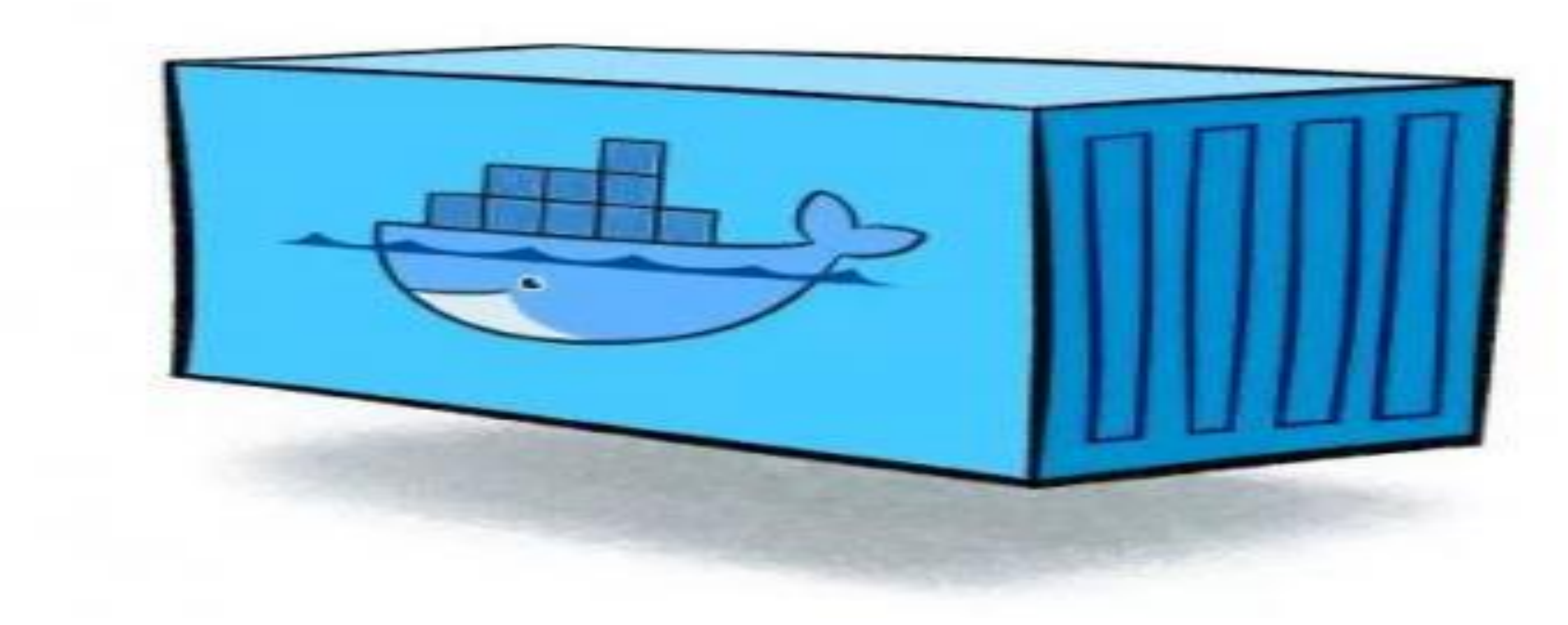
Containerizations

Unit objectives

After completing this unit, you should be able to:

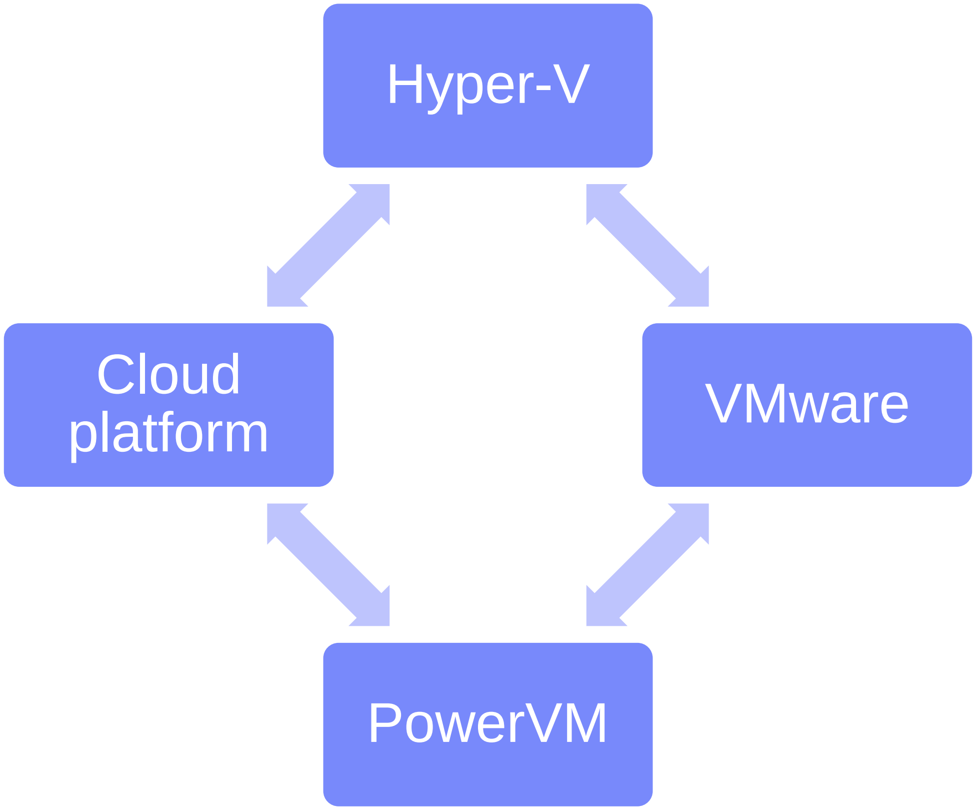
- Introduction to Containers
- Introduction to Docker
- Manage Images
- Manage Containers
- Setup real scenario
  - LAMP (Linux → Apache → MySQL → PHP)

