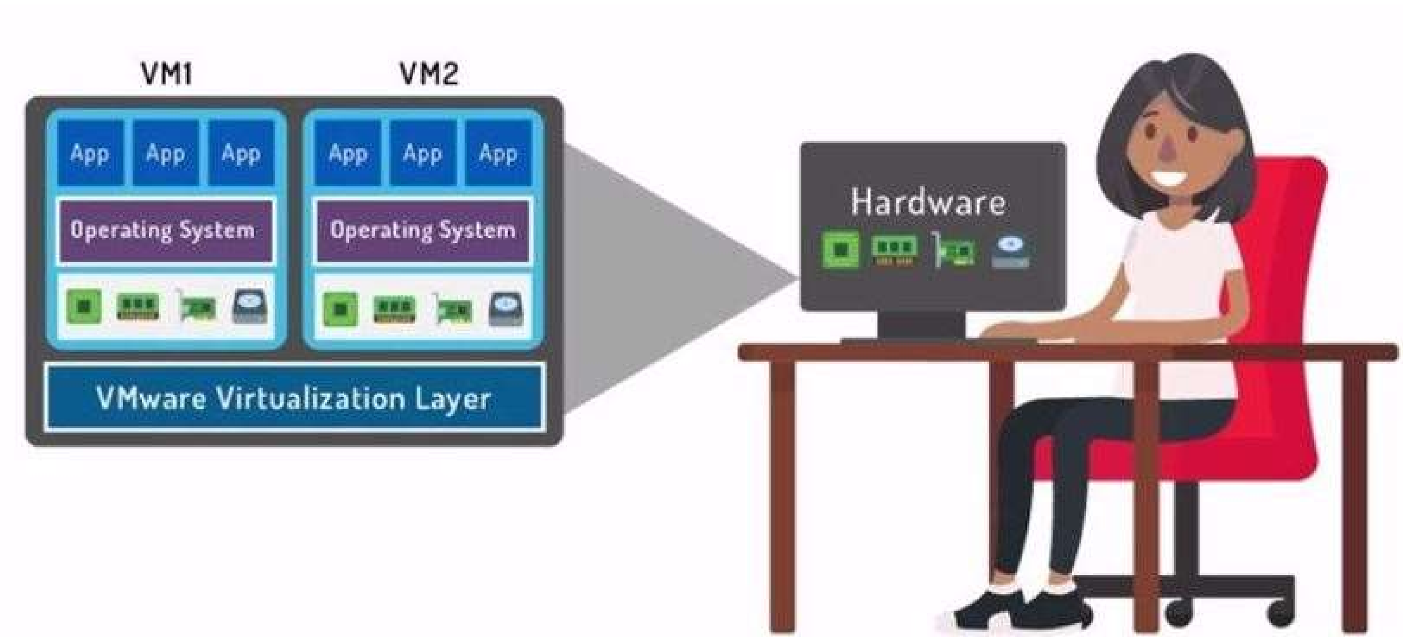
Virtualization

Dr Mukti Padhya
Assistant Professor
NFSU, Gandhinagar

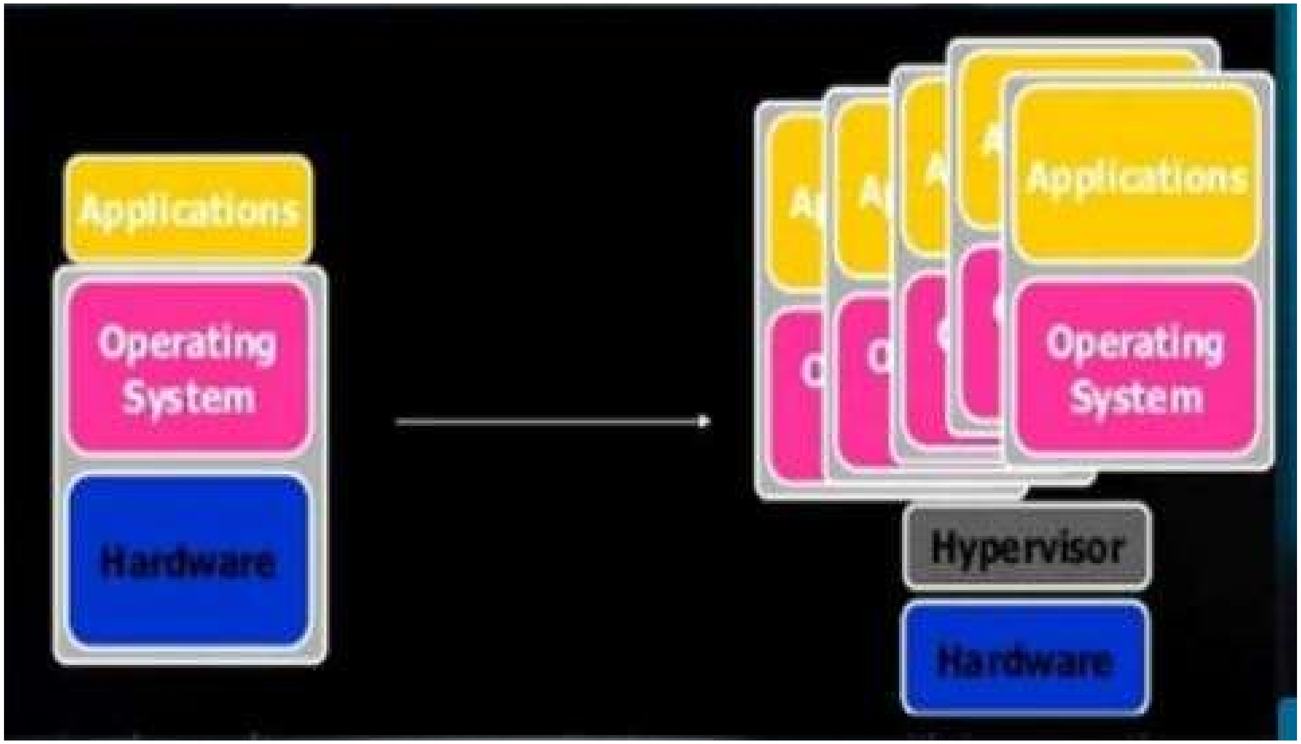
What is Virtualization??

- Virtualization is the process of creating a virtual version of a server, desktop, operating system or a storage device.



- virtualization of a computer into multiple logical computers

Virtualization - One Server for Multiple Applications/OS



Why Virtualization?

- It is the process of creating a virtual version of something like computer hardware.
- It involves using specialized software to create a virtual or software-created version of a computing resource rather than the actual version of the same resource.
- With the help of Virtualization, multiple operating systems and applications can run on same machine and its same hardware at the same time, increasing the utilization and flexibility of hardware.

BENEFITS OF VIRTUALIZATION

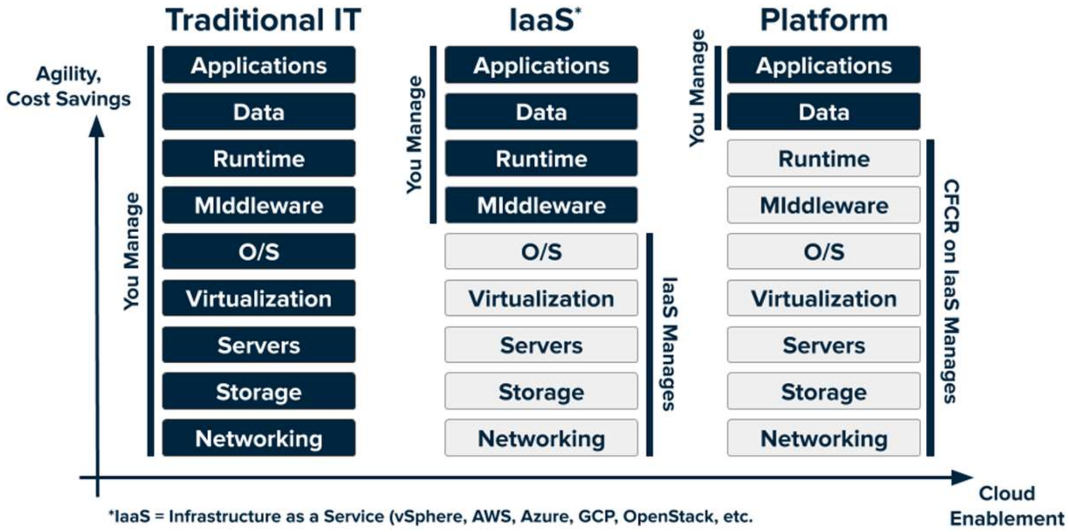
1. More flexible and efficient allocation of resources.
2. Enhance development productivity.
3. It lowers the cost of IT infrastructure.
4. Remote access and rapid scalability.
5. High availability and disaster recovery.
6. Pay peruse of the IT infrastructure on demand.
7. Enables running multiple operating systems.

