Docker Introduction

Rajesh Kumar

DevOps Architect

@RajeshKumarIN | www.RajeshKumar.xyz

What We Will Learn

- Linux Containers
  - Containers vs Virtuals Machines << FIGHT!!
  - Kernal namesspaces, cgroups, Capabilities…
- Docker Engine
  - Execution Driver" libcontainer vs LXC
  - AUFS, OverlayFS, Device Mapper…
- Docker Images
  - docker build | docker images | docker inspect
  - Union mounts, Layering, Dockerfile
- Linux Containers
  - Docker start|stop|restart
- Registers, Volumes, Networking

Prerequisites

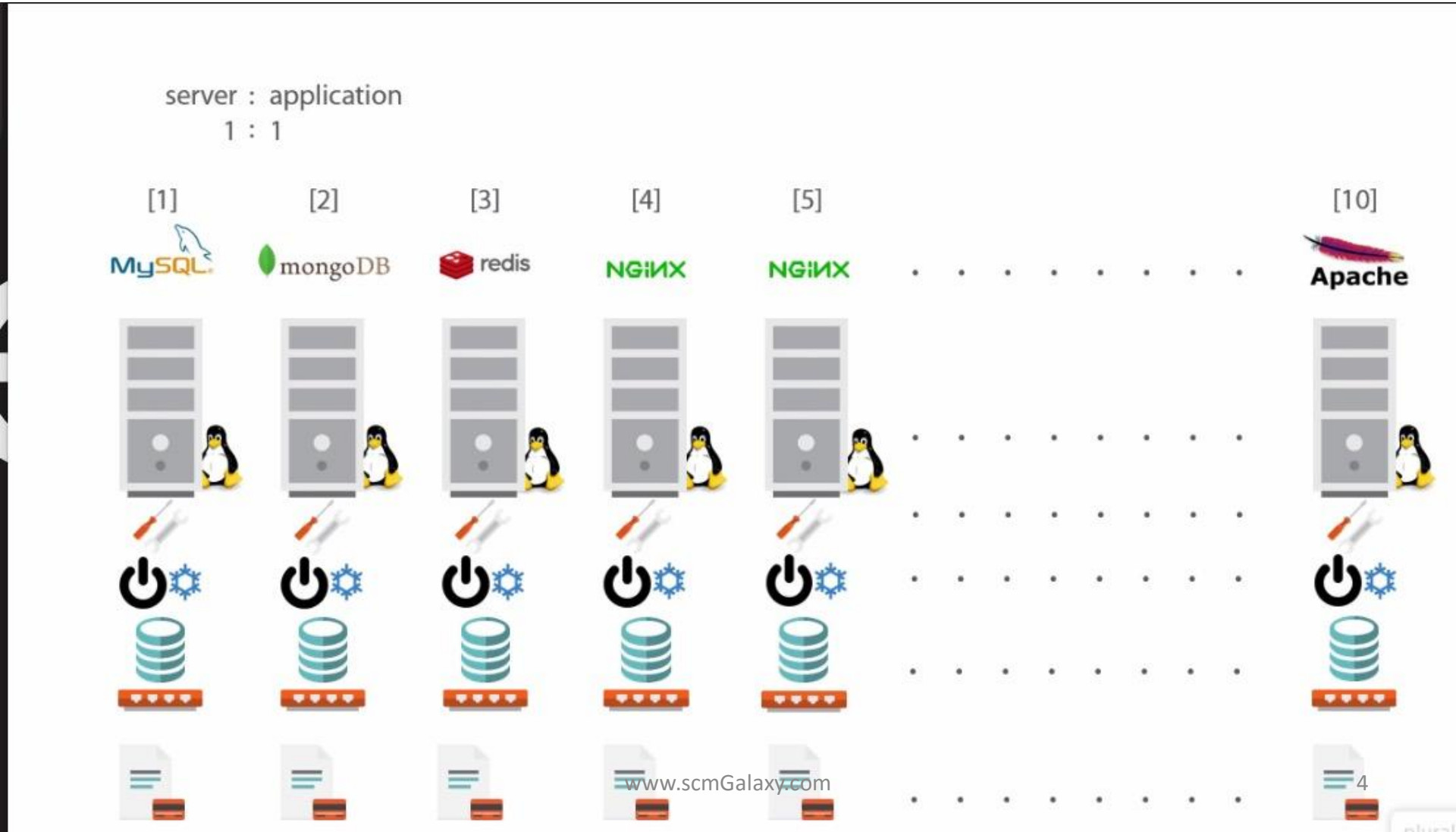
You

- Basic Computer Knowledge
- Do not need to be Linux expert

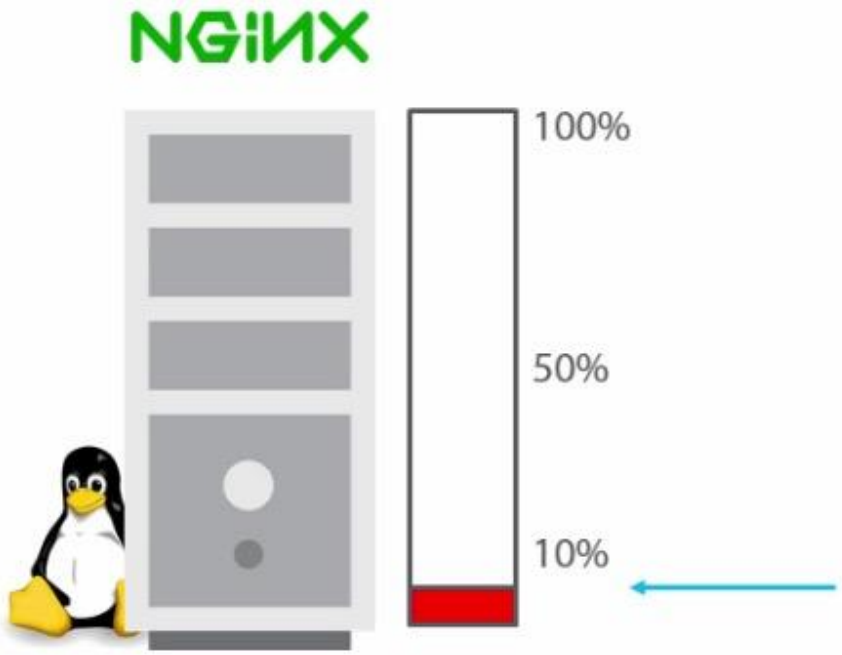
Computer

- 1 -2 Linux Machine (Can be Vms)

