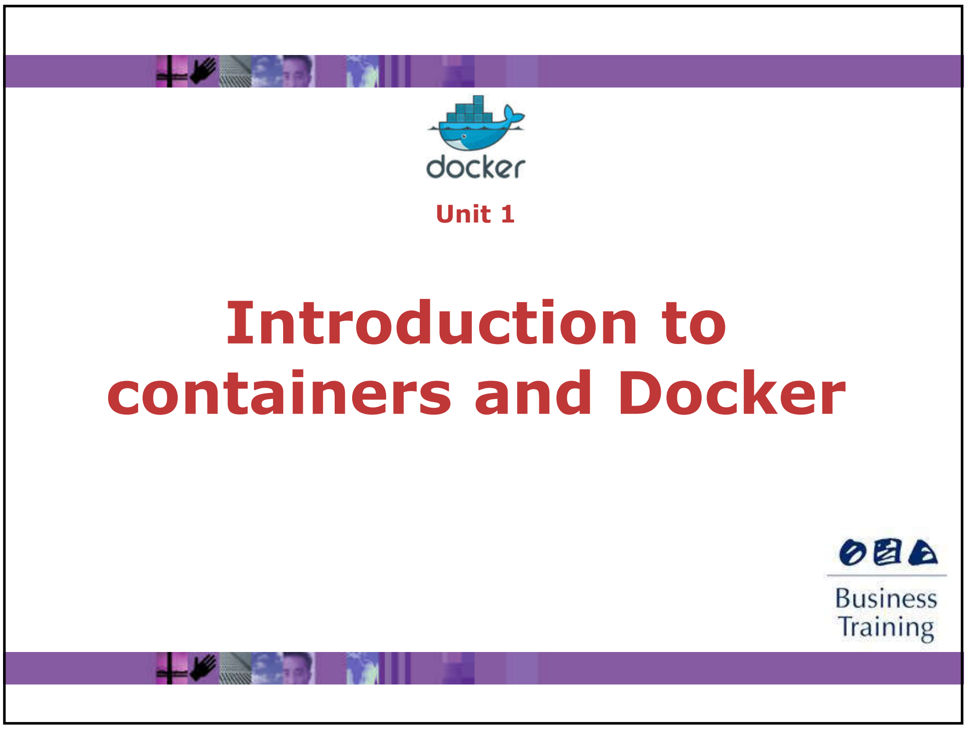


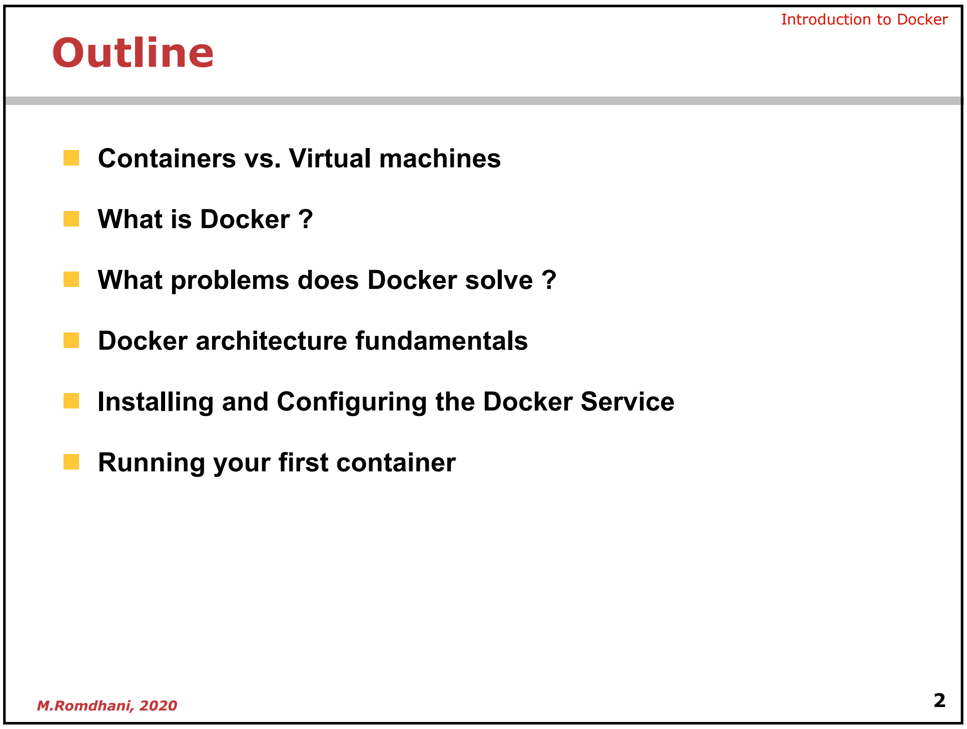
docker

Unit 1

Introduction to containers and Docker

Business Training

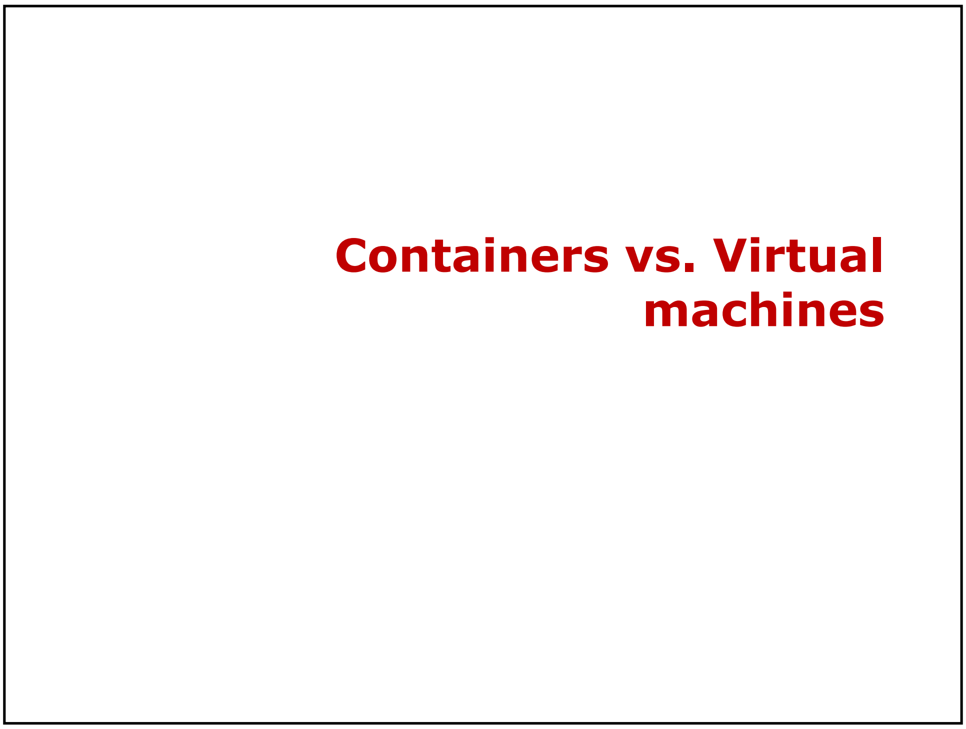
1



Outline

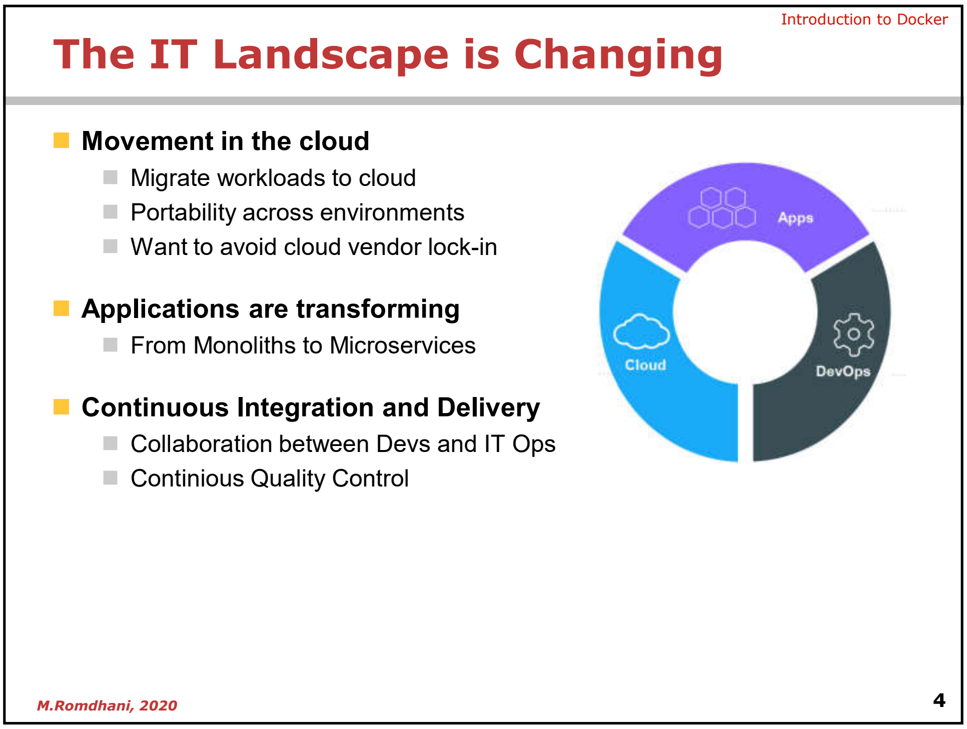
- Containers vs. Virtual machines
- What is Docker ?
- What problems does Docker solve ?
- Docker architecture fundamentals
- Installing and Configuring the Docker Service
- Running your first container

2



Containers vs. Virtual machines

3



Introduction to Docker

The IT Landscape is Changing

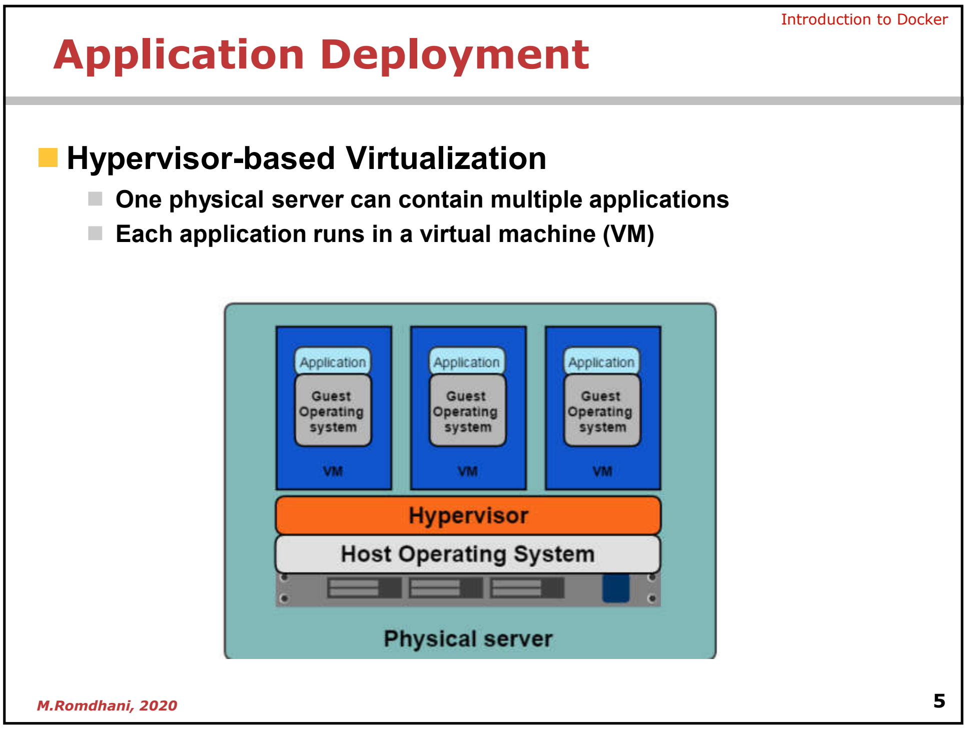
- **Movement in the cloud**
  - Migrate workloads to cloud
  - Portability across environments
  - Want to avoid cloud vendor lock-in
- **Applications are transforming**
  - From Monoliths to Microservices
- **Continuous Integration and Delivery**
  - Collaboration between Devs and IT Ops
  - Continious Quality Control