Introduction to Containers



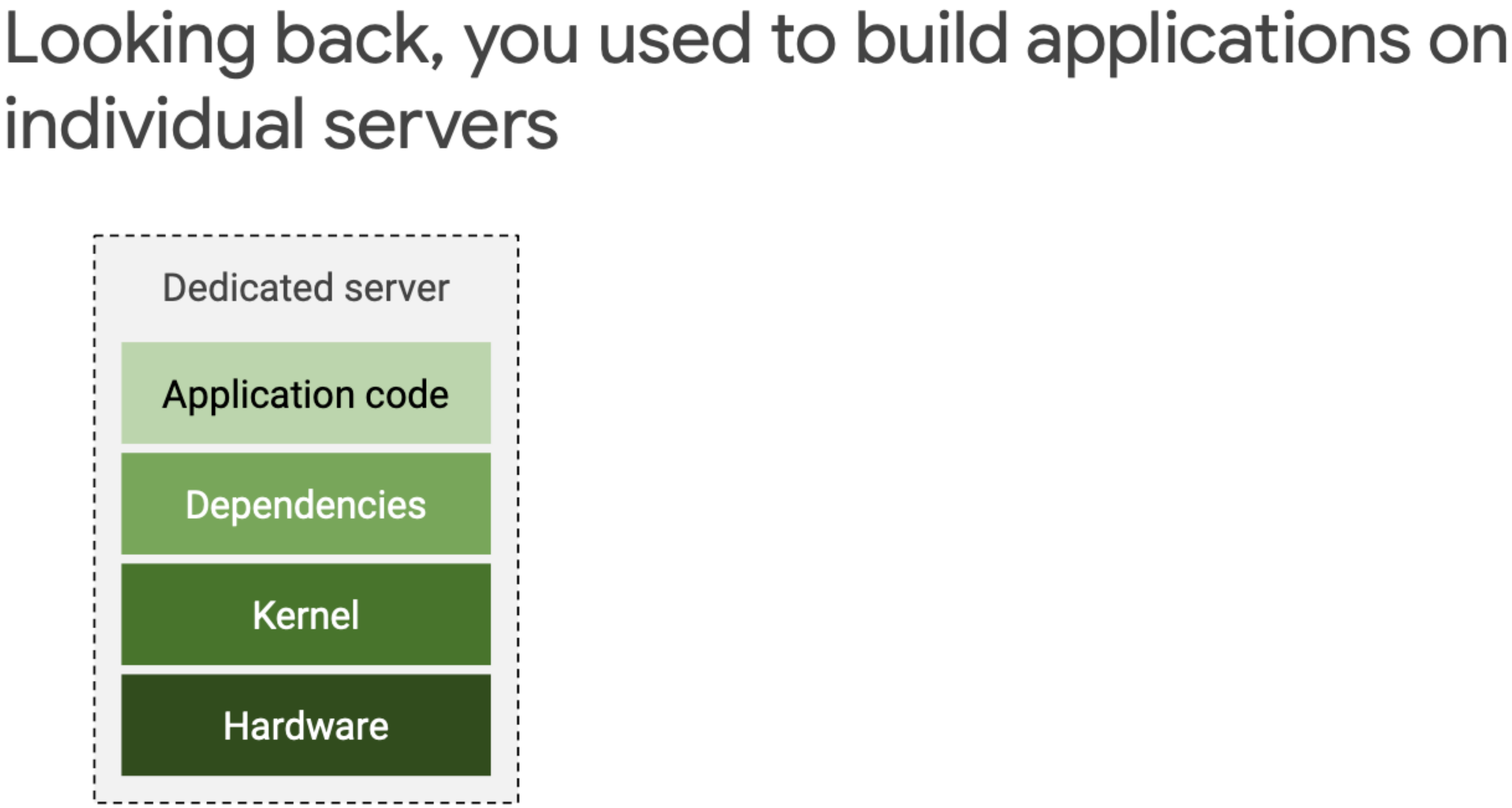
Introduction of Containers

- Small and portable
- Good Level of abstraction
- Easibly migrateable between platform or environment