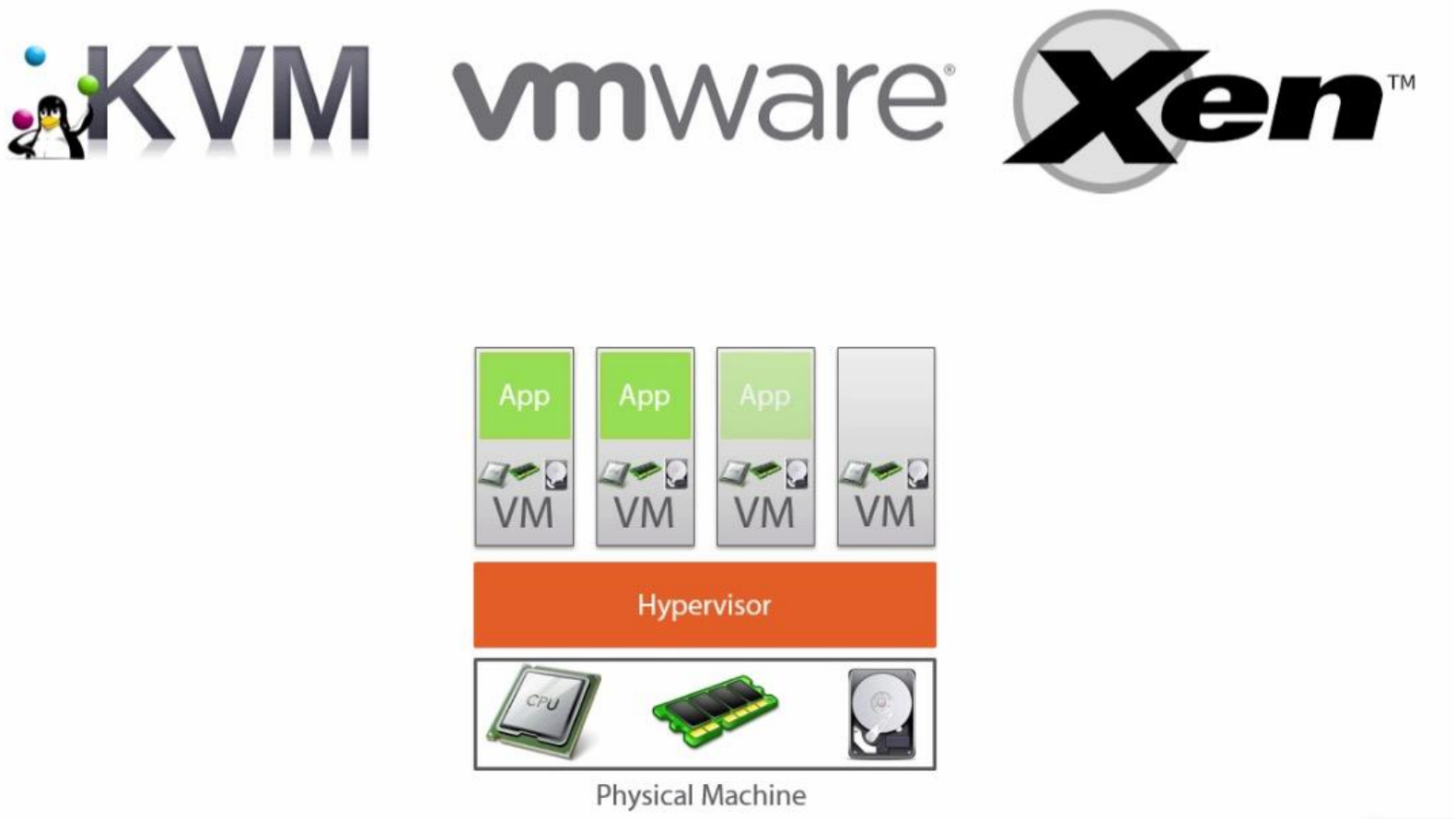
Introducing Containers

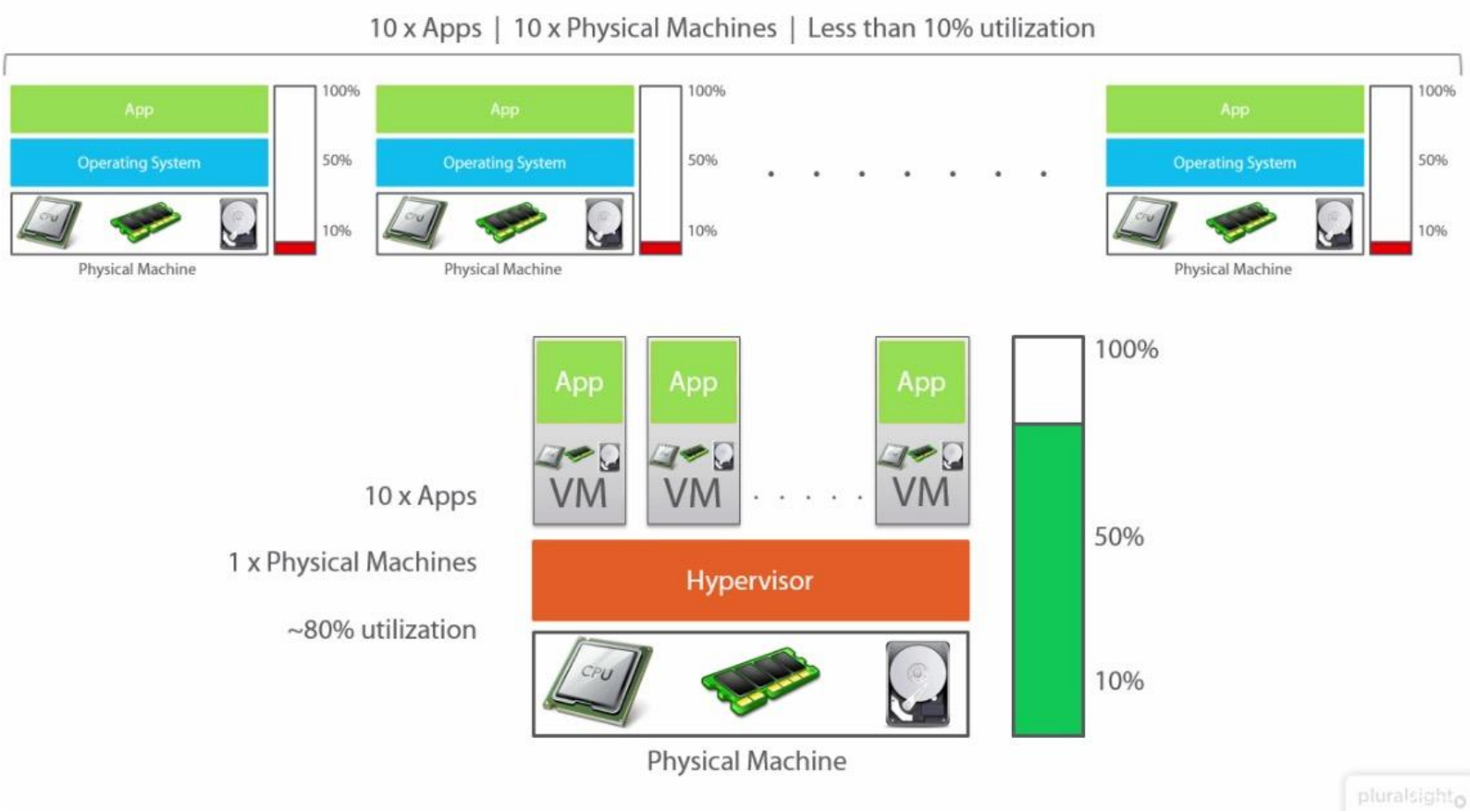


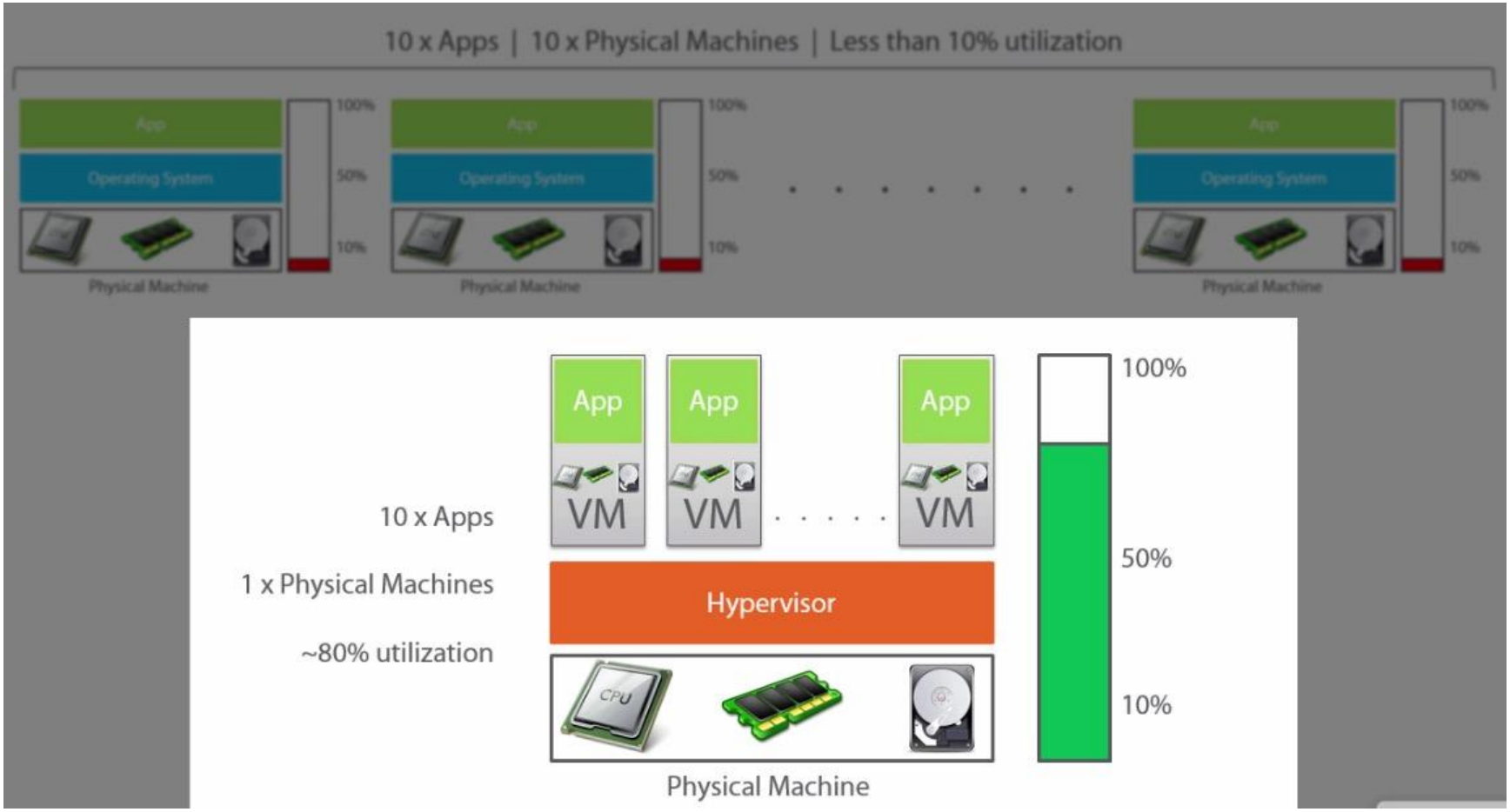
What a waste



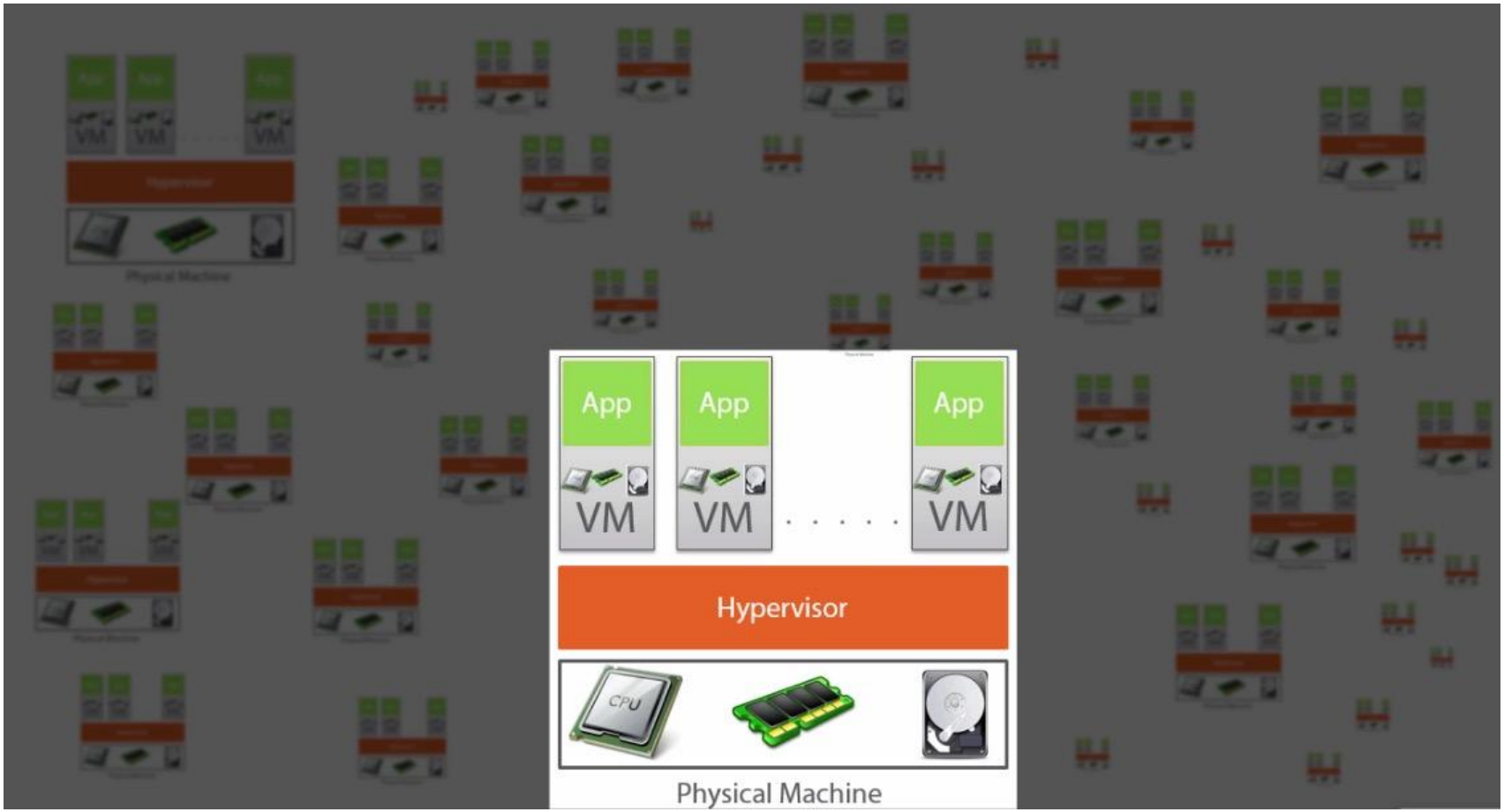
Solution - Virtualization



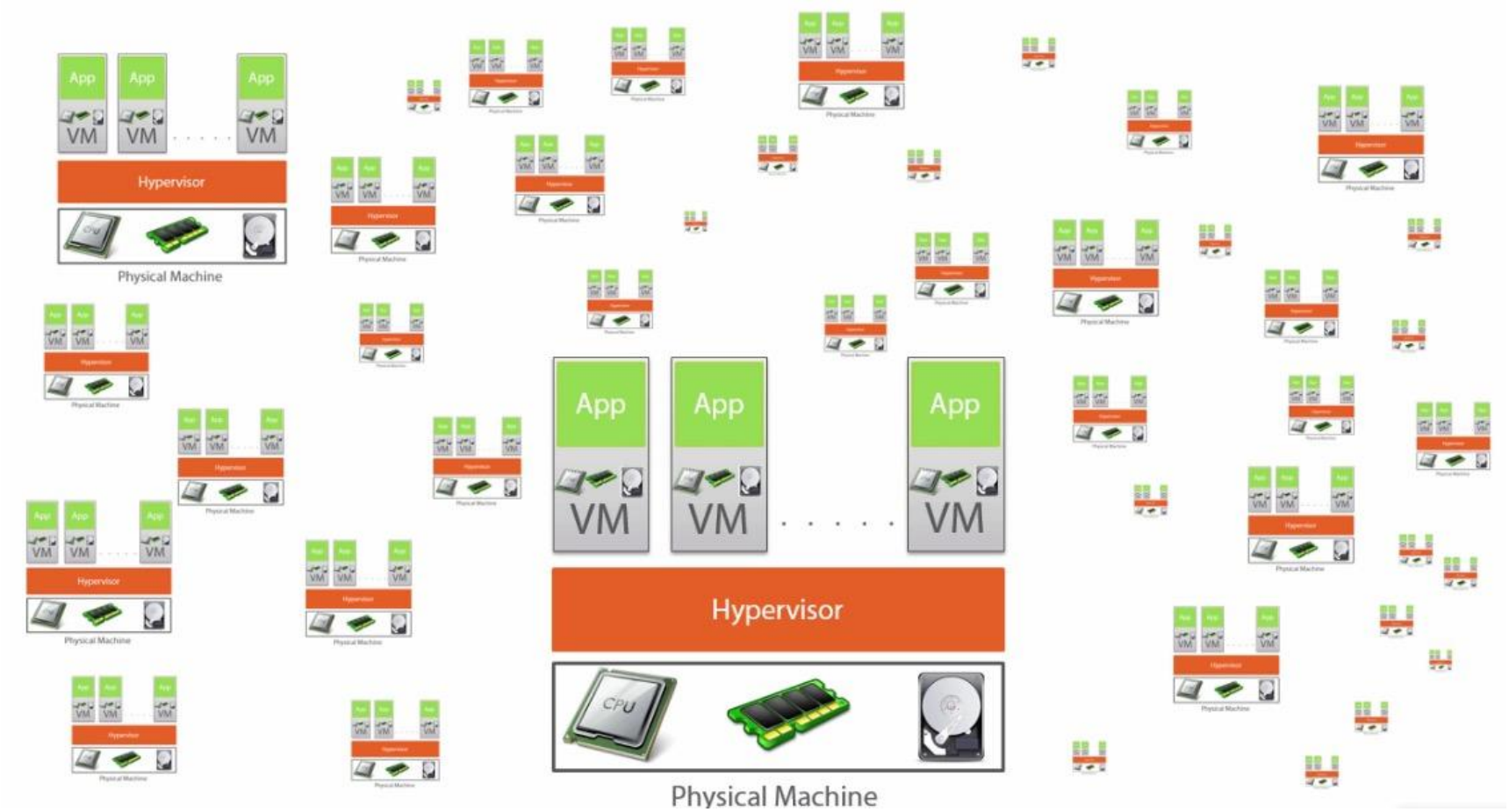

10 x Apps | 10 x Physical Machines | Less than 10% utilization
App
Operating System
100%
50%
10%
Physical Machine
App
Operating System
100%
50%
10%
Physical Machine
App
Operating System
100%
50%
10%
Physical Machine
10 x Apps
1 x Physical Machines
~80% utilization
App
App
App
VM
VM
VM
Hypervisor
CPU
100%
50%
10%
Physical Machine
pluralsight

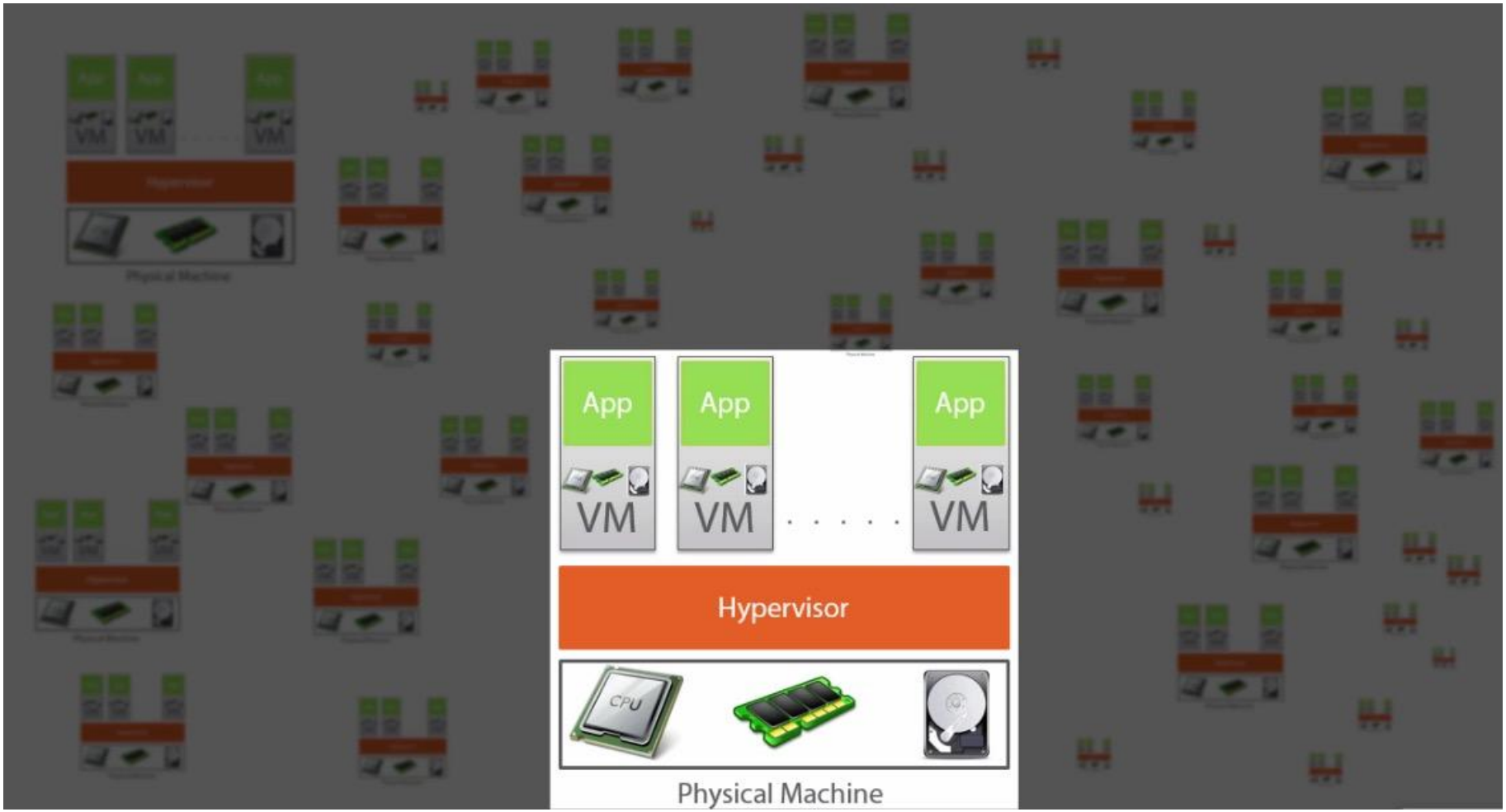

10 x Apps | 10 x Physical Machines | Less than 10% utilization
App
Operating System
100%
50%
10%
Physical Machine
10 x Apps
1 x Physical Machines
~80% utilization
App
VM
Hypervisor
CPU
100%
50%
10%
Physical Machine

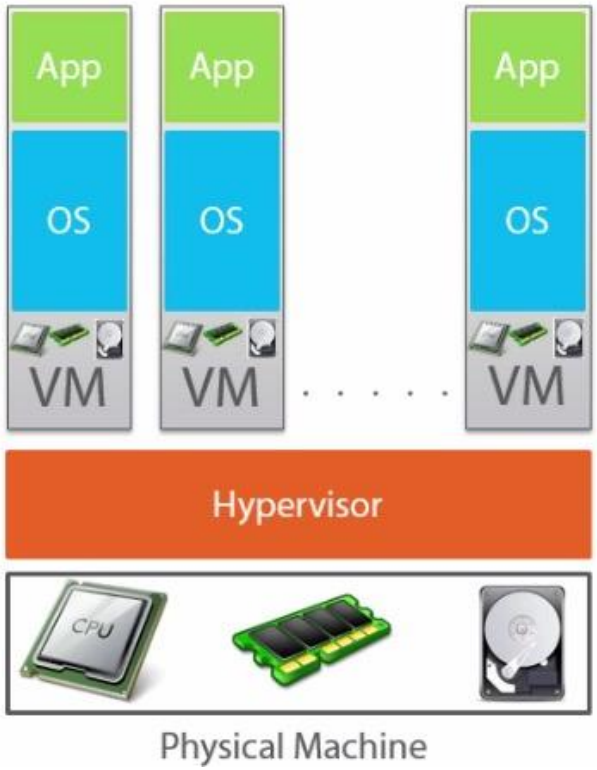
Virtualization is not perfect – Why?



