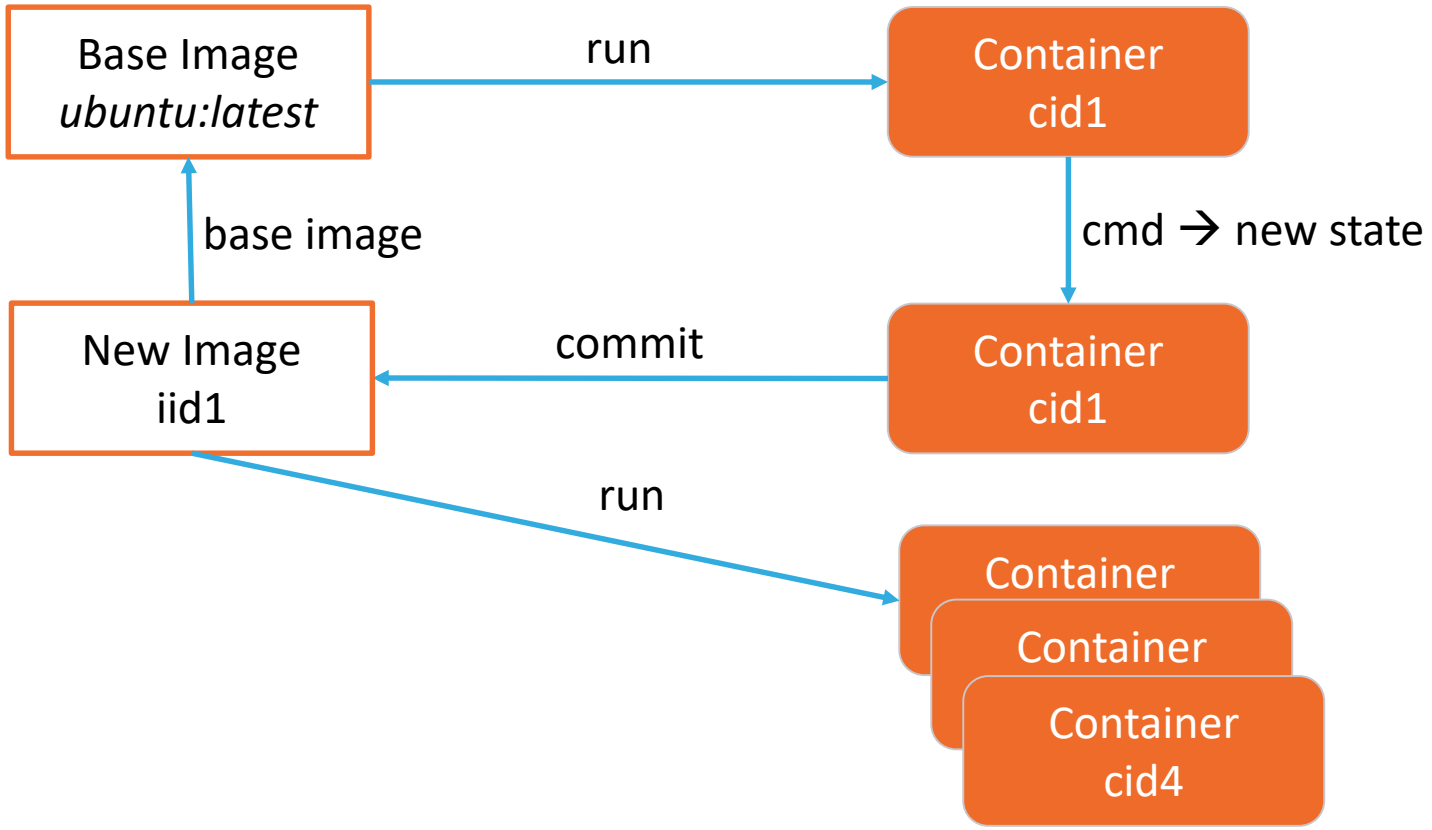
```
docker pull microsoft/WindowsServerCore
```

- Start your Container

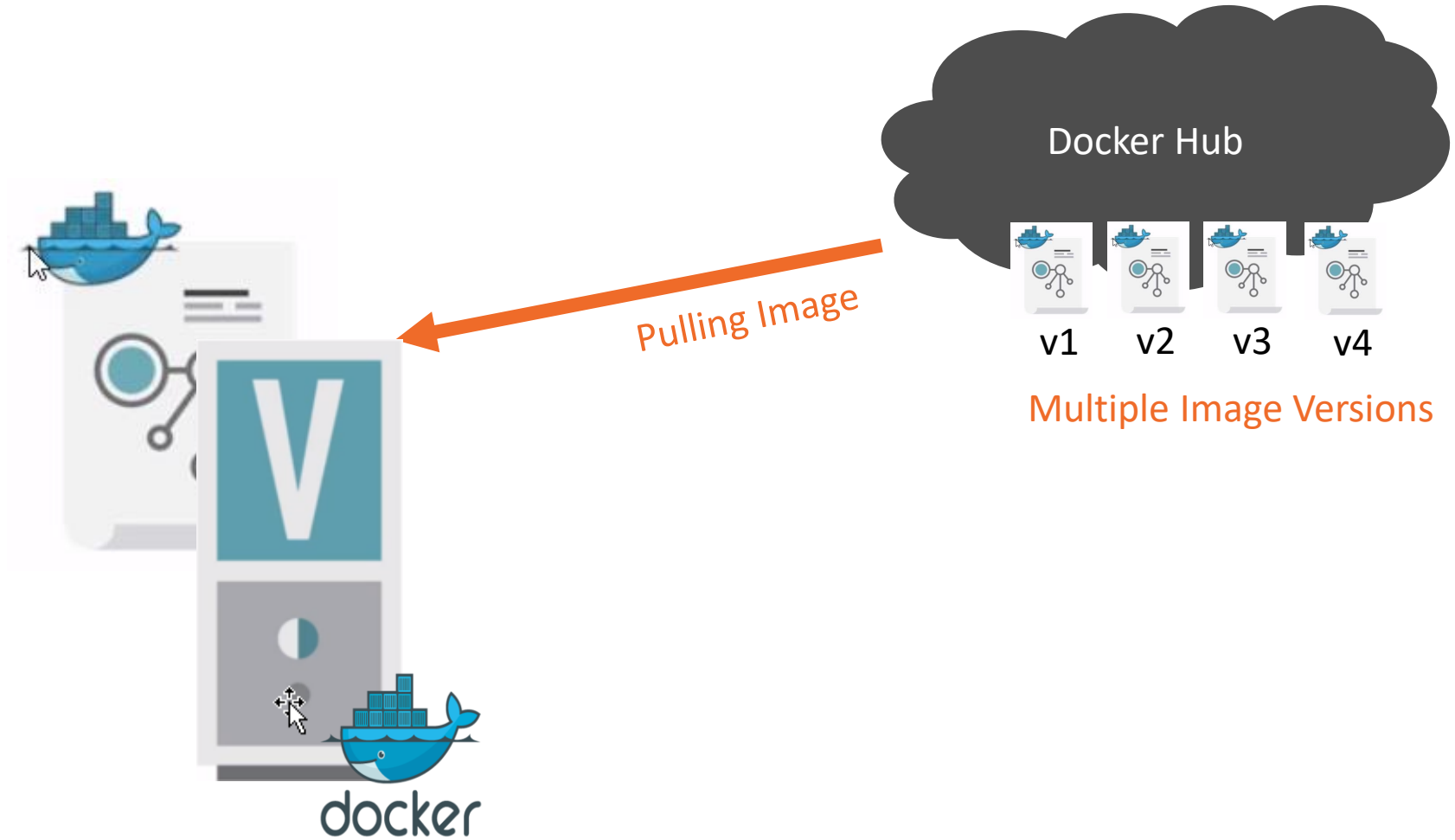
```
docker run -it microsoft/WindowsServerCore cmd
```