Introduction of Containers

- Small and portable
- Good Level of abstraction



Looking back, you used to build applications on individual servers

Dedicated server

Application code

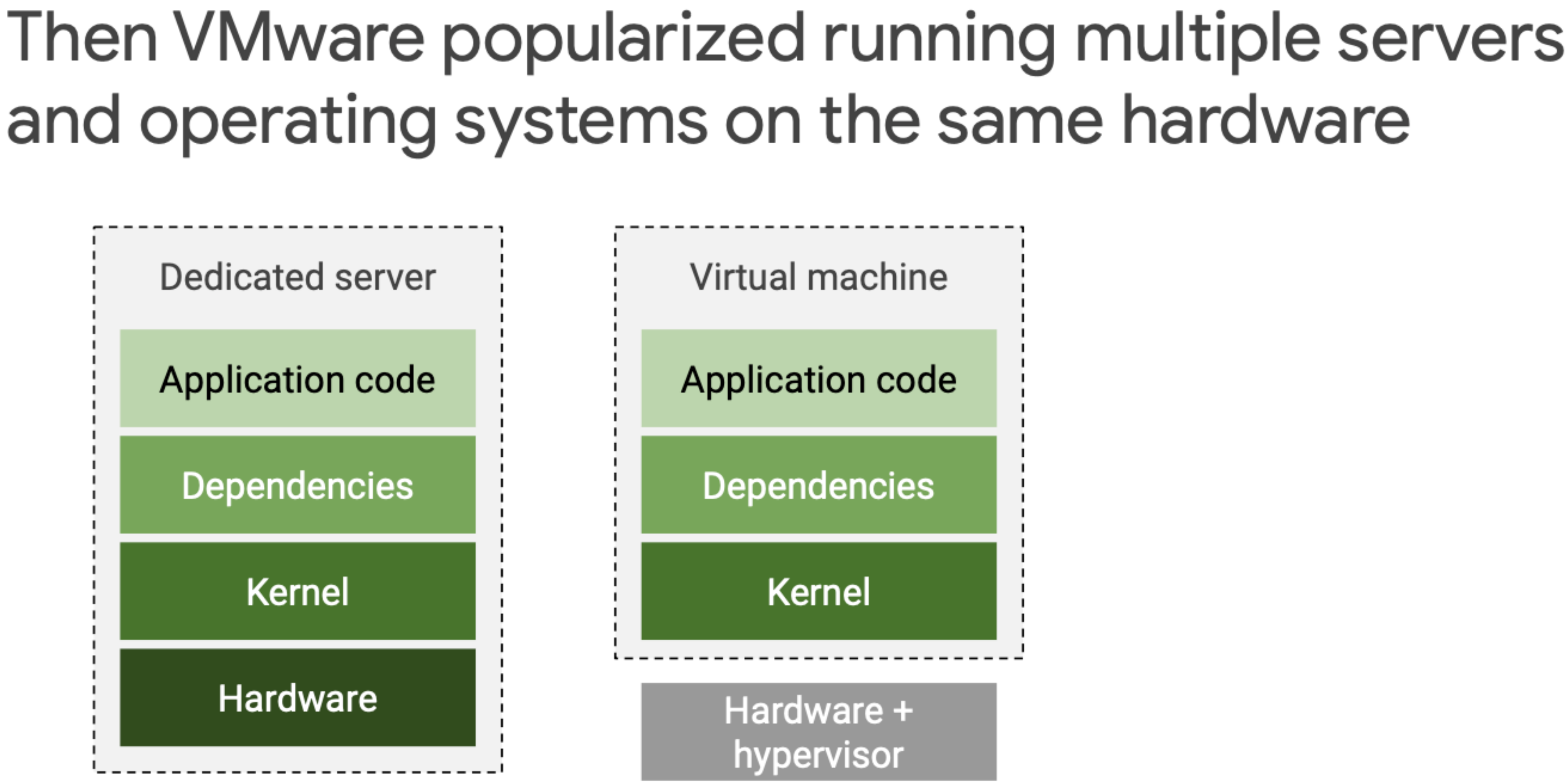
Dependencies

Kernel

Hardware

Introduction of Containers

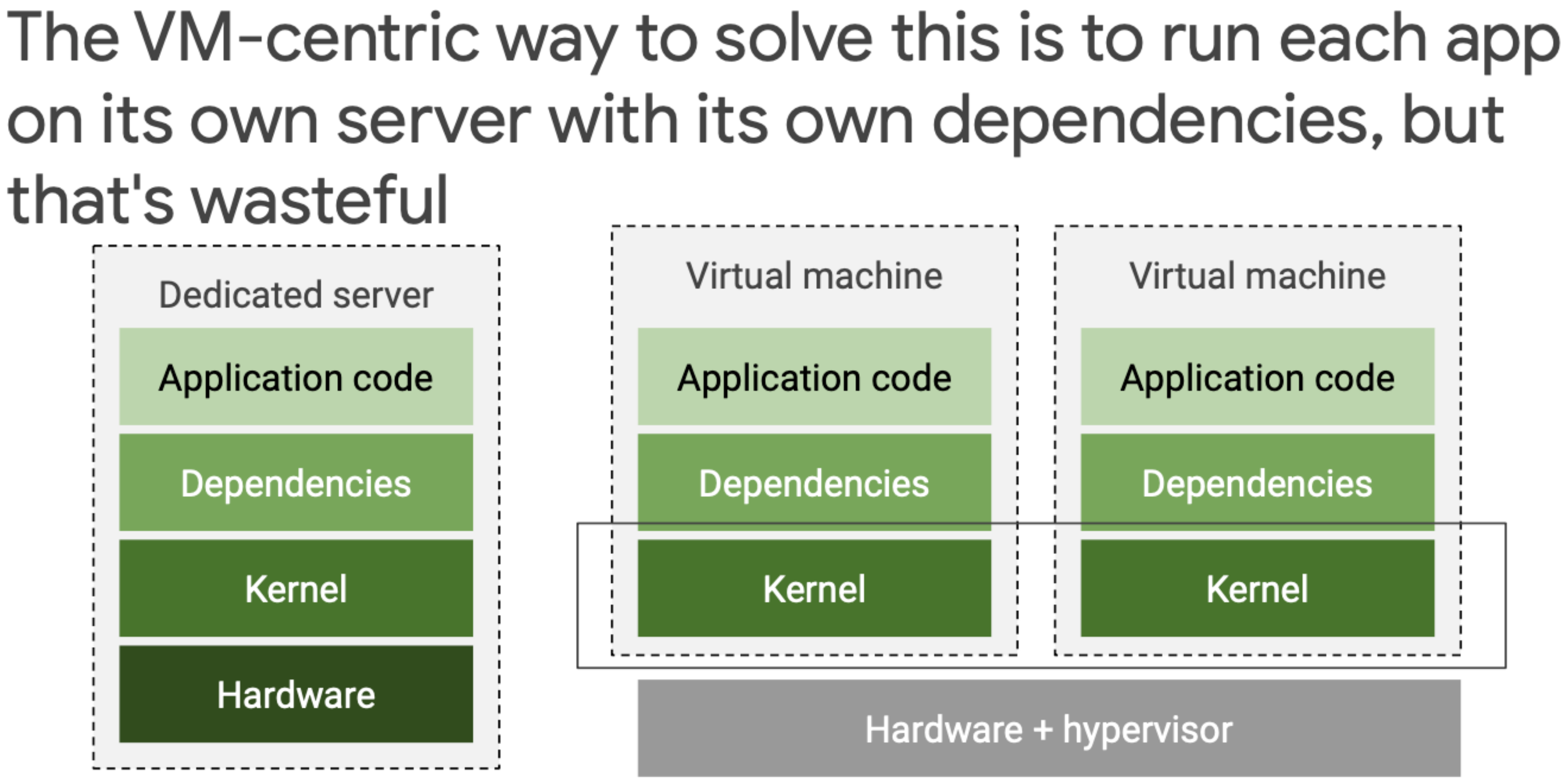
- Small and portable