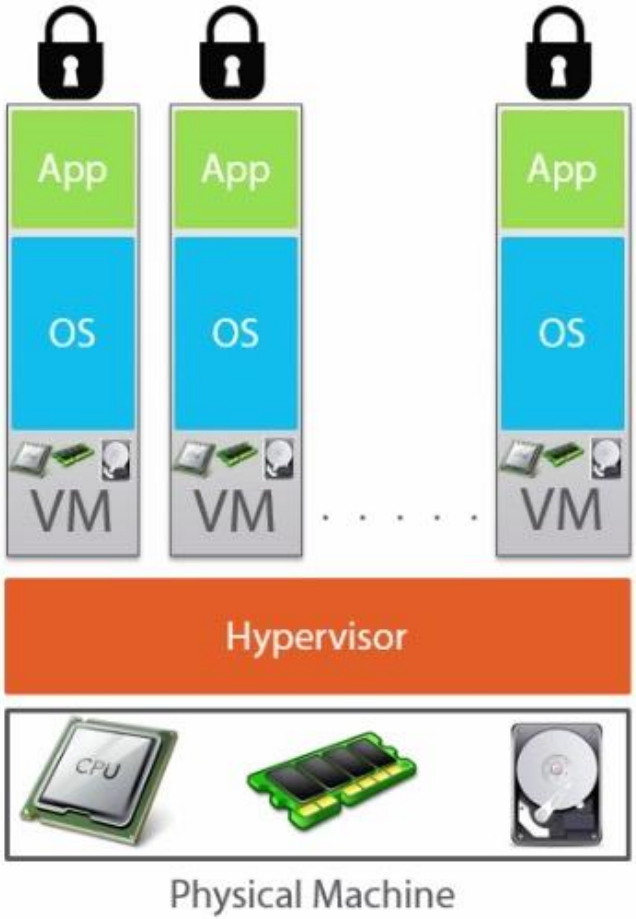
S



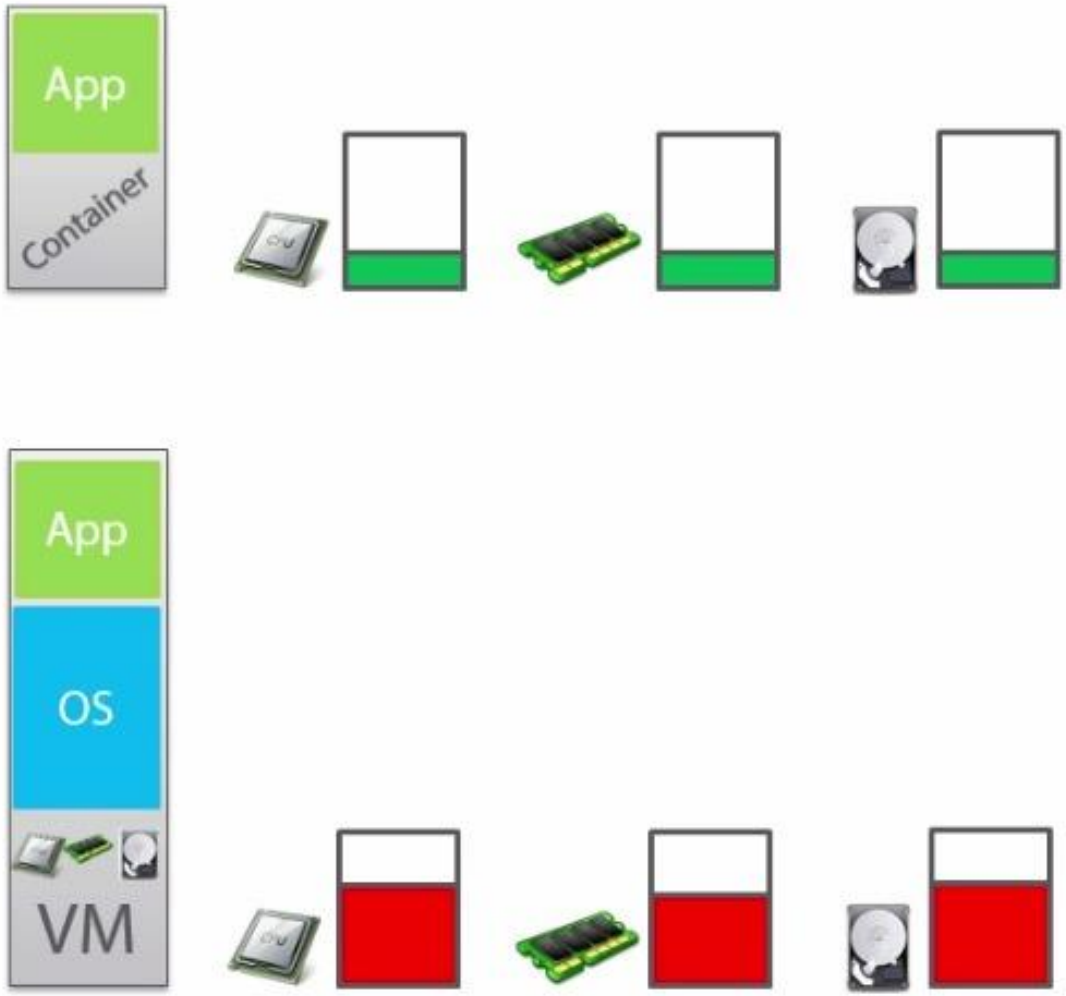

App
App
App
VM
VM
VM
Hypervisor
CPU
Physical Machine

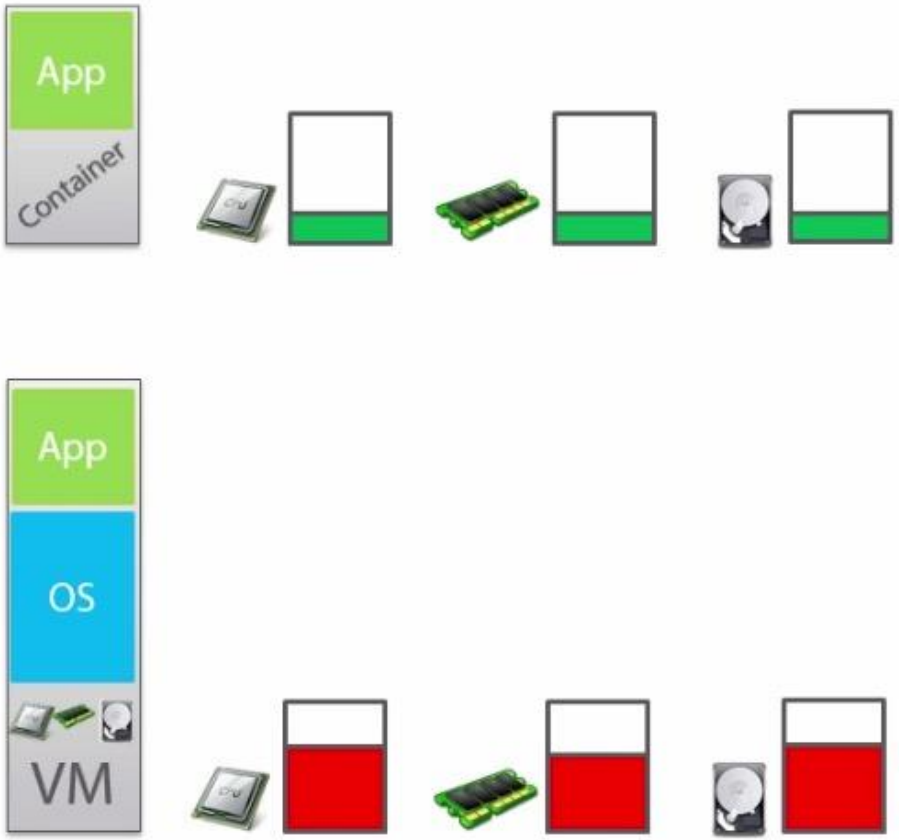


> OS != Business Value

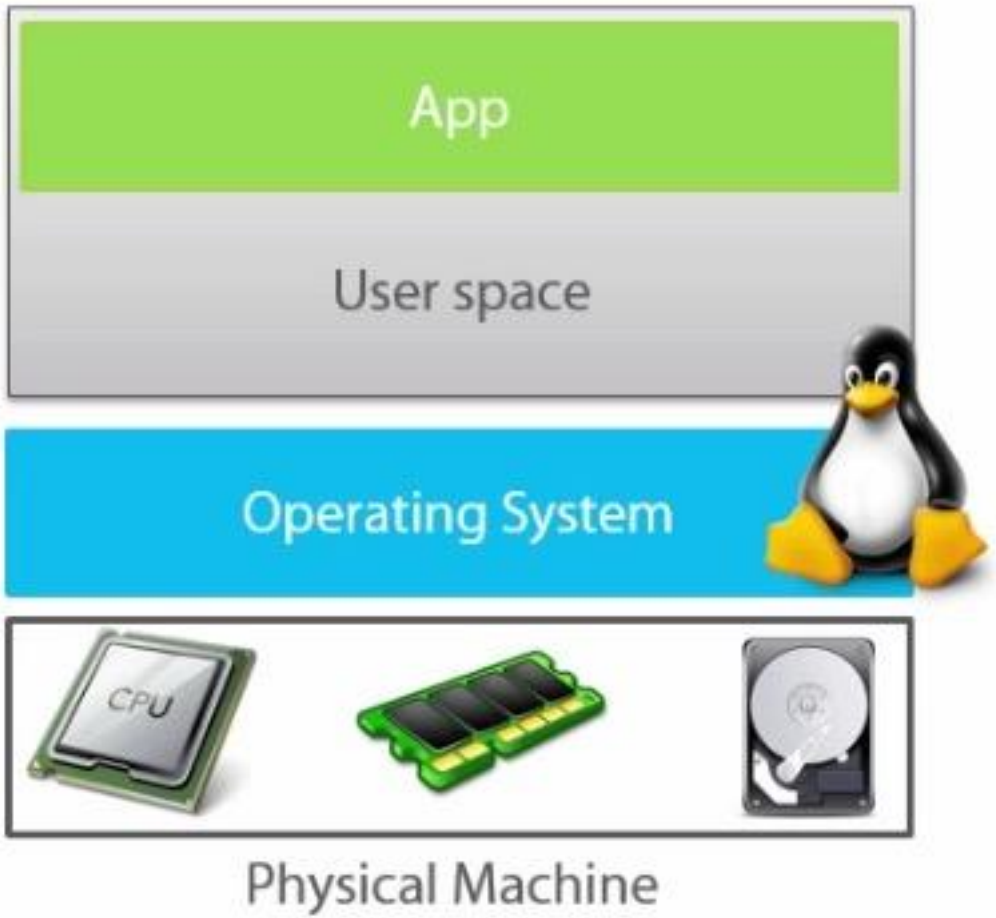

App
App
App
OS
OS
OS
VM
VM
VM
Hypervisor
CPU
Physical Machine

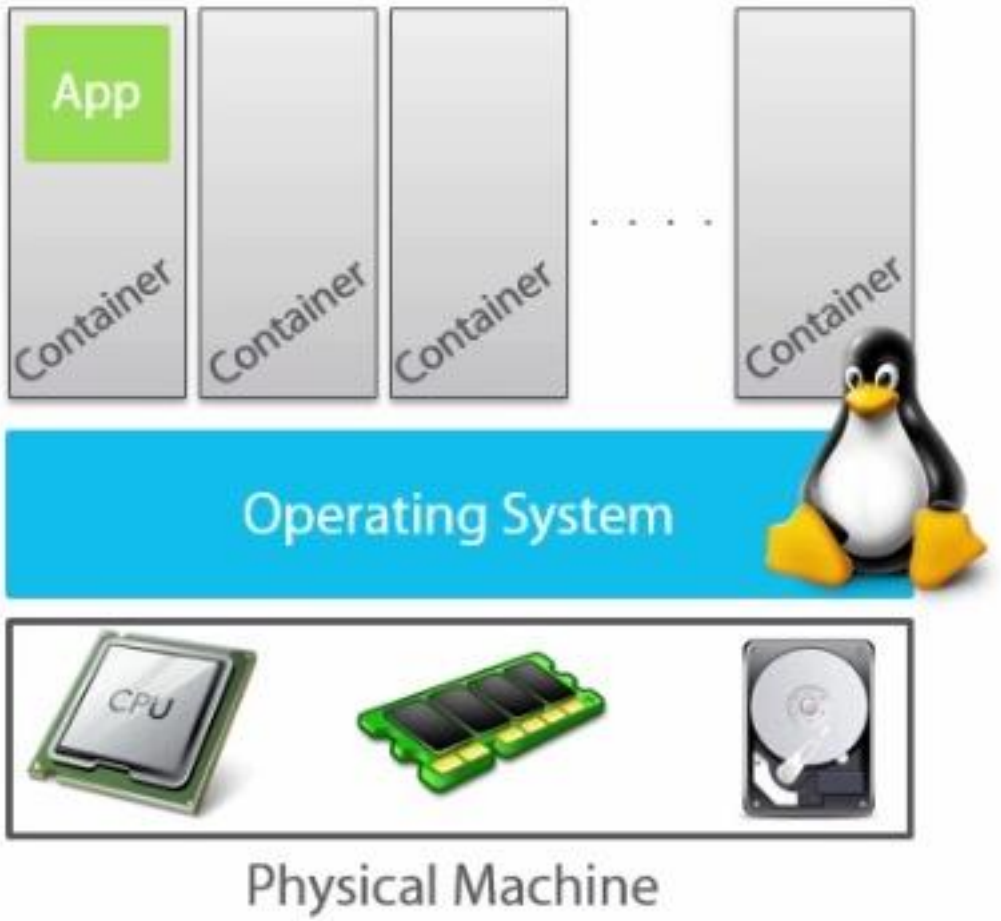

OS
25
50
75

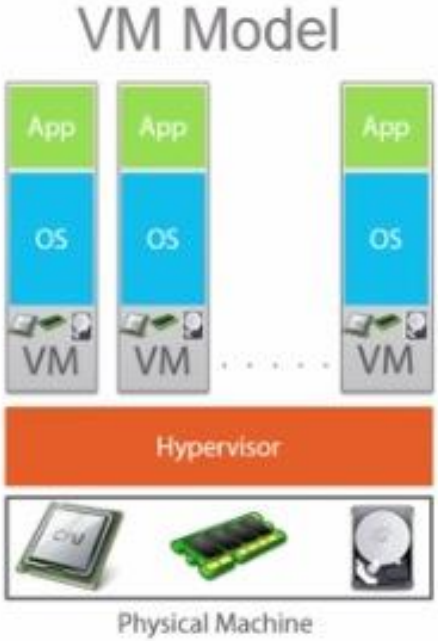

App
Container
App
OS
VM

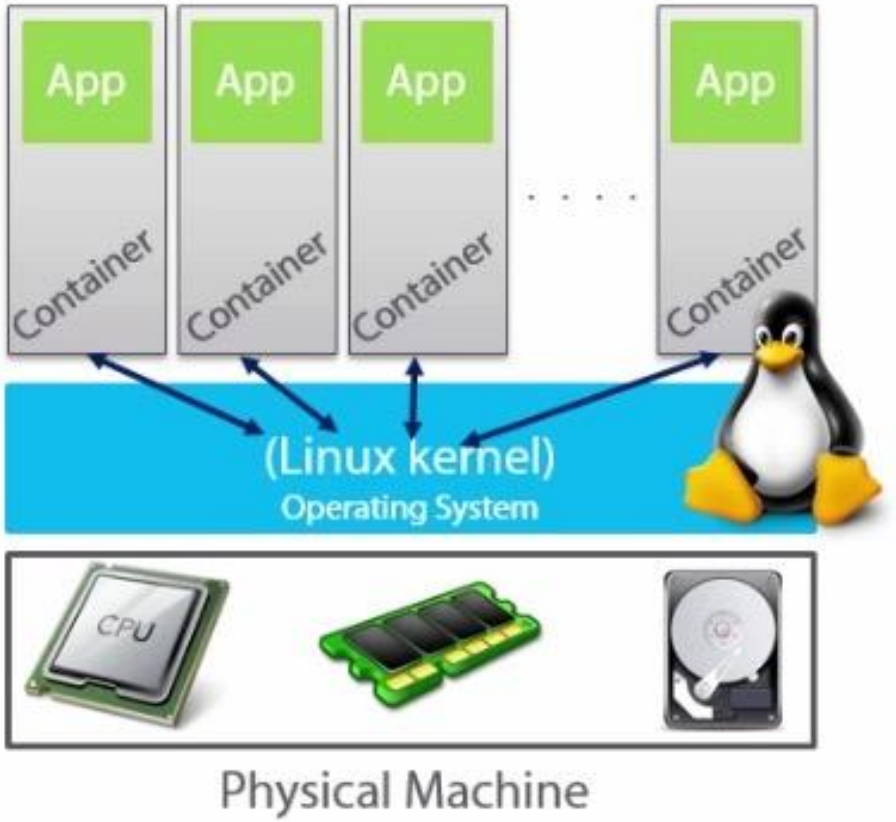


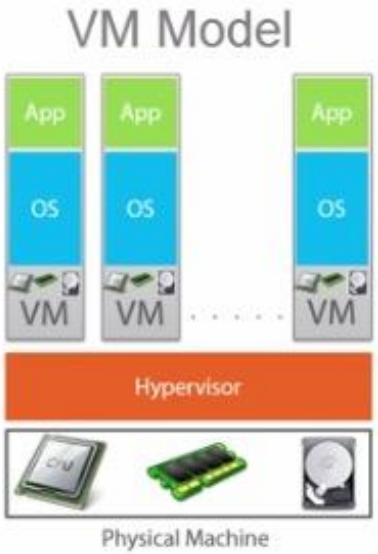
Containers are more lightweight than Virtual Machines

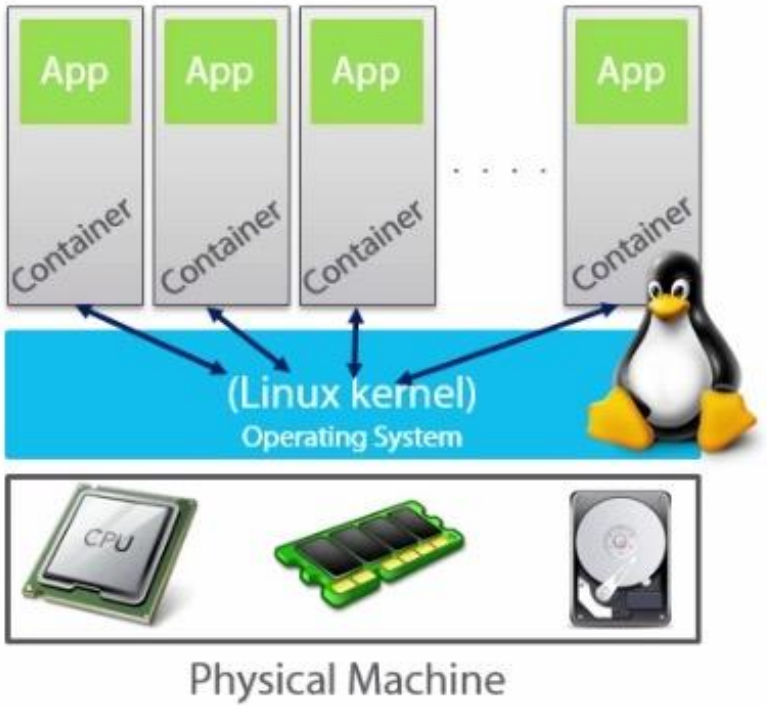

App
User space
Operating System
CPU
Physical Machine

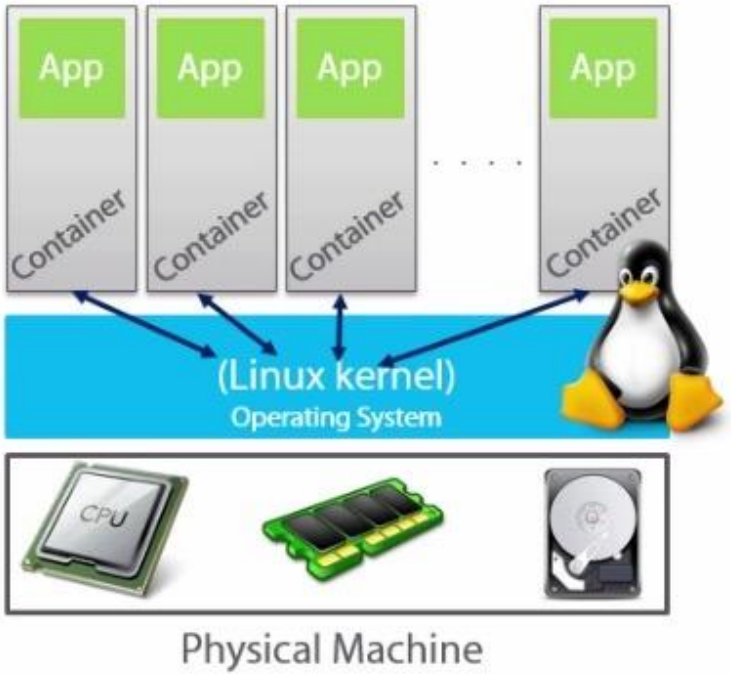

App
Container
Container
Container
Container
Operating System
CPU
Physical Machine


VM Model
App
App
App
OS
OS
OS
VM
VM
VM
Hypervisor
Physical Machine


App
App
App
App
Container
Container
Container
Container
(Linux kernel)
Operating System
CPU
Physical Machine

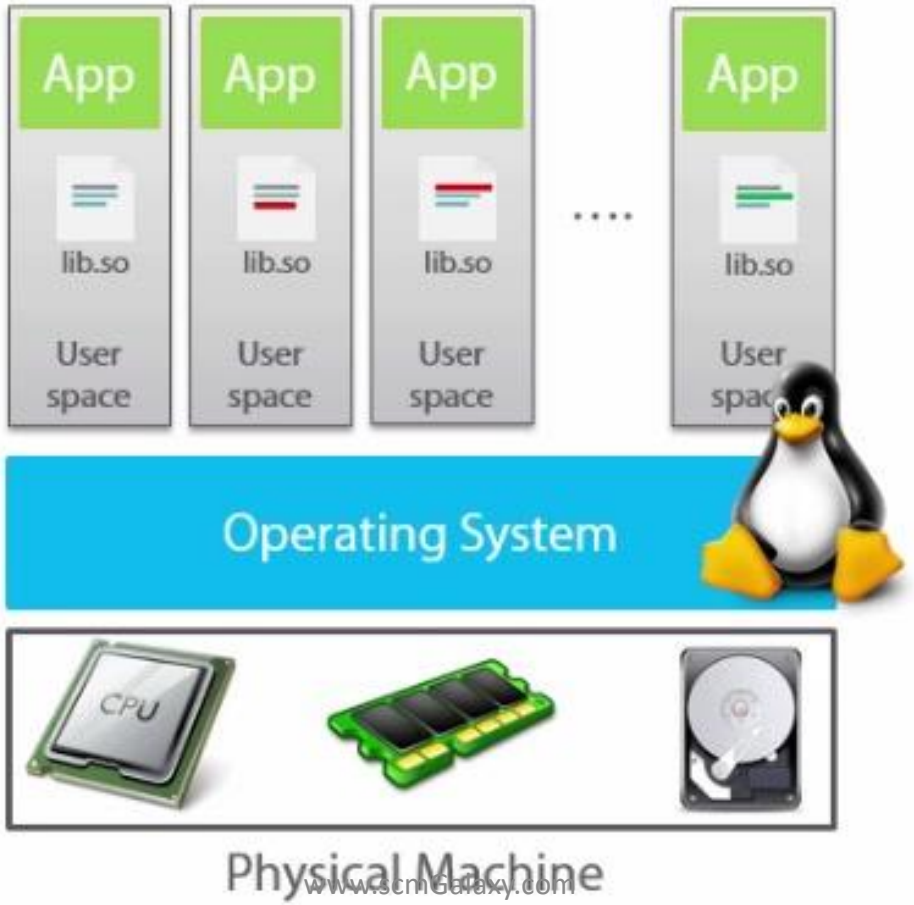

VM Model
App
App
App
OS
OS
OS
VM
VM
VM
Hypervisor
Physical Machine


App
App
App
App
Container
Container
Container
Container
(Linux kernel)
Operating System
CPU
Physical Machine

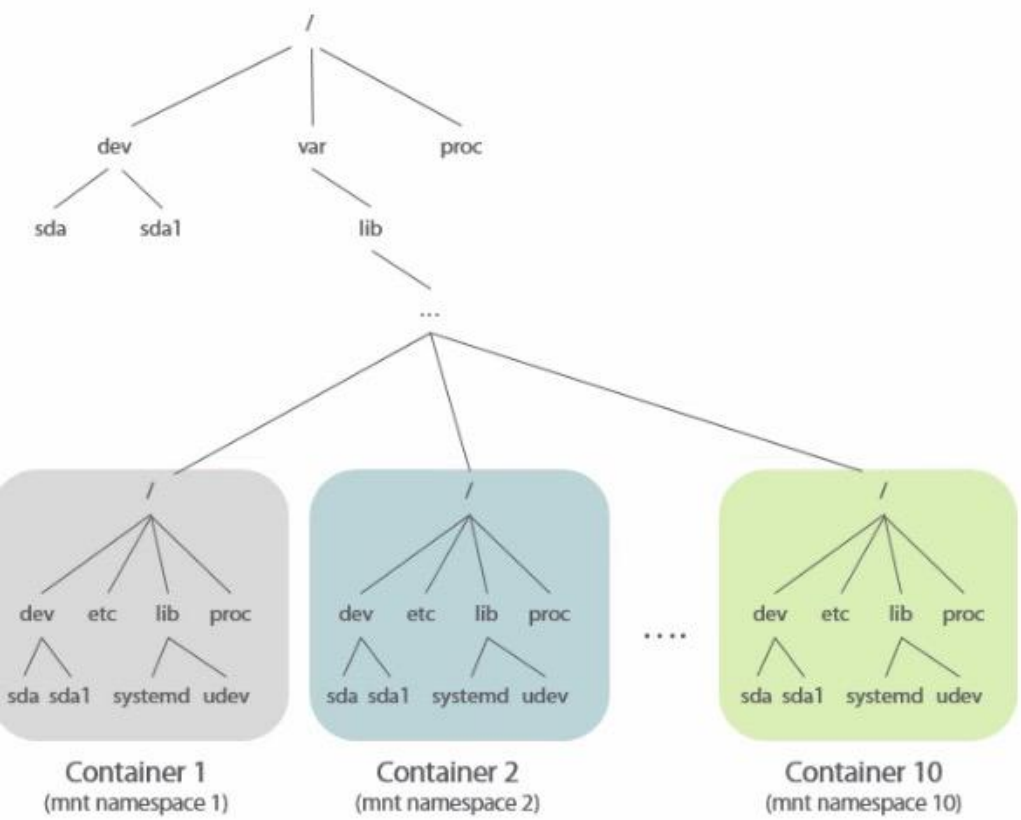


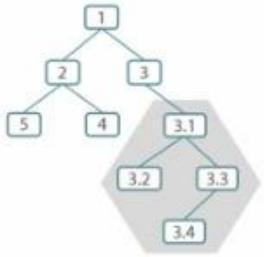
Containers consume less CPU, RAM and disk resource than Virtual Machines