Virtualization and Cloud Computing

- Virtualization is one of the main cost effective, hardware reducing, and energy saving techniques used by cloud providers.
- Virtualization allows to share a single physical instance of a resource or an application among multiple customers and organizations at one time.
- It does this by assigning a logical name to a physical storage and providing a pointer to that physical resource on demand.
- Virtualization technologies provide a virtual environment for not only executing applications but also for storage, memory, and networking.

Virtualization and Cloud Computing

- In cloud services such as IaaS (Infrastructure as a service), virtualization plays a crucial role in providing the services from the cloud provider end to the cloud subscriber.
- Cloud provider has a set of virtual machines which are allocated to the clients.
- Iaas component stack contains various layers such as hardware, operating system, middleware, and application.



Traditional IT

IaaS*

Platform

Agility, Cost Savings

You Manage

IaaS Manages

CFCR on IaaS Manages

Applications

Data

Runtime

MIddleware

O/S

Virtualization

Servers

Storage

Networking

Cloud Enablement

*IaaS = Infrastructure as a Service (vSphere, AWS, Azure, GCP, OpenStack, etc.

VMM (Virtual Machine Monitor or Hypervisor)

- The operating system layer is divided into two parts – Lower layer and higher layer.
- The lower layer is utilized by the VMM (Virtual Machine Monitor or Hypervisor)
- A hypervisor is a form of virtualization software used in Cloud hosting to divide and allocate the resources on various pieces of hardware.
- The program which provides partitioning, isolation or abstraction is called virtualization hypervisor.
- The hypervisor is a hardware virtualization technique that allows multiple guest operating systems (OS) to run on a single host system at the same time.

Types of Operating System

- Host Operating System: The operating system actually running on the hardware itself.

- Guest Operating System: The operating system running in the simulated environment likeVMware,virtualbox,ect.

- The higher layer is captured by an operating system running inside a VM referred to as a guest operating system.

VMM (Virtual Machine Monitor or Hypervisor)

- Hypervisor or Virtual Machine Monitor located just over the hardware, intended to manage all the hardware resources.
- A hypervisor is accountable for running the guest OS directly on the CPU.
- This only functions well if the guest OS is running on the same instruction set as of the host OS; otherwise, an instruction translation takes place.

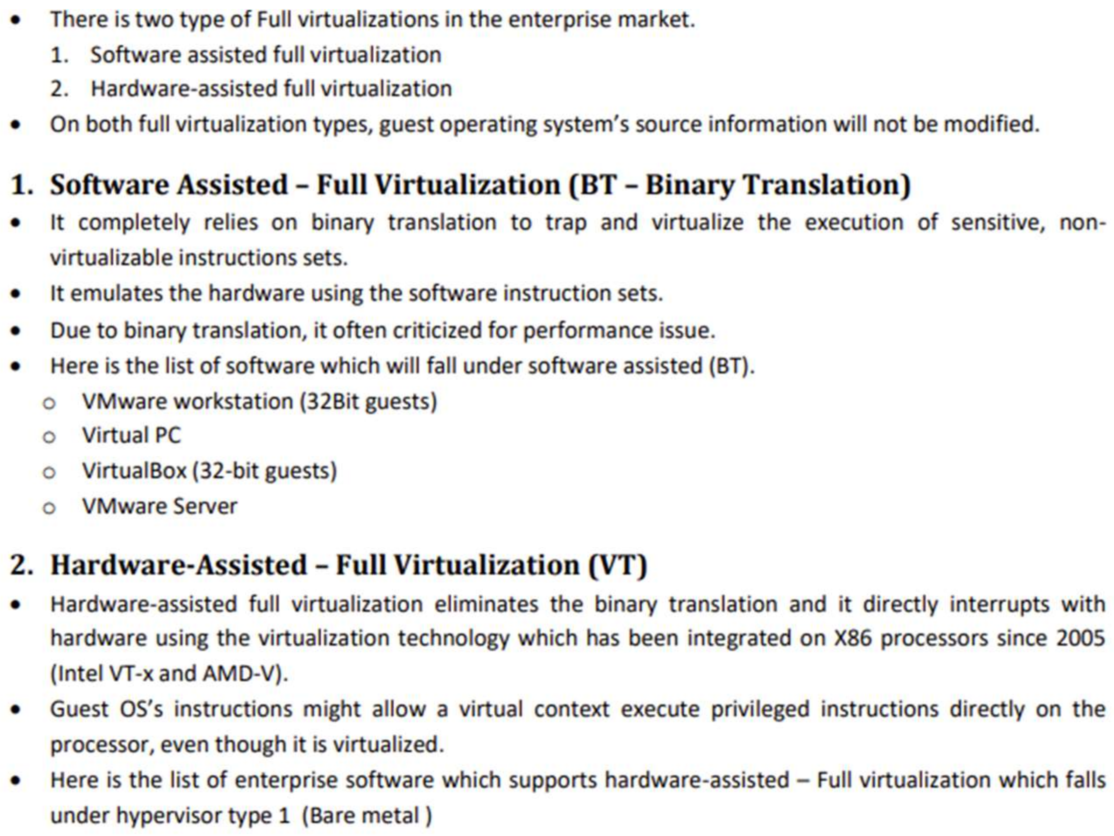
Virtualization Techniques

1. Para-virtualization
2. Full-virtualization
3. OS level-virtualization

Full virtualization

- Full virtualization is the first generation of the software solution regarding server virtualization and developed in the year of 1966 by IBM.
- A host operating system runs directly on the hardware while a guest operating system runs on the virtual machine.
- In full virtualization, the guest operating systems do not concern about the presence of a hypervisor.
- Each virtual machine and its guest OS operate as independent computers.
- Multiple guest OS execute on a single host OS in an isolated manner using direct execution and binary translation.

Full virtualization



- There is two type of Full virtualizations in the enterprise market.
  1. Software assisted full virtualization
  2. Hardware-assisted full virtualization
- On both full virtualization types, guest operating system's source information will not be modified.

1. Software Assisted – Full Virtualization (BT – Binary Translation)

- It completely relies on binary translation to trap and virtualize the execution of sensitive, non-virtualizable instructions sets.
- It emulates the hardware using the software instruction sets.
- Due to binary translation, it often criticized for performance issue.
- Here is the list of software which will fall under software assisted (BT).
  - VMware workstation (32Bit guests)
  - Virtual PC
  - VirtualBox (32-bit guests)
  - VMware Server

2. Hardware-Assisted – Full Virtualization (VT)

- Hardware-assisted full virtualization eliminates the binary translation and it directly interrupts with hardware using the virtualization technology which has been integrated on X86 processors since 2005 (Intel VT-x and AMD-V).
- Guest OS's instructions might allow a virtual context execute privileged instructions directly on the processor, even though it is virtualized.
- Here is the list of enterprise software which supports hardware-assisted – Full virtualization which falls under hypervisor type 1 (Bare metal)

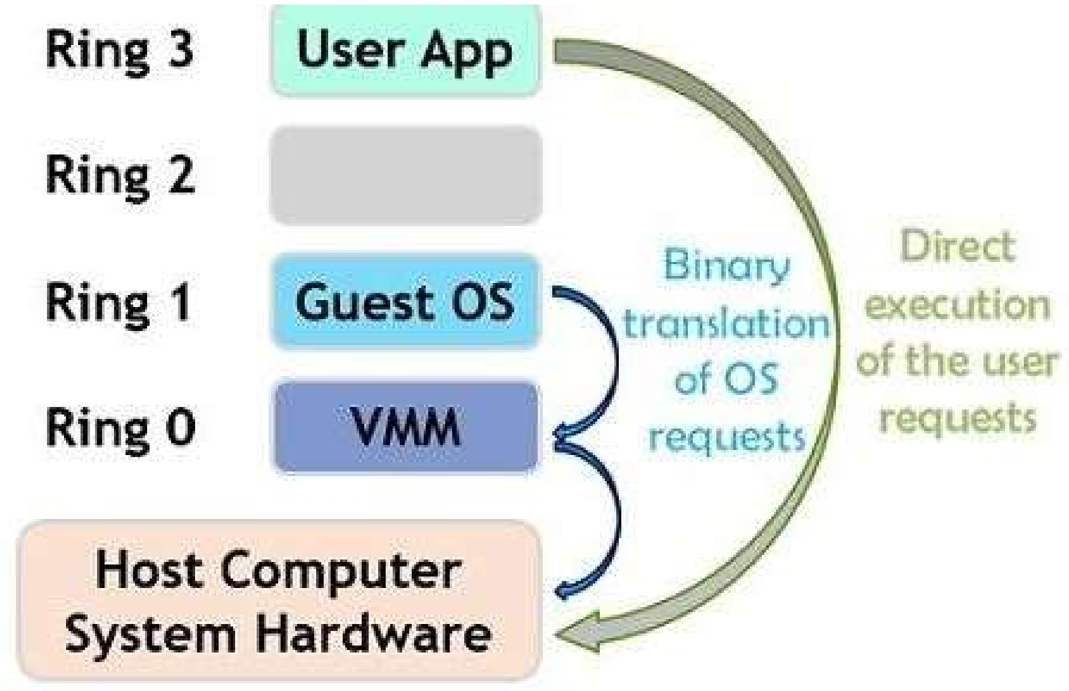
Full virtualization

- Software Assisted Virtualization:
  - whatever the virtual machines are producing a dynamic translator rewrites to the underlining hardware.
  - It involves a lack of awareness at the guest OS end about its virtualization and modification is inevitable.