Apps
Cloud
DevOps

M.Romdhani, 2020

4

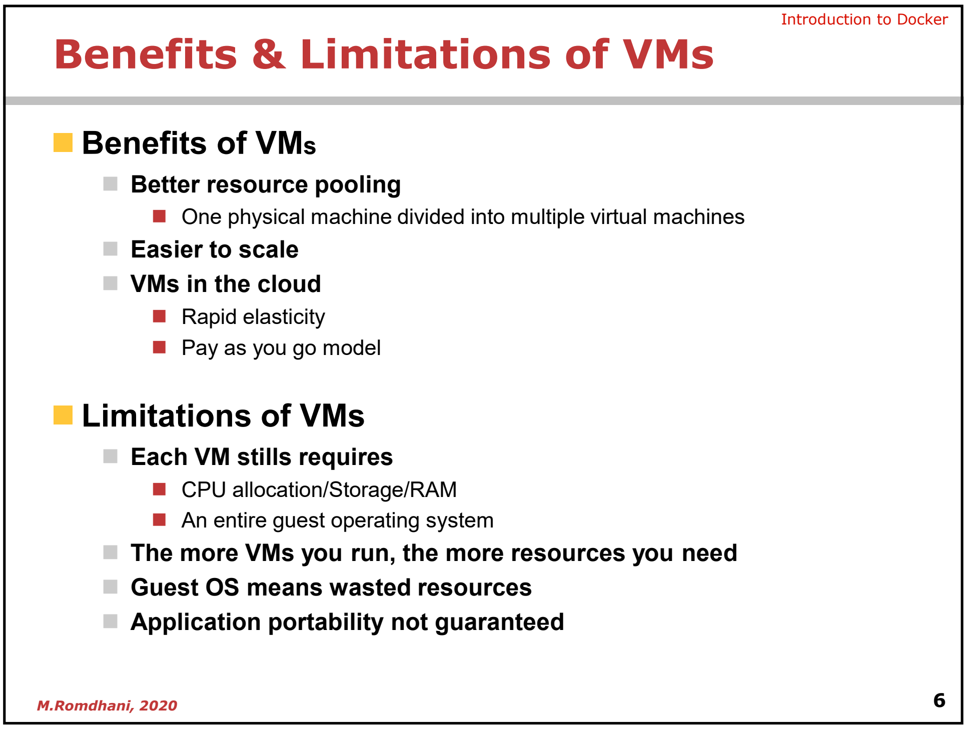
4



Application Deployment

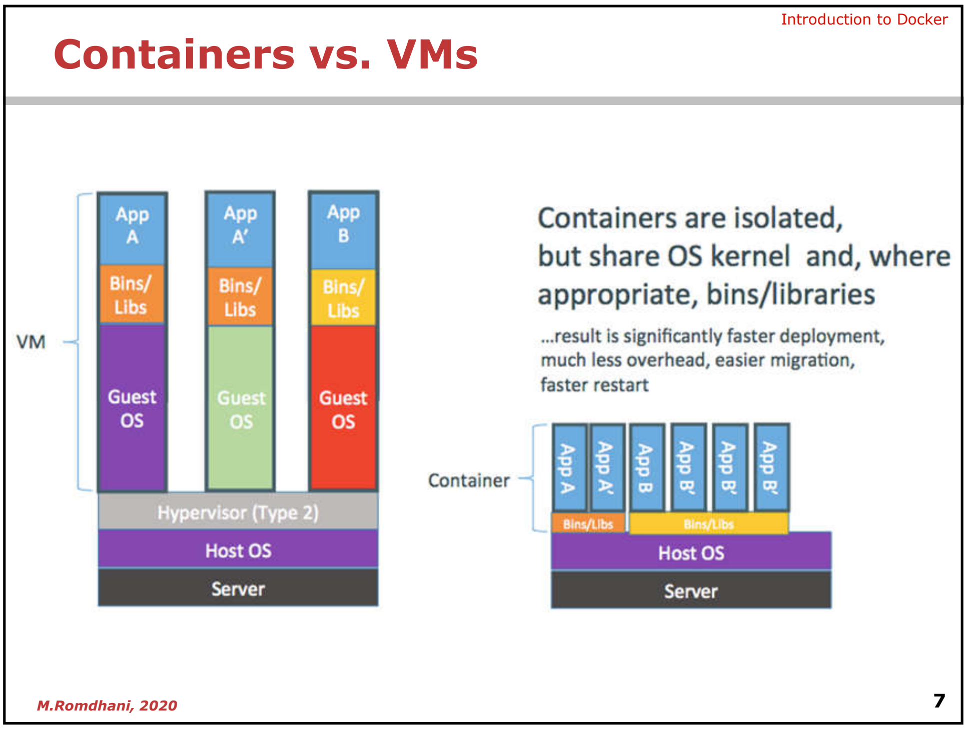
- Hypervisor-based Virtualization
  - One physical server can contain multiple applications
  - Each application runs in a virtual machine (VM)

Application
Guest Operating system
VM
Application
Guest Operating system
VM
Application
Guest Operating system
VM
Hypervisor
Host Operating System
Physical server



Benefits & Limitations of VMs

- Benefits of VMs
  - Better resource pooling
    - One physical machine divided into multiple virtual machines
  - Easier to scale
  - VMs in the cloud
    - Rapid elasticity
    - Pay as you go model
- Limitations of VMs
  - Each VM stills requires
    - CPU allocation/Storage/RAM
    - An entire guest operating system
  - The more VMs you run, the more resources you need
  - Guest OS means wasted resources
  - Application portability not guaranteed



Introduction to Docker

Containers vs. VMs

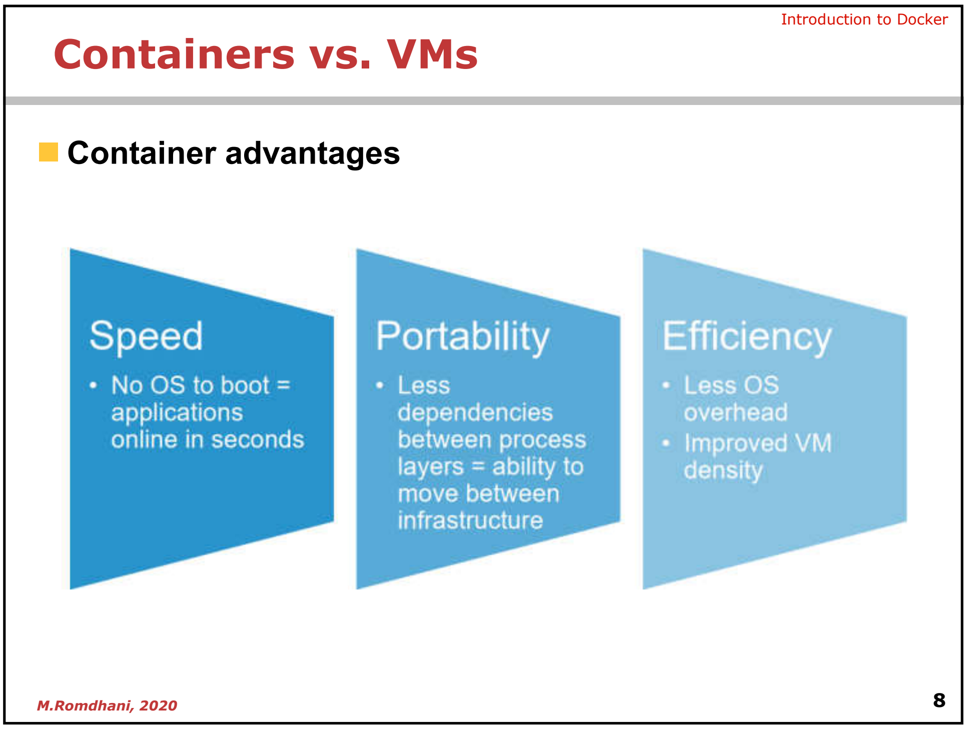
Containers are isolated, but share OS kernel and, where appropriate, bins/libraries

...result is significantly faster deployment, much less overhead, easier migration, faster restart

M.Romdhani, 2020

7

7



Introduction to Docker

Containers vs. VMs

- Container advantages

Speed

- No OS to boot = applications online in seconds

Portability

- Less dependencies between process layers = ability to move between infrastructure

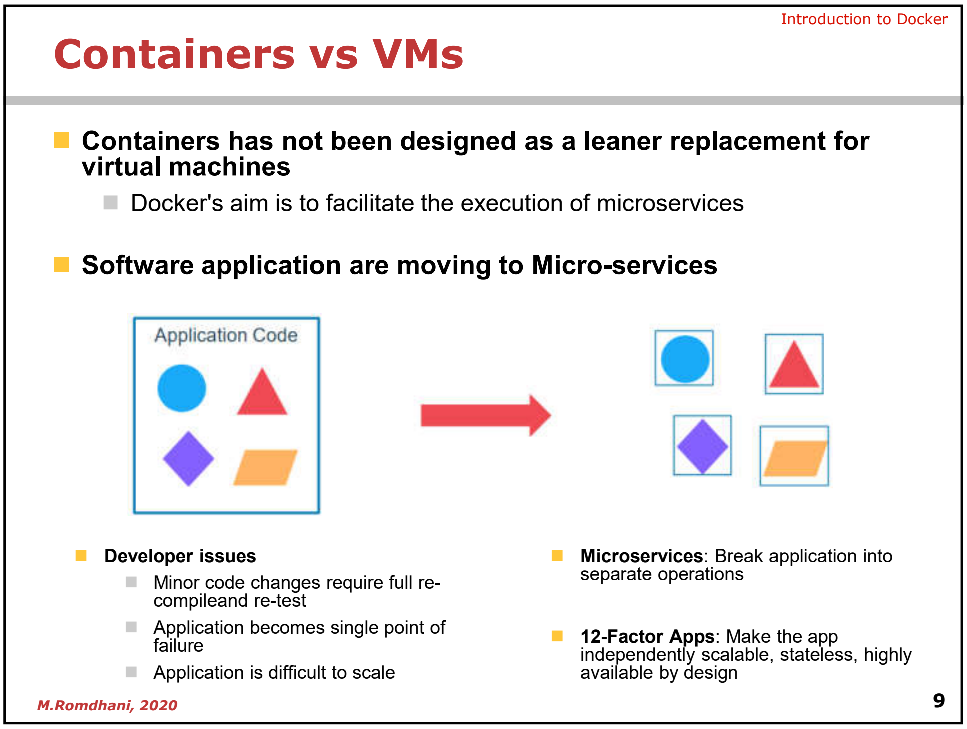
Efficiency

- Less OS overhead
- Improved VM density

M.Romdhani, 2020

8

8



Introduction to Docker

Containers vs VMs

- **Containers has not been designed as a leaner replacement for virtual machines**
  - Docker's aim is to facilitate the execution of microservices
- **Software application are moving to Micro-services**

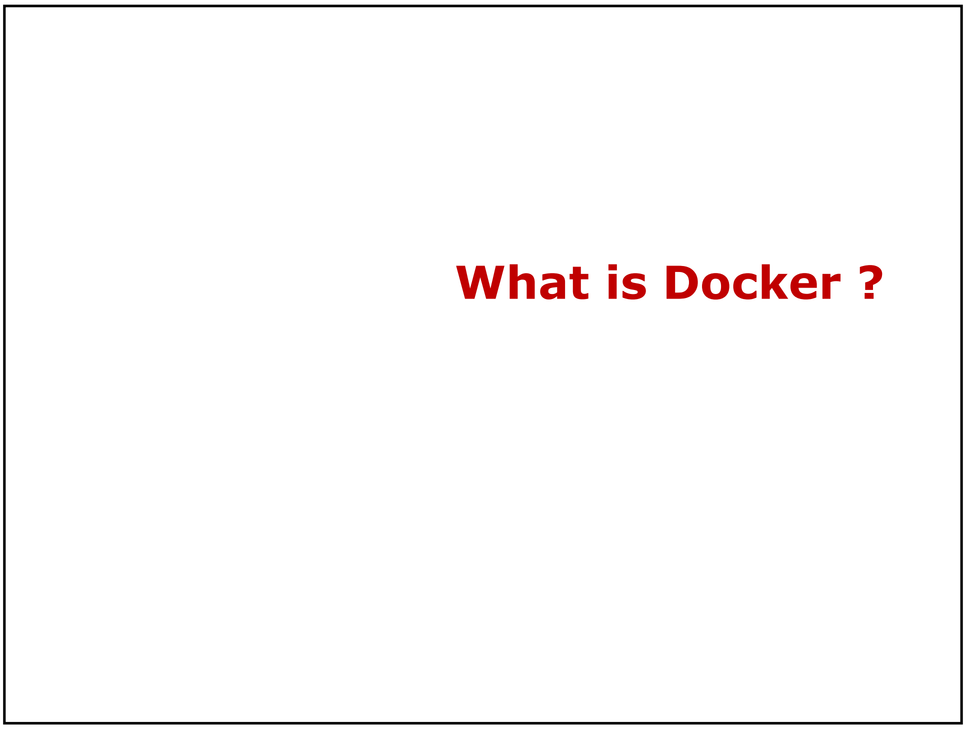
Application Code

- **Developer issues**
  - Minor code changes require full re-compileand re-test
  - Application becomes single point of failure
  - Application is difficult to scale
- **Microservices**: Break application into separate operations
- **12-Factor Apps**: Make the app independently scalable, stateless, highly available by design