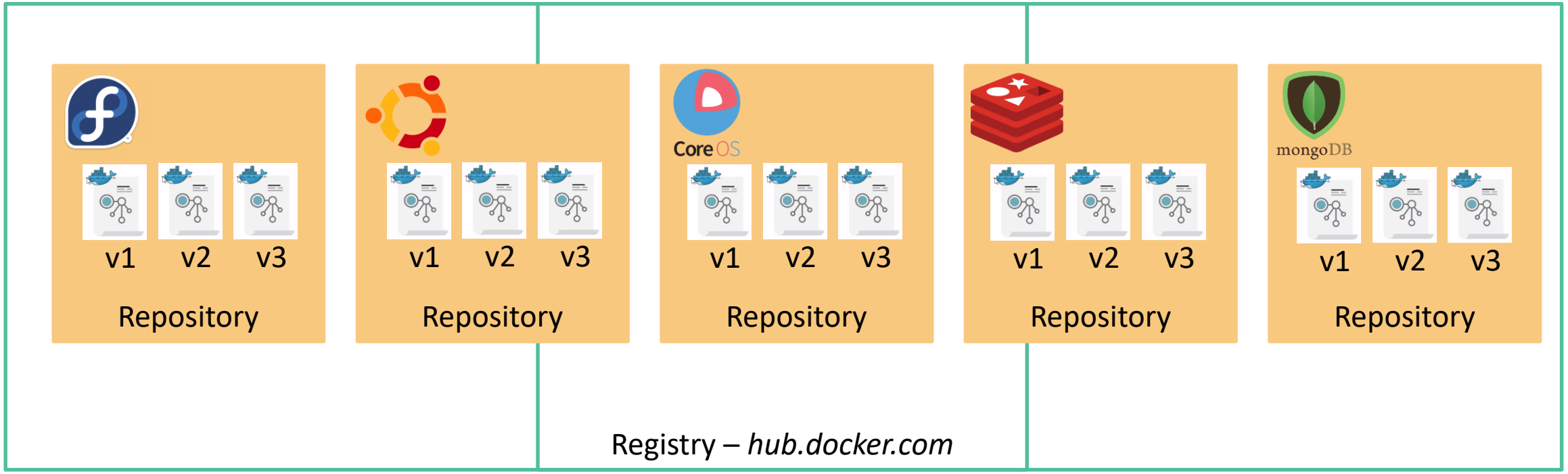
Images & Containers



Docker Images



Docker Registries



Summary

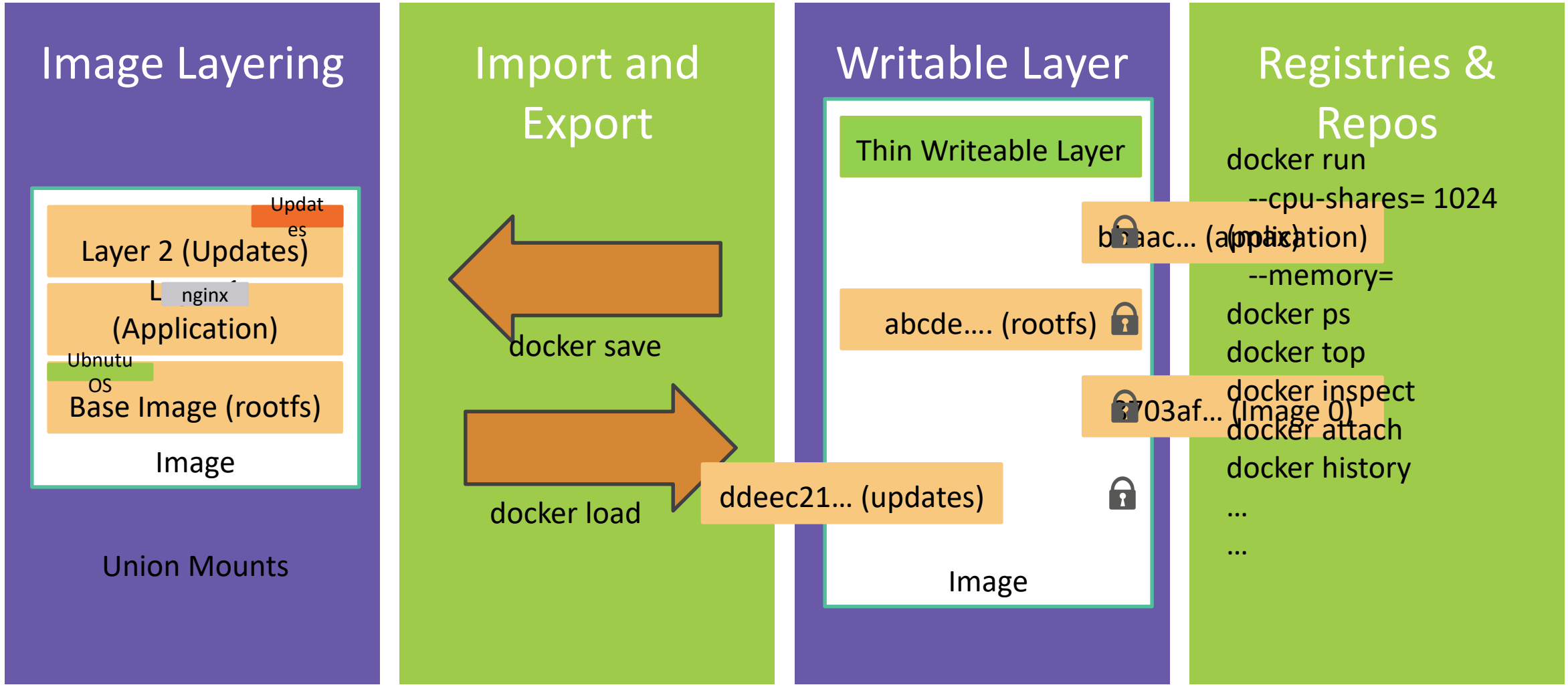
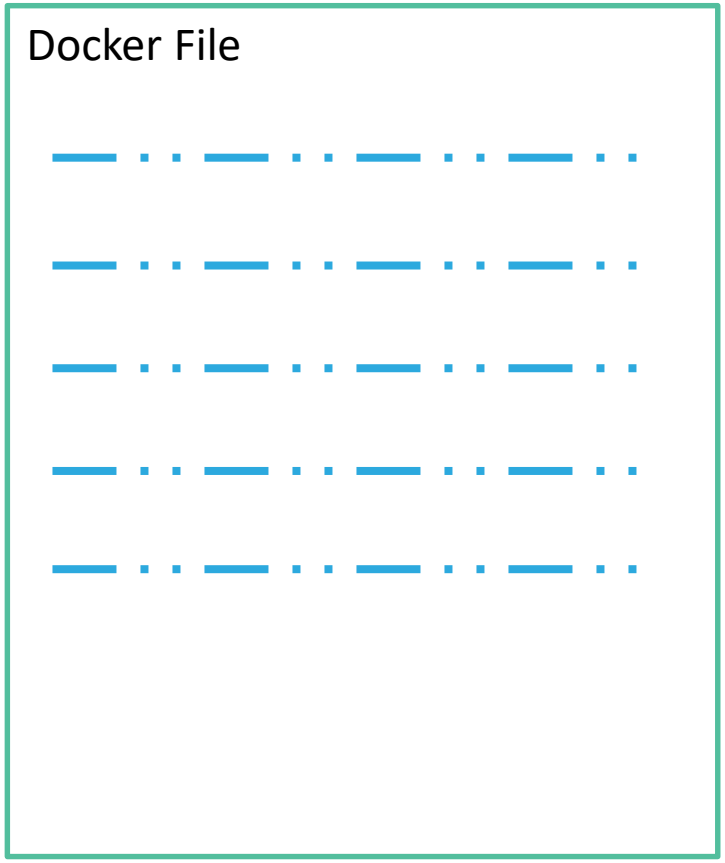