  - The technologies provide full virtualization support are VMWare, ESXi and Microsoft virtual servers.
  - Each time an OS instruction is generated the hypervisor translates it during run-time quickly and caches the outcome for the future references.
  - While the user-level instructions are executed without modification at native speed.

Full virtualization

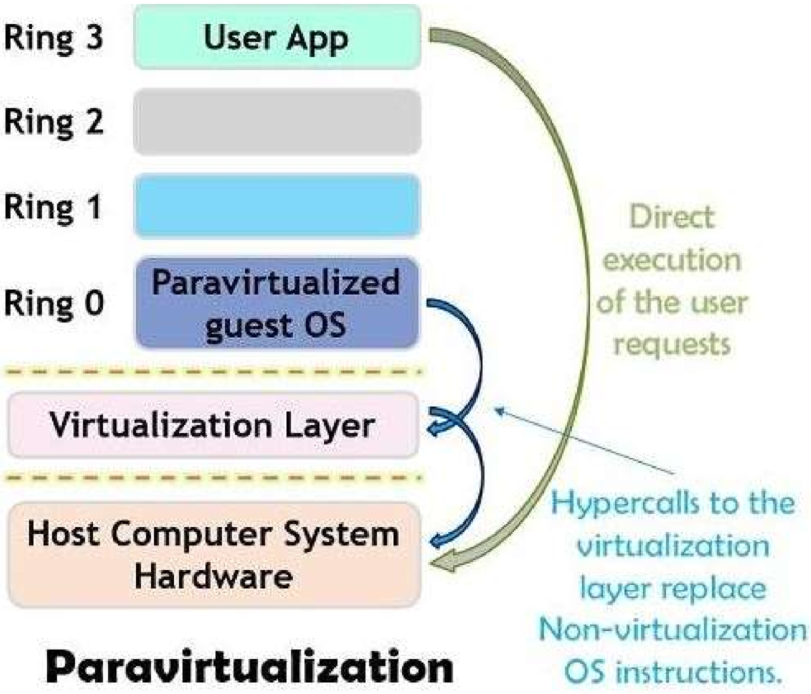
- Binary translation also takes much time and can achieve a huge performance overhead. I/O intensive applications are very challenging to employ full virtualization.


Ring 3
User App
Ring 2
Ring 1
Guest OS
Ring 0
VMM
Host Computer System Hardware
Binary translation of OS requests
Direct execution of the user requests

Para-virtualization

- Para virtualization is nothing but the interaction of the guest OS to the hypervisor in order to boost performance and productivity.
- Unlike full virtualization, paravirtualization does not implement complete isolation; instead, partial isolation is implemented in the approach.
- It also alters OS kernel to substitute the hypercalls in place of non-virtualizable instructions.
- The purpose of hypercalls is to interact with the virtualization layer hypervisor directly.

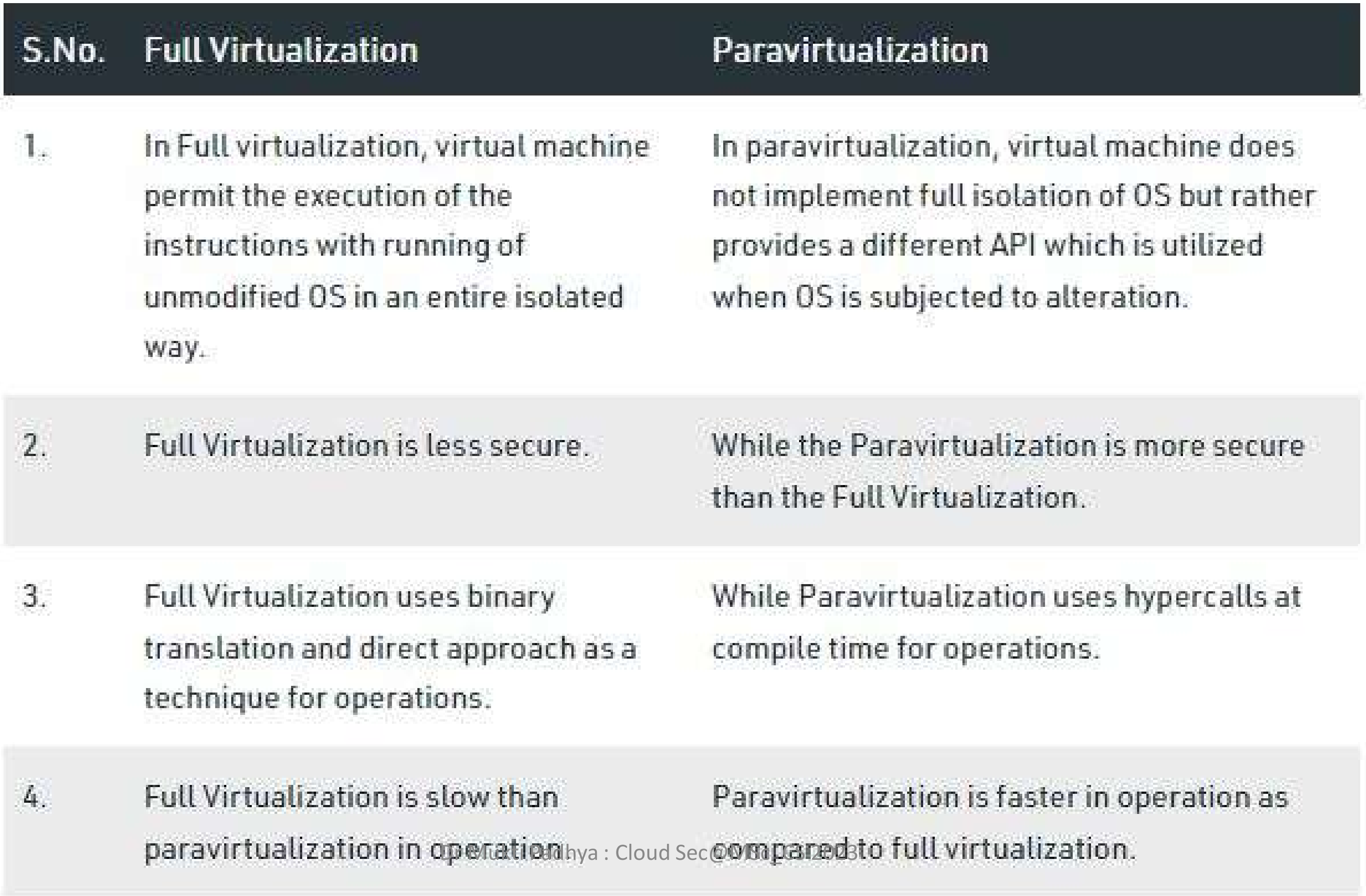
Para-virtualization


Ring 3
User App
Ring 2
Ring 1
Ring 0
Paravirtualized guest OS
Virtualization Layer
Host Computer System Hardware
Direct execution of the user requests
Hypercalls to the virtualization layer replace Non-virtualization OS instructions.
Paravirtualization

Para-virtualization

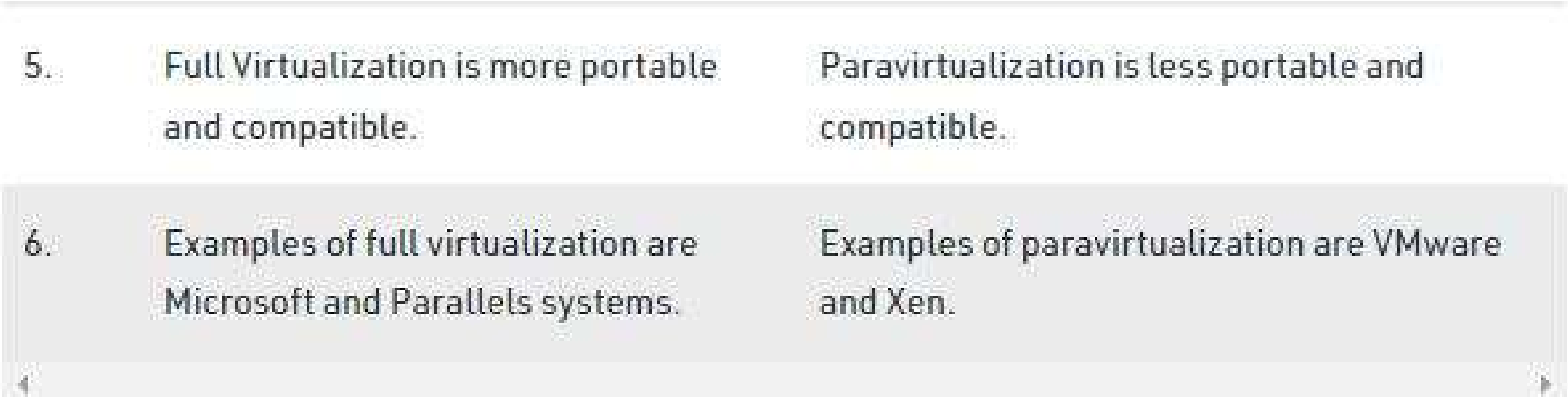
- In paravirtualization, there are various functions performed by hypervisor such as the arrangement of hypercalls interface for other crucial kernel functions like memory management, time keeping and interrupt handling.
- The major merit of paravirtualization is that it can easily reduce the virtualization overhead.
- it is less compatible and portable as it does not support the unmodified OS.
- It could also arise some crucial support and maintainability problems in the production environ due to the need for deep OS kernel modifications.