M.Romdhani, 2020

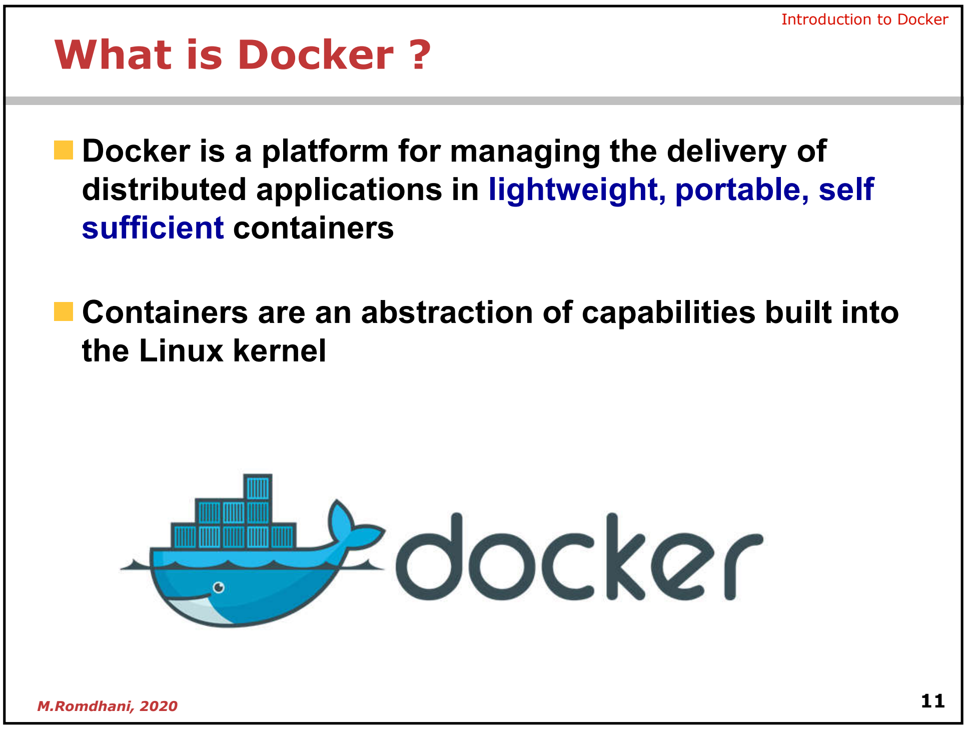
9

9



What is Docker ?

10



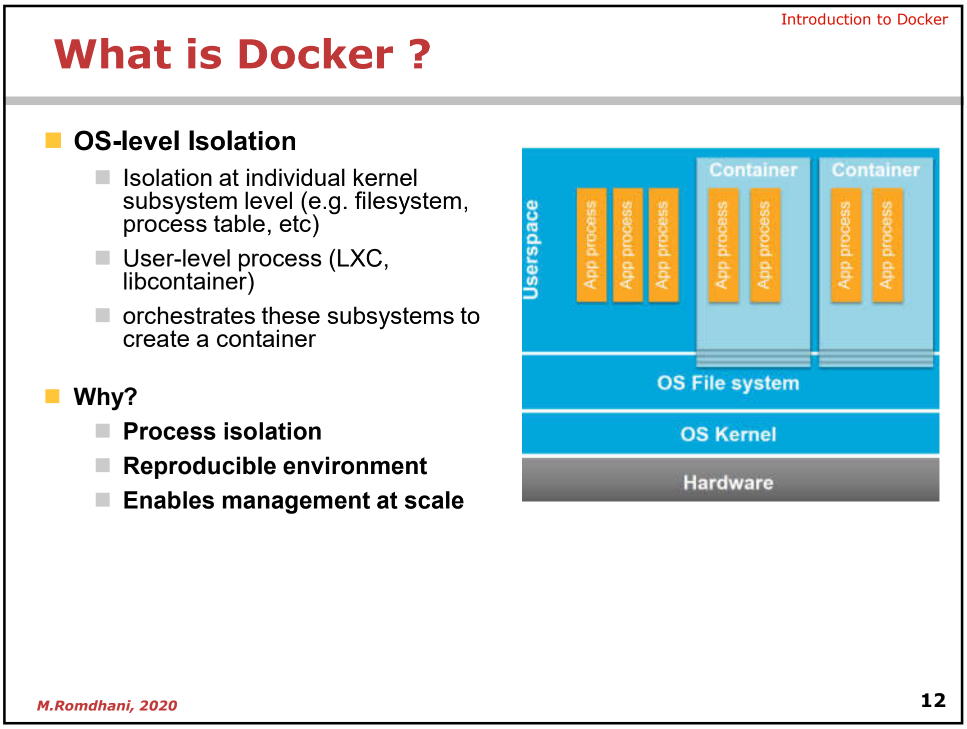
Introduction to Docker

What is Docker ?

- Docker is a platform for managing the delivery of distributed applications in lightweight, portable, self sufficient containers

- Containers are an abstraction of capabilities built into the Linux kernel

M.Romdhani, 2020 11



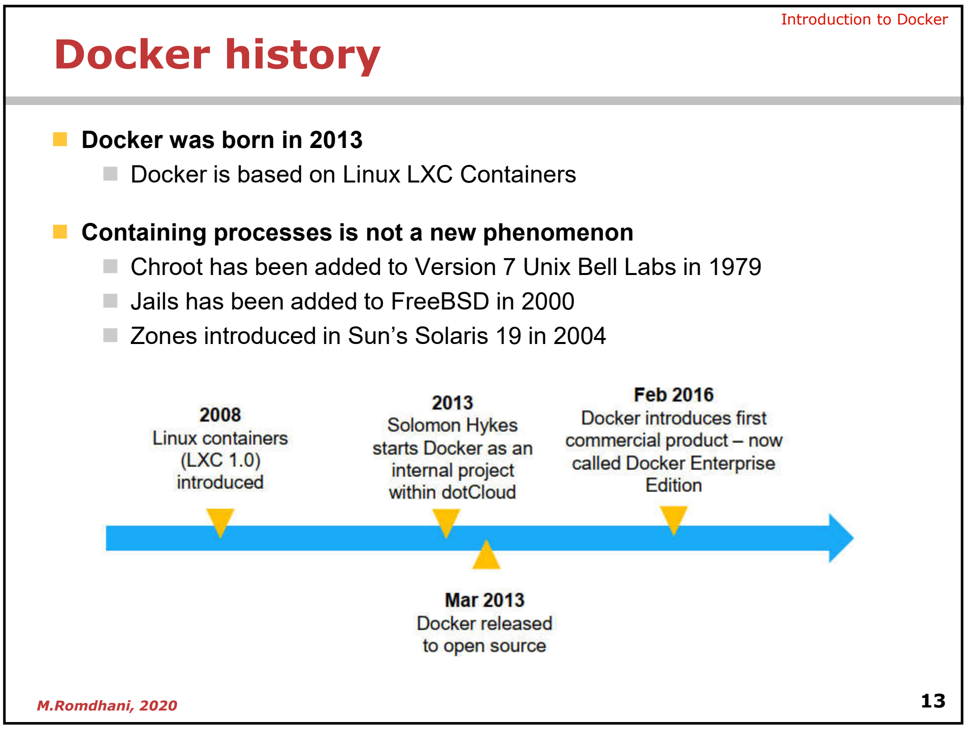
Introduction to Docker

What is Docker ?

- **OS-level Isolation**
  - Isolation at individual kernel subsystem level (e.g. filesystem, process table, etc)
  - User-level process (LXC, libcontainer)
  - orchestrates these subsystems to create a container
- **Why?**
  - **Process isolation**
  - **Reproducible environment**
  - **Enables management at scale**

Userspace
App process
App process
App process
Container
App process
App process
Container
App process
App process
OS File system
OS Kernel
Hardware

M.Romdhani, 2020 12



Introduction to Docker

Docker history

- **Docker was born in 2013**
  - Docker is based on Linux LXC Containers
- **Containing processes is not a new phenomenon**
  - Chroot has been added to Version 7 Unix Bell Labs in 1979
  - Jails has been added to FreeBSD in 2000
  - Zones introduced in Sun's Solaris 19 in 2004

2008
Linux containers
(LXC 1.0)
introduced

2013
Solomon Hykes
starts Docker as an
internal project
within dotCloud

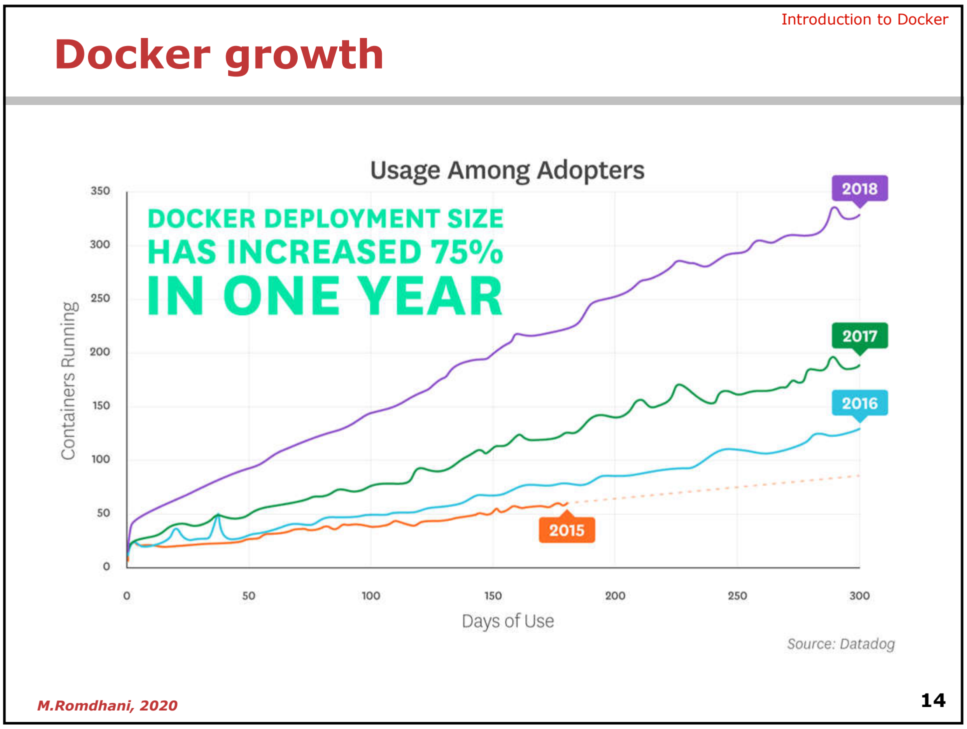
Feb 2016
Docker introduces first
commercial product – now
called Docker Enterprise
Edition

Mar 2013
Docker released
to open source

M.Romdhani, 2020

13

13



Introduction to Docker

Docker growth

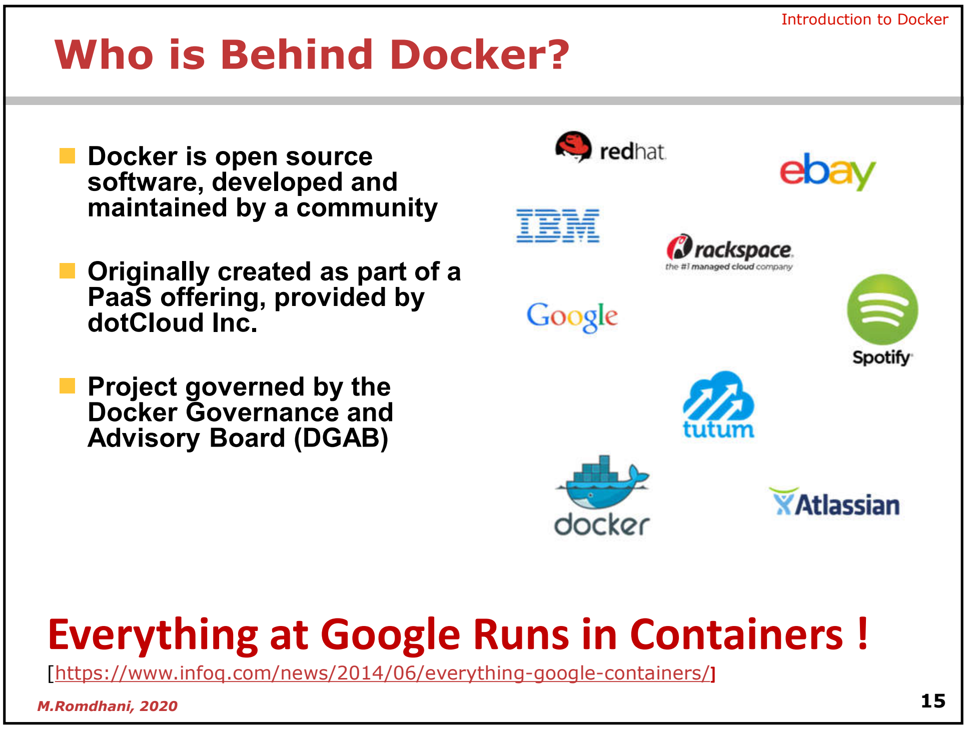
Usage Among Adopters

Source: Datadog

M.Romdhani, 2020

14

14



Who is Behind Docker?