- Good Level of abstraction



Then VMware popularized running multiple servers and operating systems on the same hardware

Introduction of Containers

- Small and portable
- Good Level of abstraction

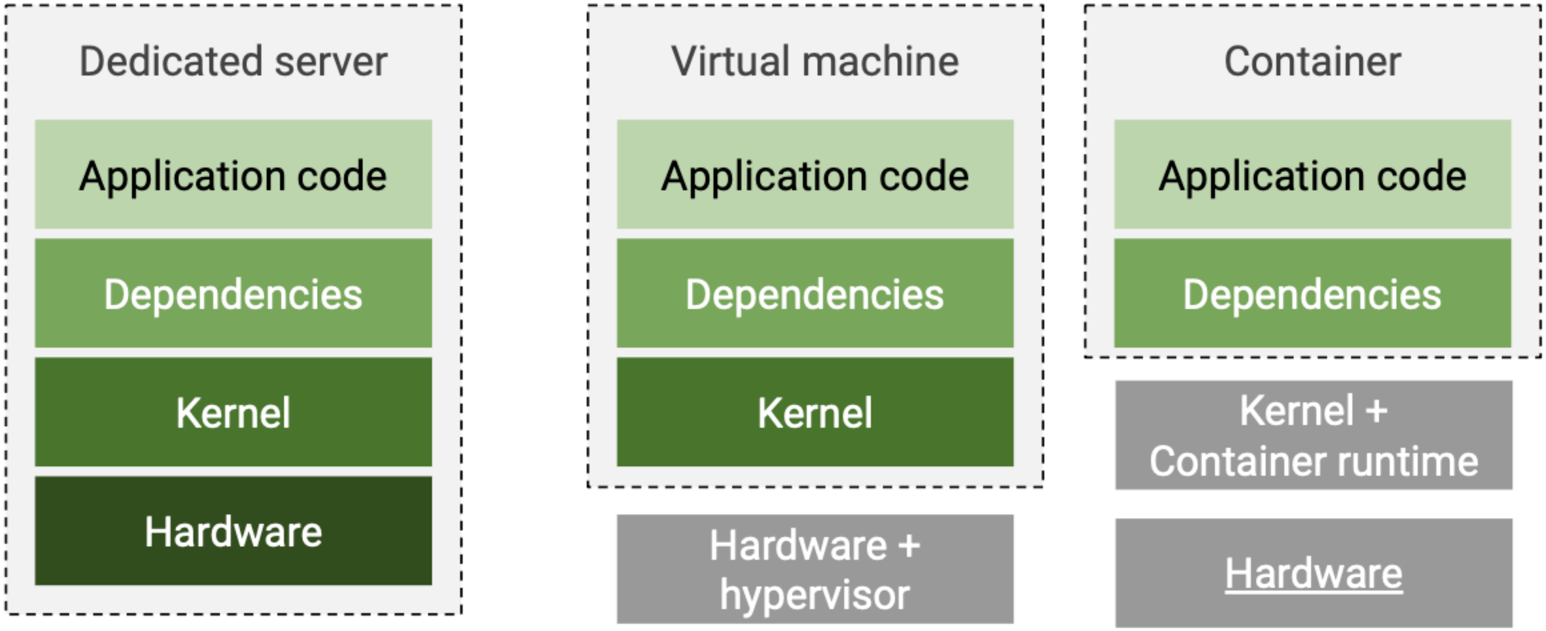


The VM-centric way to solve this is to run each app on its own server with its own dependencies, but that's wasteful