Full Virtualization vs. Para Virtualization



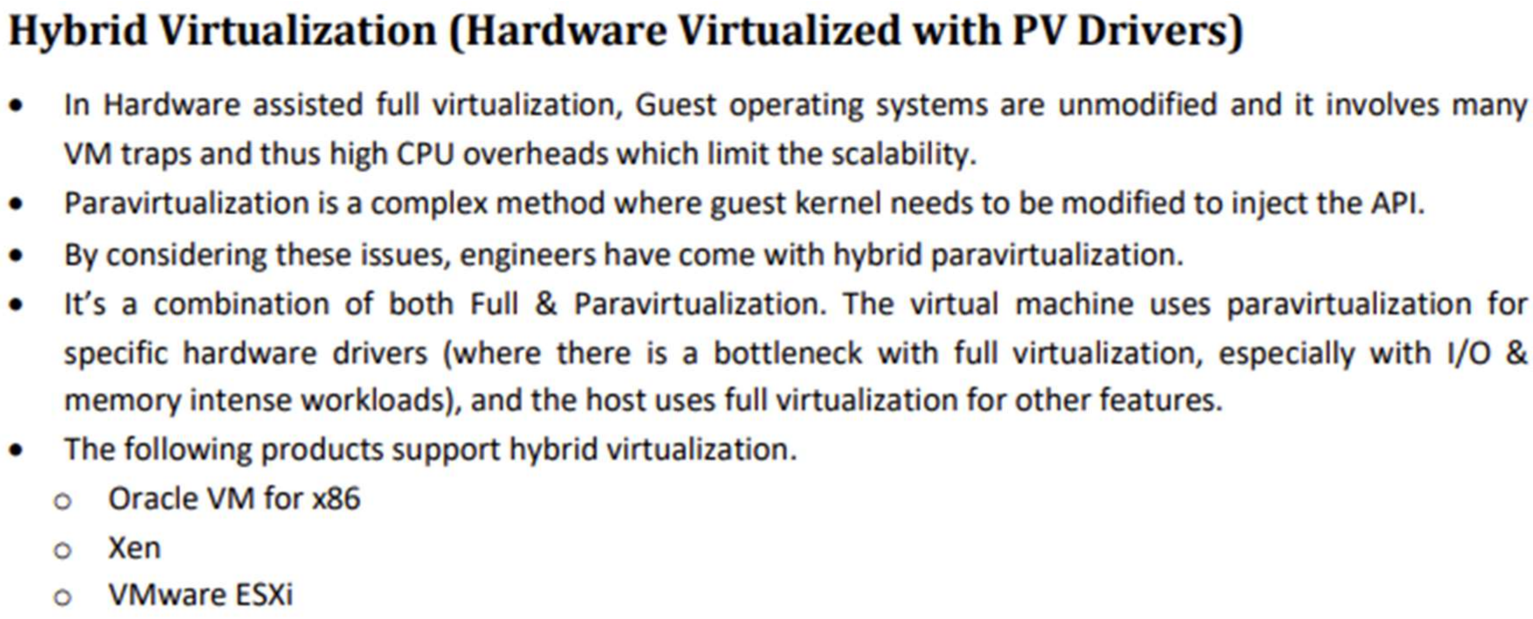
S.No.	Full Virtualization	Paravirtualization
1.	In Full virtualization, virtual machine permit the execution of the instructions with running of unmodified OS in an entire isolated way.	In paravirtualization, virtual machine does not implement full isolation of OS but rather provides a different API which is utilized when OS is subjected to alteration.
2.	Full Virtualization is less secure.	While the Paravirtualization is more secure than the Full Virtualization.
3.	Full Virtualization uses binary translation and direct approach as a technique for operations.	While Paravirtualization uses hypercalls at compile time for operations.
4.	Full Virtualization is slow than paravirtualization in operation.	Paravirtualization is faster in operation as compared to full virtualization.

Full Virtualization vs. Para Virtualization



5.	Full Virtualization is more portable and compatible.	Paravirtualization is less portable and compatible.
6.	Examples of full virtualization are Microsoft and Parallels systems.	Examples of paravirtualization are VMware and Xen.

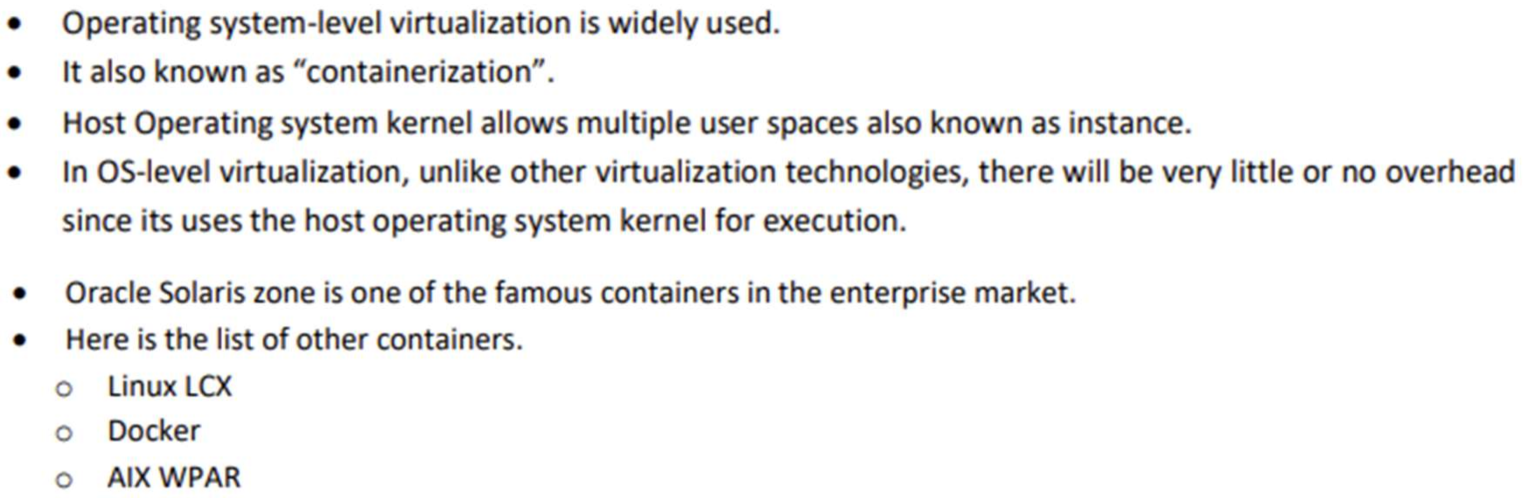
Hybrid-virtualization



Hybrid Virtualization (Hardware Virtualized with PV Drivers)

- In Hardware assisted full virtualization, Guest operating systems are unmodified and it involves many VM traps and thus high CPU overheads which limit the scalability.
- Paravirtualization is a complex method where guest kernel needs to be modified to inject the API.
- By considering these issues, engineers have come with hybrid paravirtualization.
- It's a combination of both Full & Paravirtualization. The virtual machine uses paravirtualization for specific hardware drivers (where there is a bottleneck with full virtualization, especially with I/O & memory intense workloads), and the host uses full virtualization for other features.
- The following products support hybrid virtualization.
  - Oracle VM for x86
  - Xen
  - VMware ESXi

OS Level -virtualization



- Operating system-level virtualization is widely used.
- It also known as “containerization”.
- Host Operating system kernel allows multiple user spaces also known as instance.
- In OS-level virtualization, unlike other virtualization technologies, there will be very little or no overhead since its uses the host operating system kernel for execution.
- Oracle Solaris zone is one of the famous containers in the enterprise market.
- Here is the list of other containers.
  - Linux LCX
  - Docker
  - AIX WPAR

Types of Virtualization:

1.Application Virtualization. 2.Network Virtualization.

3.Desktop Virtualization. 4.Storage Virtualization.

5.Server Virtualization. 6.Data virtualization.

Application Virtualization.

- Application virtualization helps a user to have remote access of an application from a server.
- The server stores all personal information and other characteristics of the application but can still run on a local workstation through the internet.
- Example of this would be a user who needs to run two different versions of the same software.
- Technologies that use application virtualization are hosted applications and packaged applications.