- Docker is open source software, developed and maintained by a community
- Originally created as part of a PaaS offering, provided by dotCloud Inc.
- Project governed by the Docker Governance and Advisory Board (DGAB)

redhat
ebay
IBM
rackspace
the #1 managed cloud company
Google
Spotify
tutum
docker
Atlassian

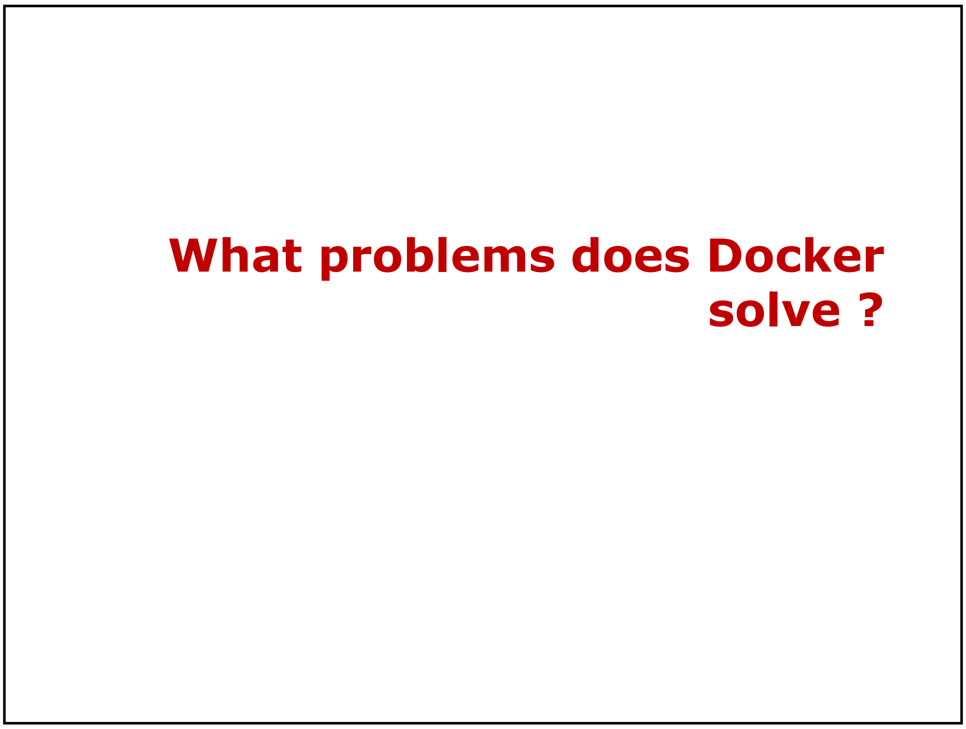
Everything at Google Runs in Containers !

[https://www.infoq.com/news/2014/06/everything-google-containers/]

M.Romdhani, 2020

15

15



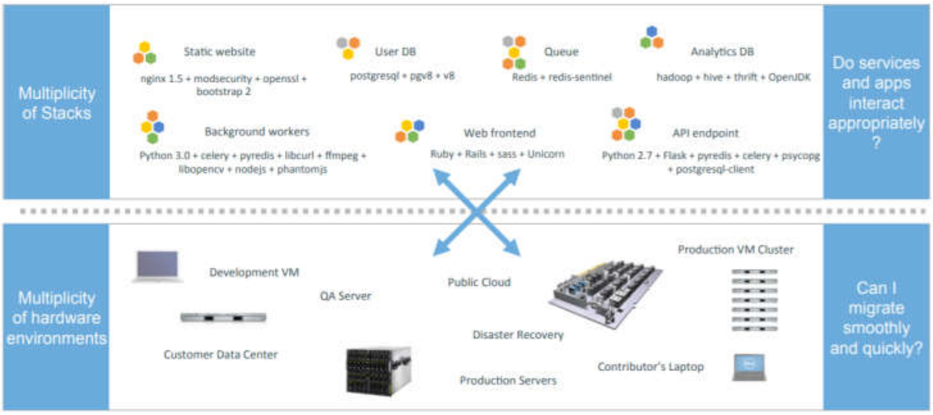
What problems does Docker solve ?

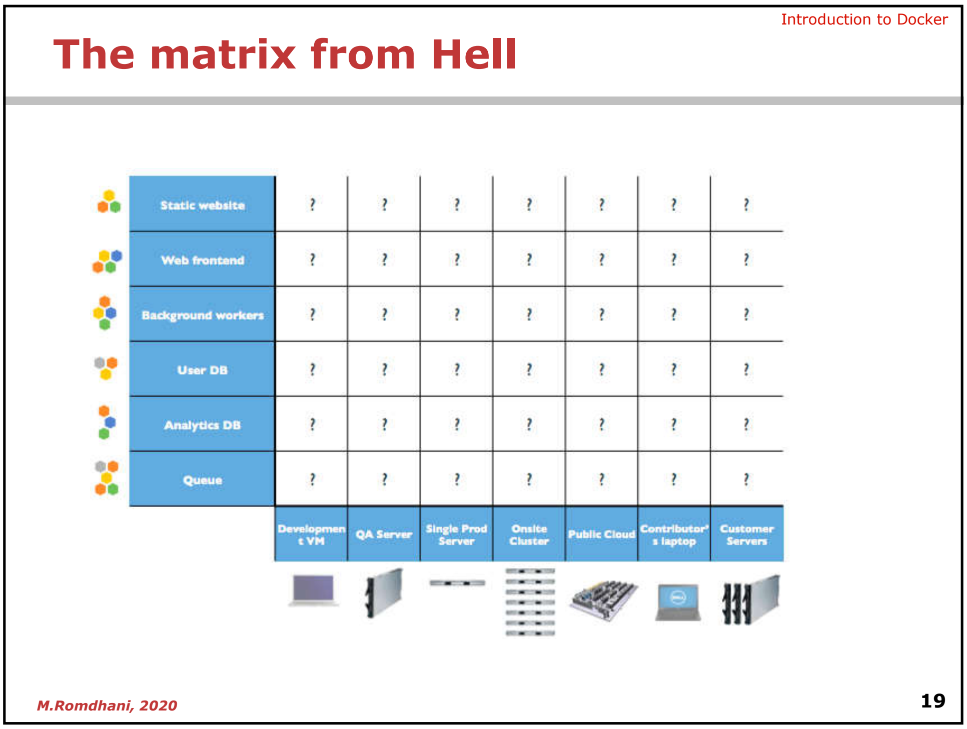
16

Docker tackles these problems

- **Deployment problems (aka Dependency matrix Hell)**
  - Applications have direct dependencies. Each of these dependencies has their own dependencies, and so on.
- **“Works on my machine”**
  - Says your coworker (IT Ops), as you struggle to deploy the latest code from the source repository.
- **Application maturity**
  - As the application matures, and you upgrade dependencies, you might need to recheck compatibilities with underlying OS infrastructure.
- **Integration Challenges**
  - As you migrate the application to different environments, you have to be aware of other Line-of-Business applications running on the target host, and resolve those one-at-a-time in each environment as part of every migration.

The Dependency matrix


Multiplicity of Stacks
Static website
nginx 1.5 + modsecurity + openssl + bootstrap 2
User DB
postgresql + pgv8 + v8
Queue
Redis + redis-sentinel
Analytics DB
hadoop + hive + thrift + OpenJDK
Background workers
Python 3.0 + celery + pyredis + libcurl + ffmpeg + libopencv + nodejs + phantomjs
Web frontend
Ruby + Rails + sass + Unicorn
API endpoint
Python 2.7 + Flask + pyredis + celery + psycopg + postgresql-client
Do services and apps interact appropriately ?
Multiplicity of hardware environments
Development VM
QA Server
Public Cloud
Production VM Cluster
Customer Data Center
Disaster Recovery
Contributor's Laptop
Production Servers
Can I migrate smoothly and quickly?



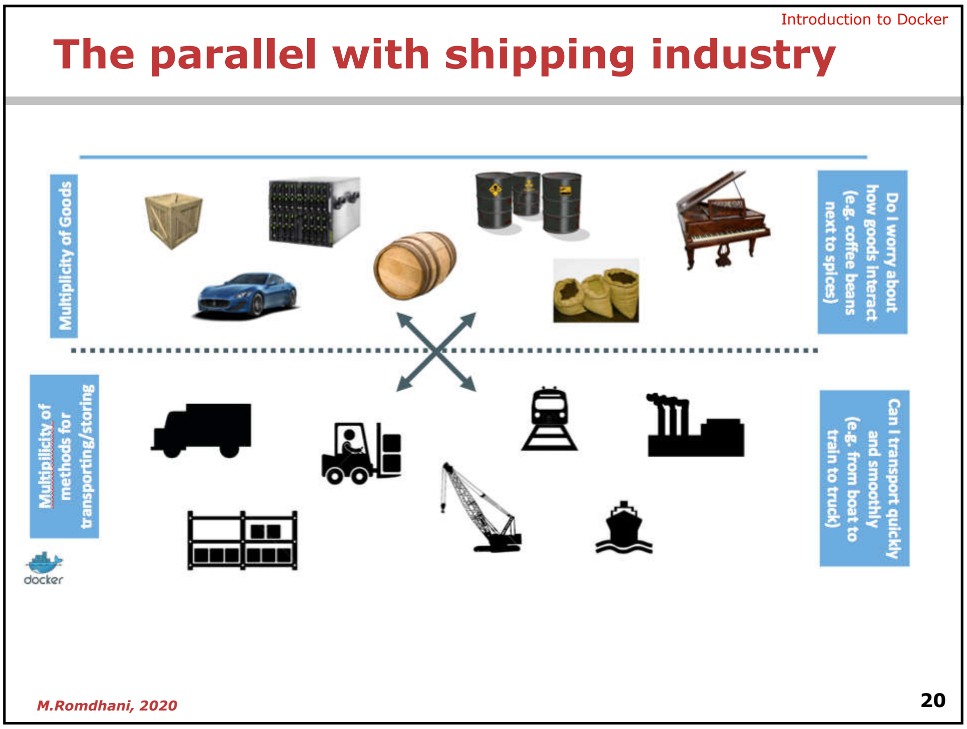
Introduction to Docker

The matrix from Hell

	Development VM	QA Server	Single Prod Server	Onsite Cluster	Public Cloud	Contributor's laptop	Customer Servers
Static website	?	?	?	?	?	?	?
Web frontend	?	?	?	?	?	?	?
Background workers	?	?	?	?	?	?	?
User DB	?	?	?	?	?	?	?
Analytics DB	?	?	?	?	?	?	?
Queue	?	?	?	?	?	?	?

M.Romdhani, 2020 19

19



Introduction to Docker

The parallel with shipping industry

Multiplicity of Goods

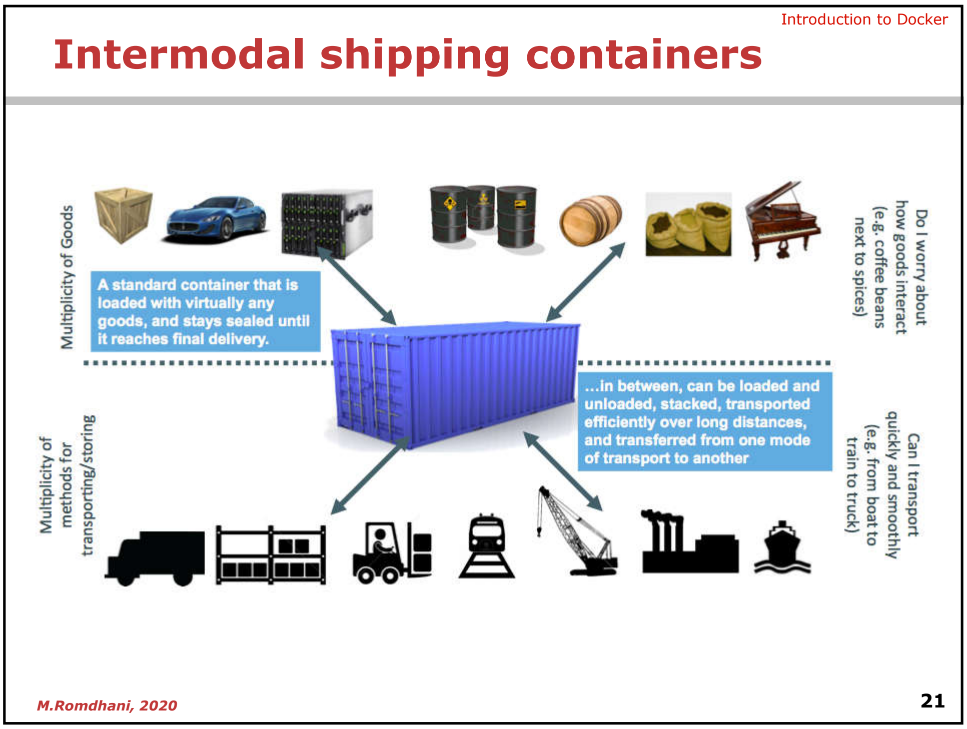
Do I worry about how goods interact (e.g. coffee beans next to spices)

Multiplicity of methods for transporting/storing

Can I transport quickly and smoothly (e.g. from boat to train to truck)

M.Romdhani, 2020 20

20



Introduction to Docker

Intermodal shipping containers

Multiplicity of Goods

A standard container that is loaded with virtually any goods, and stays sealed until it reaches final delivery.