Introduction of Containers

- Small and portable
- Good Level of abstraction

So you raise the abstraction one more level and virtualize the OS



Dedicated server

Application code

Dependencies

Kernel

Hardware

Virtual machine

Application code

Dependencies

Kernel

Hardware + hypervisor

Container

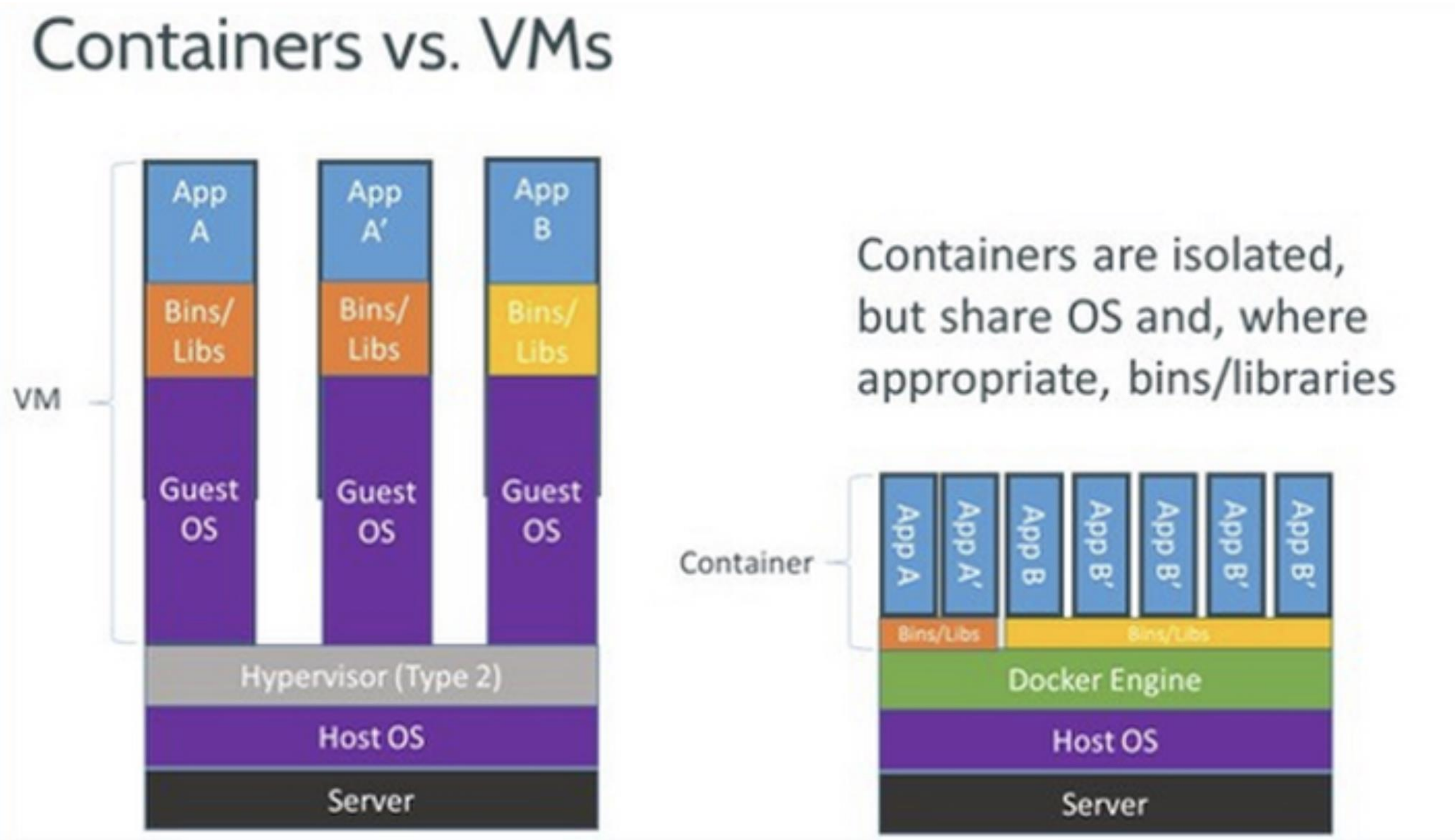
Application code

Dependencies

Kernel + Container runtime

Hardware

Containers vs VMs

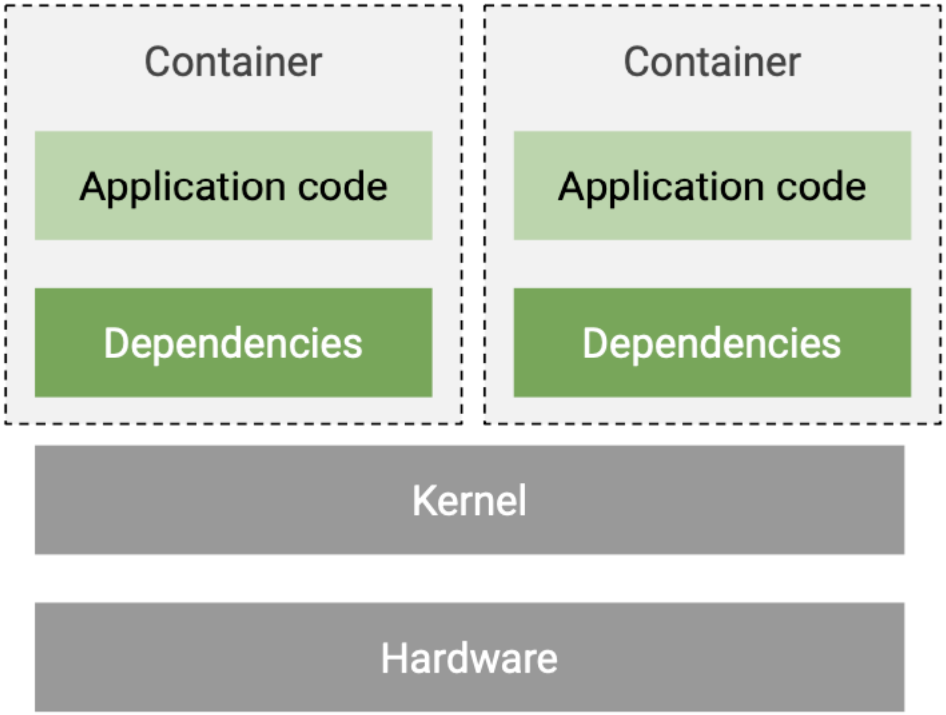


Containers vs. VMs

Containers are isolated, but share OS and, where appropriate, bins/libraries

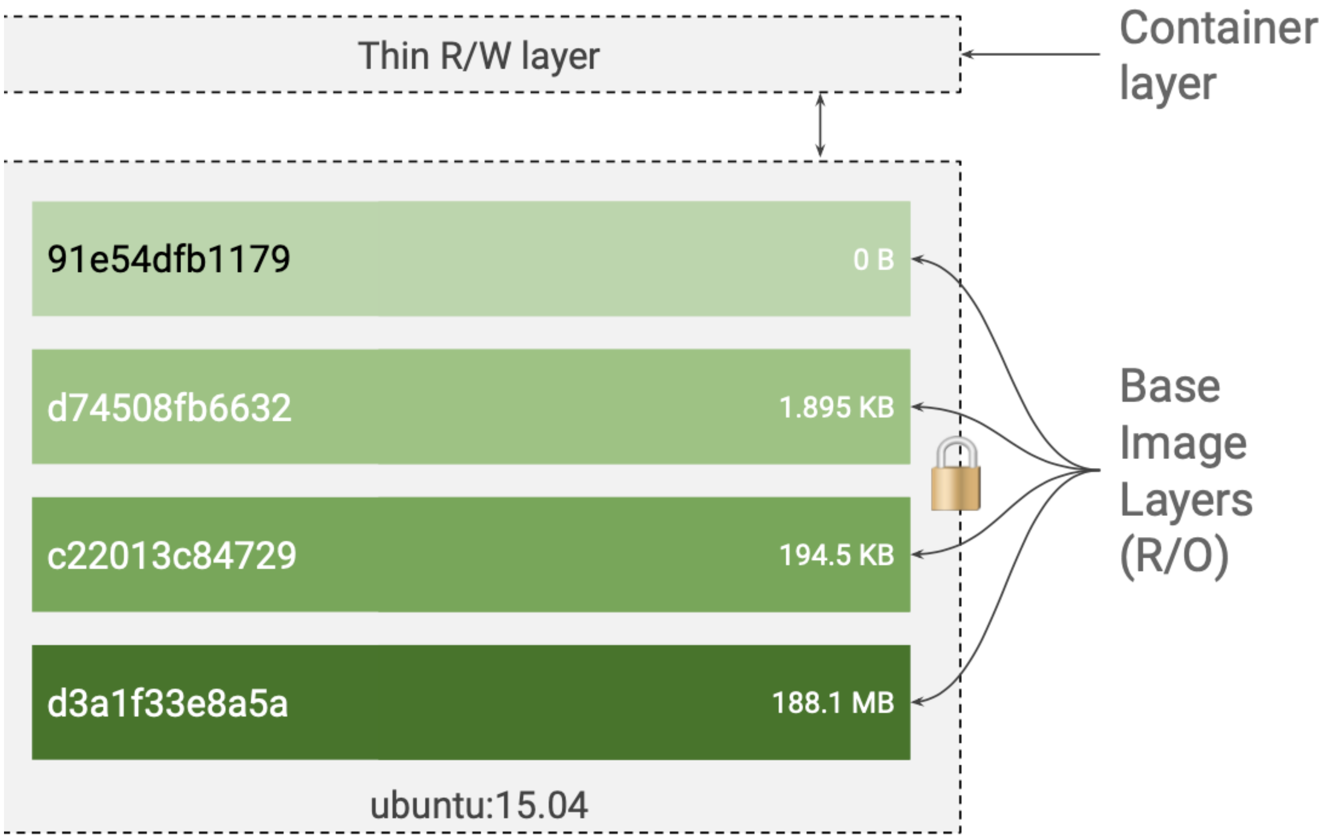
Why developers like containers

- Code works the same everywhere:
  - Across dev, test & production
  - Across bare-metal, VMs, cloud
- Packaged apps speed development
  - Rapid creation and deployment
  - CI/CD
  - Single file copy
- Provide best path to micro-services environment
- Isolated & elastic