Image Layering
Updates
Layer 2 (Updates)
nginx
(Application)
Ubnutu OS
Base Image (rootfs)
Image
Union Mounts
Import and Export
docker save
docker load
Writable Layer
Thin Writeable Layer
abcde…. (rootfs)
ddeec21… (updates)
Image
Registries & Repos
docker run
--cpu-shares= 1024
(options)
--memory=
docker ps
docker top
docker inspect
docker attach
docker history
…
…

Dockerfile

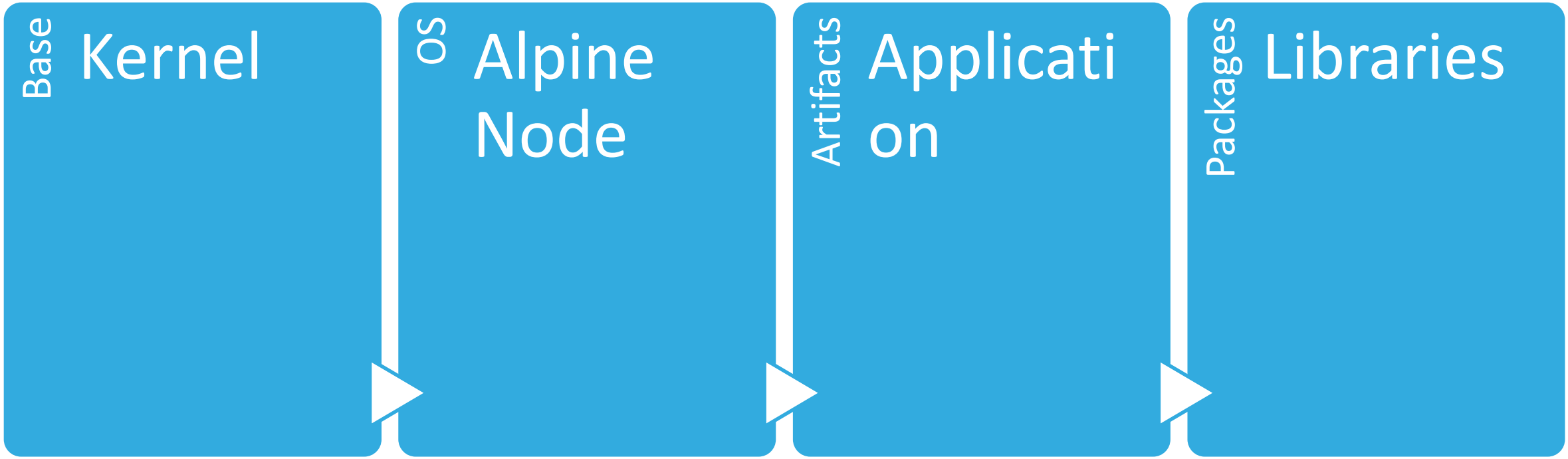
- ‘Dockerfile’
- Plain Text
- Simple Format
- Instructions to Build image
- Start in an empty directory



Dockerfile

```
# Our Base Image

FROM node:4.6

# Set our working folder

WORKDIR /app

# Add any files in our directory recursively to the image

ADD . /app

# Start the Node Package manager and install requested packages

RUN npm install

# What port number the container should expose

EXPOSE 3000

# run the application

CMD npm start
```

Each command creates a layer



Dockerfile

```
docker build -t helloworld:0.1 .
```

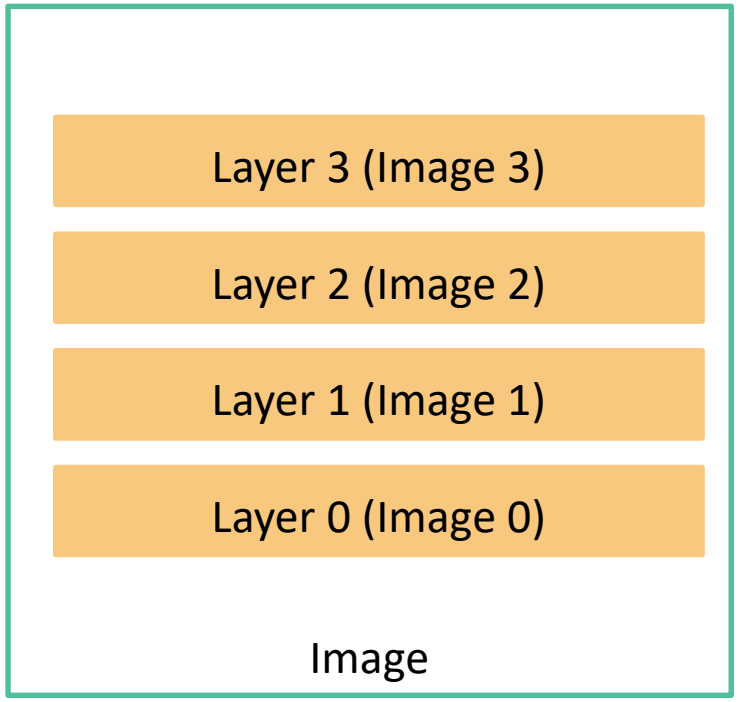
Layers on a Physical Disk

- Logical file system by grouping different file system primitives into branches
  - Directories, File Systems, Sub volumes, Snapshots
- Each Branch represents a layer in a docker image
- Containers will share common layers on the host
- Enabled images to be constructed as required, instead of monolithic images
- At startup the container adds a writeable layer on top of this file system