Do I worry about how goods interact (e.g. coffee beans next to spices)

Multiplicity of methods for transporting/storing

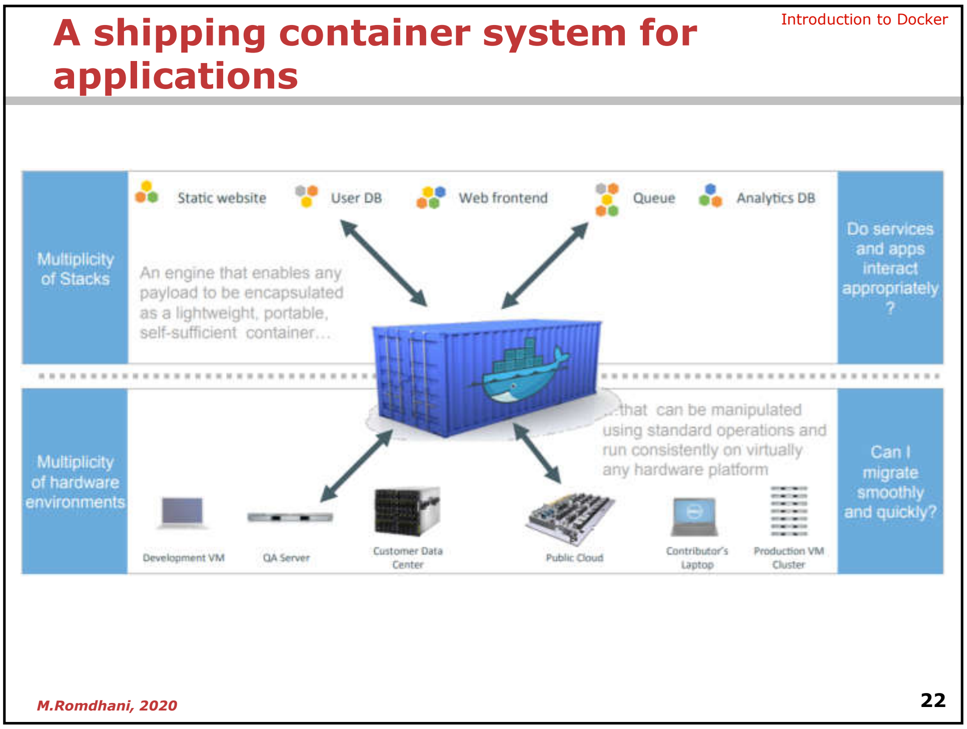
…in between, can be loaded and unloaded, stacked, transported efficiently over long distances, and transferred from one mode of transport to another

Can I transport quickly and smoothly (e.g. from boat to train to truck)

M.Romdhani, 2020

21

21



Introduction to Docker

A shipping container system for applications

Multiplicity of Stacks

Static website

User DB

Web frontend

Queue

Analytics DB

An engine that enables any payload to be encapsulated as a lightweight, portable, self-sufficient container…

Do services and apps interact appropriately ?

Multiplicity of hardware environments

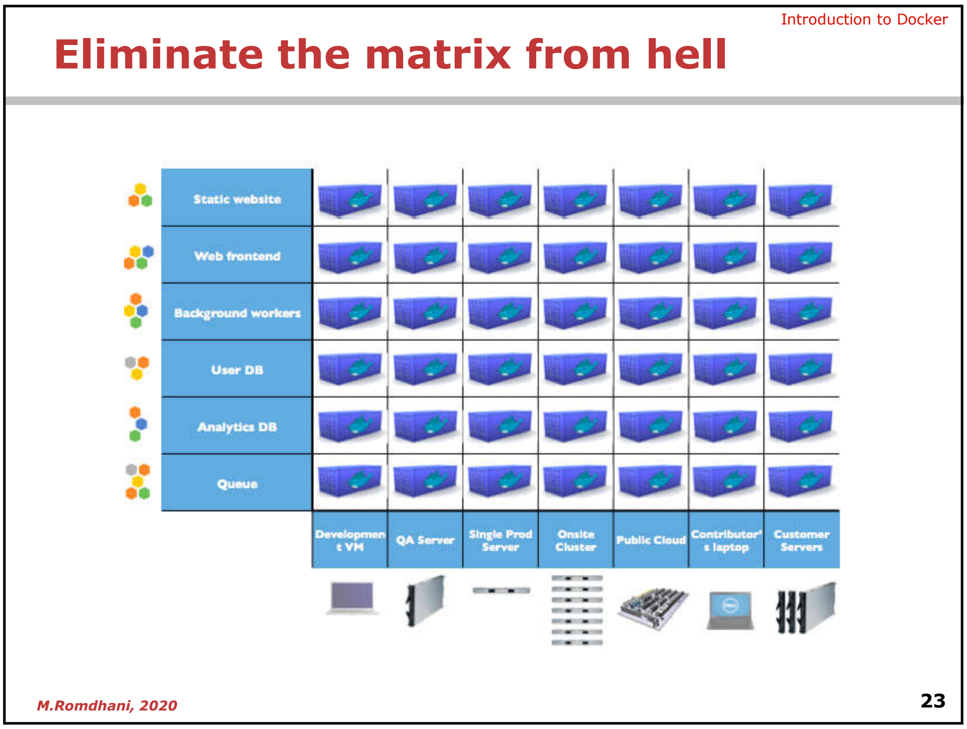
…that can be manipulated using standard operations and run consistently on virtually any hardware platform

Can I migrate smoothly and quickly?

M.Romdhani, 2020

22

22



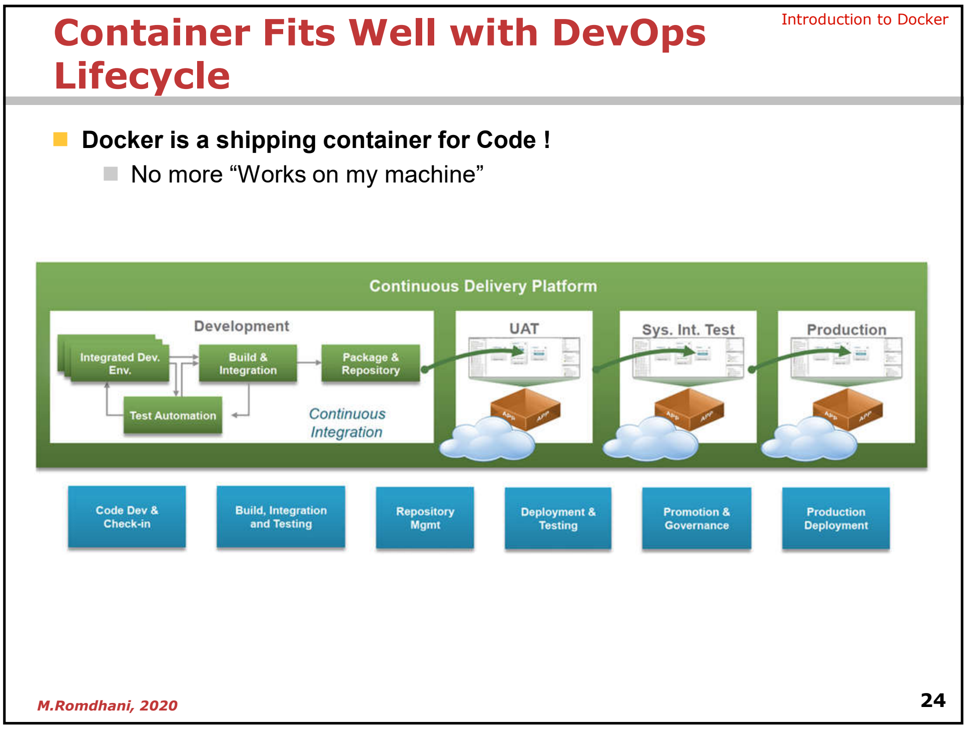
Introduction to Docker

Eliminate the matrix from hell

M.Romdhani, 2020

23

23



Introduction to Docker

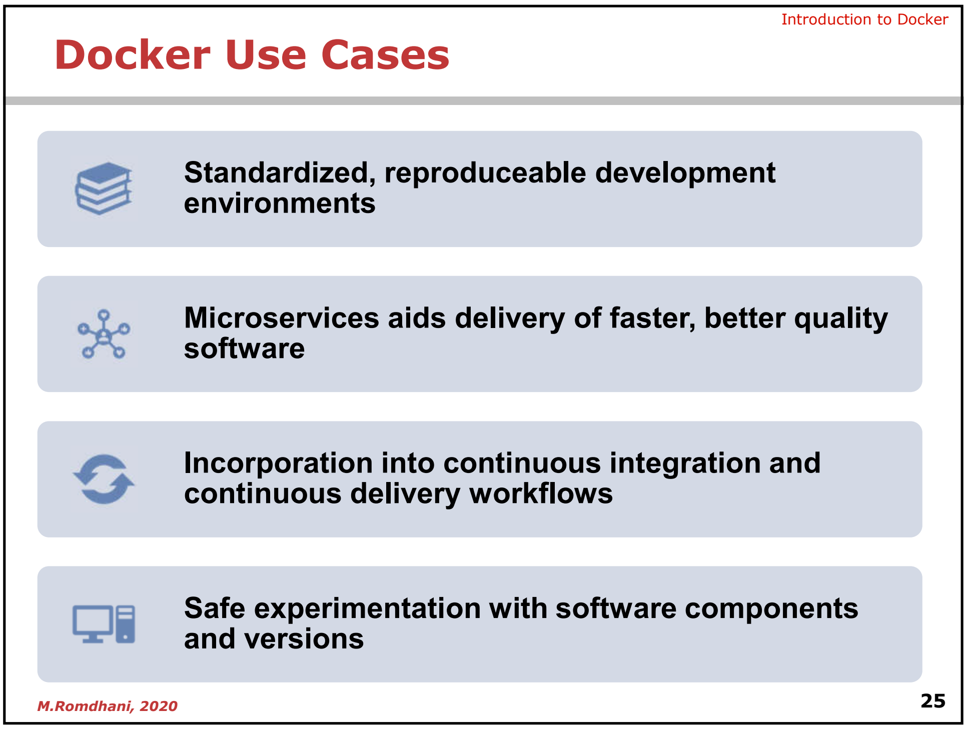
Container Fits Well with DevOps Lifecycle

- Docker is a shipping container for Code !
  - No more "Works on my machine"

M.Romdhani, 2020

24

24



Introduction to Docker

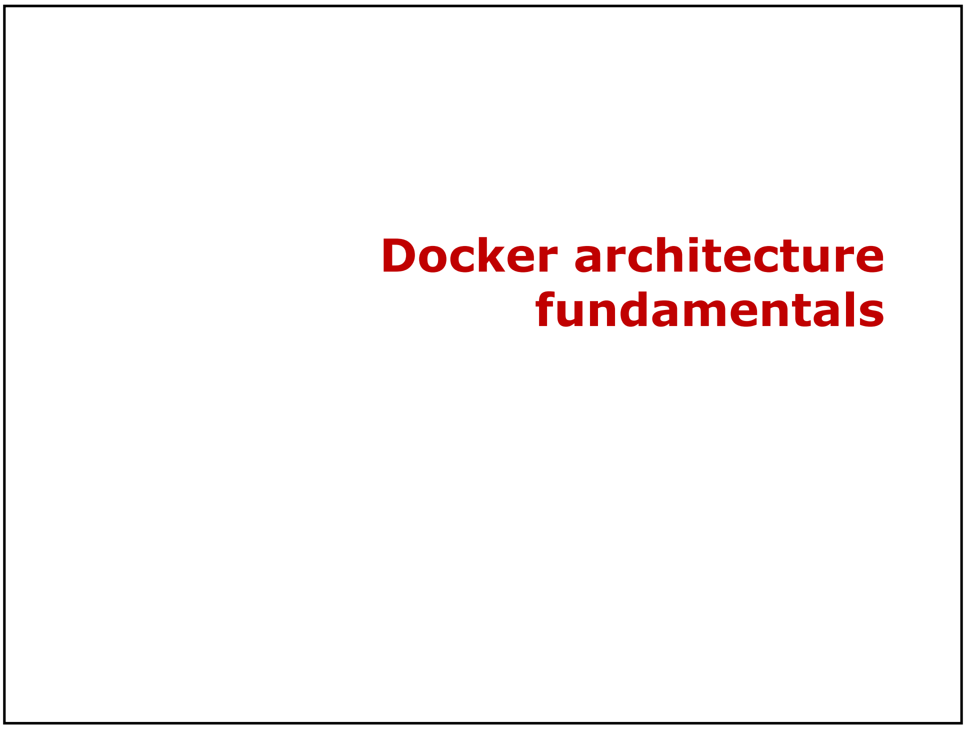
Docker Use Cases

- Standardized, reproduceable development environments
- Microservices aids delivery of faster, better quality software
- Incorporation into continuous integration and continuous delivery workflows
- Safe experimentation with software components and versions