Network Virtualization:

- The ability to run multiple virtual networks with each has a separate control and data plan.
- It co-exists together on top of one physical network.
- It can be managed by individual parties that potentially confidential to each other.
- Network virtualization provides a facility to create and provision virtual networks—logical switches, routers, firewalls, load balancer, Virtual Private Network (VPN), and workload security within days or even in weeks.

Desktop Virtualization:

- Desktop virtualization allows the users' OS to be remotely stored on a server in the data centre.
- It allows the user to access their desktop virtually, from any location by a different machine.
- Users who want specific operating systems other than Windows Server will need to have a virtual desktop.
- Main benefits of desktop virtualization are user mobility, portability, easy management of software installation, updates, and patches.

Storage Virtualization:

- Storage virtualization is an array of servers that are managed by a virtual storage system.
- The servers aren’t aware of exactly where their data is stored, and instead function more like worker bees in a hive.
- It makes managing storage from multiple sources to be managed and utilized as a single repository.
- Storage virtualization software maintains smooth operations, consistent performance and a continuous suite of advanced functions despite changes, break down and differences in the underlying equipment.

Server Virtualization:

- This is a kind of virtualization in which masking of server resources takes place.
- The central-server(physical server) is divided into multiple different virtual servers by changing the identity number, processors.
- Each system can operate its own operating systems in isolate manner. Where each sub-server knows the identity of the central server.
- It causes an increase in the performance and reduces the operating cost by the deployment of main server resources into a sub-server resource.
- It’s beneficial in virtual migration, reduce energy consumption, reduce infrastructural cost, etc.

Data virtualization:

- This is the kind of virtualization in which the data is collected from various sources and managed that at a single place without knowing more about the technical information like how data is collected, stored & formatted
- then arranged that data logically so that its virtual view can be accessed by its interested people and stakeholders, and users through the various cloud services remotely.
- Many big giant companies are providing their services like Oracle, IBM, At scale, Cdata, etc.

- It can be used to performing various kind of tasks such as:
  - Data-integration
  - Business-integration
  - Service-oriented architecture data-services
  - Searching organizational data

Hypervisor

- It is software that partitions, abstracts, and isolates the operating system and applications of the underlying computer hardware.

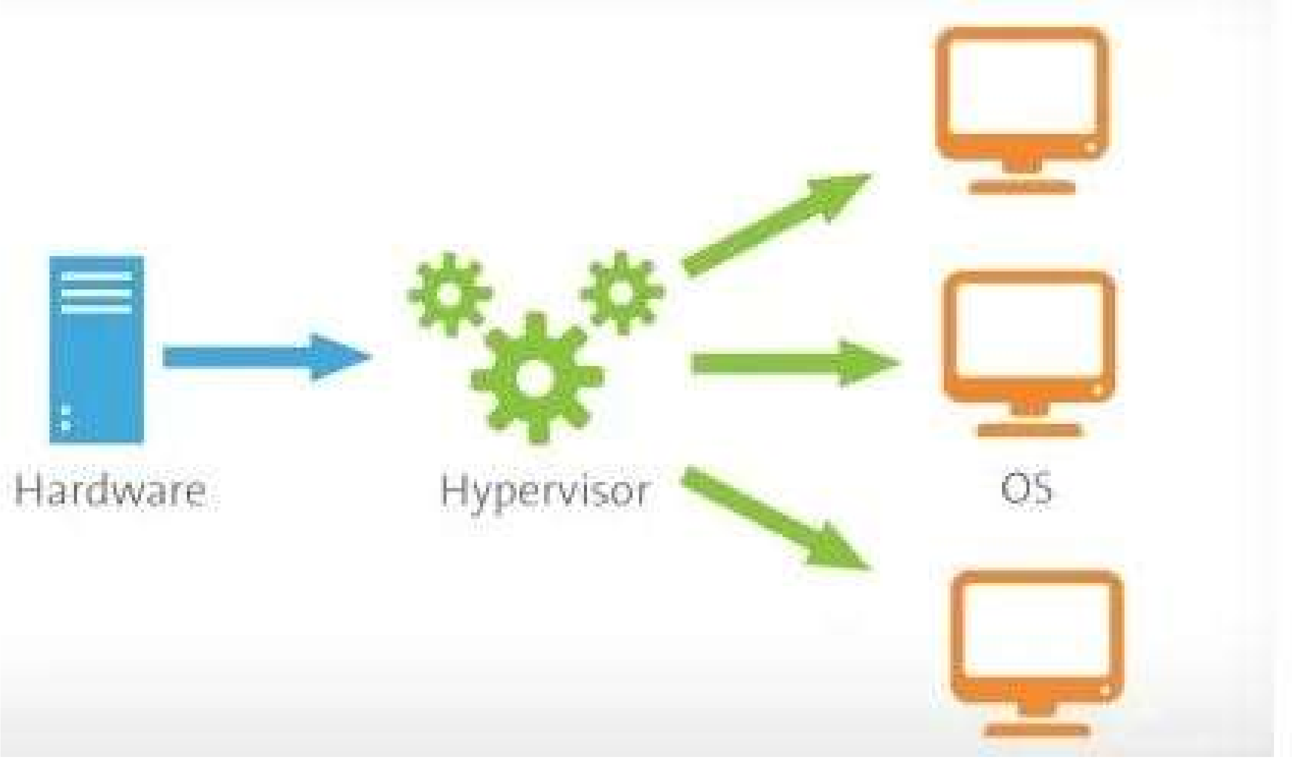
- It is a form of virtualization software used in cloud storage to divide and allocate resources across different hardware the program that provides partitioning, isolation, or abstraction is hypervisor virtualization.

- The hypervisor is a hardware virtualization technique that allows multiple guest operating systems (OS) to run on a host system at the same time.

- A hypervisor is also sometimes a virtual machine manager (VMM).

- A hypervisor can also be thought of as the operating system of virtualized systems.

Hypervisor



Features of Hypervisor

- There are two main characteristics of the hypervisor:
  - Partitioning/Partitions
  - Resource Allocation
- **Partitions:**
  - Hardware partition supervisor -Partitioning is a method of making efficient use of abundant hardware resources by allowing multiple independent software to run concurrently on the same hardware.
- **Resource Allocation:**
  - The hypervisor manages independent virtual machines by distributing resources like memory, network bandwidth, etc. between them.

How does a hypervisor work?

- Supervisor works at the server level and can be physical or virtual and is defined by hardware or software.

- Virtual machine guest OS load monitor

- Computer resource distribution monitor such as processor, memory, bandwidth, and disk storage for each virtual machine. It does this by creating pools of hardware resources, which it then allocates to virtual machines.

- The virtual machine can make requests to the hypervisor through API calls.

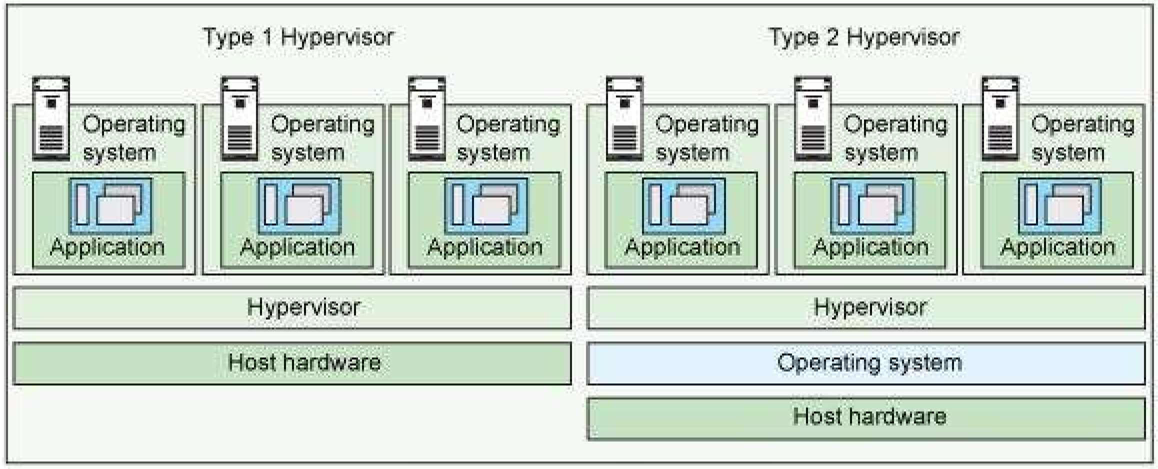
TYPES OF HYPERVISOR

- There are two types of Hypervisors we have, they are:
  - Type I
  - Type II
  - KVM

TYPES OF HYPERVISOR

- **Type 1 hypervisor:** hypervisors run directly on the system hardware – A “bare metal” embedded hypervisor,

- **Type 2 hypervisor:** hypervisors run on a host operating system that provides virtualization services, such as I/O device support and memory management.