Copy on Write

- Super efficient
  - Sub second instantiation times from containers
  - New container can consume < 1Mb of Space
- Containers appears to be a copy of the original image
  - Really its just a link to the original shared image
- A change to the file system, will presented as “Copied Up”

Images


Layer 3 (Image 3)
Layer 2 (Image 2)
Layer 1 (Image 1)
Layer 0 (Image 0)
Image

Images

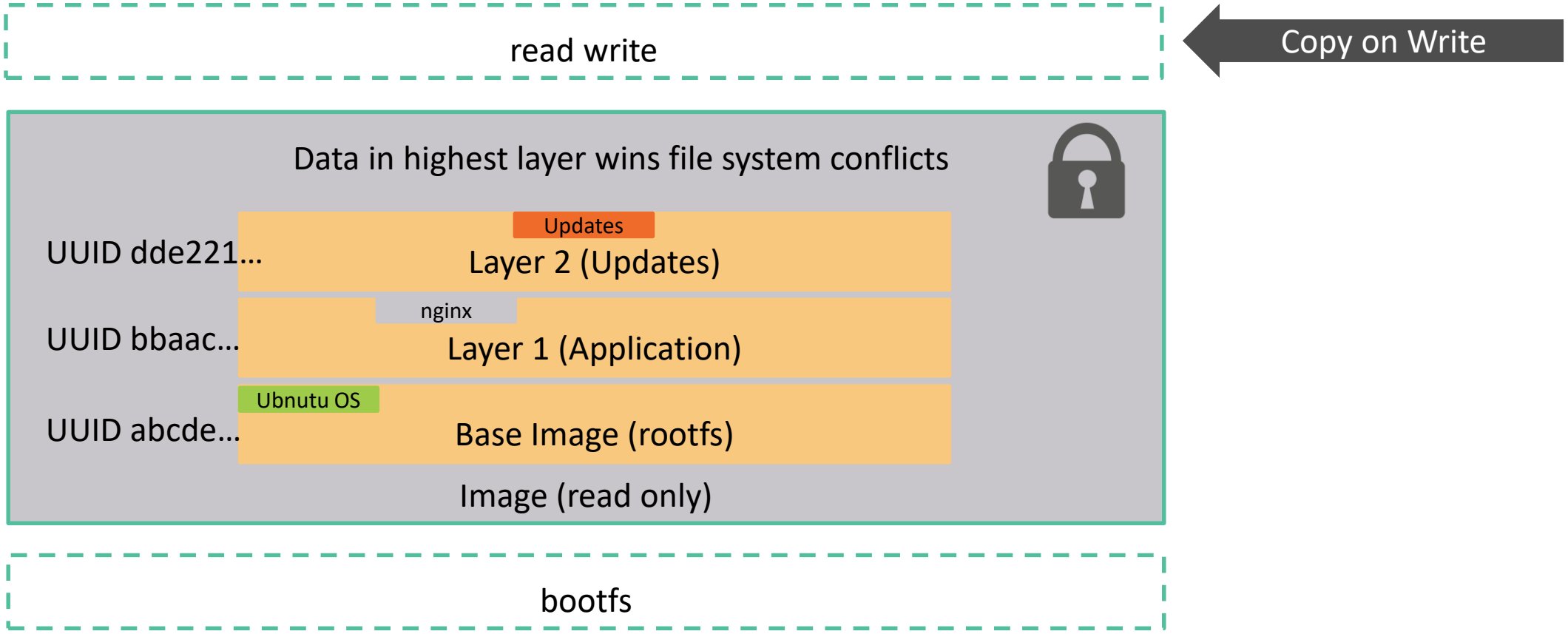
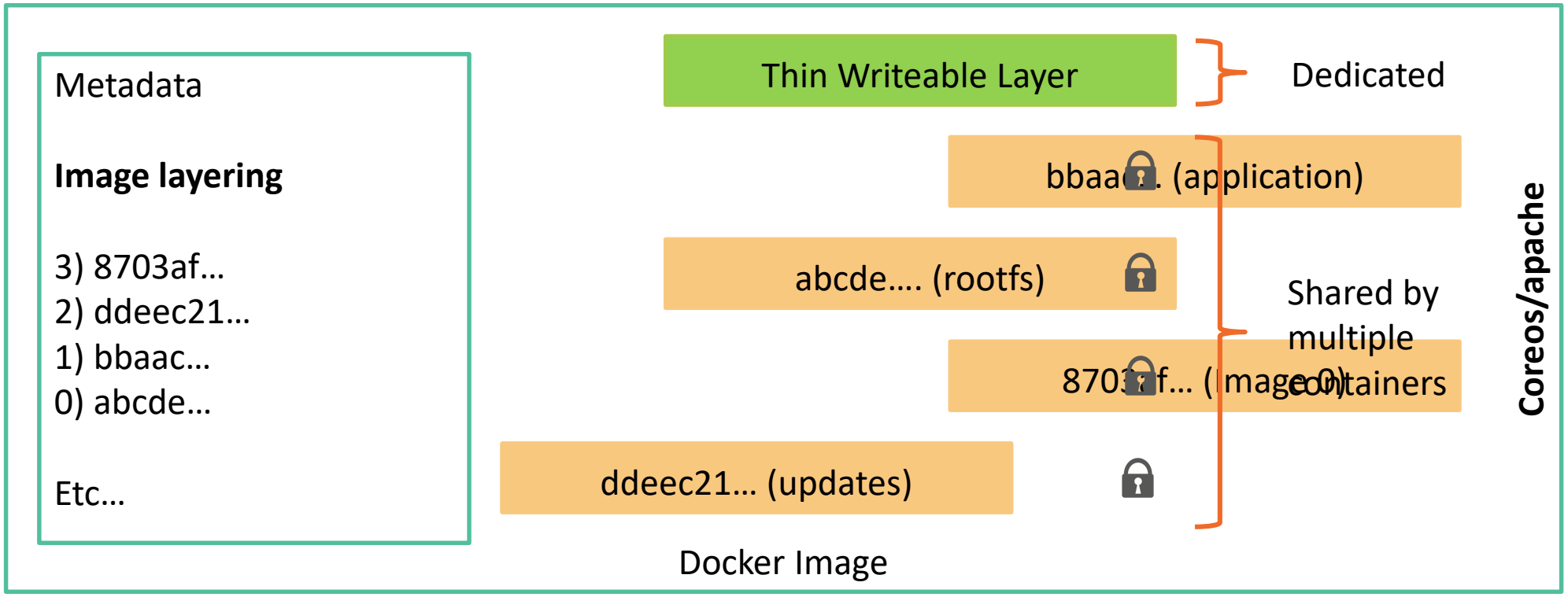