M.Romdhani, 2020

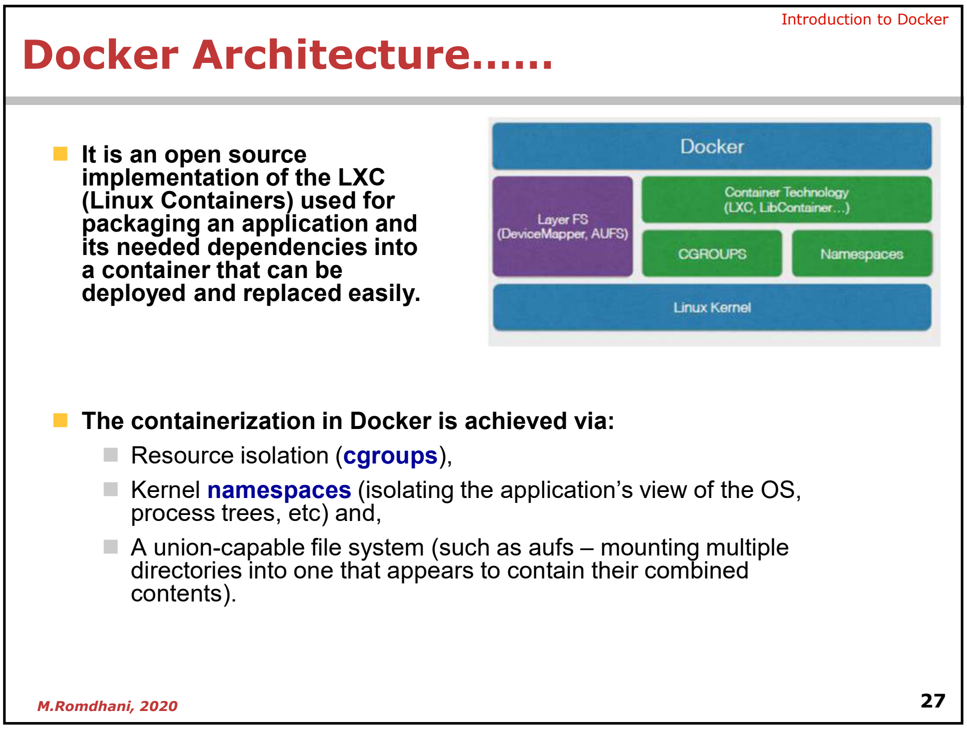
25

25



Docker architecture fundamentals

26

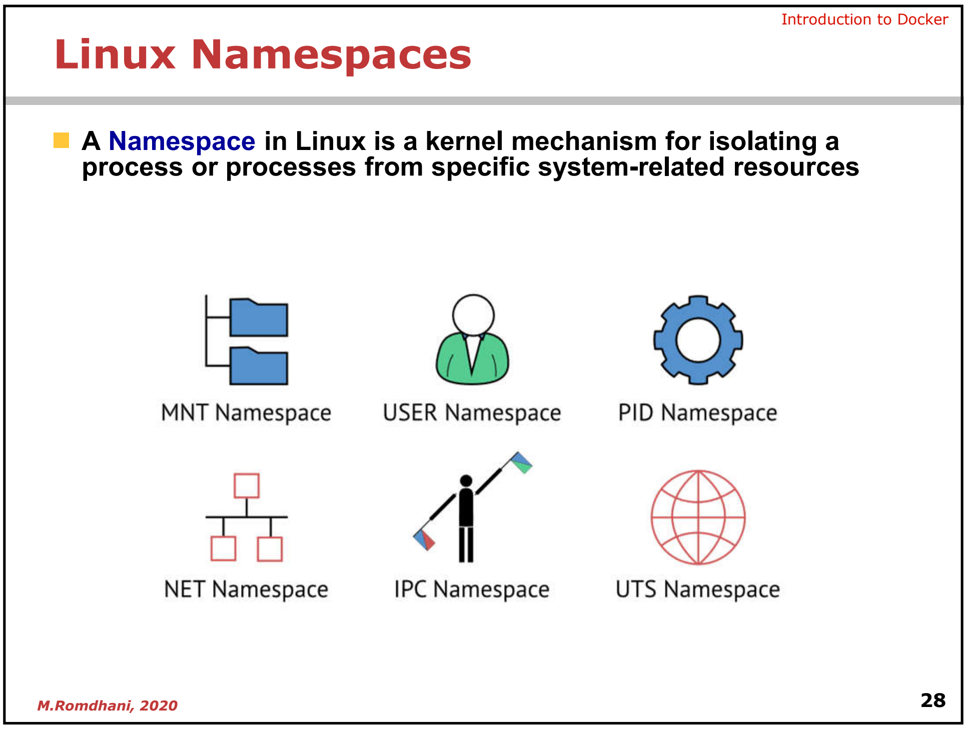


Docker Architecture……

- It is an open source implementation of the LXC (Linux Containers) used for packaging an application and its needed dependencies into a container that can be deployed and replaced easily.

Docker
Layer FS (DeviceMapper, AUFS)
Container Technology (LXC, LibContainer…)
CGROUPS
Namespaces
Linux Kernel

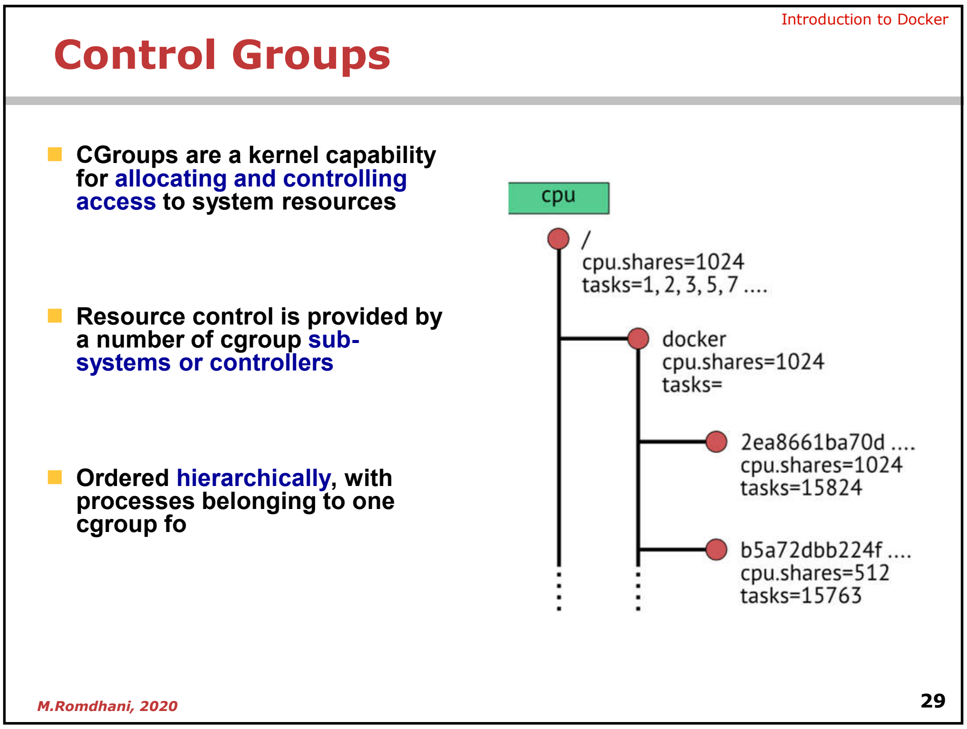
- The containerization in Docker is achieved via:
  - Resource isolation (**cgroups**),
  - Kernel **namespaces** (isolating the application's view of the OS, process trees, etc) and,
  - A union-capable file system (such as aufs – mounting multiple directories into one that appears to contain their combined contents).



Linux Namespaces

- A **Namespace** in Linux is a kernel mechanism for isolating a process or processes from specific system-related resources

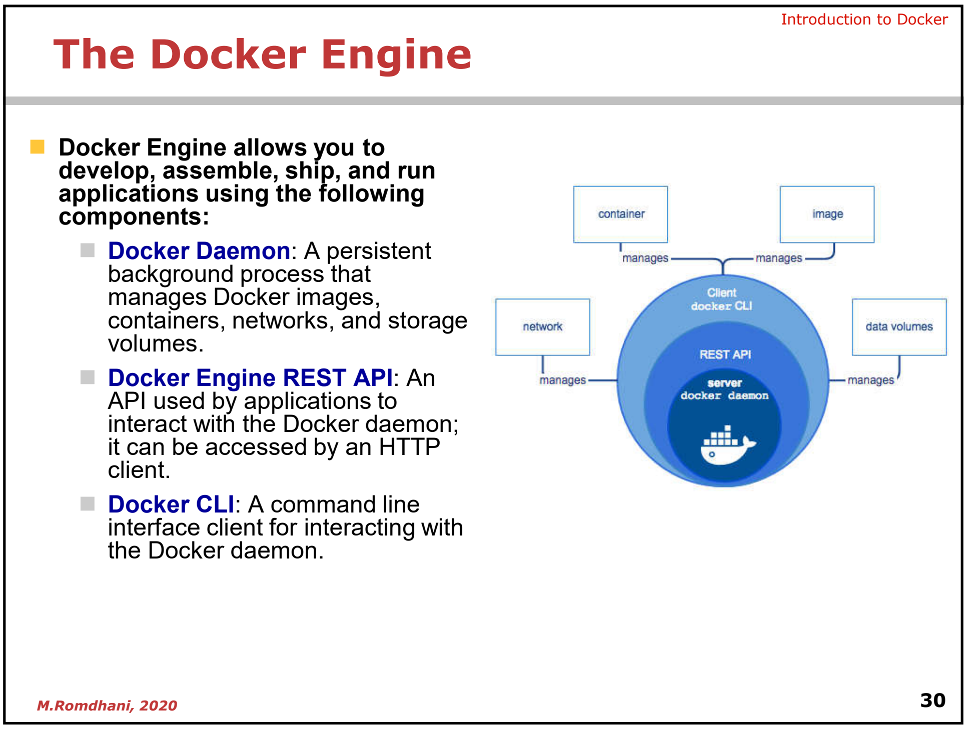
MNT Namespace
USER Namespace
PID Namespace
NET Namespace
IPC Namespace
UTS Namespace



Control Groups

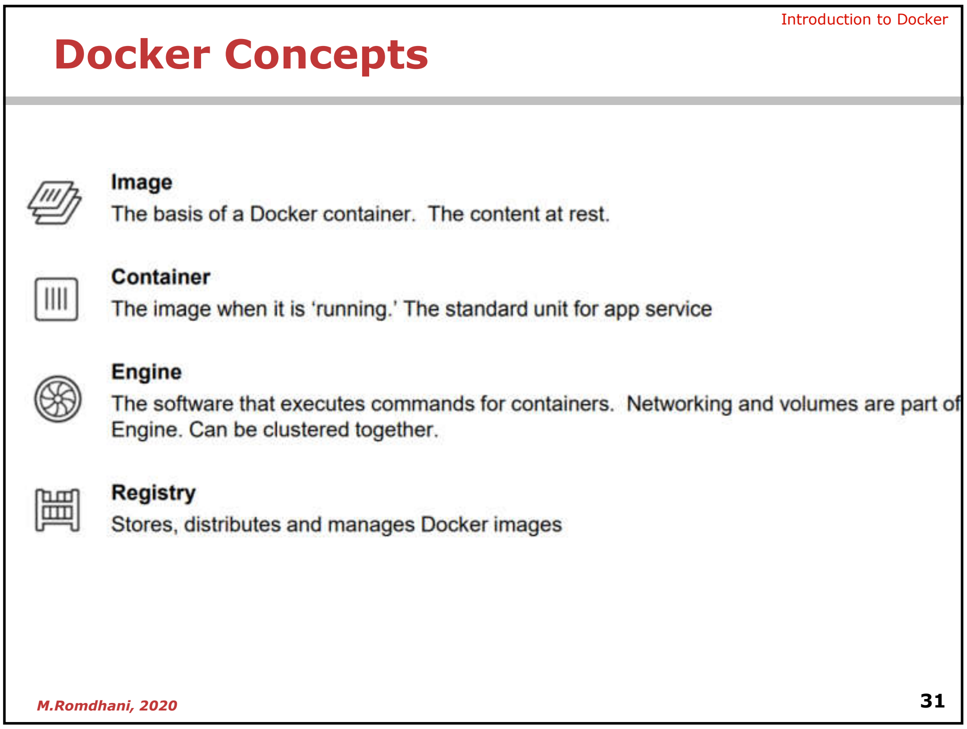
- CGroups are a kernel capability for **allocating and controlling access** to system resources
- Resource control is provided by a number of cgroup **sub-systems or controllers**
- Ordered **hierarchically**, with processes belonging to one cgroup fo

cpu
/
cpu.shares=1024
tasks=1, 2, 3, 5, 7 ....
docker
cpu.shares=1024
tasks=
2ea8661ba70d ....
cpu.shares=1024
tasks=15824
b5a72dbb224f ....
cpu.shares=512
tasks=15763



The Docker Engine

- Docker Engine allows you to develop, assemble, ship, and run applications using the following components:
  - **Docker Daemon**: A persistent background process that manages Docker images, containers, networks, and storage volumes.
  - **Docker Engine REST API**: An API used by applications to interact with the Docker daemon; it can be accessed by an HTTP client.
  - **Docker CLI**: A command line interface client for interacting with the Docker daemon.