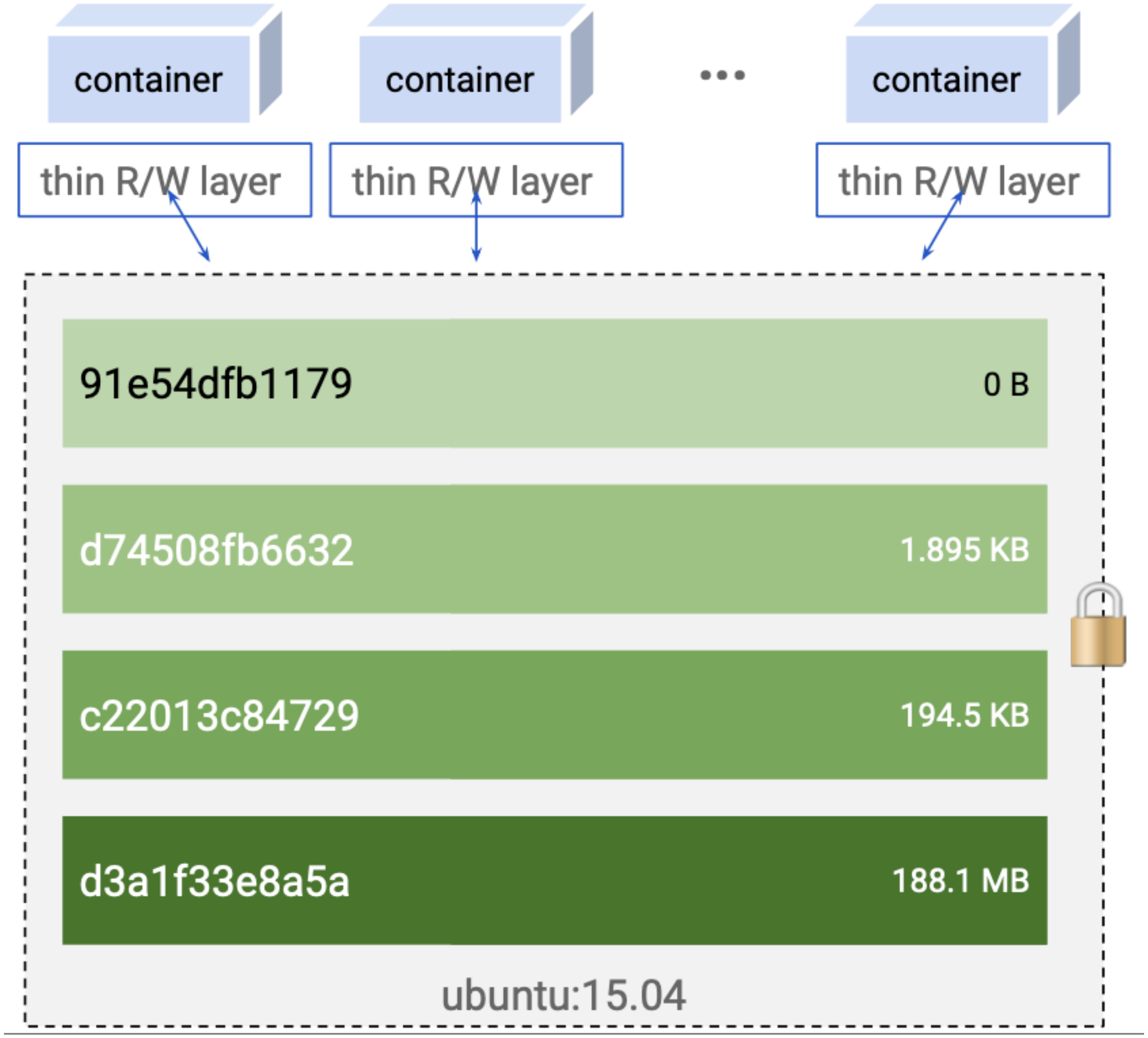
Containers use layered file system

- With top layer writable
- Start multiple containers from same Docker image



Containers promotes smaller shared images

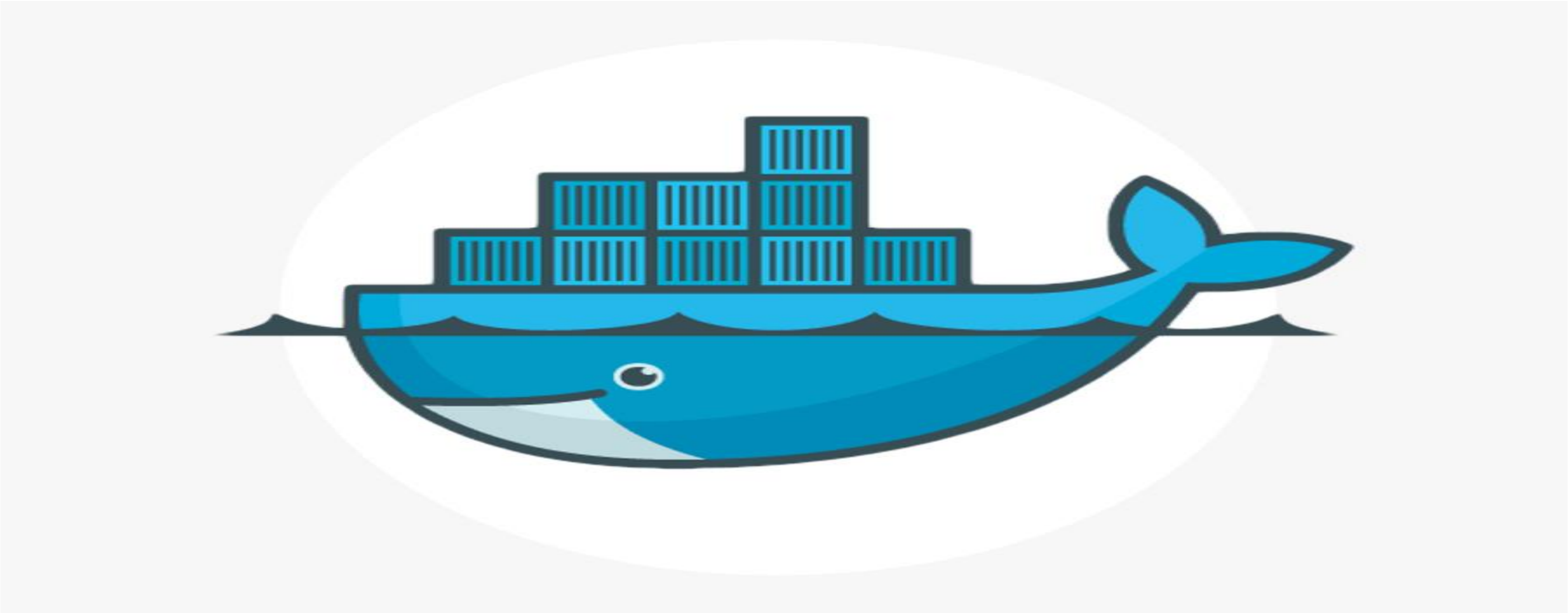
- Base image size about 200 MB.
- As container spawn up, it may consumed only 100-200 KB.
- Instead of copy whole image, it creates layer with delta data only.
- Fast boot time



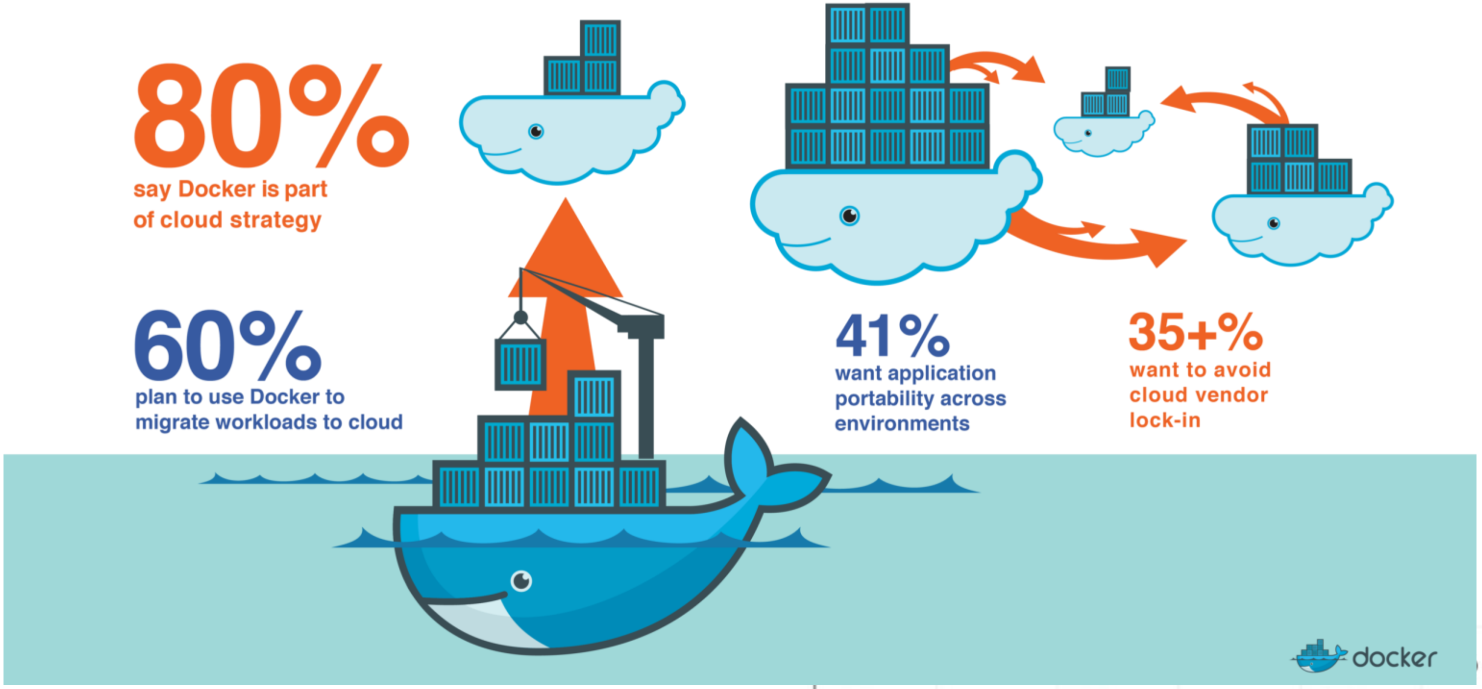
Benefits of Containers

- Return of Investment (ROI) and Cost Saving
- Standardization and Productivity
- Excellent for micro-services
- Efficiency for Continuous Integration and Continuous Deployment
- Compatibility and Maintainability
- Simplicity and Faster configurations
- Rapid Deployment
- Widely used in Multi-Cloud Platforms