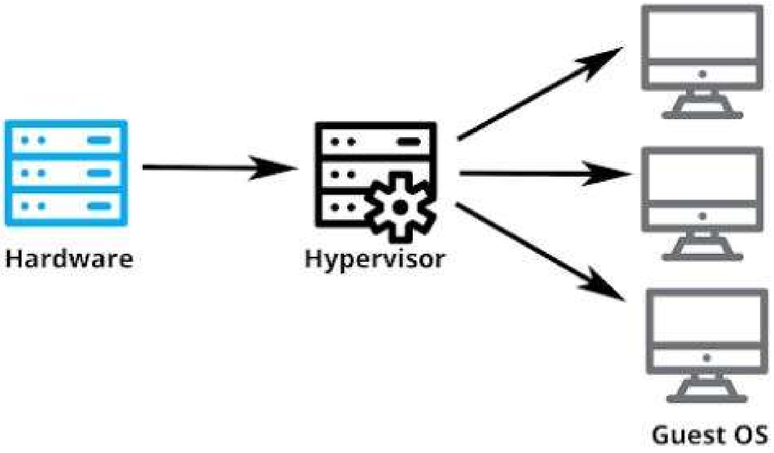
how type 1 and type 2 hypervisors differ

TYPE I HYPERVISOR:

- Type I are also called as Bare metal and Native Hypervisors.
- They directly run the server’s hardware or host machines directly , so the hypervisor software is the operating system. In other words, the hypervisor has direct access to the hardware without any other software interfering.
- Type I Hypervisor is best for enterprise computing and large scale deployments.
- Examples: VMware ESXi, Citrix XenServer and Microsoft Hyper-V hypervisor.

TYPE I HYPERVISOR:

- Type 1 is widely recognized as the highest performing and most efficient super monitor for business computing.
- The ability to directly allocate resources makes these monitors more scalable.



Advantages of Type I Hypervisor

- Physical resource optimisation: With Type 1 hypervisor, IT can leverage server hardware, freeing up data centre and real estate costs, and reducing power consumption.

- Better resource allocation: Most Type 1 monitors give administrators the ability to manually set resource allocations, based on application priority. Many Type 1 monitors also automate resource allocation as needed, allowing resource management to be a dynamic and personalised option.

- Type 1 hypervisors are mainly found in enterprise environments.

Disadvantages of Type I Hypervisor

- Cons: One problem with Type-1 hypervisors is that they usually need a dedicated separate machine to perform their operation and to instruct different VMs and control the host hardware resources

Type 1 Vendors

- VMware ESX and ESXi
  - VMware is an industry-leading vendor of virtualization technology, and many large data centers run on their products. It may not be the most cost-effective solution for smaller IT environments. If you do not need all the advanced features VMware vSphere offers, there is a free version of this hypervisor and multiple commercial editions.
  - These hypervisors offer advanced features and scalability, but require licensing, so the costs are higher.

Type 1 Vendors

- Citrix XenServer
  - formerly known as Xen Server
  - This Server virtualization platform by Citrix is best suited for enterprise environments. It can handle all types of workloads and provides features for the most demanding tasks.
  - The core hypervisor technology is free, but like VMware’s free ESXi, it has almost no advanced features.

Type 1 Vendors

- Microsoft Hyper-V
- Microsoft also offers a free edition of their hypervisor, but if you want a GUI and additional functionalities, you will have to go for one of the commercial versions. Hyper-V may not offer as many features as VMware vSphere package, but you still get live migration, replication of virtual machines, dynamic memory and many other features.

Type 1 Hypervisors

VMware vSphere

Microsoft Hyper-V

Xen Server™

KVM

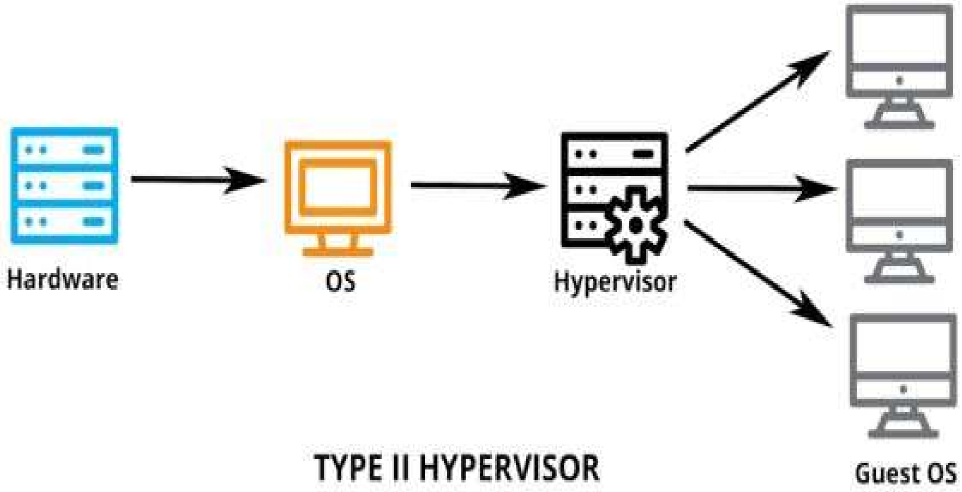
Red Hat Virtualization

TYPE 2 HYPERVISOR:

- Type II hypervisors are embedded or hosted hypervisors.
- Usually, these hypervisors are on top of the operating system. Due to its reliance on the underlying host operating system – unlike type 1 – it is known as “hosted hypervisor”.
- Hypervisor runs as an application in the operating system, then runs directly on the host computer.
- Type 2 hypervisor supports multiple clients but is not a direct access to the server hardware and its resources.
- The pre-existing operating system processes calls to the CPU to obtain memory, network resources, and memory. All of these can create some latency.
- Examples: Oracle Solaris Zones, Oracle VM Server for x86, Oracle VM Virtual Box, VMWare Workstation, VMware Fusion and more.

TYPE 2 HYPERVISOR:

- Type 2 hypervisors are typically found in environments with a small number of servers.



Advantages of Type II Hypervisor

- Easier configuration: These hypervisors are easier to configure and manage as there is an underlying operating system to work with.

- Simplified management: Type 2 supervisors do not require a dedicated administrator.

- Compatibility: Type 2 hypervisors are compatible with a wider range of hardware because they run on an operating system, rather than specific hardware machines.

- Cons: Here there is no direct access to the physical hardware resources so the efficiency of these hypervisors lags in performance as compared to the type-1 hypervisors, and potential security risks are also there an attacker can compromise the security weakness if there is access to the host operating system so he can also access the guest operating system.

Type 2 Hypervisors

vmware Workstation
VirtualBox
VMware Fusion
QEMU
Microsoft Virtual PC

KVM Hypervisor

- KVM stands for Kernel-based Virtual Machine

- It is a mixture of Type I and Type 2 hypervisors.

- It is on Linux and makes Linux a Type I hypervisor.

- This type of hypervisor:
  - is secure.
  - Offers plenty of storage
  - Full hardware support and memory management capabilities
  - Provides low latency
  - Allow apps to take priority
  - Provides better scalability, planning, and control of resources

Criteria for choosing Hypervisor

- Type 1 hypervisors offer much better performance than Type 2
- Type 2 are much simpler to set up,
- To determine which hypervisor meets your needs is to compare their performance metrics.
  - CPU overhead,
  - the amount of maximum host
  - guest memory,
  - support for virtual processors.

Criteria for choosing Hypervisor

- Cost
- Understand your needs:
  - Flexibility
  - Scalability
  - Usability
  - Availability
  - Reliability
  - Efficiency
  - Reliable support

Criteria for choosing Hypervisor

- Performance
- Ecosystem : the availability of documentation, support, training, third-party developers and consultancies, and so on – in determining whether or not a solution is cost-effective in the long term.
- Team expertise
  - expertise of your staff in managing and maintaining a particular hypervisor platform. The more familiar your team is with a given product, its configuration, and its eccentricities, the fewer the configuration mistakes.
- Certifications and attestations
  - One additional consideration when selecting a hypervisor is the availability of various formal certifications and attestations. While they may not be requirements for your specific organization, these certifications and attestations speak to the maturity, production readiness, and thoroughness of the testing a particular hypervisor platform has been subjected to.