Introduction to Docker

Docker Concepts

Image
The basis of a Docker container. The content at rest.

Container
The image when it is 'running.' The standard unit for app service

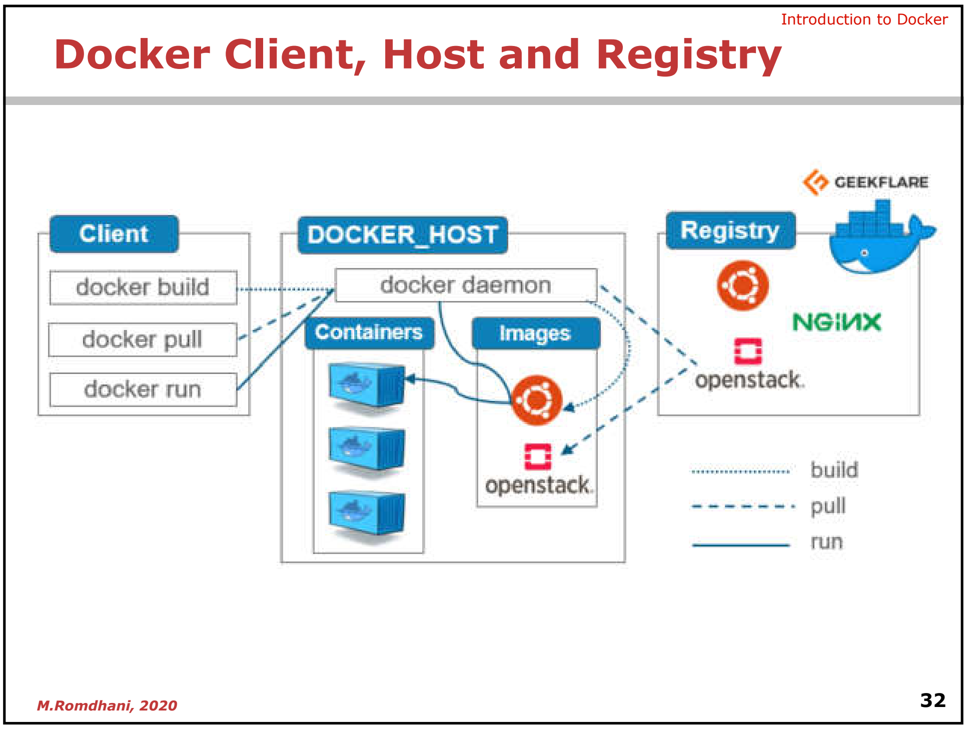
Engine
The software that executes commands for containers. Networking and volumes are part of Engine. Can be clustered together.

Registry
Stores, distributes and manages Docker images

M.Romdhani, 2020

31

31



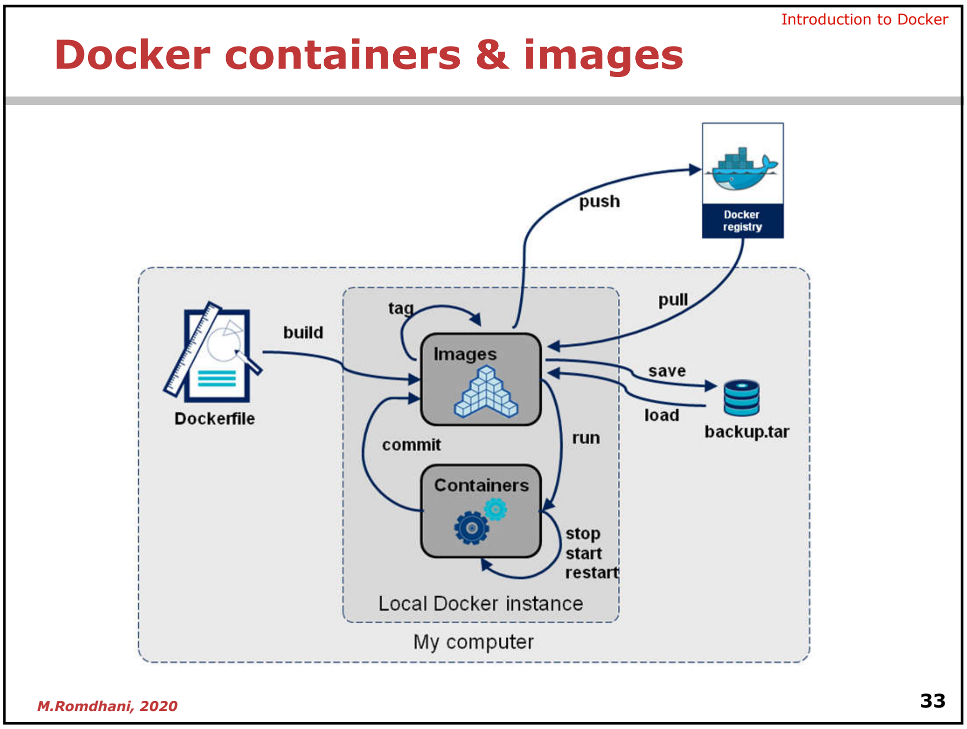
Introduction to Docker

Docker Client, Host and Registry

M.Romdhani, 2020

32

32



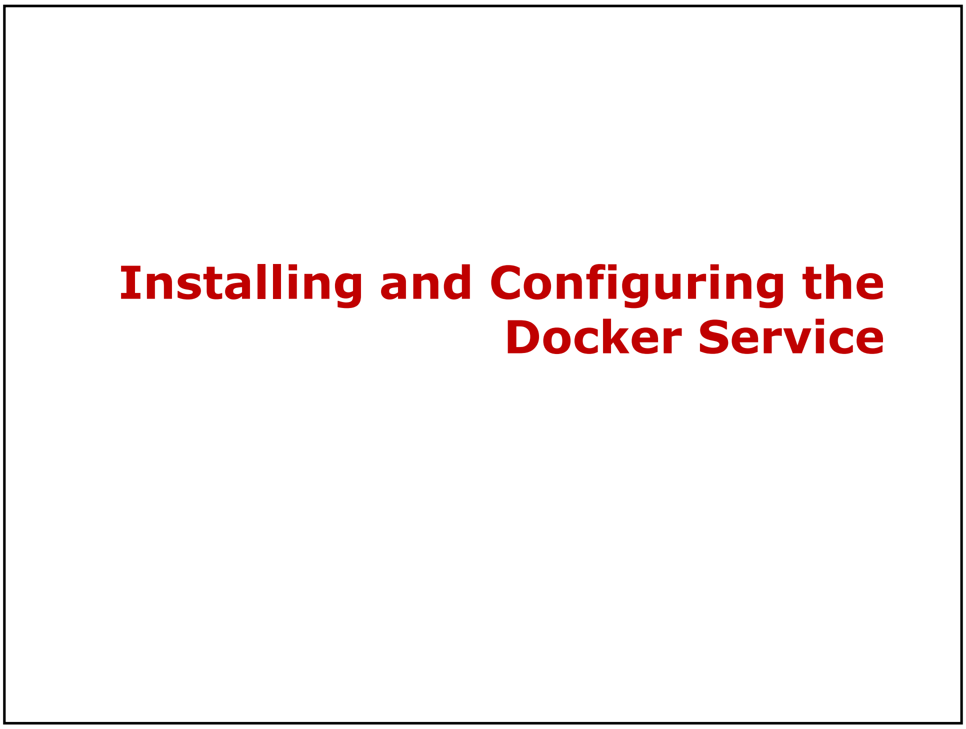
Introduction to Docker

Docker containers & images

M.Romdhani, 2020

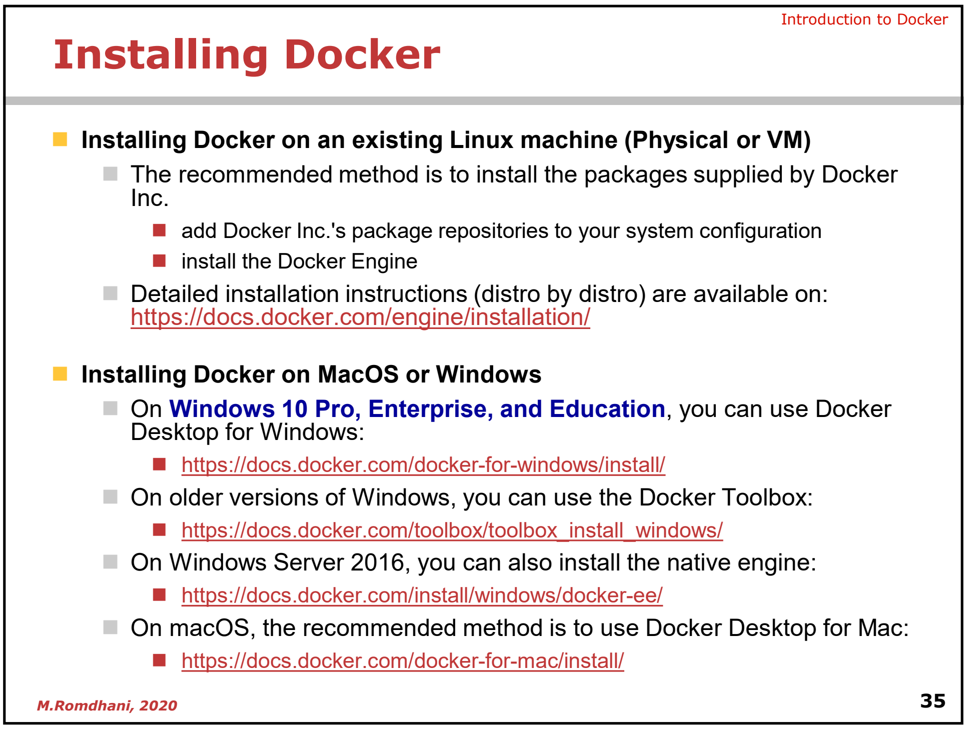
33

33



Installing and Configuring the Docker Service

34



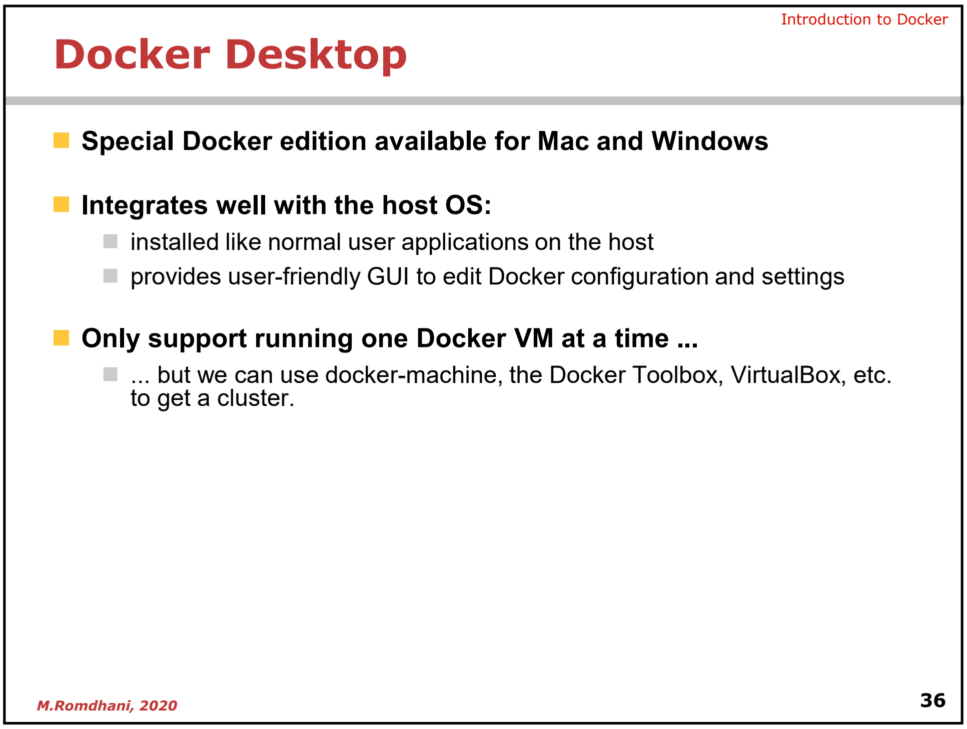
Introduction to Docker

Installing Docker

- **Installing Docker on an existing Linux machine (Physical or VM)**
  - The recommended method is to install the packages supplied by Docker Inc.
    - add Docker Inc.'s package repositories to your system configuration
    - install the Docker Engine
  - Detailed installation instructions (distro by distro) are available on: https://docs.docker.com/engine/installation/
- **Installing Docker on MacOS or Windows**
  - On **Windows 10 Pro, Enterprise, and Education**, you can use Docker Desktop for Windows:
    - https://docs.docker.com/docker-for-windows/install/
  - On older versions of Windows, you can use the Docker Toolbox:
    - https://docs.docker.com/toolbox/toolbox_install_windows/
  - On Windows Server 2016, you can also install the native engine:
    - https://docs.docker.com/install/windows/docker-ee/
  - On macOS, the recommended method is to use Docker Desktop for Mac:
    - https://docs.docker.com/docker-for-mac/install/