Introduction to Docker



Docker – Fastest growing technology

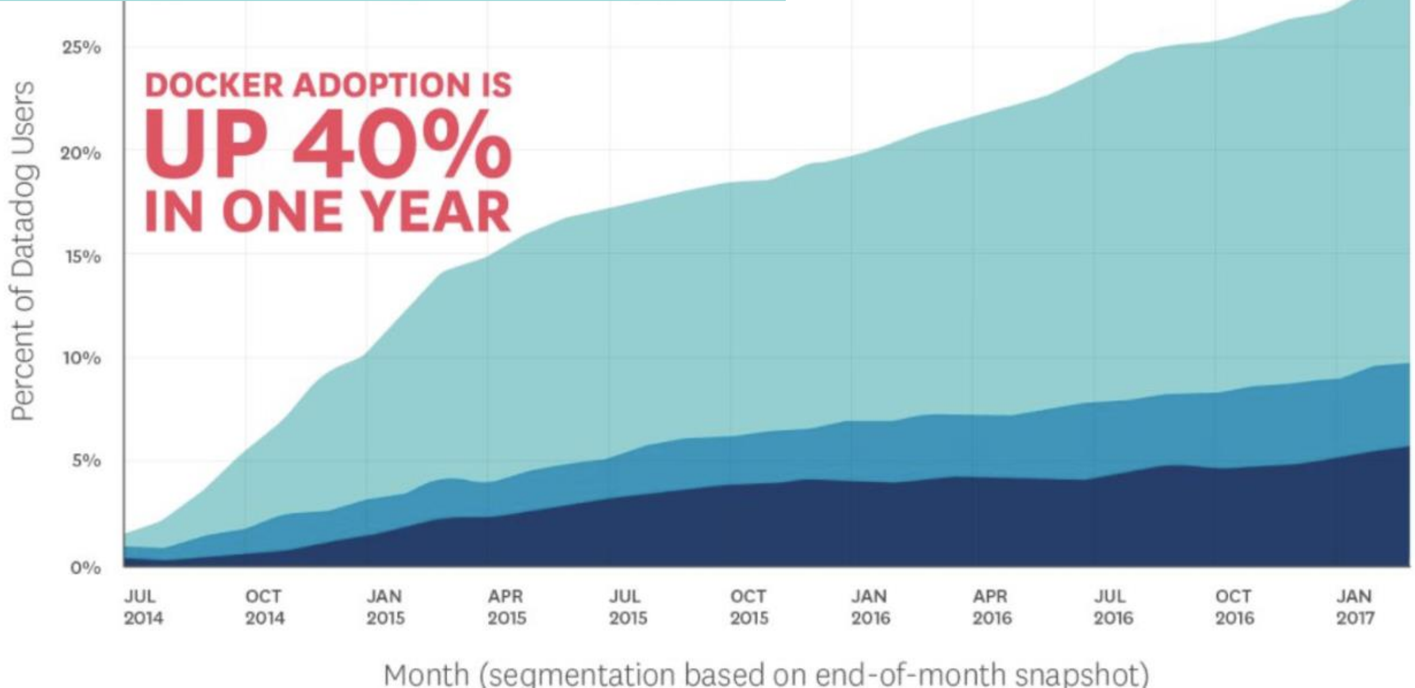


80% say Docker is part of cloud strategy

60% plan to use Docker to migrate workloads to cloud

41% want application portability across environments

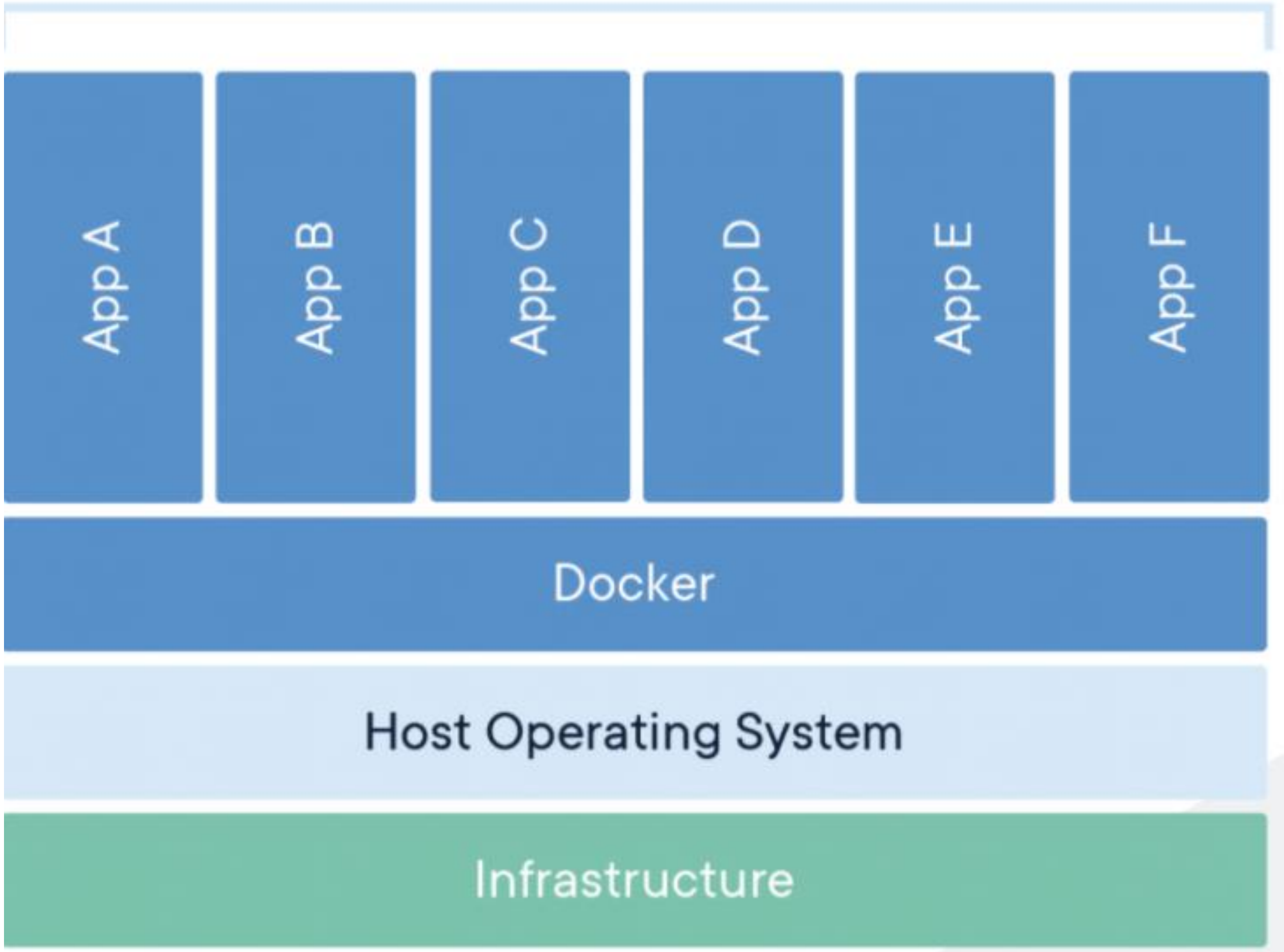
35+% want to avoid cloud vendor lock-in



DOCKER ADOPTION IS UP 40% IN ONE YEAR

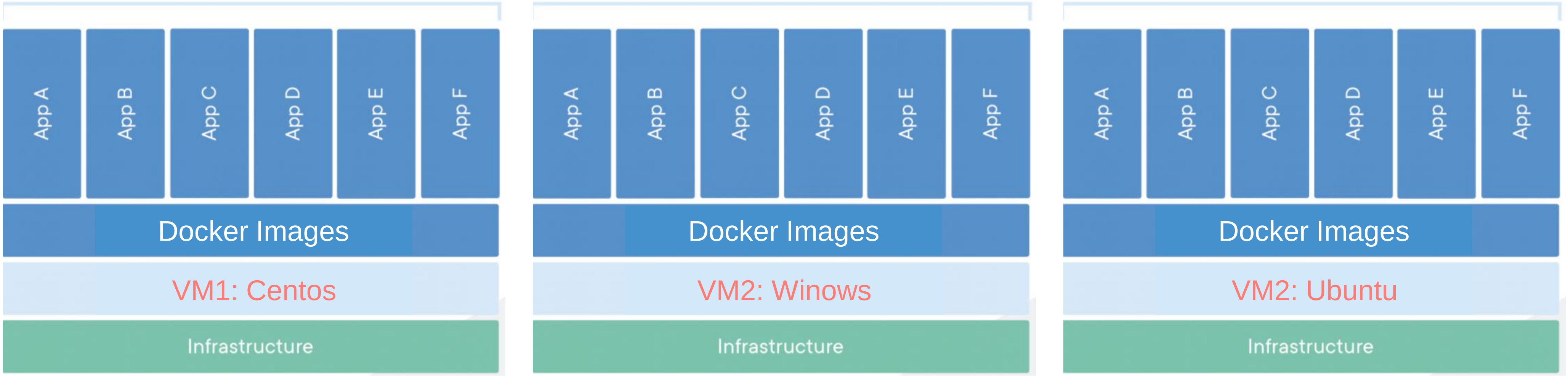
Introduction to Docker

- Open source
- Standardized image format
- Easily package, distribute, and manage applications within containers



Introduction to Docker

- Open source
- Standardized image format
- Easily package, distribute, and manage applications within containers



Management

- Docker Registry
  - docker.io
  - redhat.io
  - gcr.io
- In v7 and before, docker engine is to installed
  - docker command
- in v8, docker engine is built-in the kernel. no need installation
  - podman command
- For backward compatibility
  - alias docker="podman"
- Search for image
  - skopeo command (must install first)
  - or podman search

Search for image

- Explore docker hub
- Using podman

```
# podman search ubuntu
# podman search lamp
```

- Install and use skopeo

```
# dnf –y install skopeo
# skopeo inspect docker://docker.io/library/mysql
# skopeo inspect docker://docker.io/library/python
# skopeo inspect docker://docker.io/library/ubuntu
# skopeo inspect docker://docker.io/library/centos
```

Download image

- Search for any repo and download latest Ubuntu

```
# podman pull ubuntu
```

- Search for specific repo and download latest CentOS

```
# podman pull docker.io/library/centos
# podman pull docker.io/library/nginx
```

- Download latest but specific tag of Ubuntu

```
# podman pull docker.io/library/centos:zesty
```

- Download apache built with Fedora Core libraries only

Manage image

- List all downloaded images

```
# podman images [-a]
```

- Inspect image

```
# podman inspect ubuntu | more
# podman inspect ubuntu:zesty | more
```

- Remove image

```
# podman image rm ubuntu:zesty
```

- Remove all un-used images

```
# podman image rm -a
```

Start container from image

- Boot into ubuntu, view /etc/passwd file, quickly remove upon completion

```
# podman run --rm ubuntu cat /etc/*release
# podman run --rm nginx cat /proc/cpuinfo
```

- Boot into nginx with interactive shell

```
# podman run -it nginx /bin/bash
nginx-id:/# uname –a
nginx-id:/# ls /dev
nginx-id:/# apt list
nginx-id:/# exit
```

Inspect container

- List all running containers

```
# podman ps
or # podman container ls
```

Inspect a container for more information

```
# podman inspect <container-id> | more
```

- List all processes running in the container

```
# podman container top <container-id>
```

- Fetch logs from the container

```
# podman container logs <container-id>
```

Multiple login session into container

- On Terminal 1, start up a container on centos

```
# podman run –it centos /bin/bash
centos-id# cat /etc/*release
centos-id# tty
```

- On Terminal 2, access the same centos container

```
# podman container ls
# podman exec –it <container-id> /bin/bash
centos-id# echo "test" > /dev/pts/0
centos-id# top
```

On both terminal , exit out

Example : Setup a LAMP

- Linux with Apache, MySQL, PHP

- On terminal1, download and run lamp

```
# podman search lamp
# podman pull docker.io/mattrayner/lamp
# podman run --name server1 -d lamp
# podman run --name server2 -d lamp
```

Still with terminal1 verify running lamp servers