Image layering is accomplished through **union mounts**

Containers

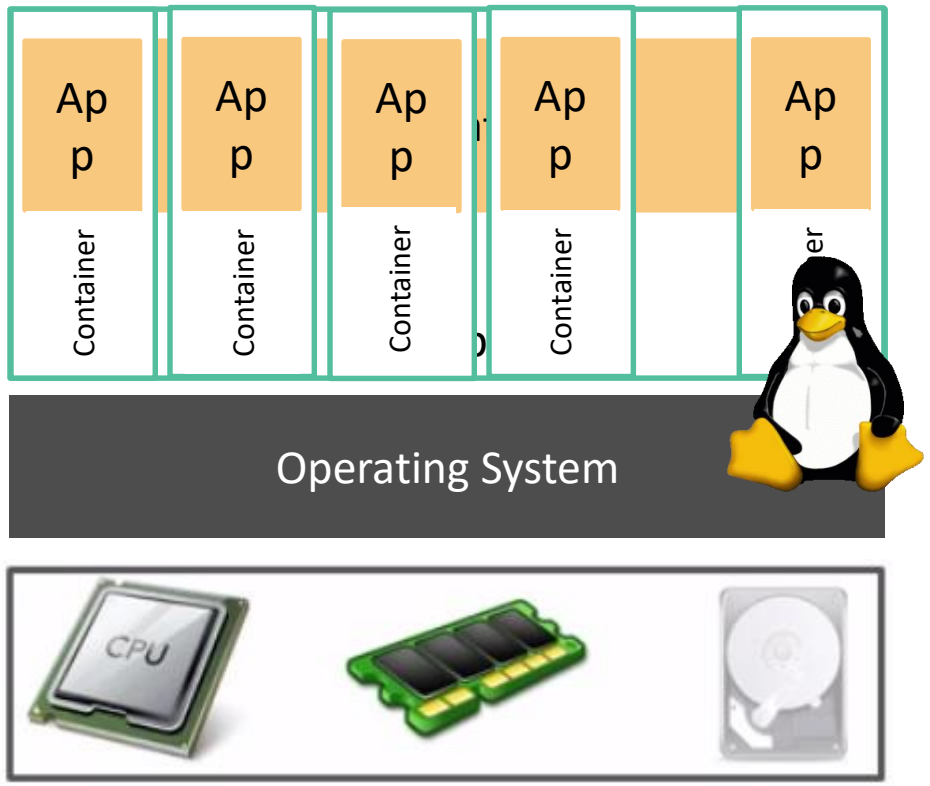


Every **container** gets its own writable top layer

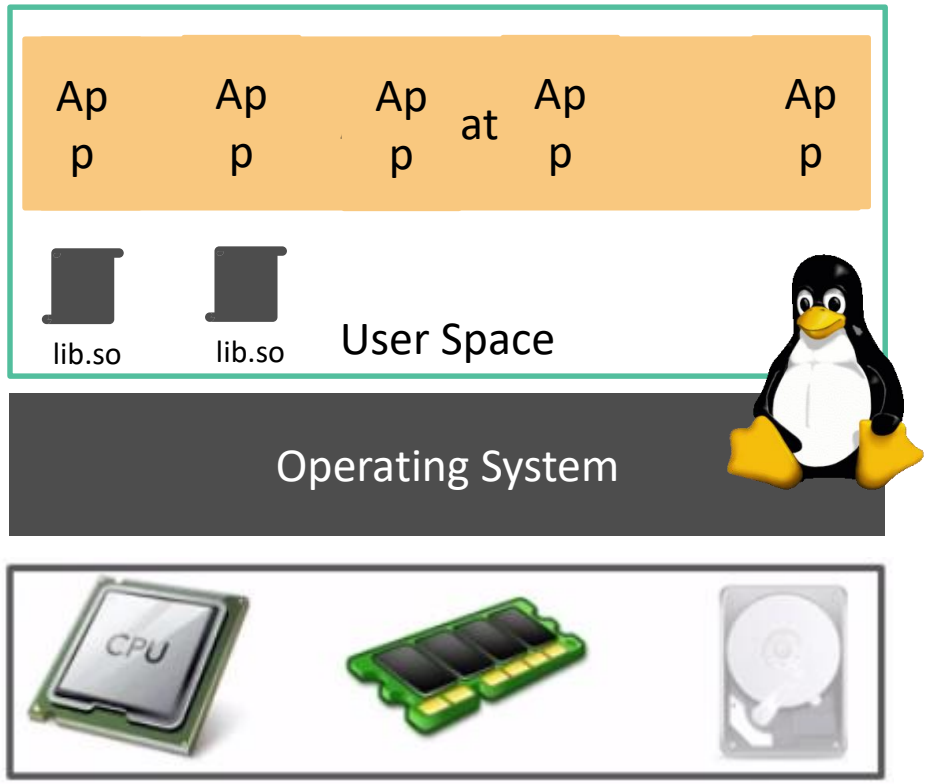
Resources

Slides and demos from the conference will be available at github.com/nordicinfrastructureconference/2018 (bit.ly/2y7JhA3)