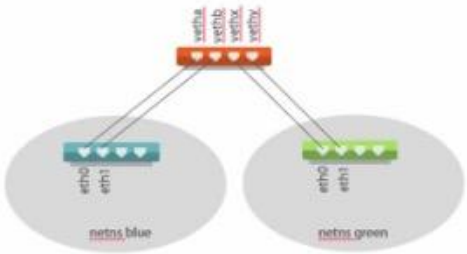
How Containers Work



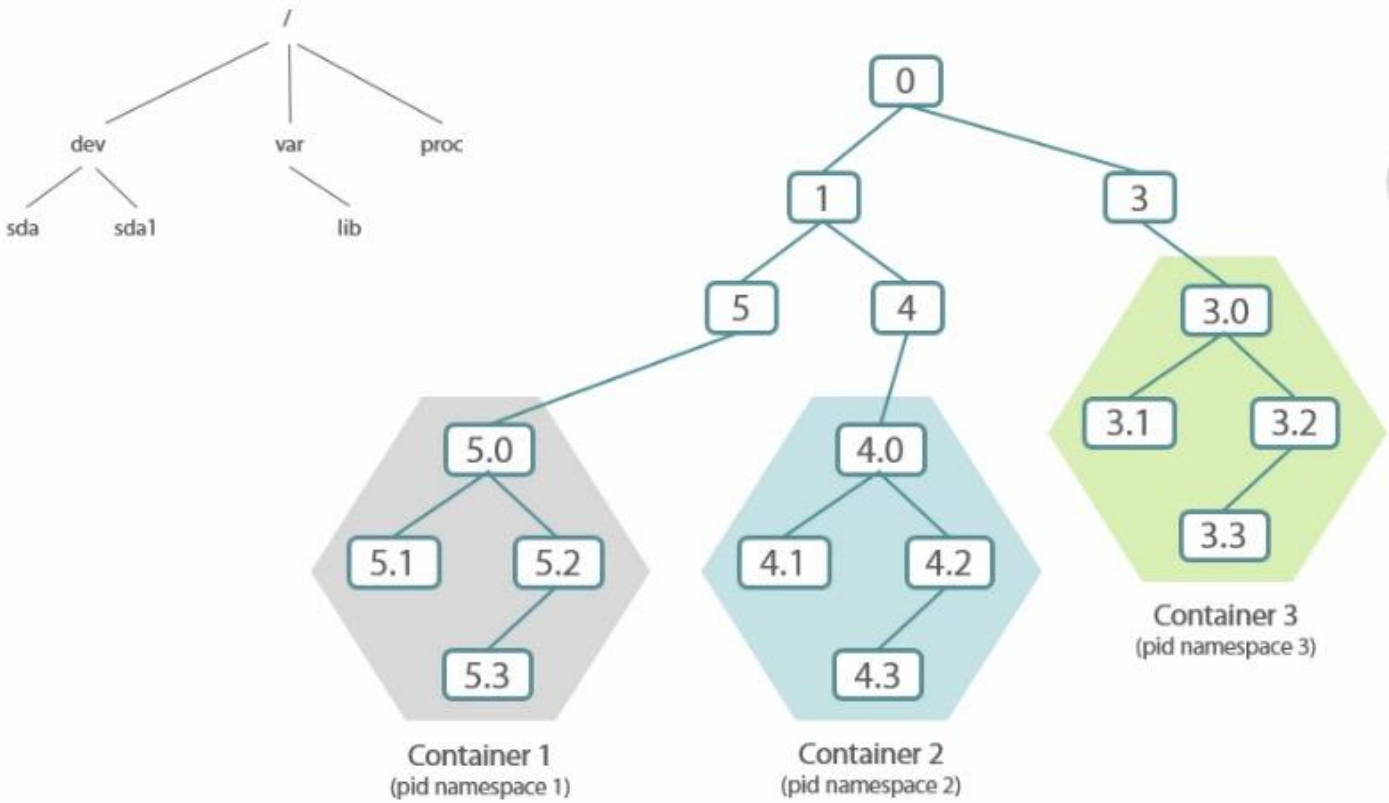
How Containers Work

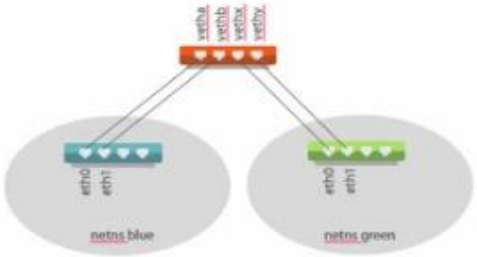




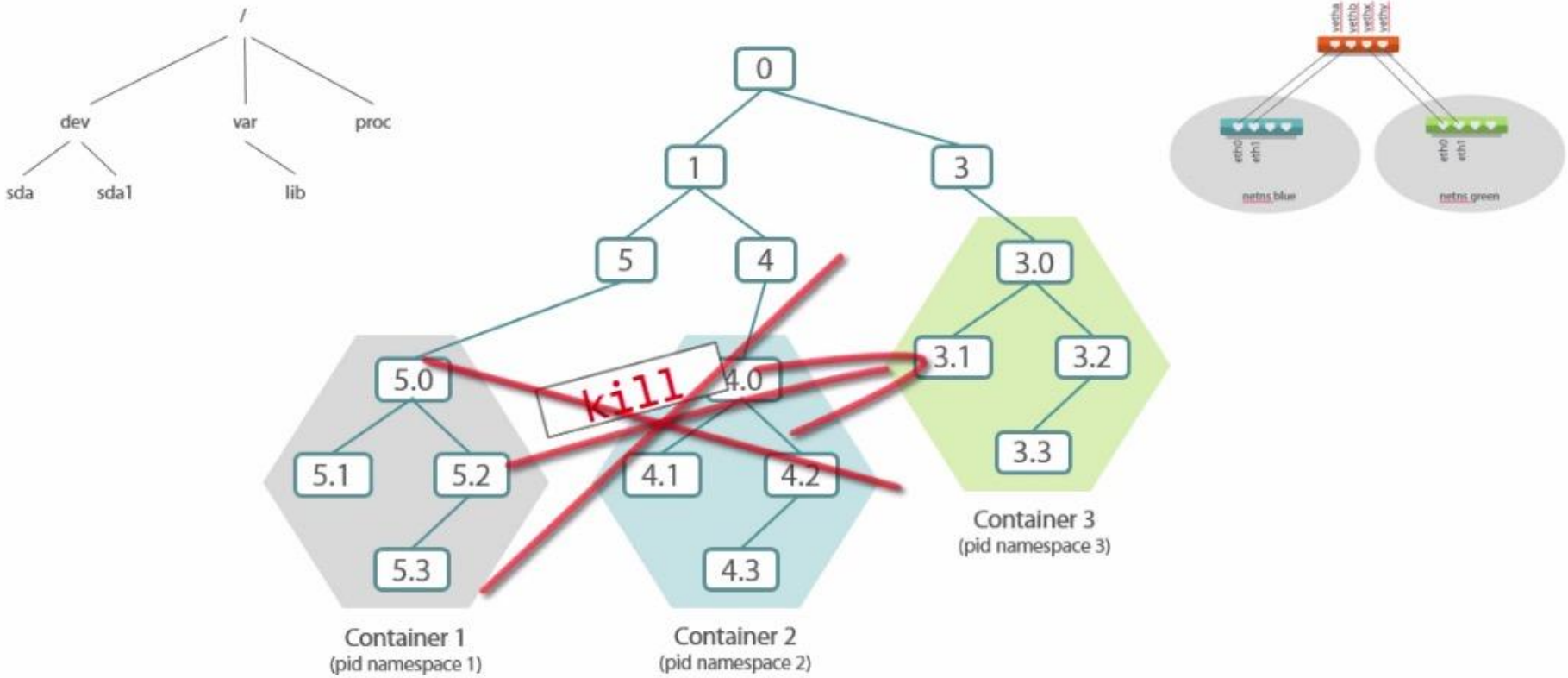


How Containers Work



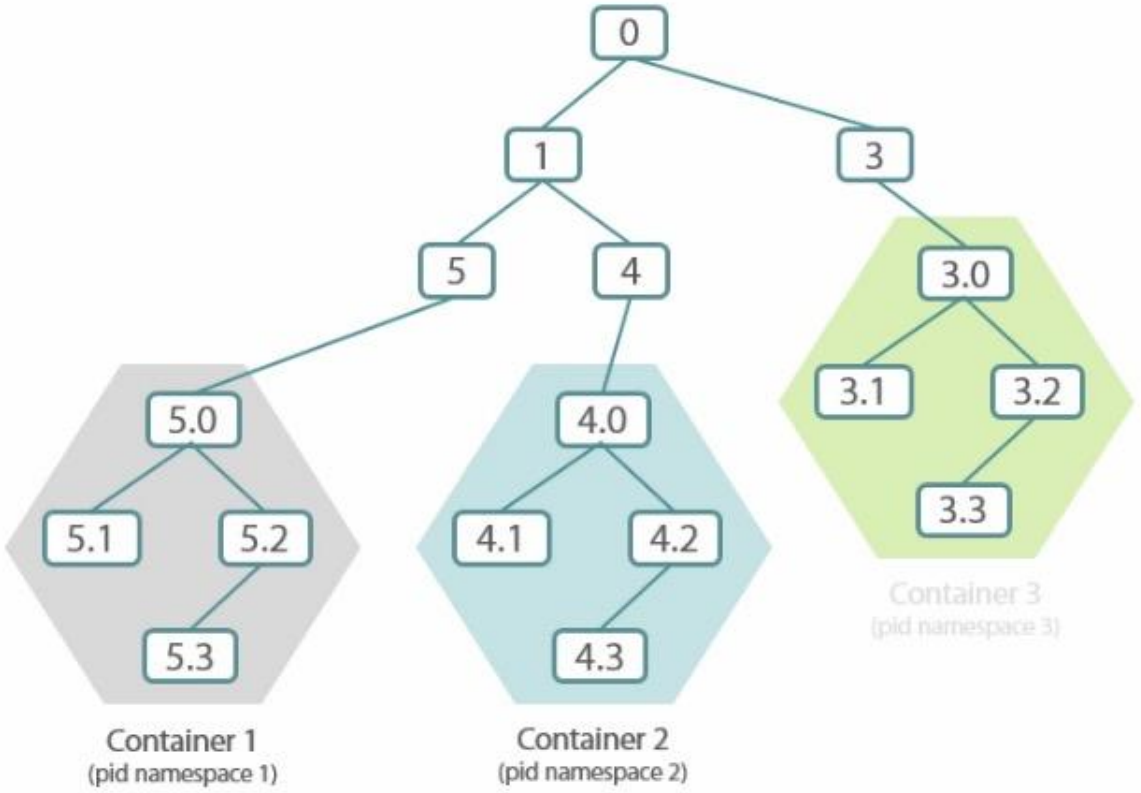


How Containers Work



Kernel Namespaces

The pid Namespace



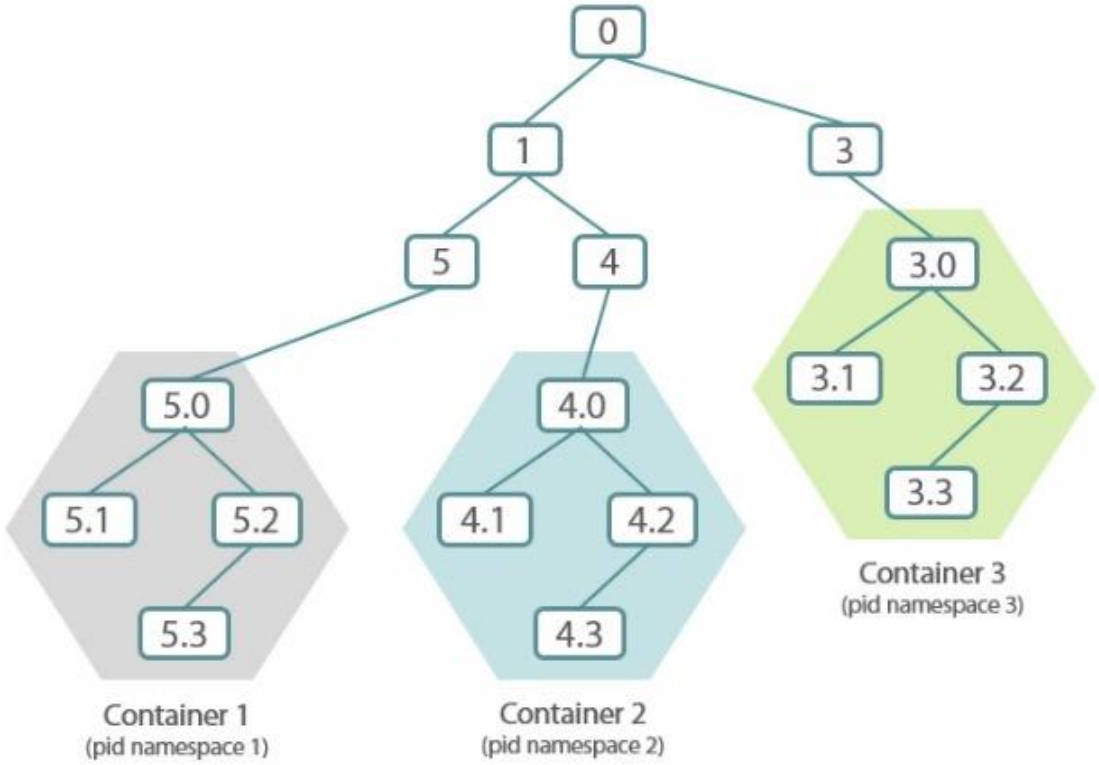
Kernel Namespaces

The pid Namespace

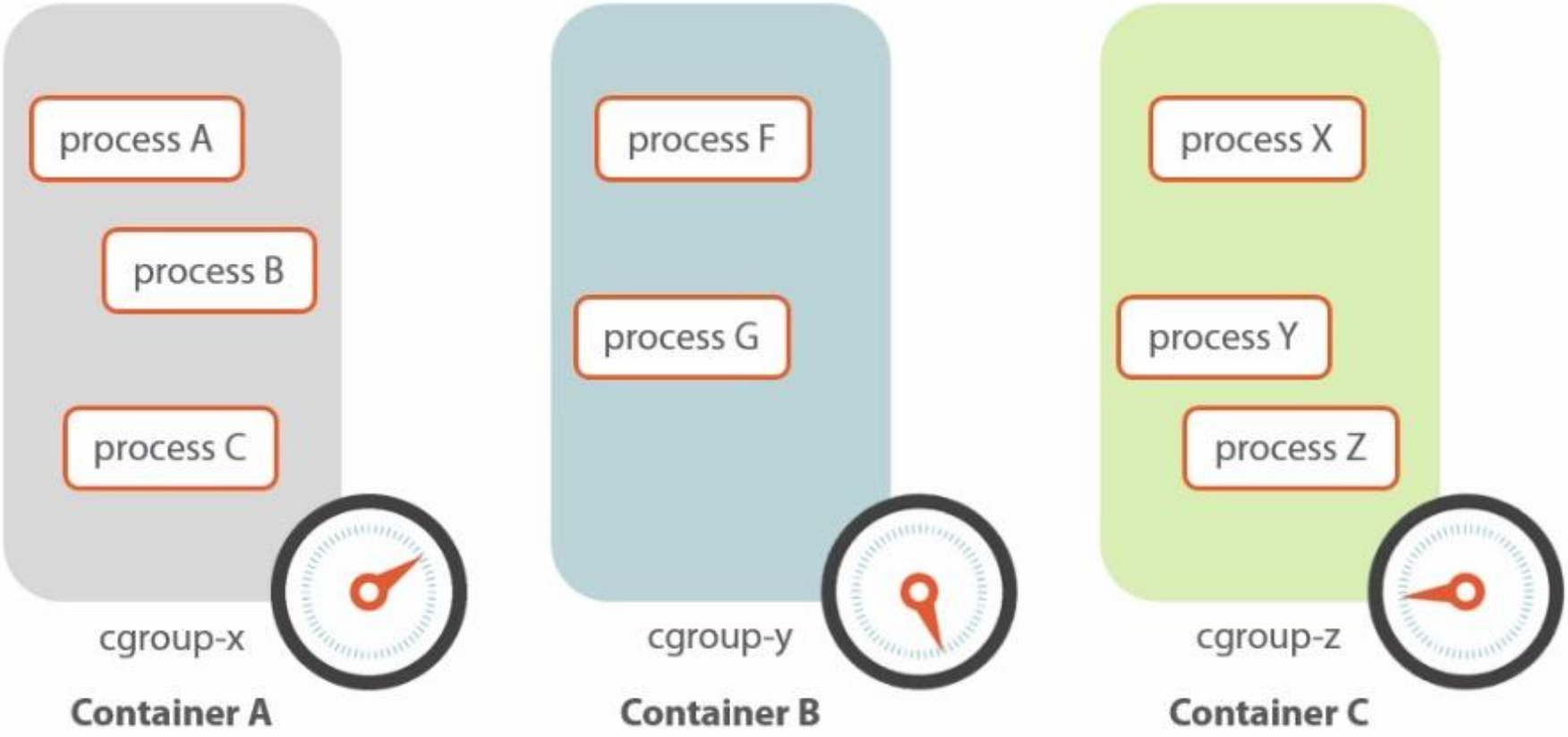
The net Namespace

The mnt Namespace

The user Namespace



Control Groups (cgroups)



Capabilities

root

CAP_AUDIT_CONTROL ✓

CAP_CHOWN ✓

CAP_DAC_OVERRIDE ✓

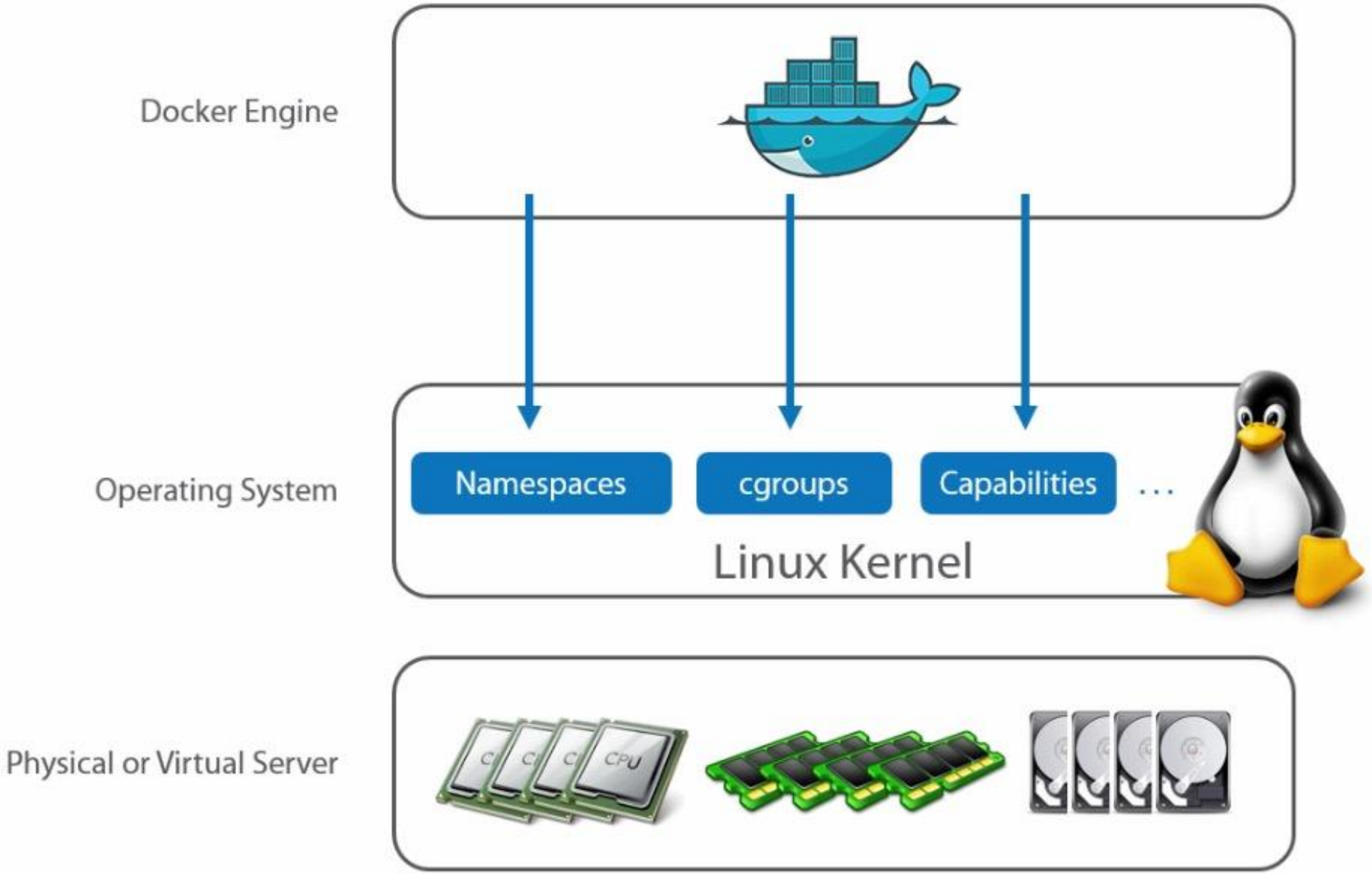
CAP_KILL ✓

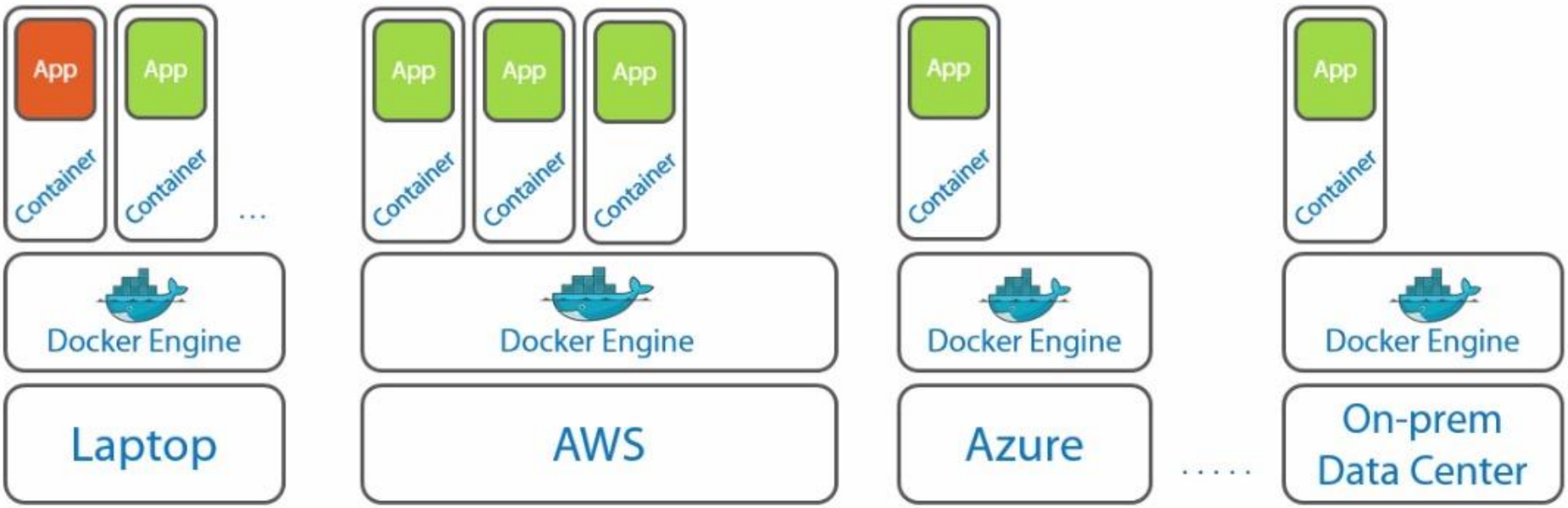
CAP_NET_BIND_SERVICE ✓

CAP_SETUID ✓

. . . . . ✓


docker


Docker Engine
Operating System
Namespaces
cgroups
Capabilities
...
Linux Kernel
Physical or Virtual Server
CPU

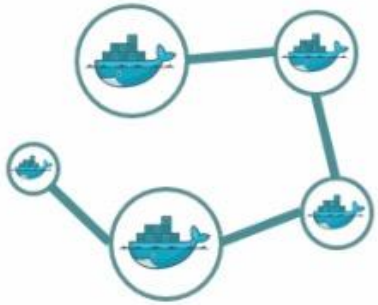

App
App
Container
Container
...
Docker Engine
Laptop
App
App
App
Container
Container
Container
Docker Engine
AWS
App
Container
Docker Engine
Azure
App
Container
Docker Engine
On-prem
Data Center

The Evolving Docker Platform

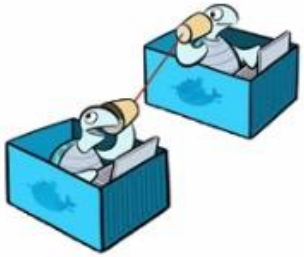
Registry
(Docker Hub)

Clustering
(Docker Swarm…)

Orchestration
(Docker Compose…)