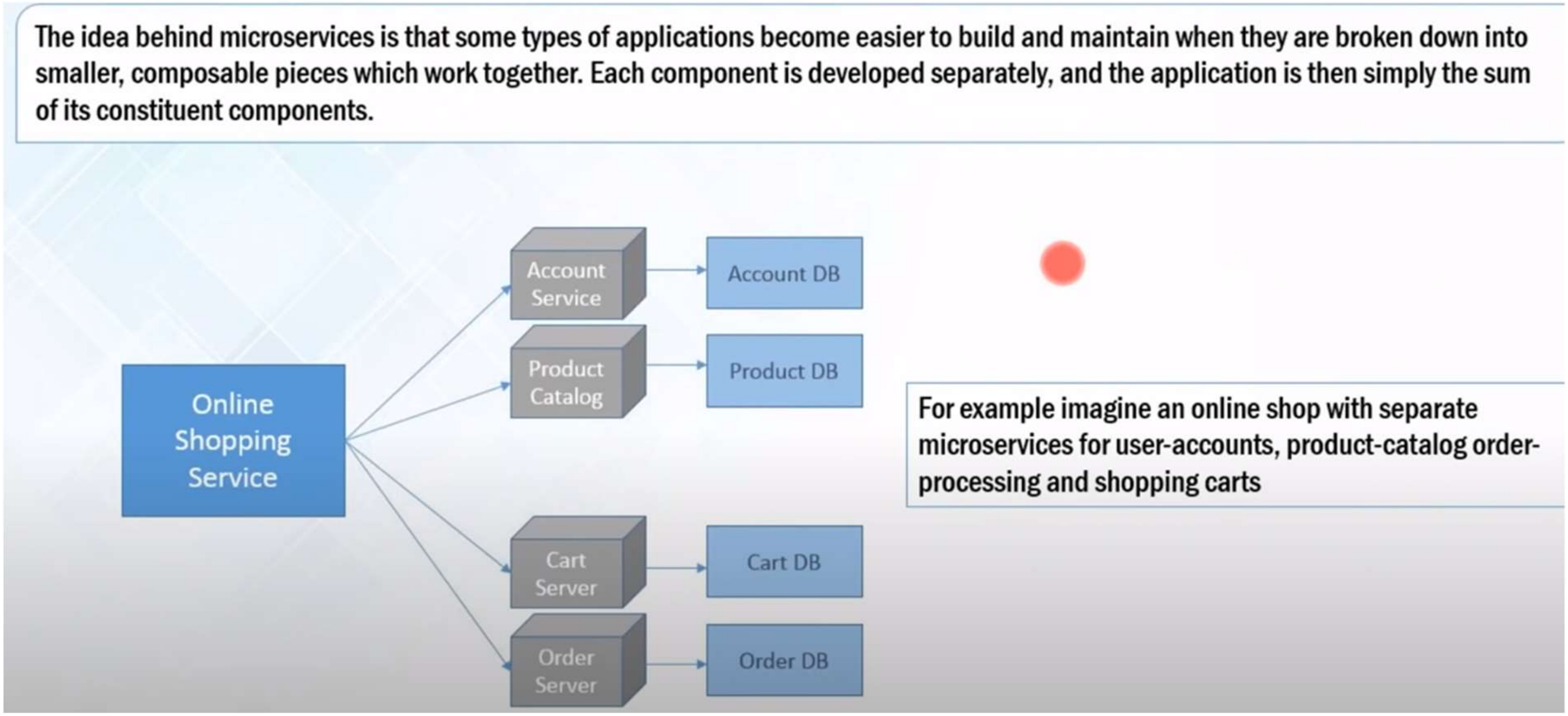
Criteria for choosing Hypervisor

- Product or project maturity

Product maturity has a number of effects once you have deployed your cloud:

  - Availability of expertise
  - Active developer and user communities
  - Timeliness and availability of updates
  - Incidence response

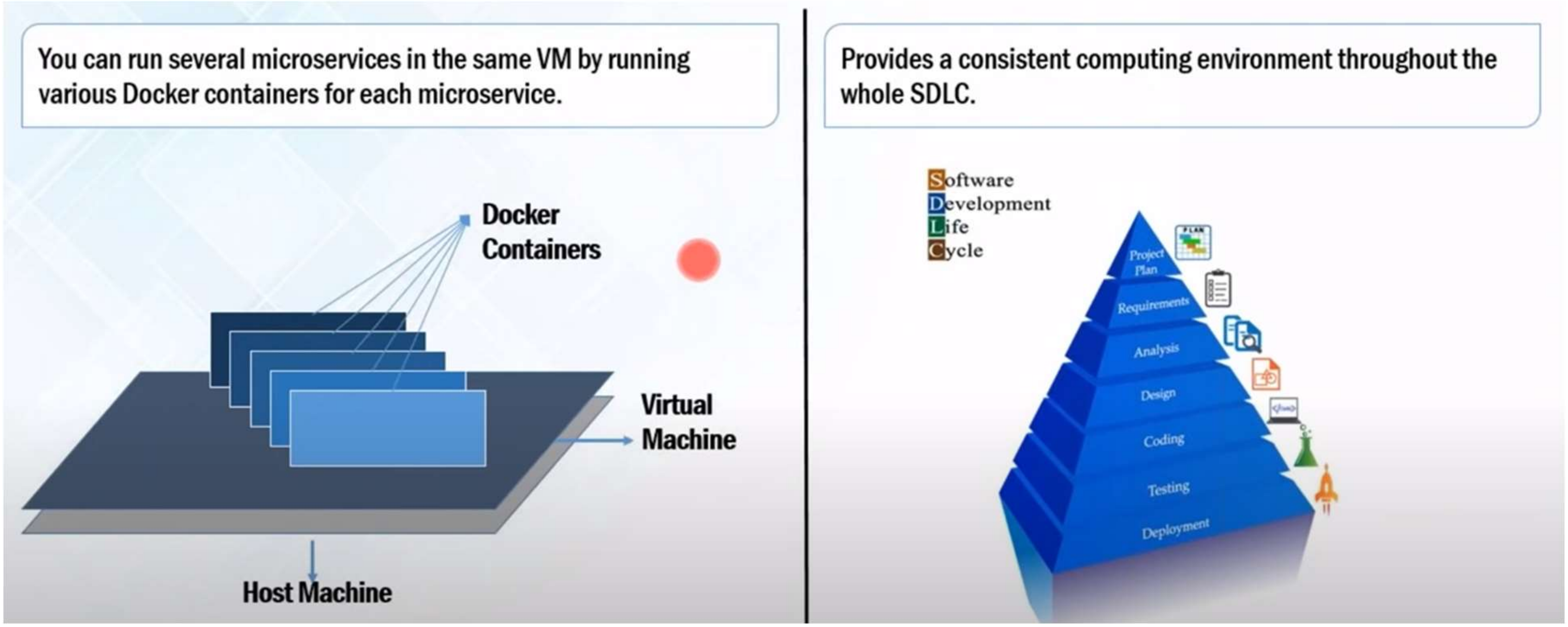
Problems of Virtual M/C



The idea behind microservices is that some types of applications become easier to build and maintain when they are broken down into smaller, composable pieces which work together. Each component is developed separately, and the application is then simply the sum of its constituent components.

For example imagine an online shop with separate microservices for user-accounts, product-catalog order-processing and shopping carts

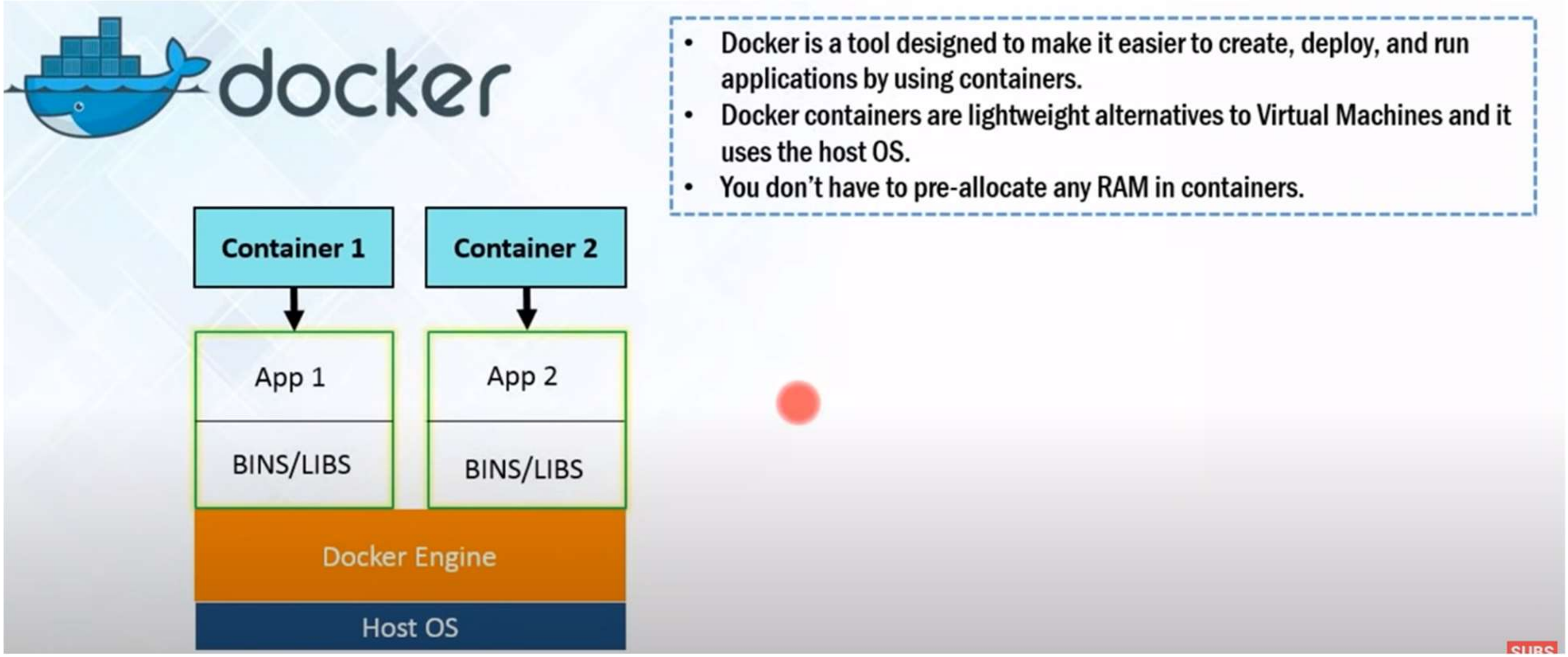
Solution : Docker??