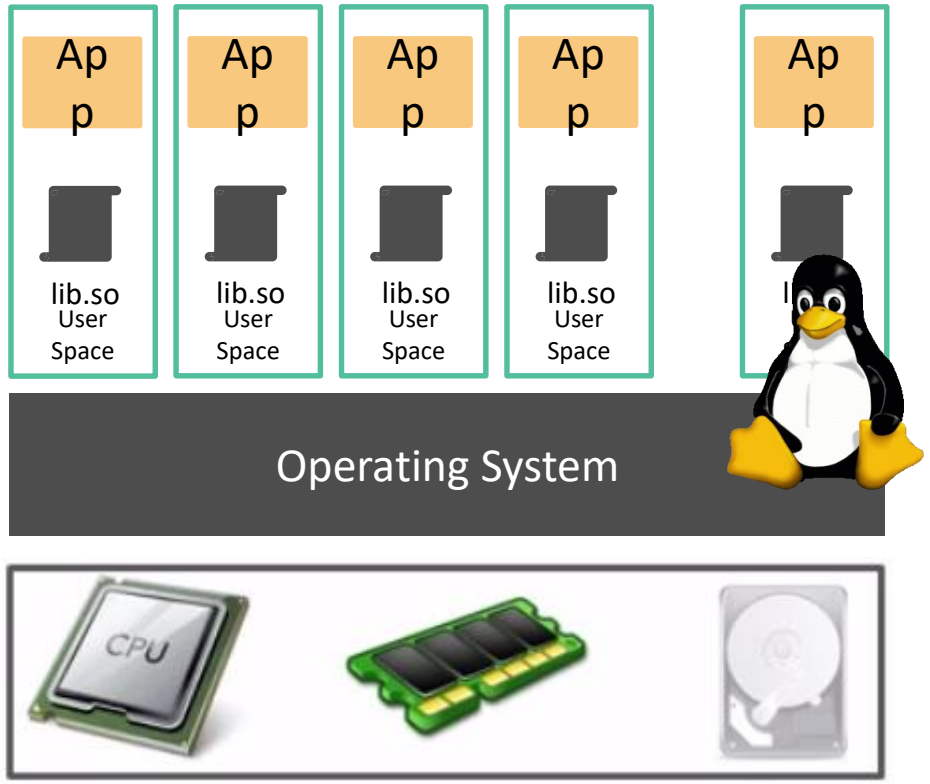
Containers


App
Container
App
Container
App
Container
App
Container
App
Operating System
CPU

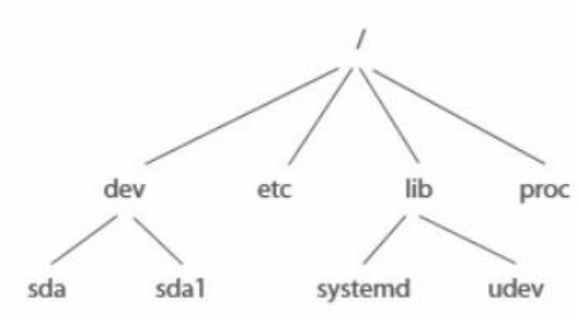
How Containers Work

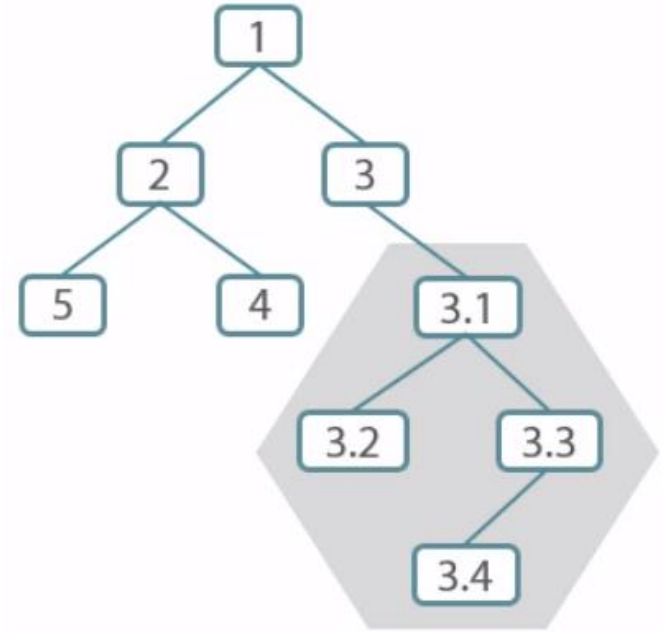