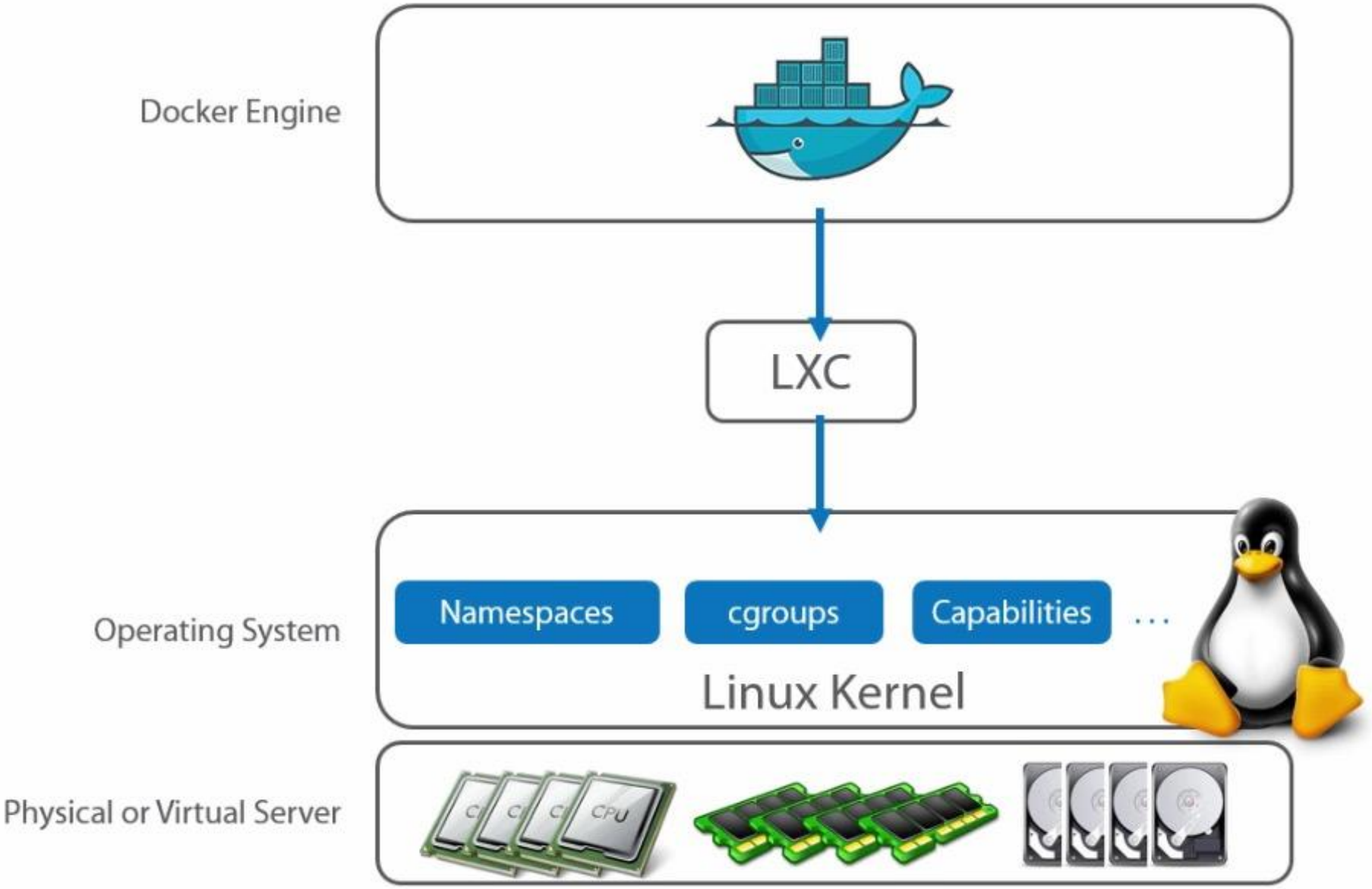
Networking
(libchan…)

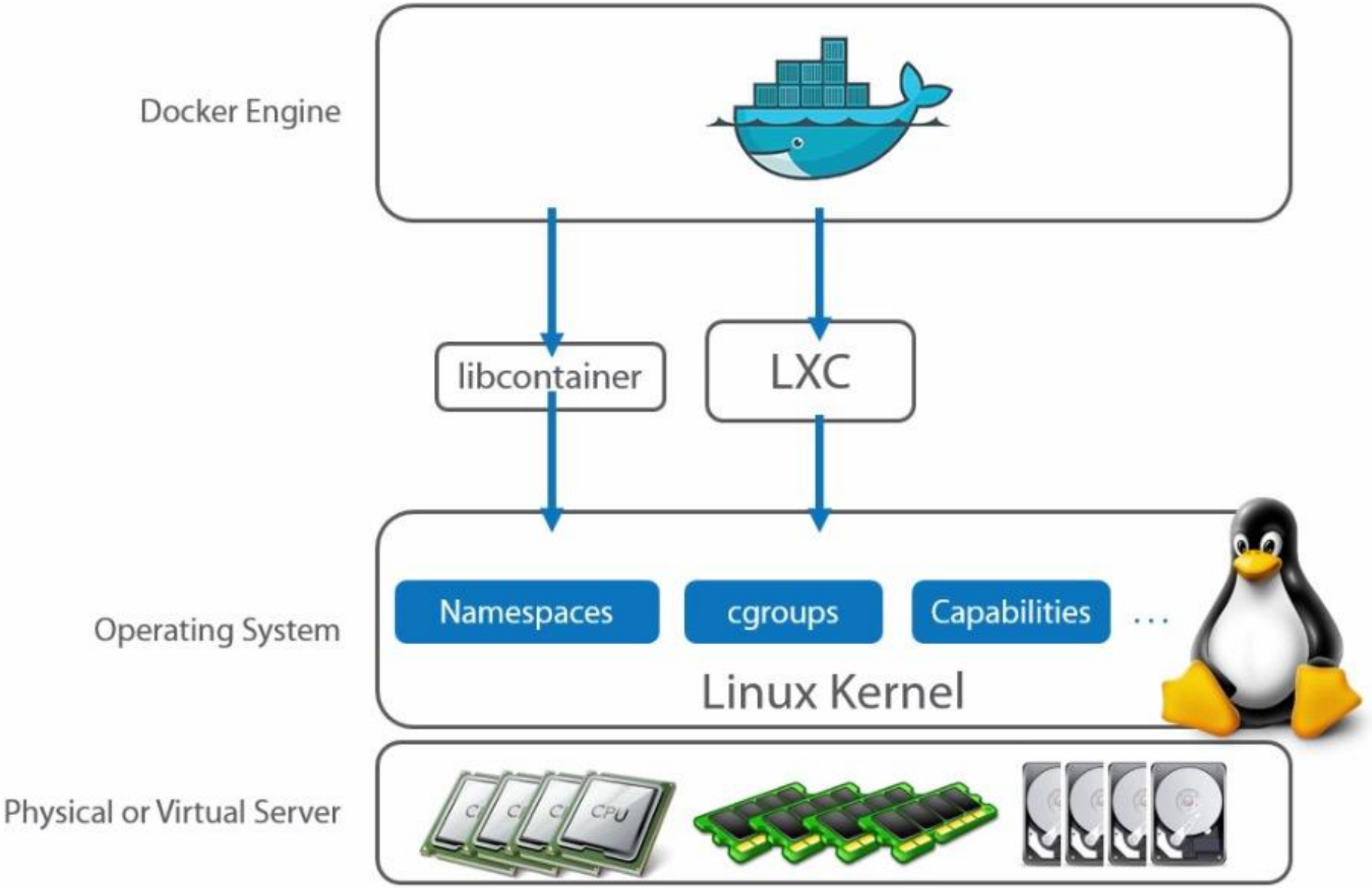


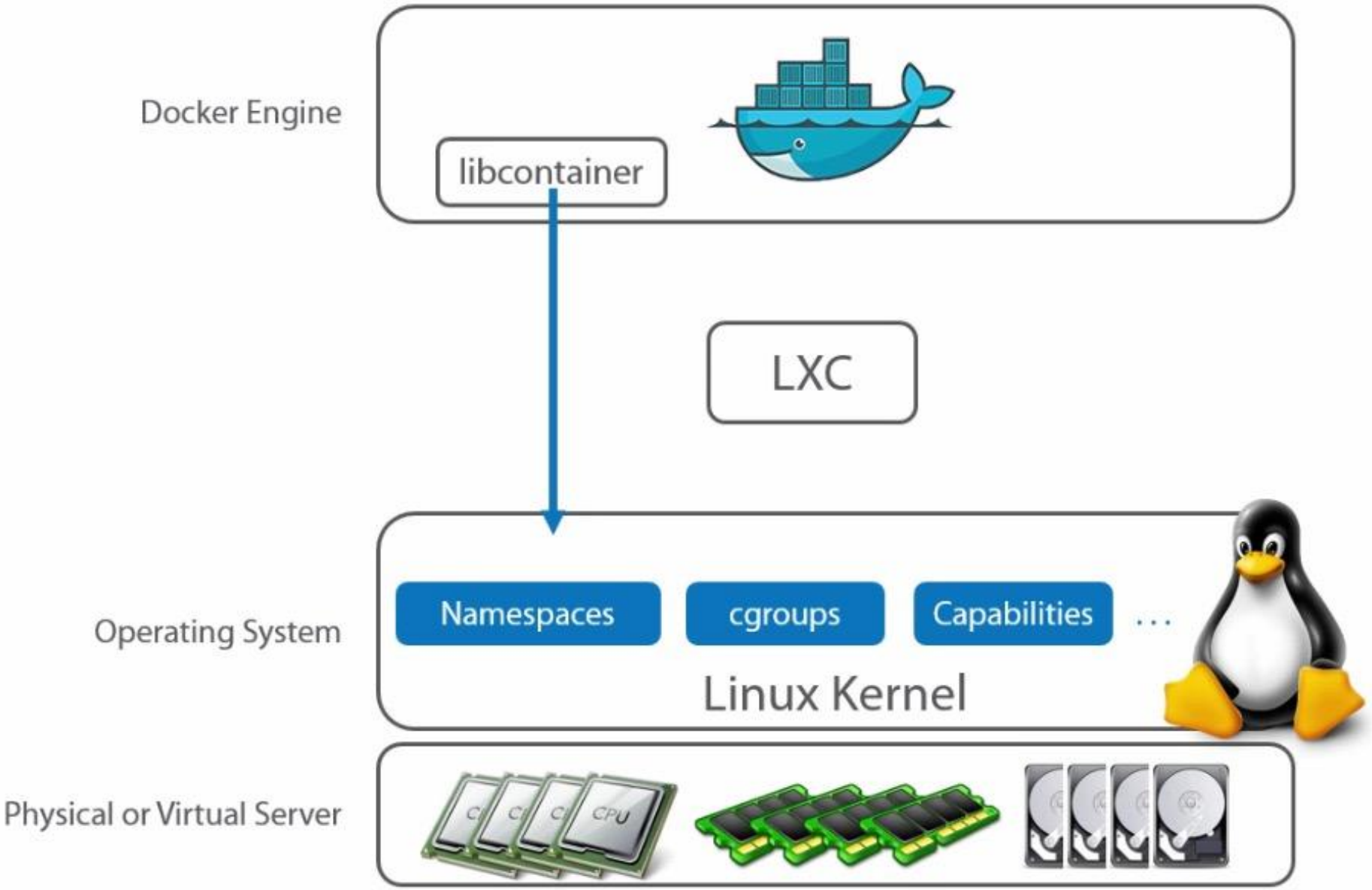
. . . . . .

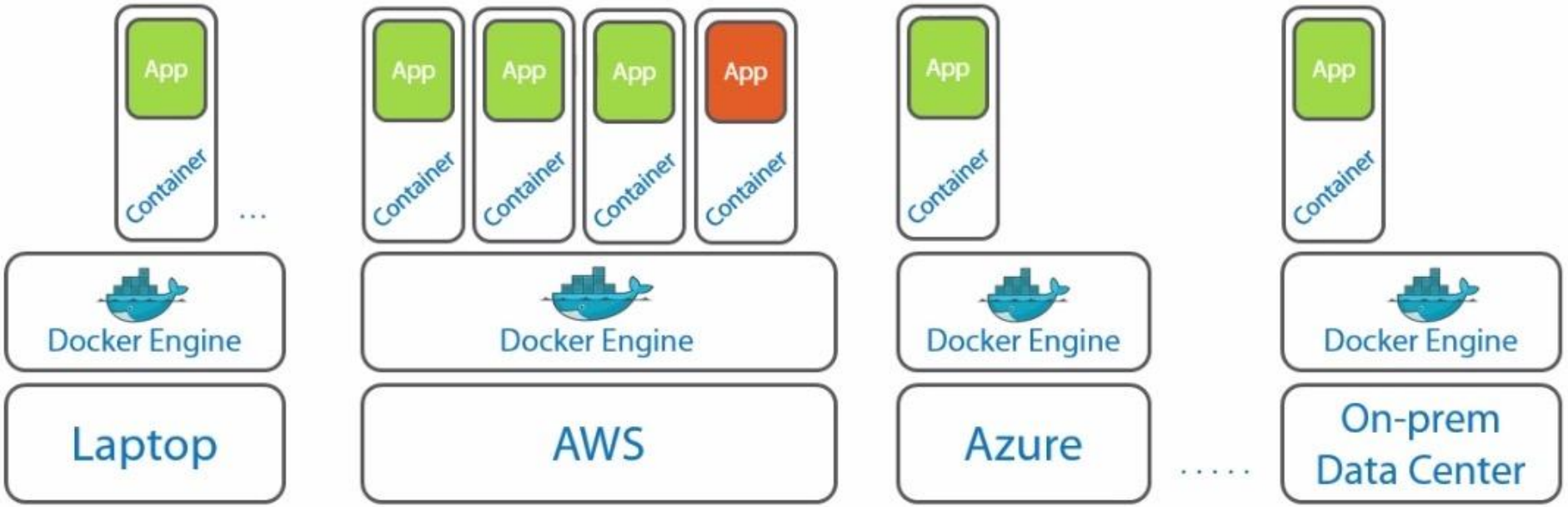

LXC

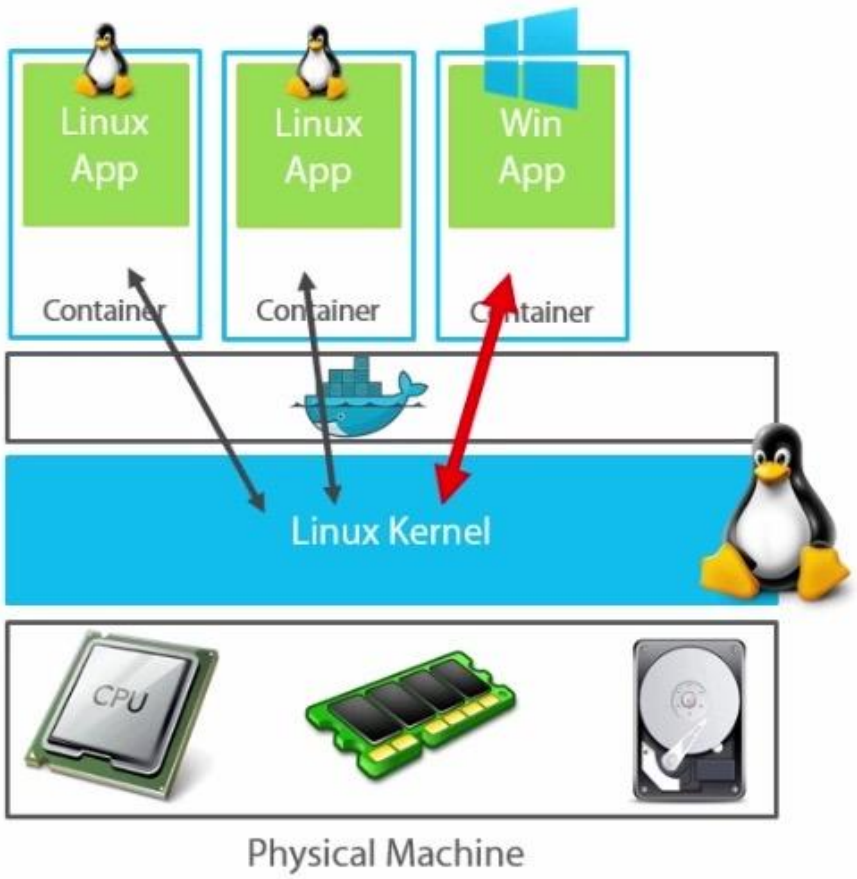
LXC and Docker…… What's the skinny?

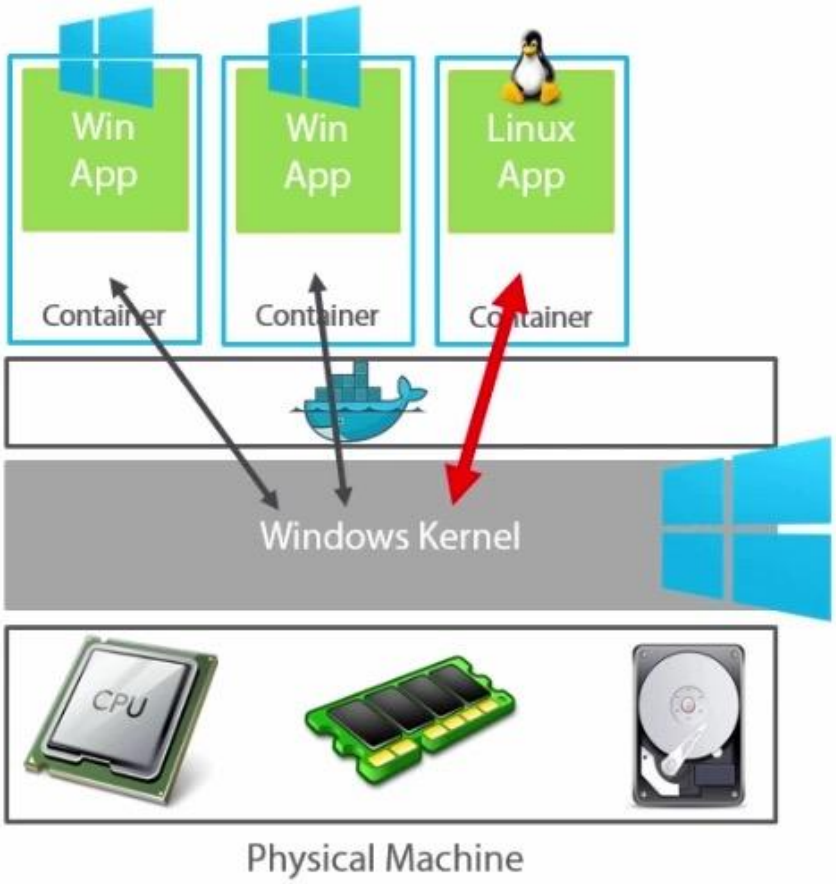

Docker Engine
LXC
Operating System
Namespaces
cgroups
Capabilities
...
Linux Kernel
Physical or Virtual Server
CPU

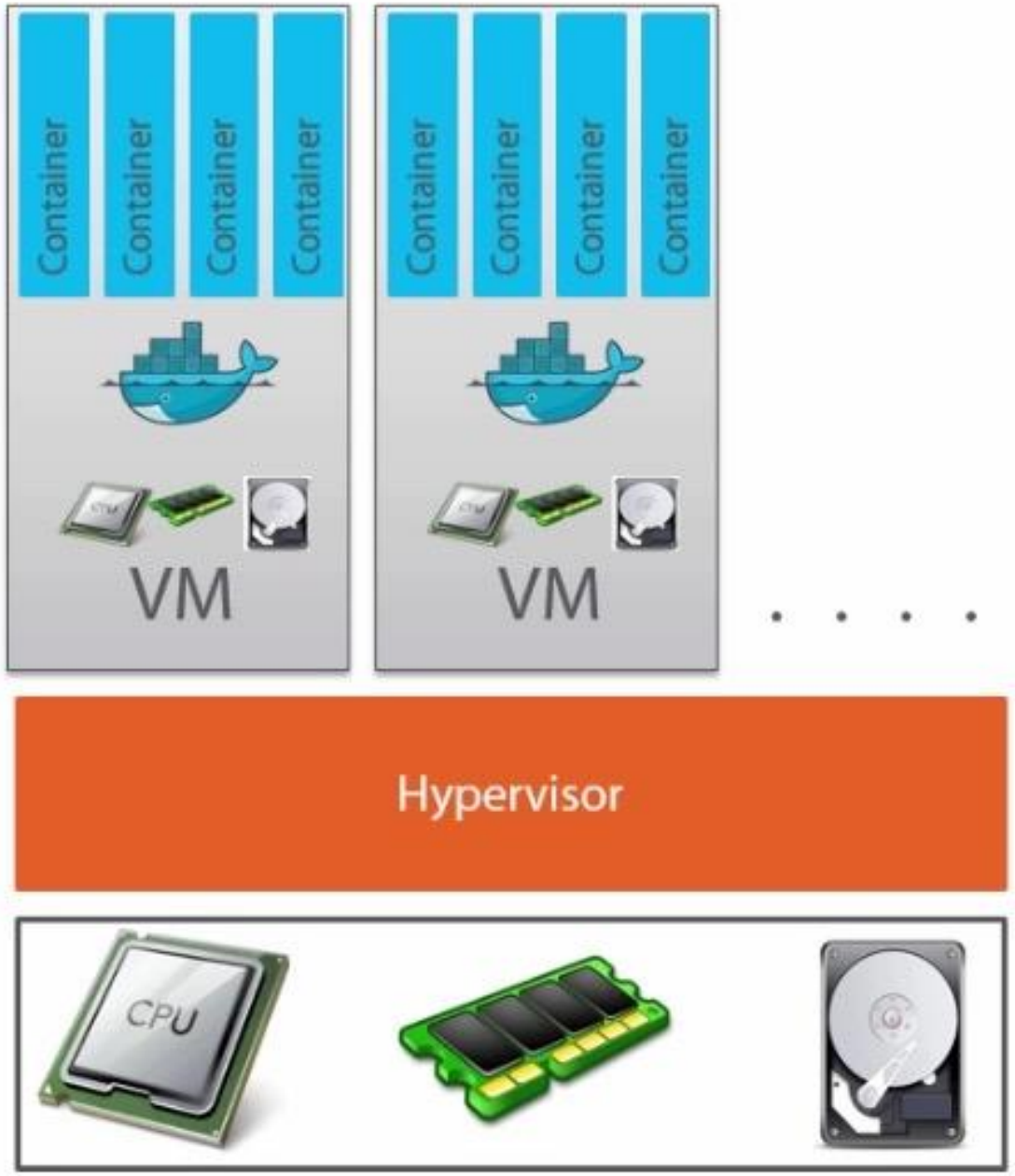

Docker Engine
libcontainer
LXC
Operating System
Namespaces
cgroups
Capabilities
...
Linux Kernel
Physical or Virtual Server
CPU