M.Romdhani, 2020 **35**

35



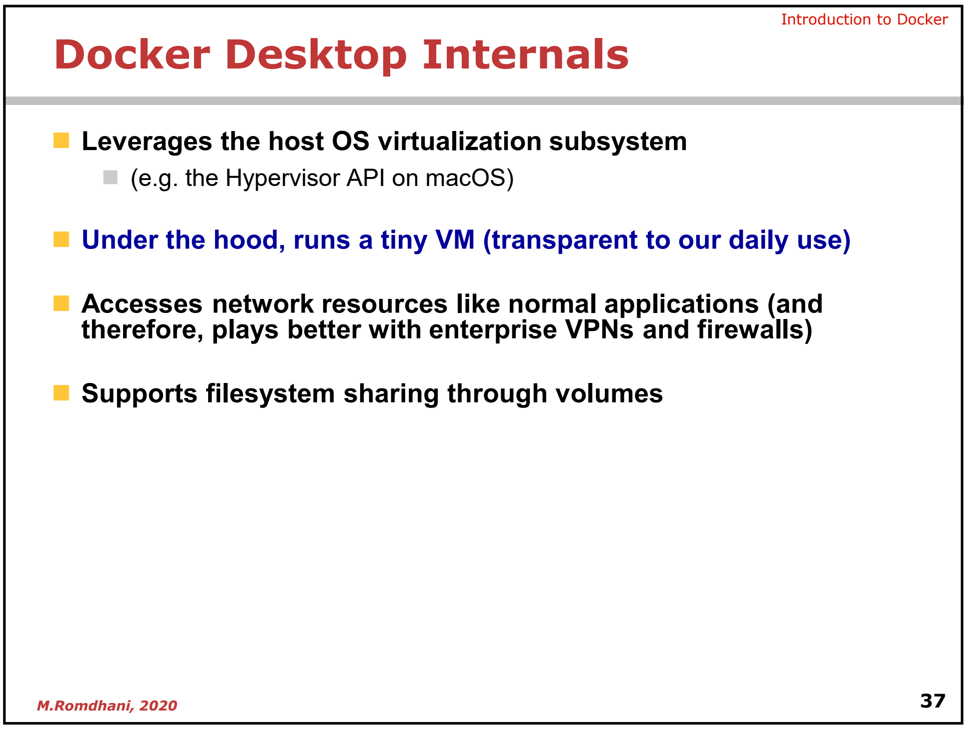
Introduction to Docker

Docker Desktop

- **Special Docker edition available for Mac and Windows**
- **Integrates well with the host OS:**
  - installed like normal user applications on the host
  - provides user-friendly GUI to edit Docker configuration and settings
- **Only support running one Docker VM at a time ...**
  - ... but we can use docker-machine, the Docker Toolbox, VirtualBox, etc. to get a cluster.

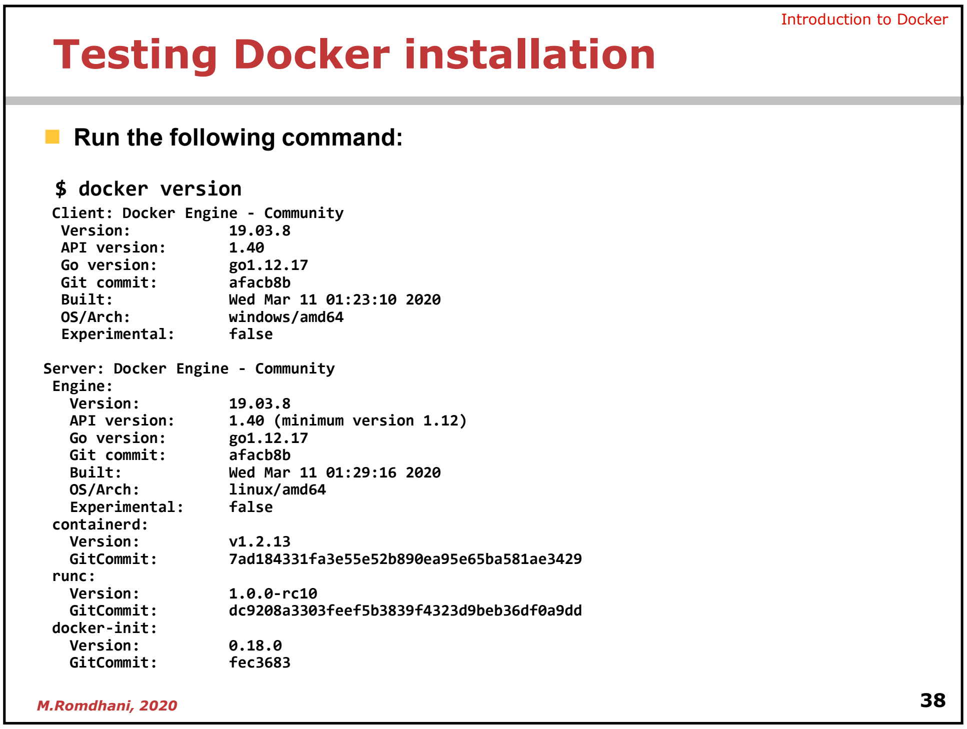
M.Romdhani, 2020 **36**

36



Docker Desktop Internals

- Leverages the host OS virtualization subsystem
  - (e.g. the Hypervisor API on macOS)
- **Under the hood, runs a tiny VM (transparent to our daily use)**
- Accesses network resources like normal applications (and therefore, plays better with enterprise VPNs and firewalls)
- Supports filesystem sharing through volumes



Testing Docker installation

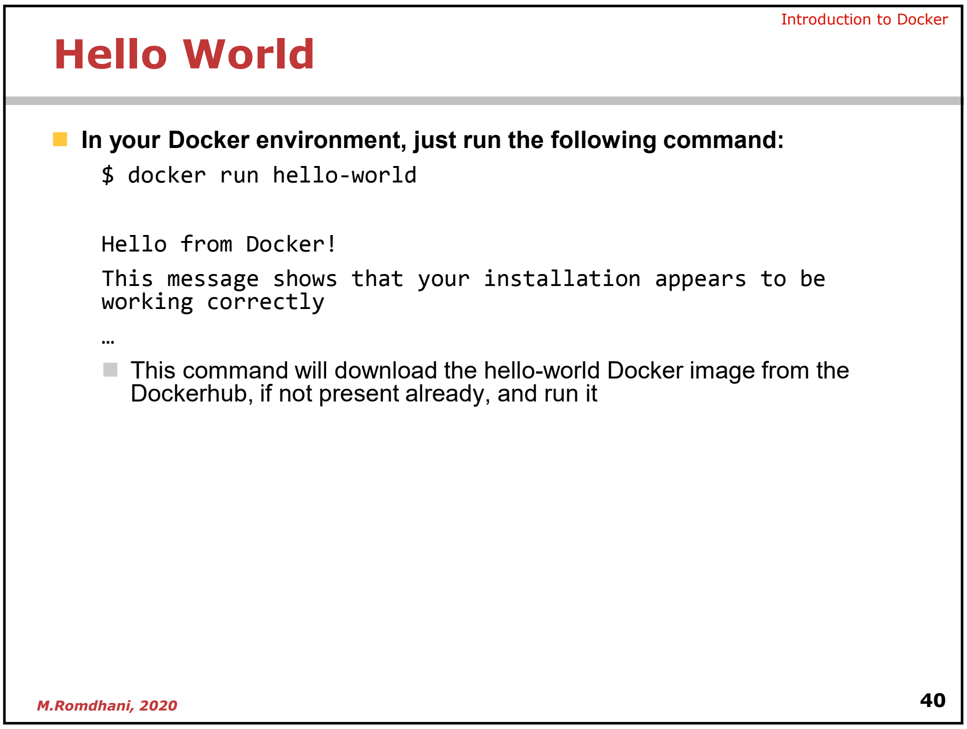
- Run the following command:

```
$ docker version
Client: Docker Engine - Community
 Version:           19.03.8
 API version:       1.40
 Go version:        go1.12.17
 Git commit:        afacb8b
 Built:             Wed Mar 11 01:23:10 2020
 OS/Arch:           windows/amd64
 Experimental:      false

Server: Docker Engine - Community
 Engine:
  Version:          19.03.8
  API version:      1.40 (minimum version 1.12)
  Go version:       go1.12.17
  Git commit:       afacb8b
  Built:            Wed Mar 11 01:29:16 2020
  OS/Arch:          linux/amd64
  Experimental:     false
 containerd:
  Version:          v1.2.13
  GitCommit:        7ad184331fa3e55e52b890ea95e65ba581ae3429
 runc:
  Version:          1.0.0-rc10
  GitCommit:        dc9208a3303feef5b3839f4323d9beb36df0a9dd
 docker-init:
  Version:          0.18.0
  GitCommit:        fec3683
```

Running your first container

39



Hello World

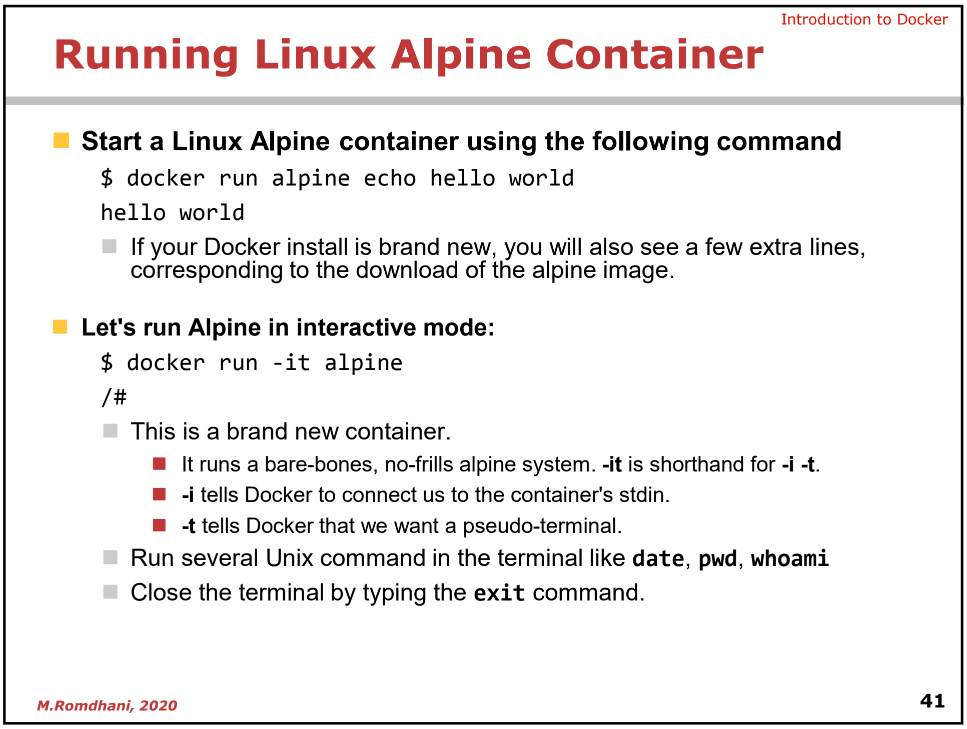
- **In your Docker environment, just run the following command:**

```
$ docker run hello-world

Hello from Docker!
This message shows that your installation appears to be
working correctly
…
```

  - This command will download the hello-world Docker image from the Dockerhub, if not present already, and run it

40



Introduction to Docker

Running Linux Alpine Container

- **Start a Linux Alpine container using the following command**

```
$ docker run alpine echo hello world
hello world
```

  - If your Docker install is brand new, you will also see a few extra lines, corresponding to the download of the alpine image.

- **Let's run Alpine in interactive mode:**

```
$ docker run -it alpine
/#
```

  - This is a brand new container.
    - It runs a bare-bones, no-frills alpine system. **-it** is shorthand for **-i -t**.
    - **-i** tells Docker to connect us to the container's stdin.
    - **-t** tells Docker that we want a pseudo-terminal.
  - Run several Unix command in the terminal like **`date`**, **`pwd`**, **`whoami`**
  - Close the terminal by typing the **`exit`** command.

M.Romdhani, 2020

41