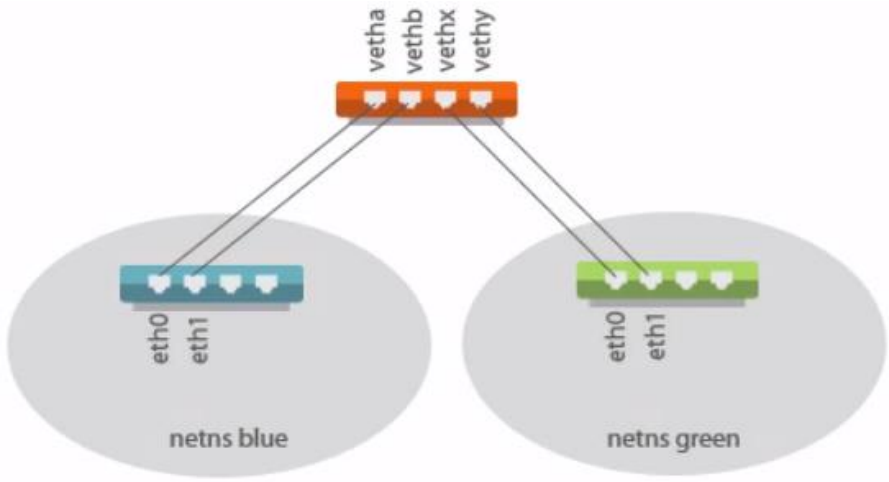
How Containers Work



How Containers Work





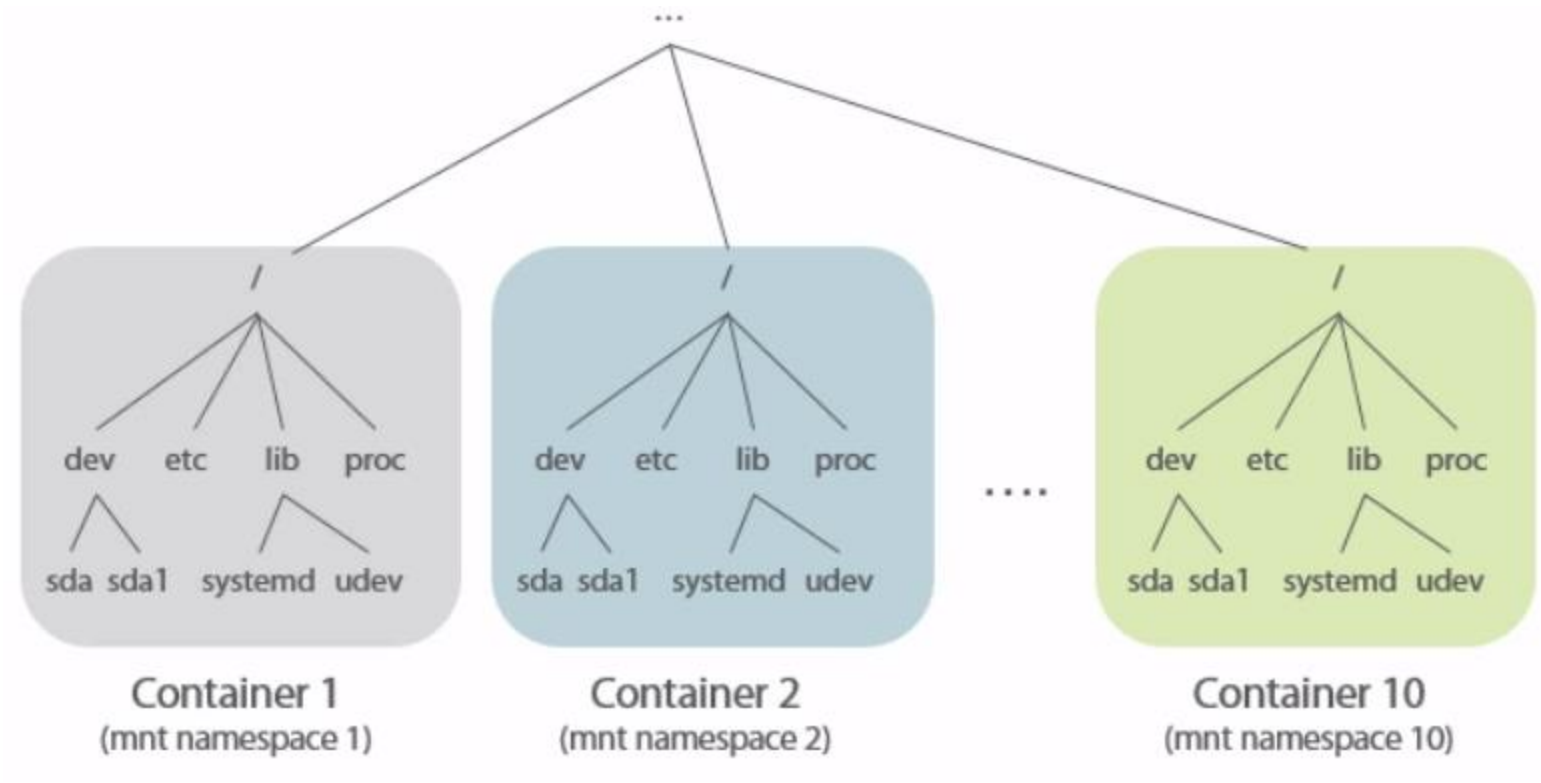


File System

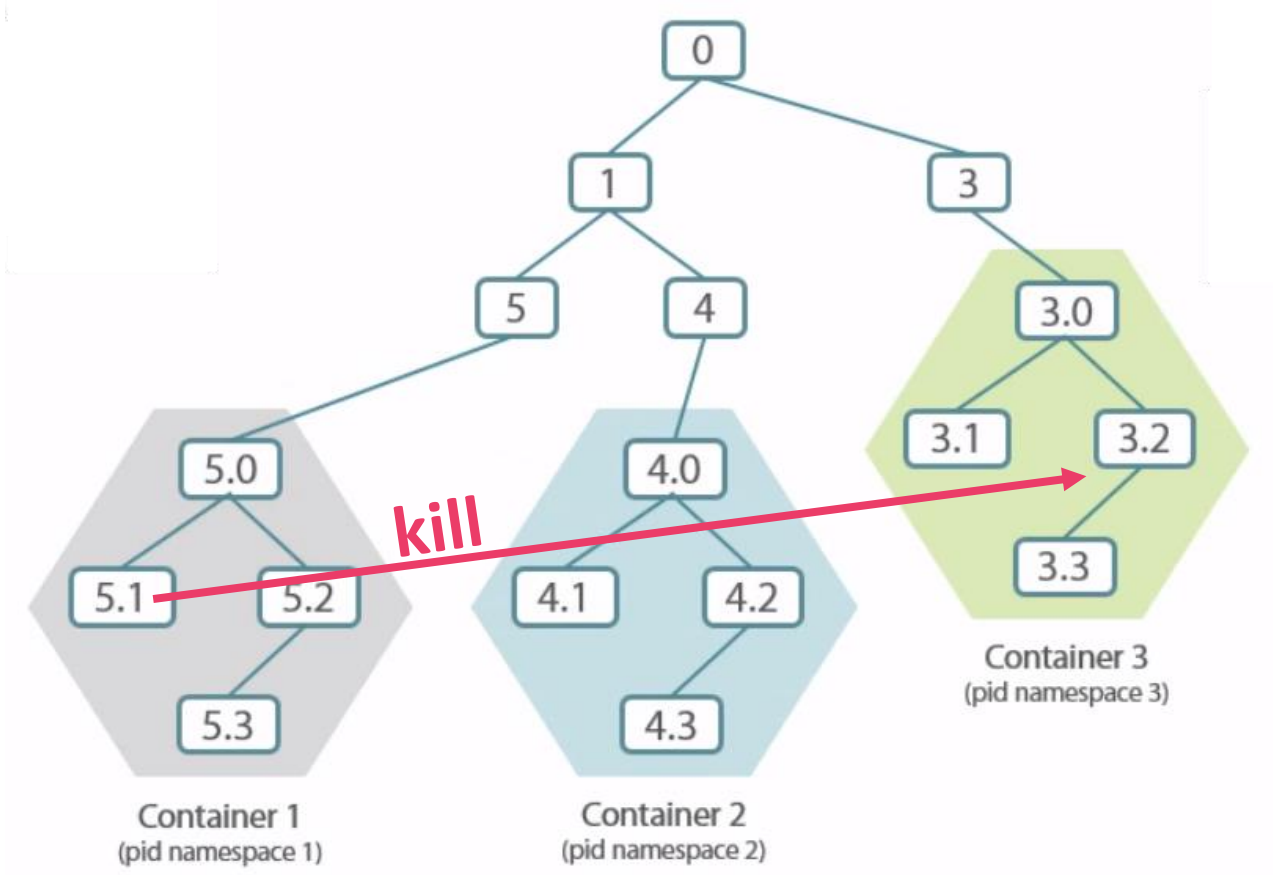
Process Tree

Networking

How Containers Work



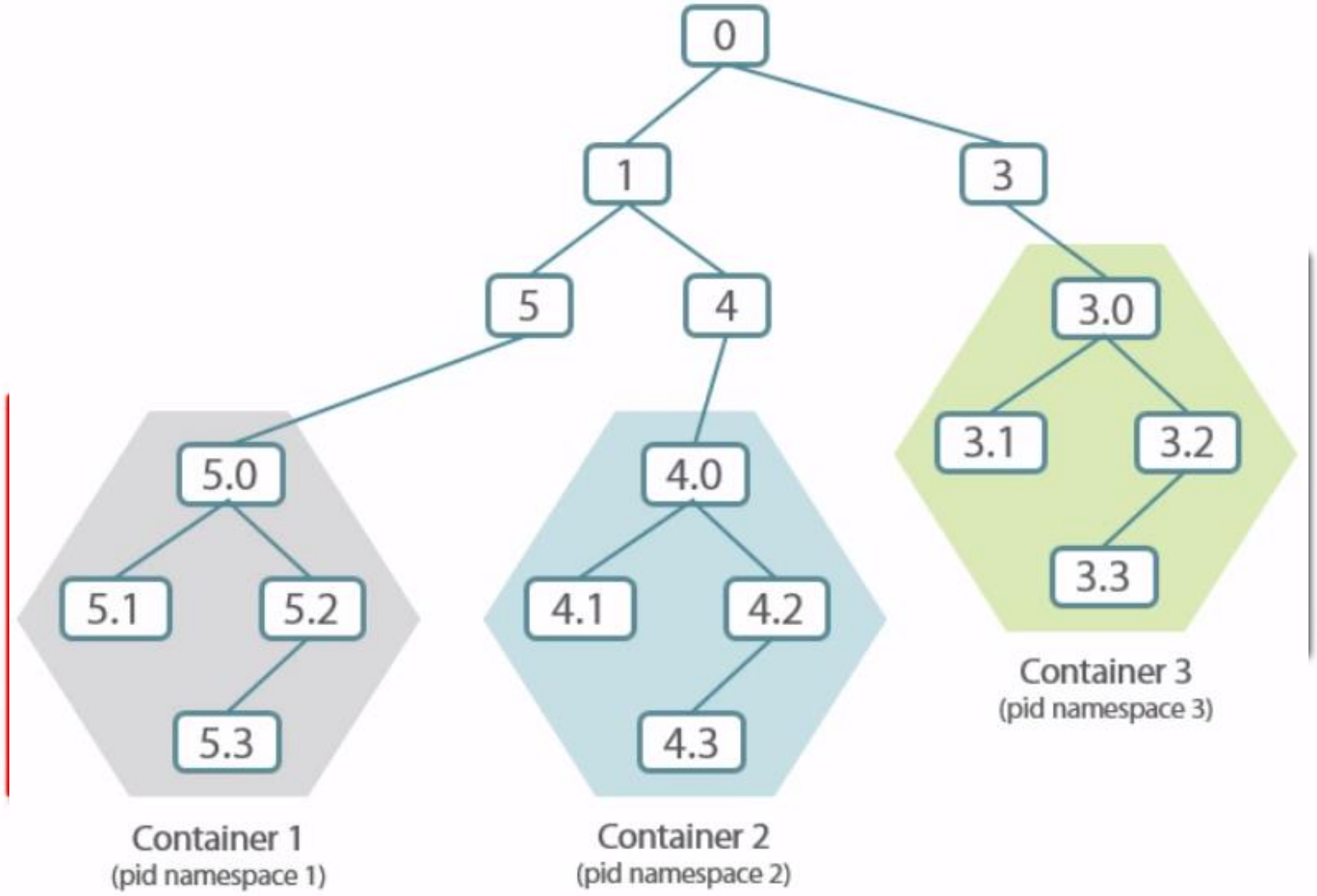