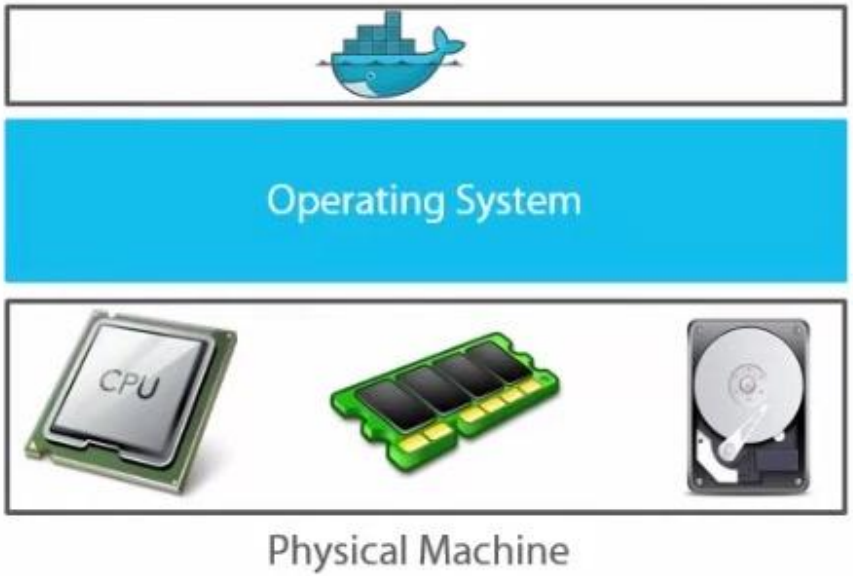

Docker Engine
libcontainer
LXC
Operating System
Namespaces
cgroups
Capabilities
...
Linux Kernel
Physical or Virtual Server
CPU

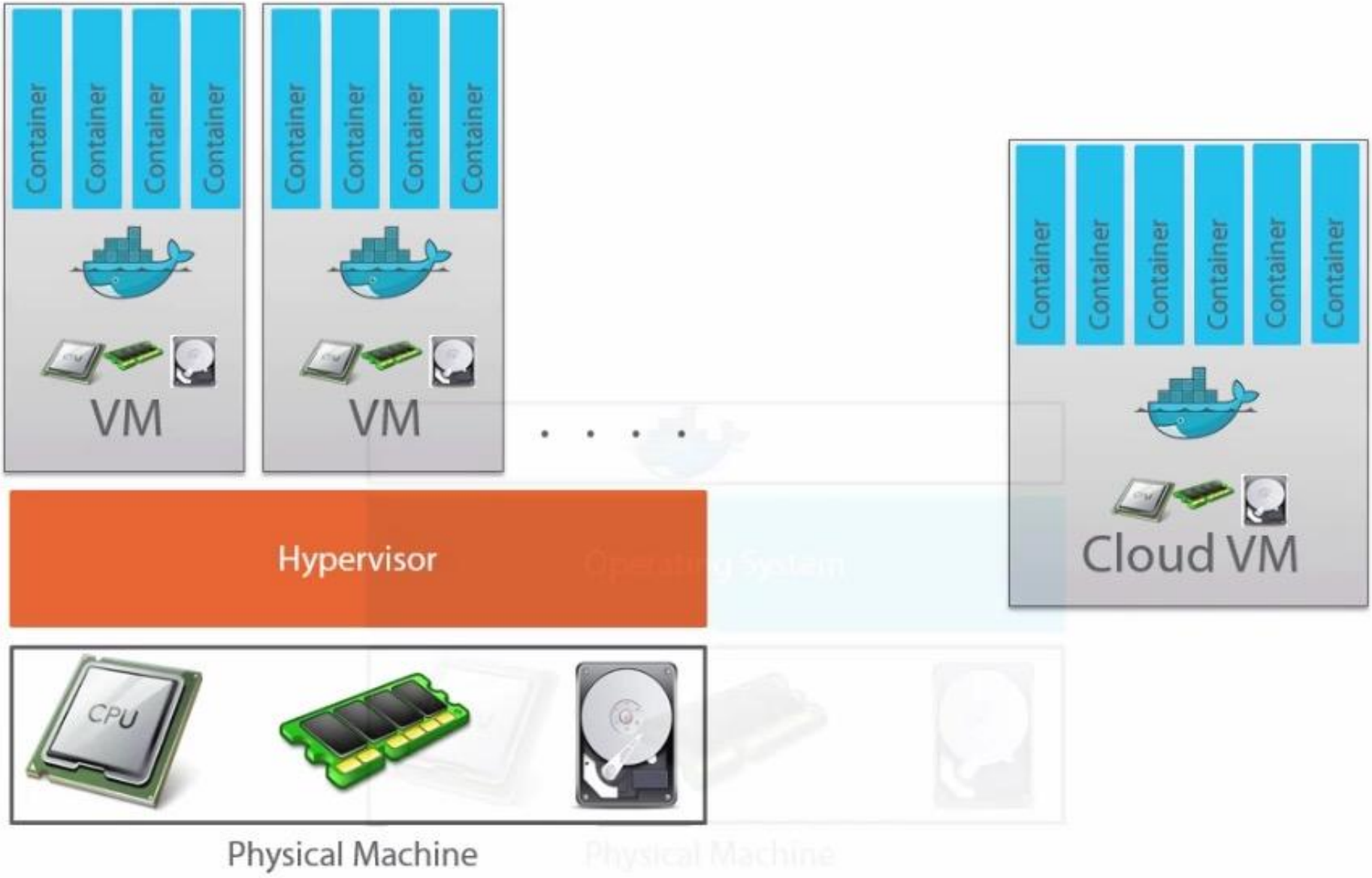

App
Container
...
App
Container
App
Container
App
Container
App
Container
App
Container
App
Container
Docker Engine
Docker Engine
Docker Engine
Docker Engine
Laptop
AWS
Azure
.....
On-prem
Data Center

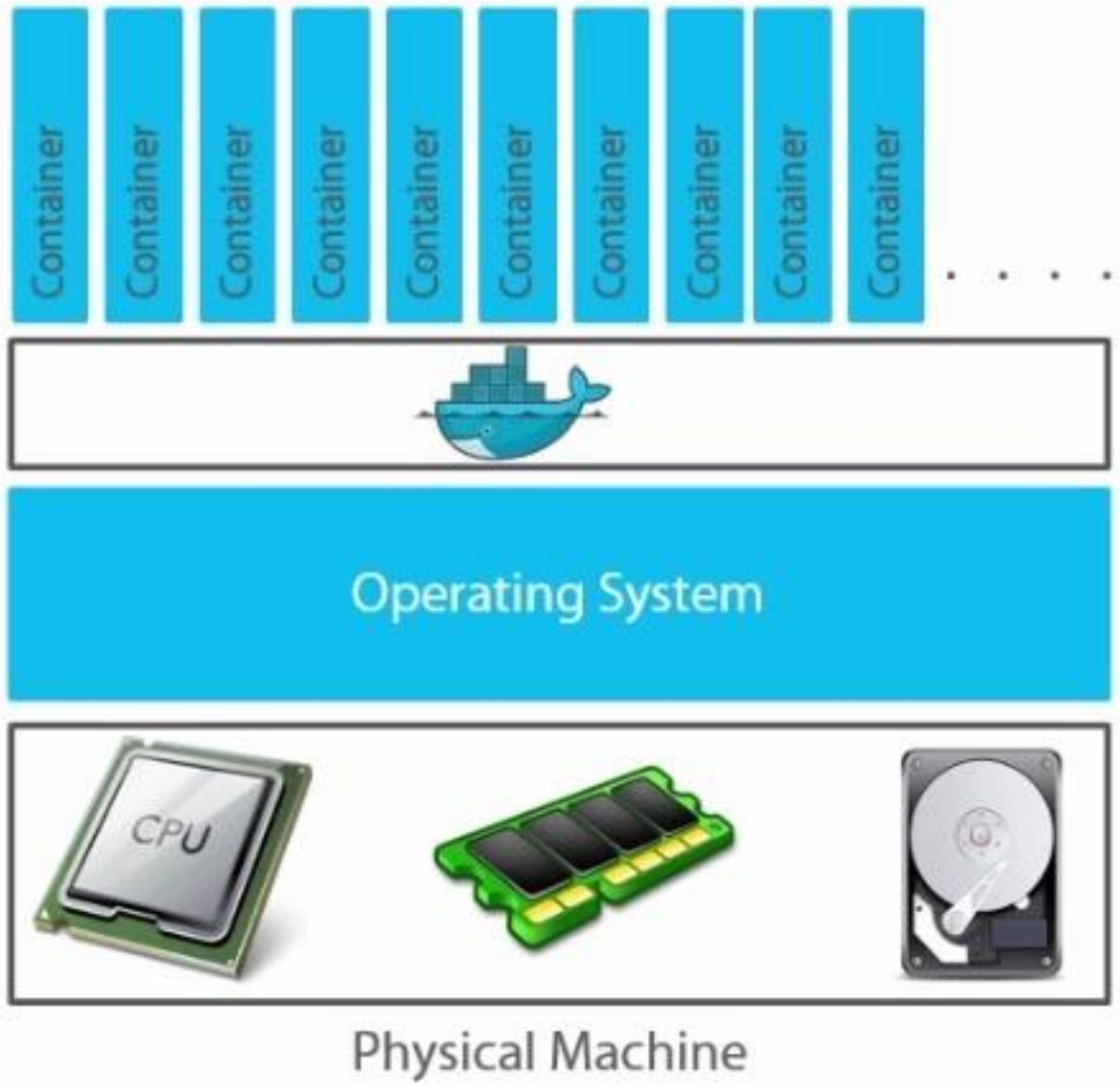

Linux App
Linux App
Win App
Container
Container
Container
Linux Kernel
CPU
Physical Machine

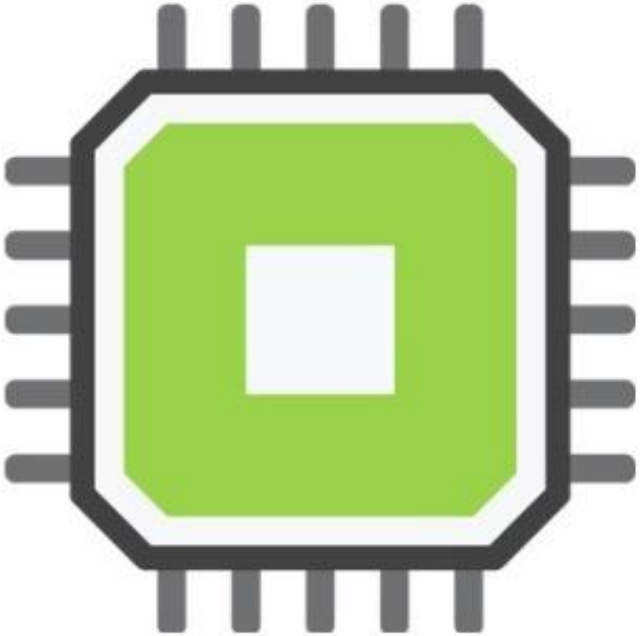

Win App
Win App
Linux App
Container
Container
Container
Windows Kernel
CPU
Physical Machine


Container
Container
Container
Container
Container
Container
Container
Container
VM
VM
Hypervisor
CPU
Physical Machine


Operating System
CPU
Physical Machine

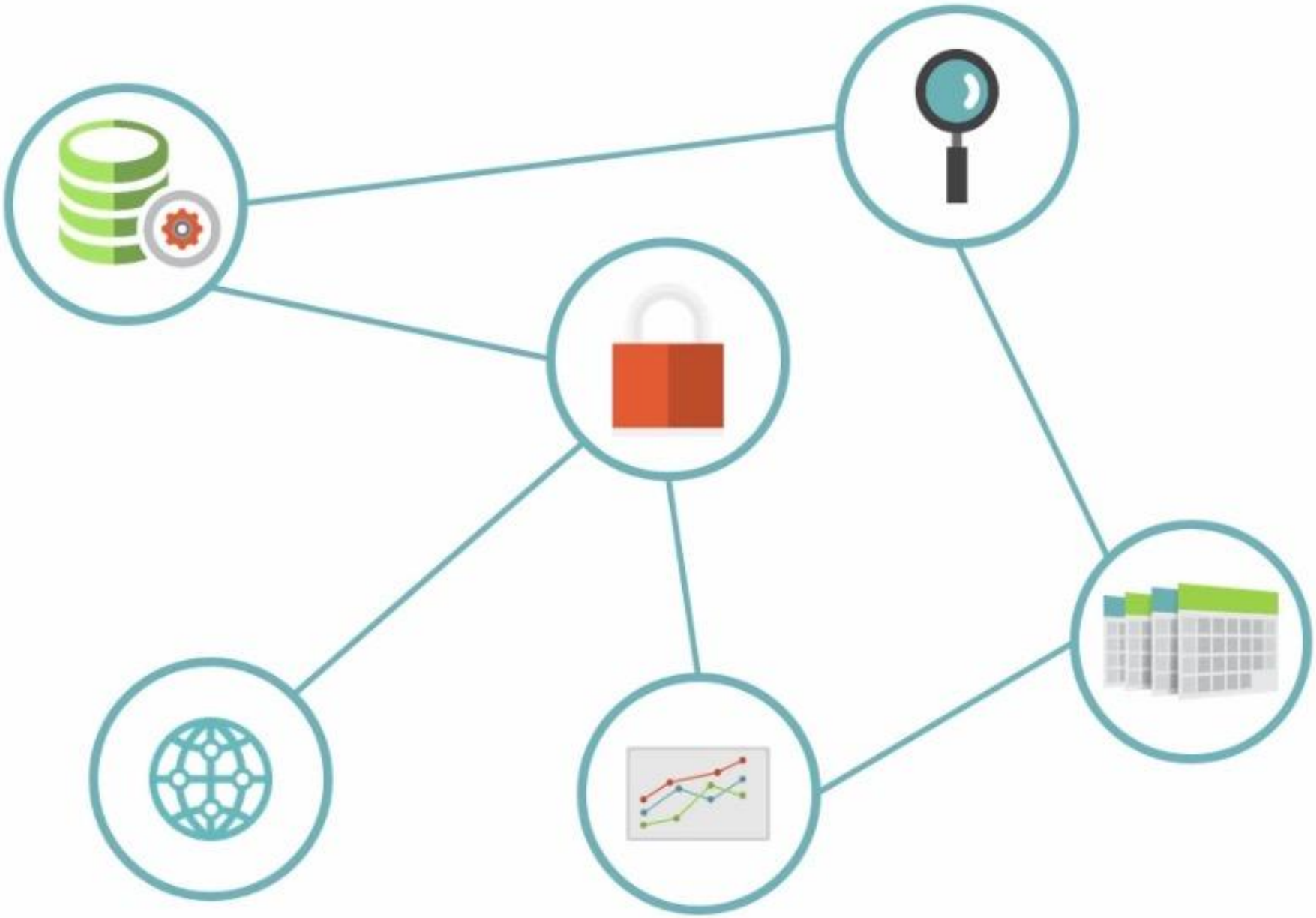

Container
Container
Container
Container
VM
Container
Container
Container
Container
VM
Container
Container
Container
Container
Container
Container
Cloud VM
Hypervisor
CPU
Physical Machine


Container
Container
Container
Container
Container
Container
Container
Container
Container
Container
Operating System
CPU
Physical Machine



Chip-level assists for Containers

- Performance offloads
- Security features
- .....