```
# podman container ls –a
```

Create database non-interactively

- On Terminal2, Perform some database stuff

```
# podman container ls

# podman logs <container-id>

# podman exec <container-id> mysql –uroot –e "show databases"

# podman exec <container-id> mysql –uroot –e "create database mydb"
```

Create table interactively

- Still with Terminal2, do following

```
# podman exec –it <container-id> /bin/bash
container-id:/# mysql –u root
mysql> show databases;
mysql> use mydb;
mysql> create table users (
• name varchar(50) NULL,
• age varchar(50) NULL,
• live varchar(50) NULL
• );
```

Insert rows into table

- Still with Terminal2, do following

```
mysql> show tables;

mysql> insert into users (name,age,live) values
("ali","35","sunnyvale");

mysql> insert into users (name, age,live)
values("barbara","59","sunnyvale");

mysql> insert into users (name, age,live)
values("nicholas","19","frankfurt");
```

- Now show all rows

```
mysql> select * from users;
```

Log into phpMyAdmin

We need the graphical

```
# systemctl isolate graphical.target
```

Get container's ip

```
# podman inspect <container-id> | grep 10.88
10.88.0.5
```

Launch firefox

Enter url ➔ http://10.88.0.5

Change url ➔ http://10.88.0.5/phpmyadmin

Login as admin with password shown earlier

Checkpoint

1. Why Docker so popular?
   a) It the most widely used open-source operating system in the whole wide world
   b) It has standardized format, industry leading in containerization
   c) It is a multi-billion company that provide cloud technology
   d) Its a leading virtualization technology on par with VMware, Hyper and so on

2. Which statements are accurate? [choose two]
   a) Docker image provides shared content to containers
   b) Container image provides shared content to Docker
   c) Docker is read writable wherelse Container is read-only image
   d) Container is read writable wherelse Docker is read-only image

3. How do you quickly spawn up new container from a downloaded Ubuntu image and get an interactive shell?
   a) docker run –it ubuntu /bin/bash
   b) docker pull docker.io/ubuntu
   c) podman run –it ubuntu /bin/bash
   d) podman pull docker.io/ubuntu

4. True or False: Best thing that developers loved about container is micro-services design

Checkpoint

1. Why Docker so popular?
   a) It the most widely used open-source operating system in the whole wide world
   b) It has standardized format, industry leading in containerization
   c) It is a multi-billion company that provide cloud technology
   d) Its a leading virtualization technology on par with VMware, Hyper and so on

2. Which statements are accurate? [choose two]
   a) Docker image provides shared content to containers
   b) Container image provides shared content to Docker
   c) Docker is read writable wherelse Container is read-only image
   d) Container is read writable wherelse Docker is read-only image

3. How do you quickly spawn up new container from a downloaded Ubuntu image and get an interactive shell?
   a) docker run –it ubuntu /bin/bash
   b) docker pull docker.io/ubuntu
   c) podman run –it ubuntu /bin/bash
   d) podman pull docker.io/ubuntu

4. True or False: Best thing that developers loved about container is micro-services design

Unit summary

Having completed this unit, you should be able to:

- Introduction to Containers
- Introduction to Docker
- Manage Images
- Manage Containers
- Setup real scenario
  - LAMP (Linux → Apache → MySQL → PHP)