How Containers Work



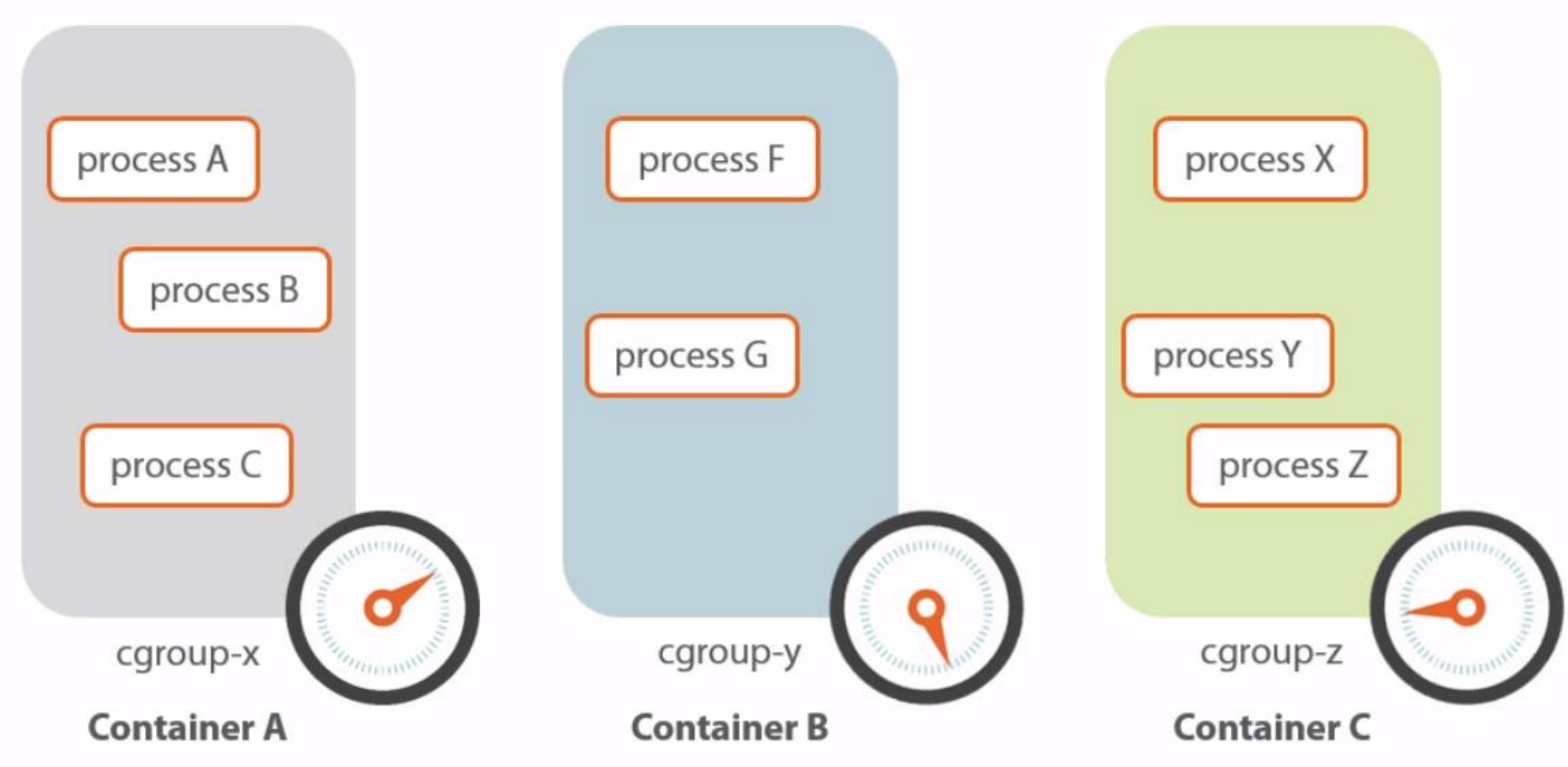
Process Tree

How Containers Work – Kernel Namespaces

The pid Namespace



How Containers Work – Control Groups



How Containers Work – Capabilities

Personal Preparation

- Knowledge
- Experience