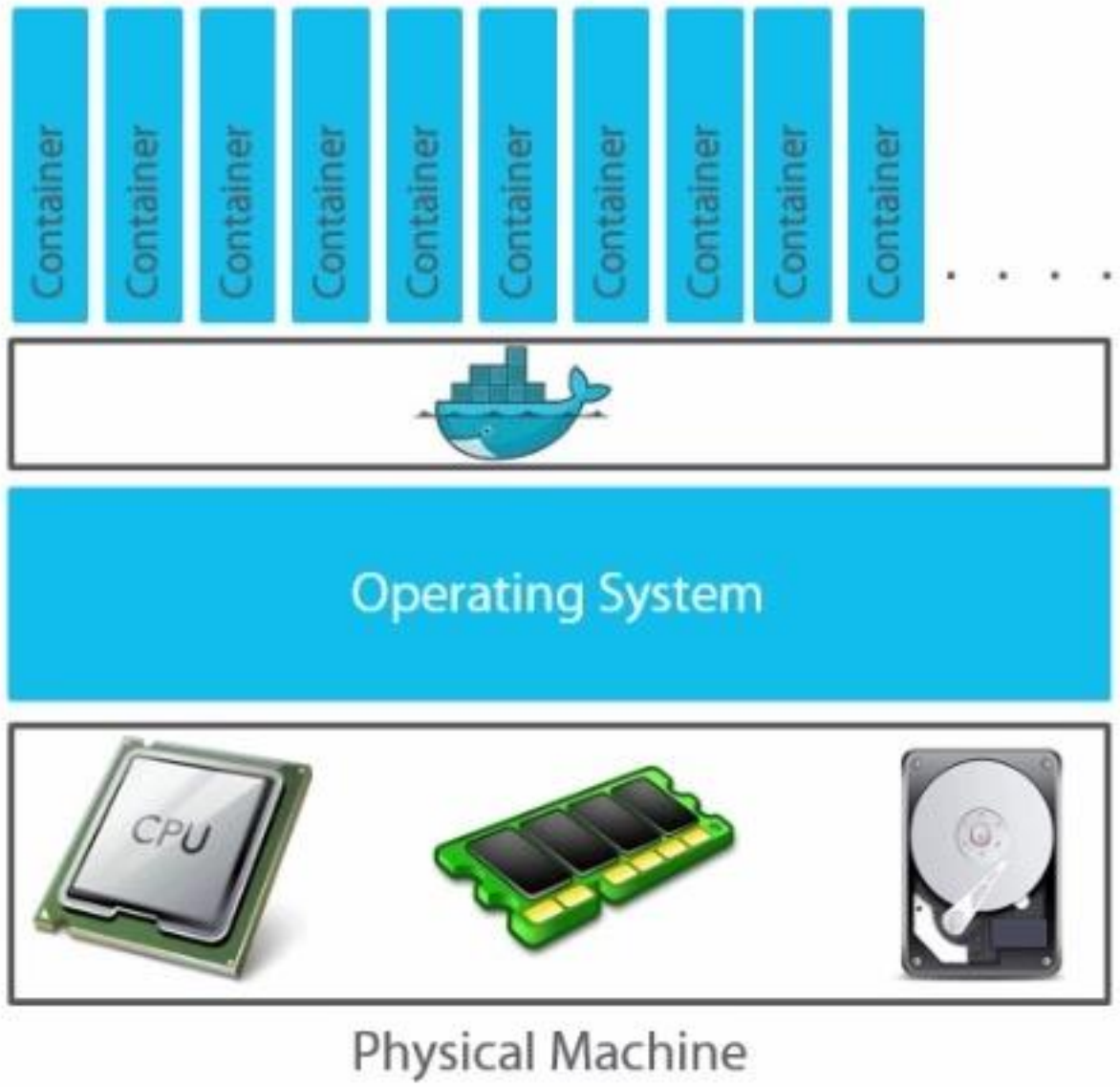


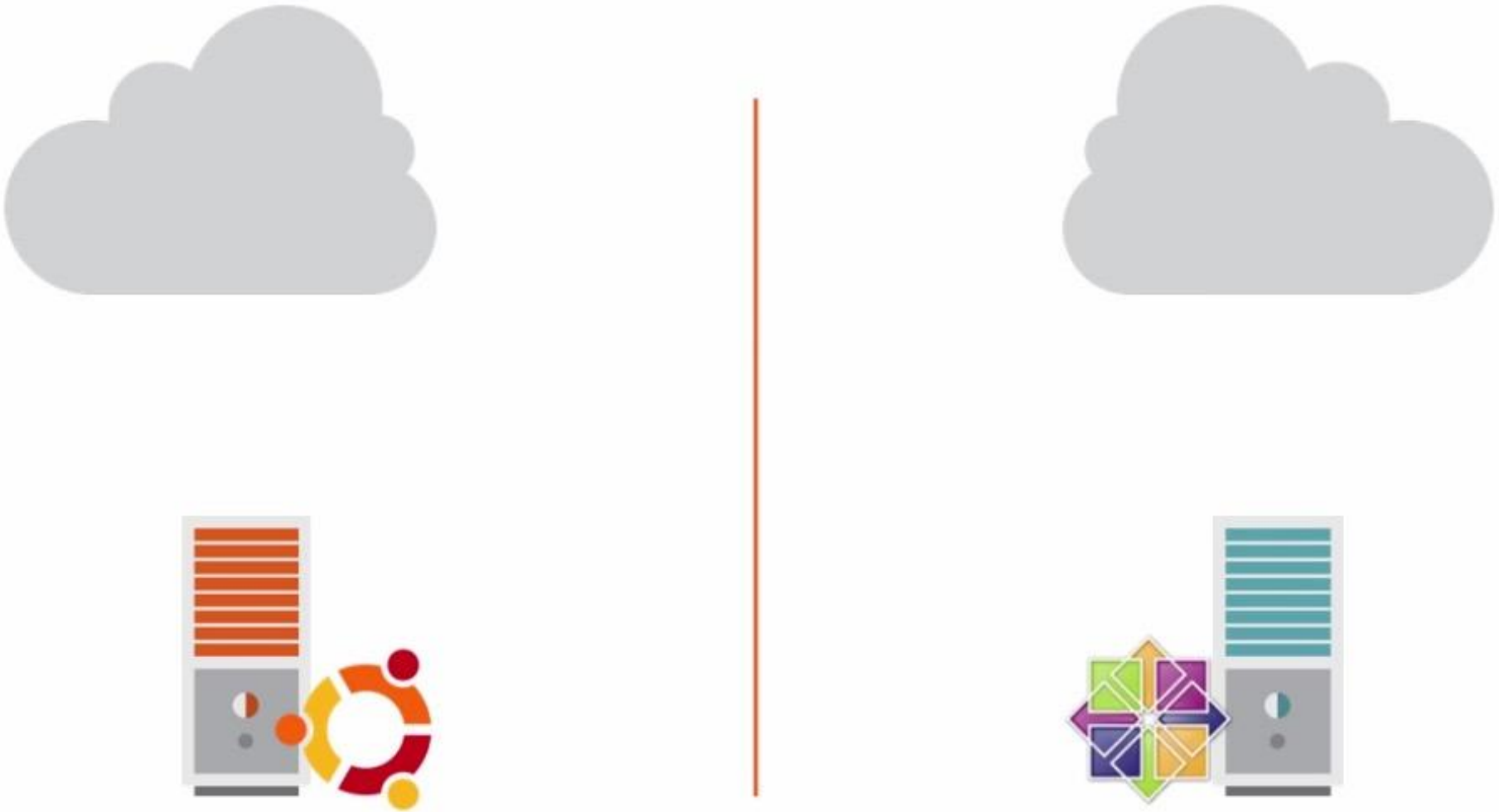
The Future of Docker

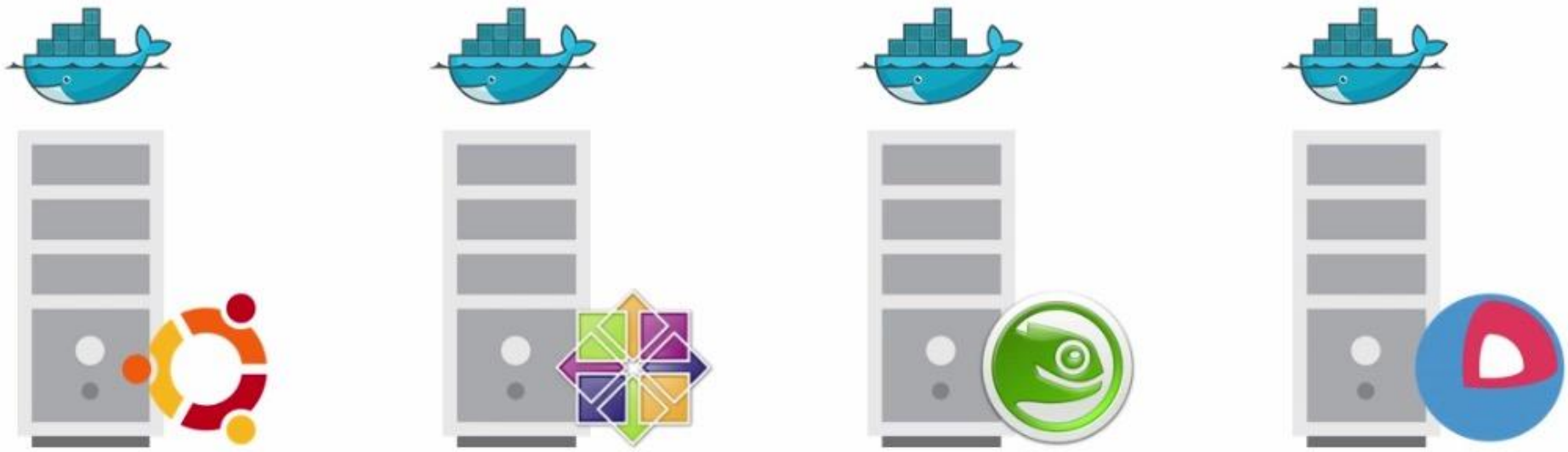
Coming up Next….

Installing Ubuntu and CentOS Linux

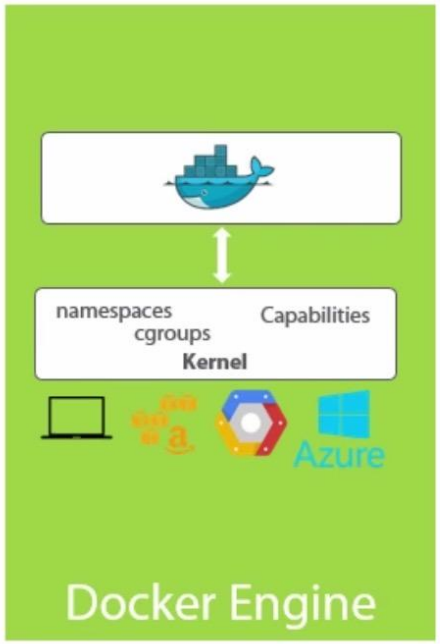
Installing and updating the Docker Daemon


Container
Container
Container
Container
Container
Container
Container
Container
Container
Container
Operating System
CPU
Physical Machine





Major Docker Components

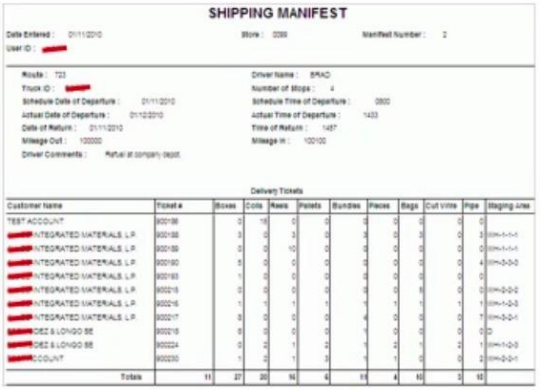


Docker Engine

(Shipping Yard)

Docker Images

(Manifests)



SHIPPING MANIFEST

Date Entered : 01/11/2010 Store : 0299 Manifest Number : 2

User ID : [redacted]

Route : 723 Driver Name : BRAD

Truck ID : [redacted] Number of Stops : 4

Schedule Date of Departure : 01/11/2010 Schedule Time of Departure : 0800

Actual Date of Departure : 01/12/2010 Actual Time of Departure : 1433

Date of Return : 01/11/2010 Time of Return : 1457

Mileage Out : 100000 Mileage In : 100100

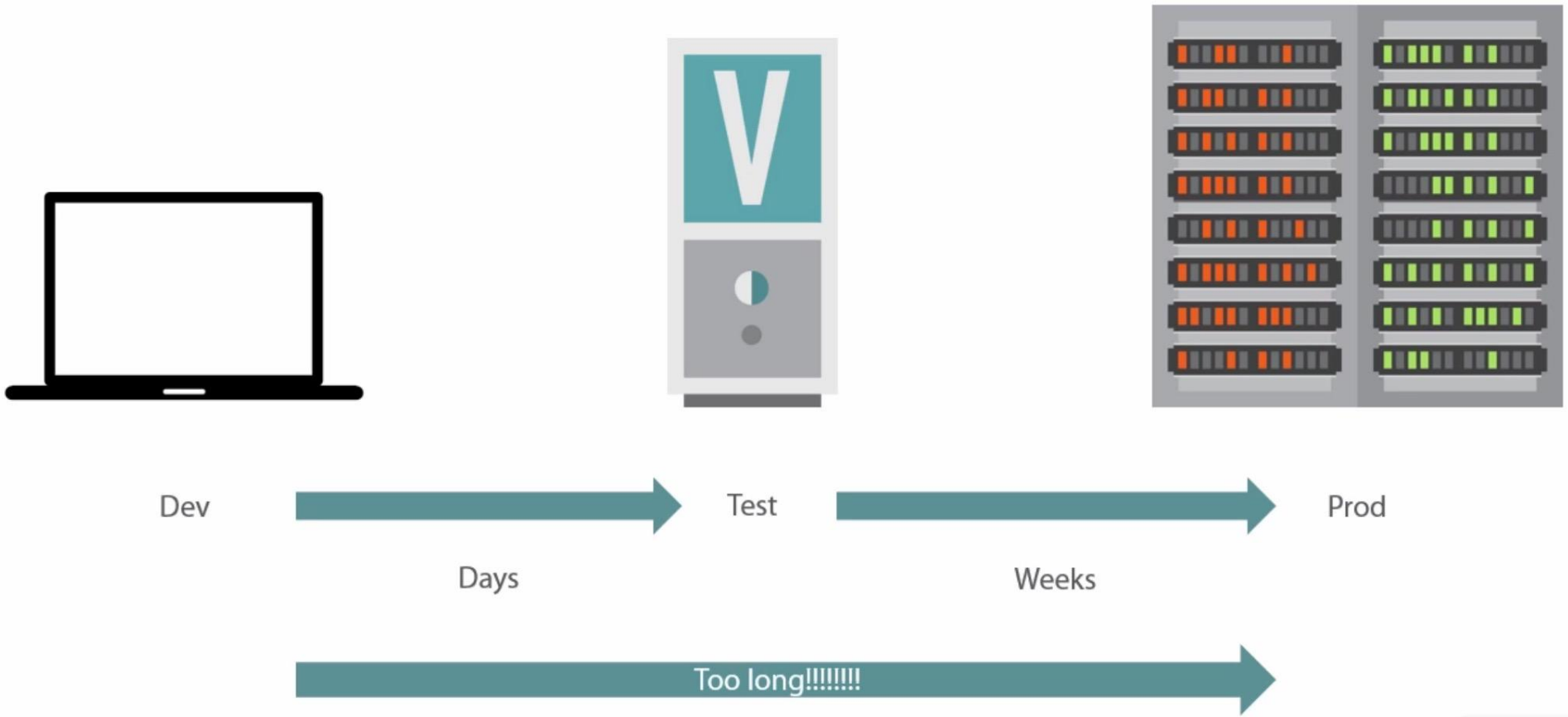
Driver Comments : Refusal at company depot.

Delivery Tickets

Customer Name	Ticket #	Boxes	Coils	Rees	Pallets	Bundles	Pieces	Bags	Cut Wire	Pipe	Staging Area
TEST ACCOUNT	900186	0	10	0	0	0	0	0	0	0	
[redacted] INTEGRATED MATERIALS, L.P.	900188	3	0	3	0	0	0	0	0	3	03A-1-1-1
[redacted] INTEGRATED MATERIALS, L.P.	900189	0	0	10	0	0	0	0	0	0	03A-1-1-1
[redacted] INTEGRATED MATERIALS, L.P.	900190	5	0	0	0	0	0	0	0	4	03A-2-2-2
[redacted] INTEGRATED MATERIALS, L.P.	900193	1	0	0	0	0	0	0	0	0	
[redacted] INTEGRATED MATERIALS, L.P.	900215	0	0	0	0	0	0	0	0	0	03A-2-2-2
[redacted] INTEGRATED MATERIALS, L.P.	900216	1	0	1	1	1	1	1	1	1	03A-1-2-3
[redacted] INTEGRATED MATERIALS, L.P.	900217	0	0	0	0	0	0	0	0	7	03A-2-2-1
[redacted] DEZ & LONGO SE	900218	0	0	0	0	0	0	0	0	0	0
[redacted] DEZ & LONGO SE	900224	0	0	1	0	0	0	0	0	1	03A-1-2-3
[redacted] ACCOUNT	900230	1	0	1	0	0	0	0	0	2	03A-2-2-1
Totals		11	27	20	10	4	10	4	10	0	10	

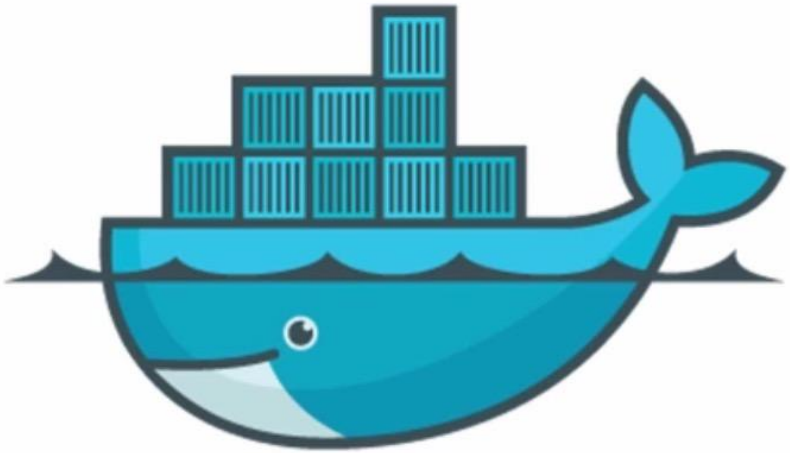
Docker Containers

(Shipping Containers)


Dev
Test
Prod
Days
Weeks
Too long!!!!!!!!

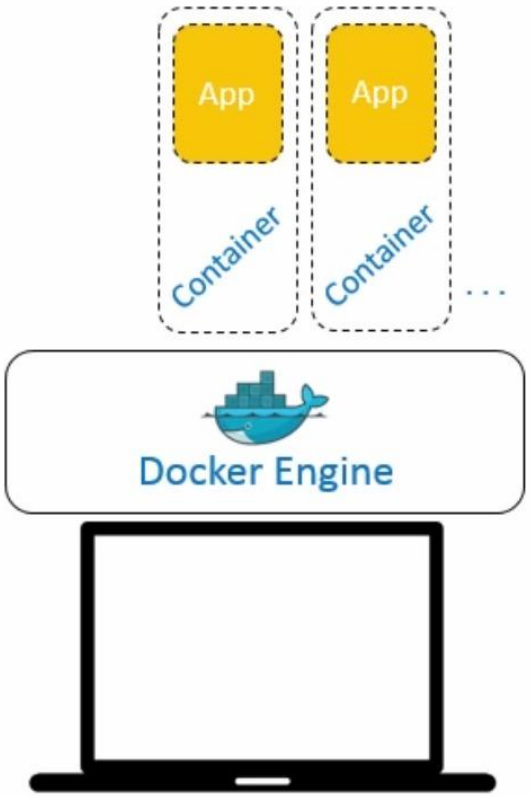
Containers can contain malicious code

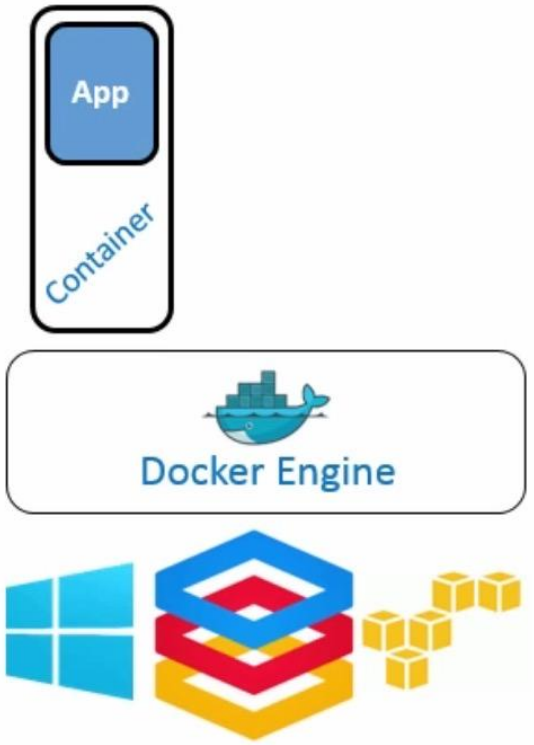
Trust the code you run!



Docker Engine


Container
Yard 2
RTG cranes
Berth 2
Rail terminal
Ship-to-shore cranes
RTG cranes
Container
Yard 1
RTG cranes
Berth 1
RTG cranes
RTG cranes


App
App
Container
Container
. . .
Docker Engine


App
Container
Docker Engine

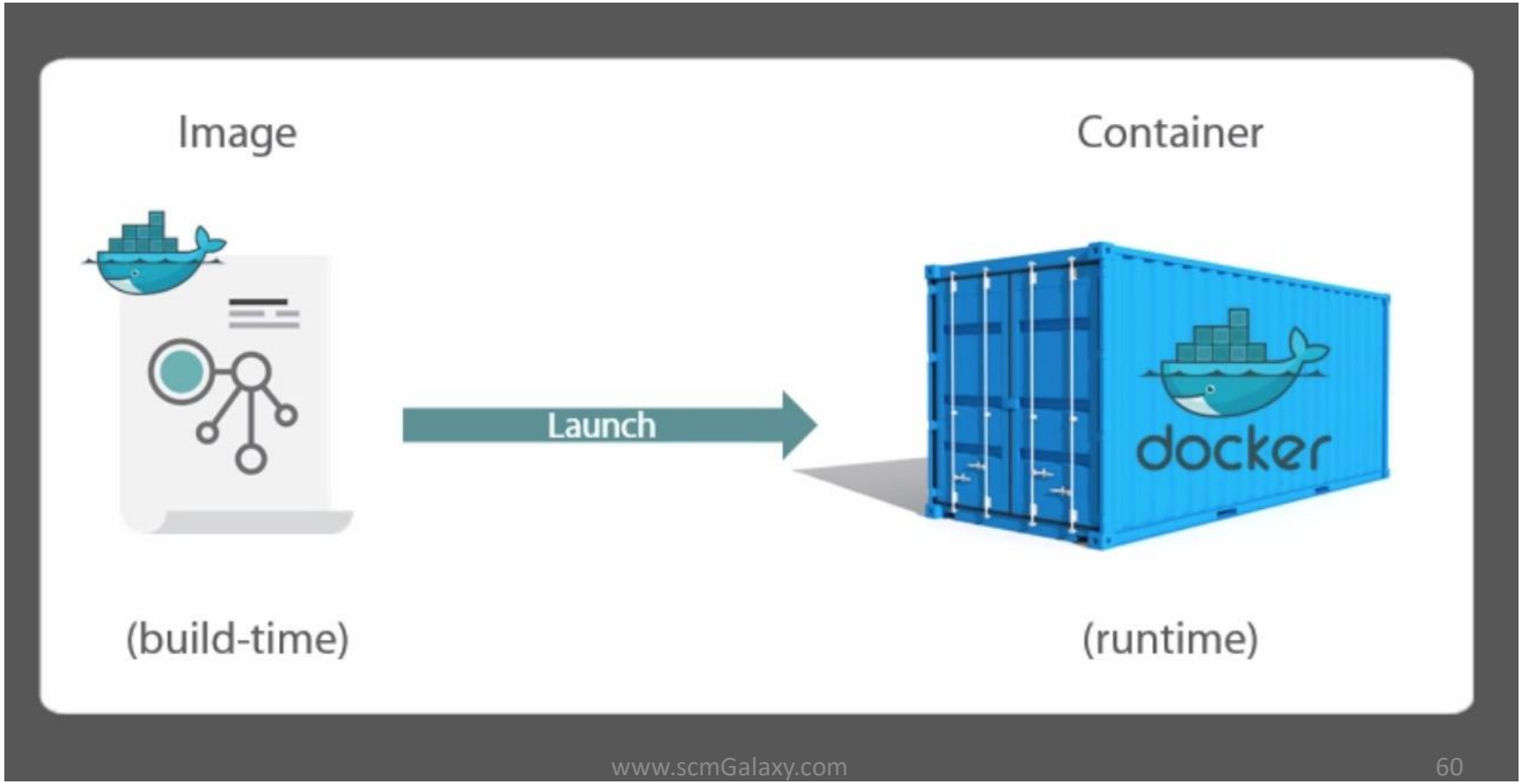


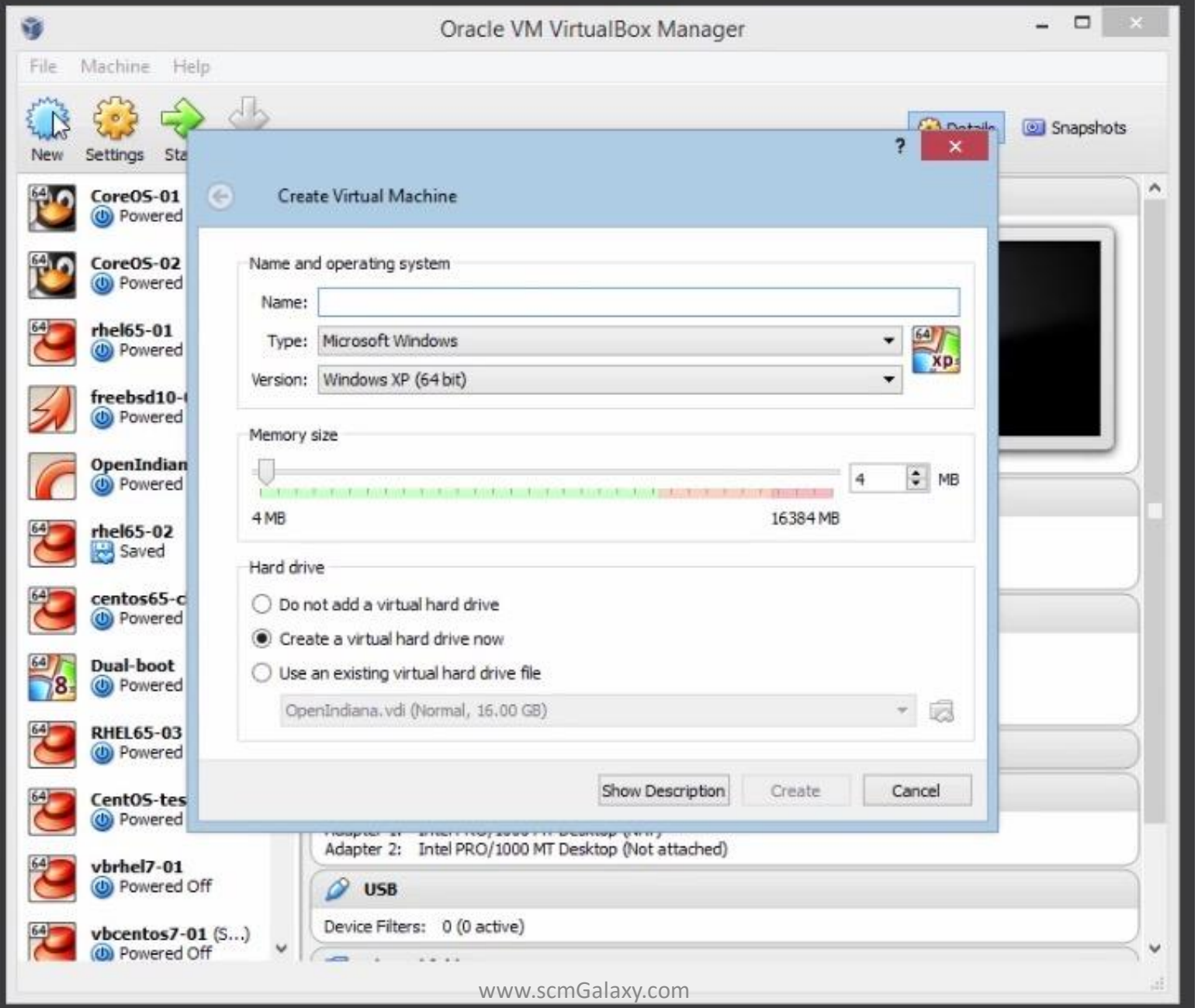
Image
Container
Launch
docker
(build-time)
(runtime)

```
root@ubuntu1404-02:~#
root@ubuntu1404-02:~#
root@ubuntu1404-02:~#
root@ubuntu1404-02:~# docker images fedora
REPOSITORY          TAG                 IMAGE ID            CREATED             VIRTUAL SIZE
fedora              20                  2e613b1df961        5 days ago          374.1 MB
fedora              heisenbug           2e613b1df961        5 days ago          374.1 MB
fedora              rawhide             9da4e4dbc6be        5 days ago          379.5 MB
fedora              21                  bfe0bb6667e4        5 days ago          250.2 MB
fedora              latest              bfe0bb6667e4        5 days ago          250.2 MB
root@ubuntu1404-02:~#
root@ubuntu1404-02:~#
root@ubuntu1404-02:~#
root@ubuntu1404-02:~# docker pull coreos/etcd
Pulling repository coreos/etcd
649a8540db5e: Pulling fs layer
a2d6848456de: Download complete
995c2b064ad7: Download complete
d4e20e5984a9: Download complete
bfd1f10d9c74: Download complete
```

Images stored locally under **/var/lib/docker/<storage driver>**

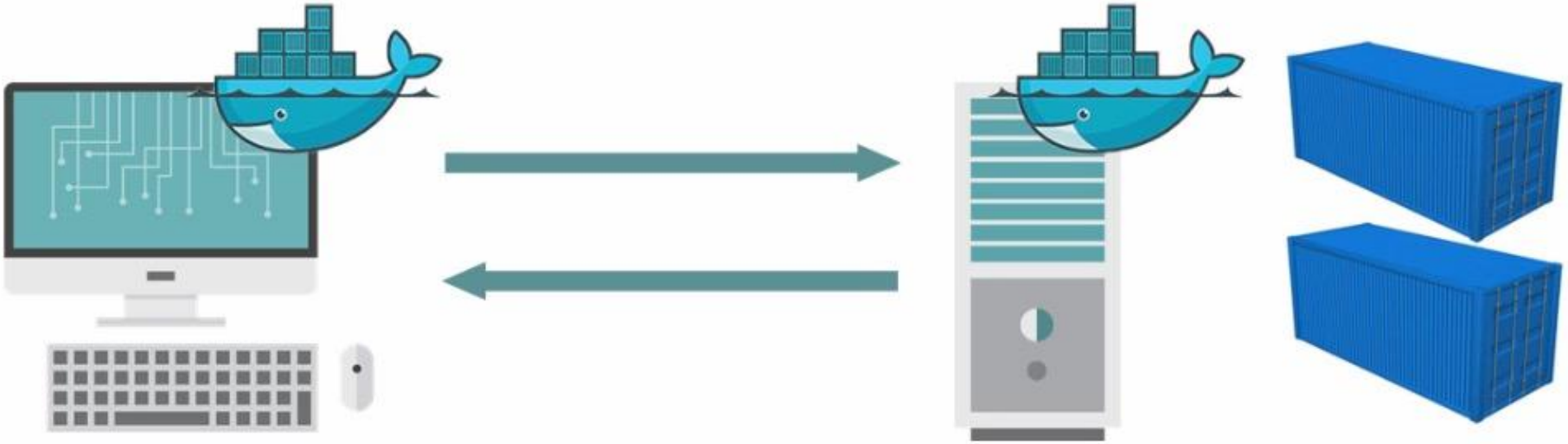

d not found
u1404-02:~#
#
#
Ctrl + P + Q

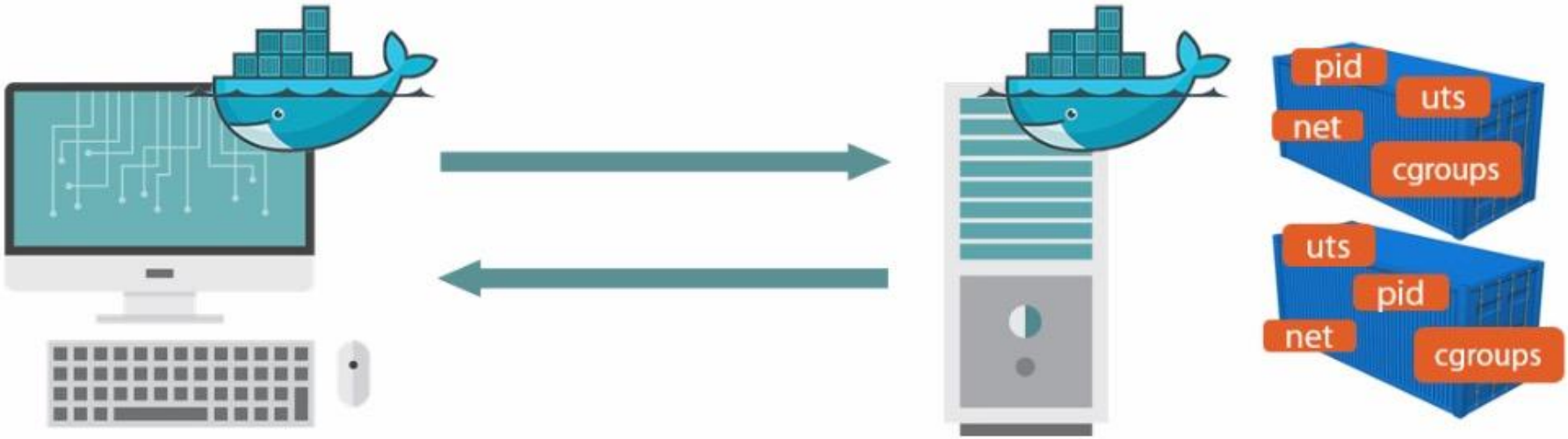

Ubuntu Community Ask! Developer Design Discourse Hardware Insights Juju Partners Shop More
ubuntu
Cloud Server Desktop Phone Tablet TV Management Download
Search
Download › Server › ARM POWER8
Download Ubuntu Server
Ubuntu Server 14.04.1 LTS
The Long Term Support version of Ubuntu Server, including the Icehouse release of OpenStack and support guaranteed until April 2019 — 64 bit only.
Recommended for most users.
Ubuntu Server 14.04.1 LTS release notes
Download
Alternative downloads and torrents ›


Oracle VM VirtualBox Manager
File Machine Help
New Settings
Snapshots
CoreOS-01 Powered
CoreOS-02 Powered
rhel65-01 Powered
freebsd10- Powered
OpenIndian Powered
rhel65-02 Saved
centos65-c Powered
Dual-boot Powered
RHEL65-03 Powered
CentOS-tes Powered
vbrhel7-01 Powered Off
vbcentos7-01 (S...) Powered Off
Create Virtual Machine
Name and operating system
Name:
Type: Microsoft Windows
Version: Windows XP (64 bit)
Memory size
4 MB
4 MB
16384 MB
Hard drive
Do not add a virtual hard drive
Create a virtual hard drive now
Use an existing virtual hard drive file
OpenIndiana.vdi (Normal, 16.00 GB)
Show Description
Create
Cancel
Adapter 2: Intel PRO/1000 MT Desktop (Not attached)
USB
Device Filters: 0 (0 active)

Installing and Updating Docker

Client & Deamon




pid
uts
net
cgroups
uts
pid
net
cgroups

Install on Ubantu / Centos

```
[root@localhost ~]# docker version
Client version: 1.3.2
Client API version: 1.15
Go version (client): go1.3.3
Git commit (client): 39fa2fa/1.3.2
OS/Arch (client): linux/amd64
Server version: 1.3.2
Server API version: 1.15
Go version (server): go1.3.3
Git commit (server): 39fa2fa/1.3.2
[root@localhost ~]#
[root@localhost ~]#
[root@localhost ~]#
[root@localhost ~]# docker info
Containers: 0
Images: 0
Storage Driver: devicemapper
 Pool Name: docker-253:1-21646475-pool
 Pool Blocksize: 65.54 kB
 Data file: /var/lib/docker/devicemapper/devicemapper/data
 Metadata file: /var/lib/docker/devicemapper/devicemapper/metadata
 Data Space Used: 307.2 MB
 Data Space Total: 107.4 GB
 Metadata Space Used: 729.1 kB
 Metadata Space Total: 2.147 GB
 Library Version: 1.02.84-RHEL7 (2014-03-26)
Execution Driver: native-0.2
Kernel Version: 3.10.0-123.el7.x86_64
Operating System: CentOS Linux 7 (Core)
[root@localhost ~]#
```

Install on Ubuntu

- `apt-get install docker.io`

Install on Red Hat/CentOS

- `yum install docker`

Basic Commands

`docker -v`

- Basic version info

`docker version`

- Shows more detailed version information

`docker info`

- Number of containers
- Number of images
- Storage driver
- Execution Driver
- etc....

Docker needs root

Install Docker on Ubantu using root

> apt-get update

> apt-get install -y docker.io

> service docker.io start

Install Docker on Redhar / Centos

> yum update

> yum install -y docker.io

> systemctl start docker.service

How to verify the version of docker?

> docker -v

> docker version

How to know Docker running?

> service docker.io status

> systemctl status docker.service

How to check details of Docker clients, deamon, containers, images, drivers, etc

> docker info

Skip Next 3 Slides

Update Docker version

> wget -q0- https://get.docker.com/gpg | apt-key add -

> echo deb http://get.docker.com/ubantu docker main > /etc/apt/sources.list.d/docker.list

> apt-get update

> apt-get install lxc-docker

> docker version

Adding Users to the Docker Group (Docker Config (Need root to work)

> docker run -it ubuntu /bin/bash (as a non-root)

[permission denied]

> cat /etc/group

> sudo gpasswd -a username docker

> cat /etc/group

> docker run -it ubuntu /bin/bash (as a non-root)

> logout

> login username

Setup Network to Docker Container

> docker -v

> netstat -tlp

> service docker stop

> docker -H ipaddress:port -d &

> netstat -tlp

> export DOCKER_HOST="tcp://ipaddress:port" (from another machine)

> docker version

Images vs Container

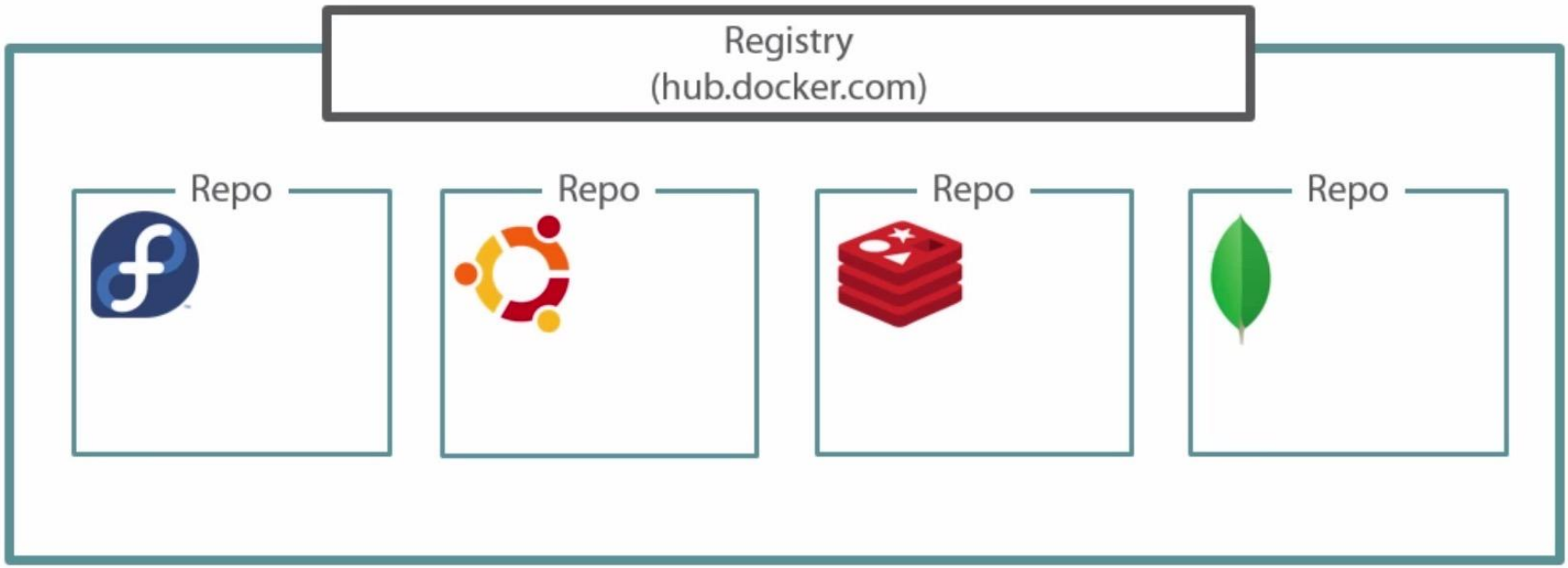
An instance of an image is called container. If you start this image, you have a running container of this image.

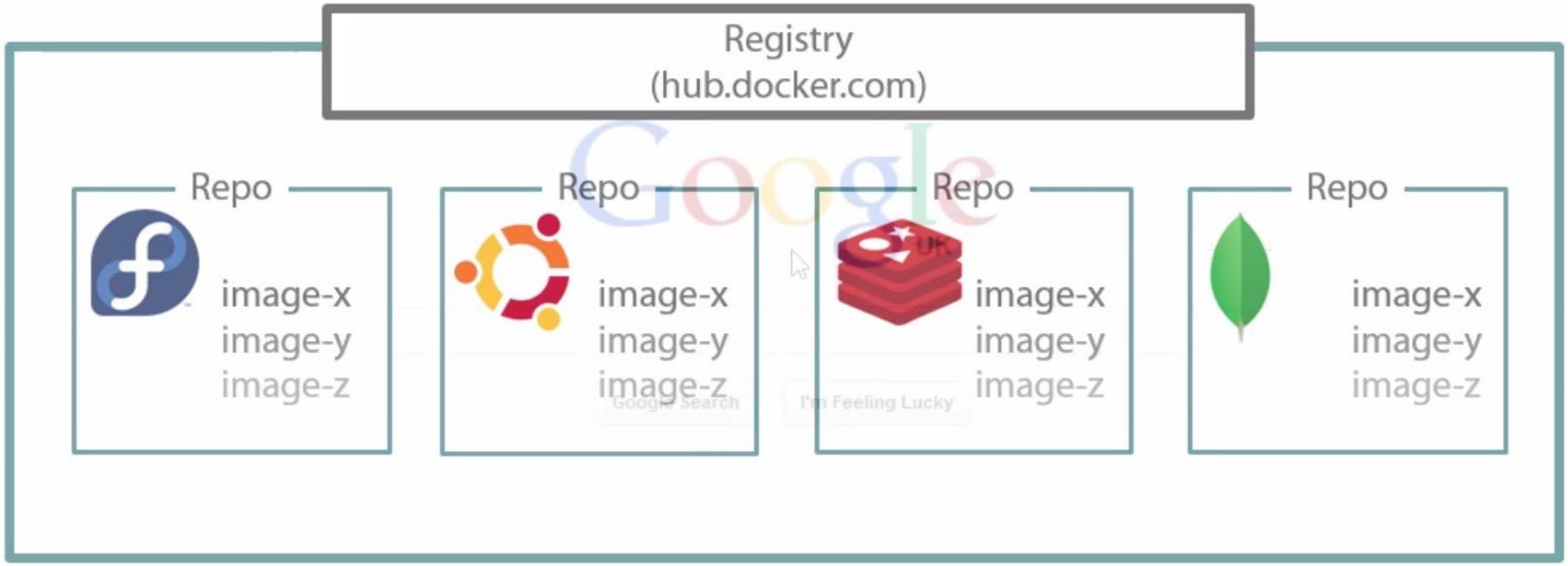
You can have many running containers of the same image. You can see all your images with docker images whereas you can see your running containers with docker ps (and you can see all containers with docker ps -a).

So a running image is a container.

Reference

http://stackoverflow.com/questions/23735149/docker-image-vs-container


Registry
(hub.docker.com)
Repo
Repo
Repo
Repo


Registry
(hub.docker.com)
Repo
image-x
image-y
image-z
Repo
image-x
image-y
image-z
Repo
image-x
image-y
image-z
Repo
image-x
image-y
image-z

Docker Images

- docker pull -a fedora
- Docker info

> docker run -it fedora /bin/bash

> docker images fedora

[Images are stored under /var/lib/docker/<storage drivers>

Docker Containers

> docker run -it ubuntu /bin/bash

> docker images

> docker ps

> docker attach <container_id>

> docker ps -a

Docker Registries and Repositories

hub.docker.com

Setup Jenkins Using Docker

Pull the official jenkins image from Docker repository.

> docker pull jenkins

Next, run a container using this image and map data directory from the container to the host; e.g in the example below /var/jenkins_home from the container is mapped to jenkins/ directory from the current path on the host. Jenkins 8080 port is also exposed to the host as 49001.

> docker run -d -p 49001:8080 -v $PWD/jenkins:/var/jenkins_home -t jenkins

Other commands

> docker run -p 8080:8080 jenkins
> docker create -v /var/jenkins_home --name jenkins-dv jenkins

> docker run -d -p 8080:8080 --volumes-from jenkins-dv --name myjenkins jenkins
> http://localhost:8080

> docker run -d -p 8080:8080 --volumes-from jenkins-dv --name myjenkins2 jenkins

Questions