You can run several microservices in the same VM by running various Docker containers for each microservice.

Provides a consistent computing environment throughout the whole SDLC.

What is Docker??

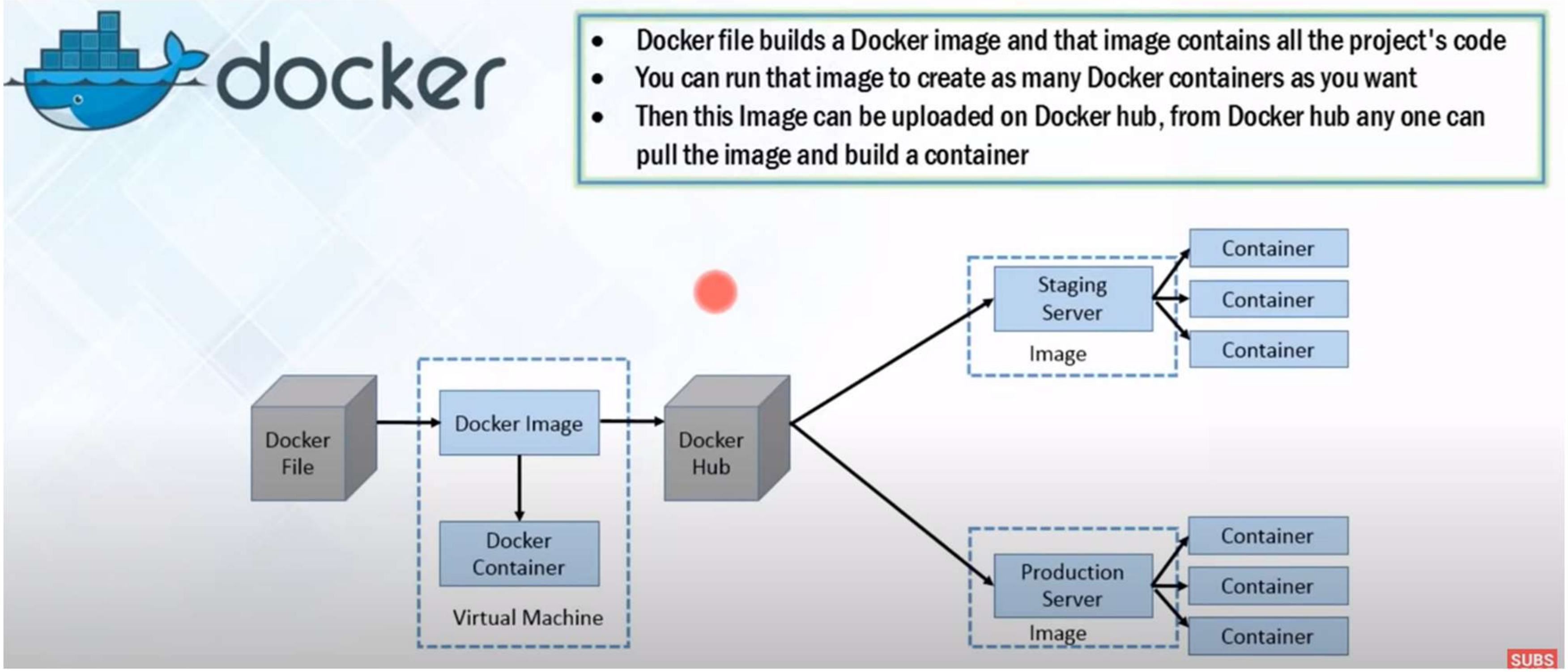


- Docker is a tool designed to make it easier to create, deploy, and run applications by using containers.
- Docker containers are lightweight alternatives to Virtual Machines and it uses the host OS.
- You don't have to pre-allocate any RAM in containers.

Docker

- Docker is a set of platforms as a service (PaaS) products that use the Operating system level virtualization to deliver software in packages called containers.
- Containers are isolated from one another and bundle their own software, libraries, and configuration files; they can communicate with each other through well-defined channels.
- All containers are run by a single operating system kernel and therefore use fewer resources than a virtual machine.

Working of Docker



- Docker file builds a Docker image and that image contains all the project's code
- You can run that image to create as many Docker containers as you want
- Then this Image can be uploaded on Docker hub, from Docker hub any one can pull the image and build a container

Important Terminologies in Docker

1. Docker Image

- It is a file, comprised of multiple layers, used to execute code in a Docker container.
- They are a set of instructions used to create docker containers.

2. Docker Container

- It is a runtime instance of an image.
- Allows developers to package applications with all parts needed such as libraries and other dependencies.

3. Docker file

- It is a text document that contains necessary commands which on execution helps assemble a Docker Image.
- Docker image is created using a Docker file.

Important Terminologies in Docker

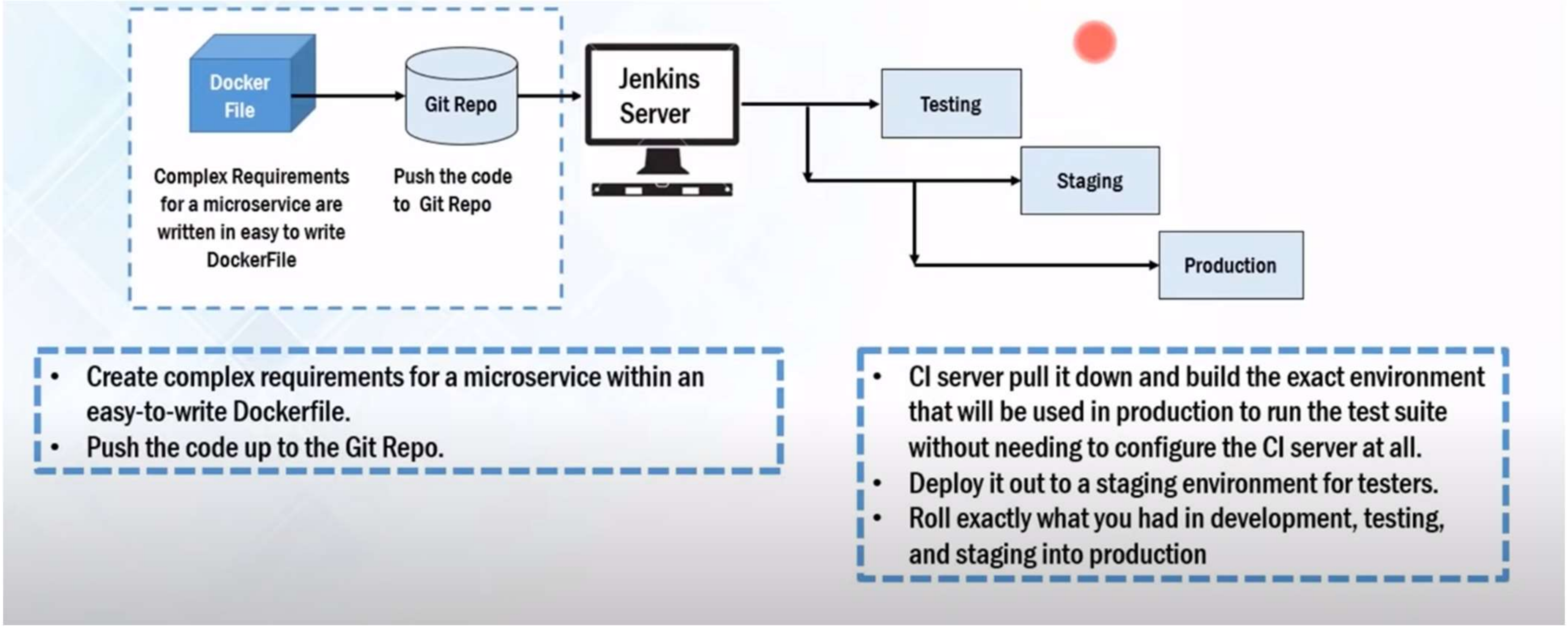
4. Docker Engine

- The software that hosts the containers is named Docker Engine.
- Docker Engine is a client-server based application
- The docker engine has **3 main** components:
  - **Server**: It is responsible for creating and managing Docker images, containers, networks, and volumes on the Docker. It is referred to as a daemon process.
  - **REST API**: It specifies how the applications can interact with the Server and instructs it what to do.
  - **Client**: The Client is a docker command-line interface (CLI), that allows us to interact with Docker using the docker commands.

5. Docker Hub

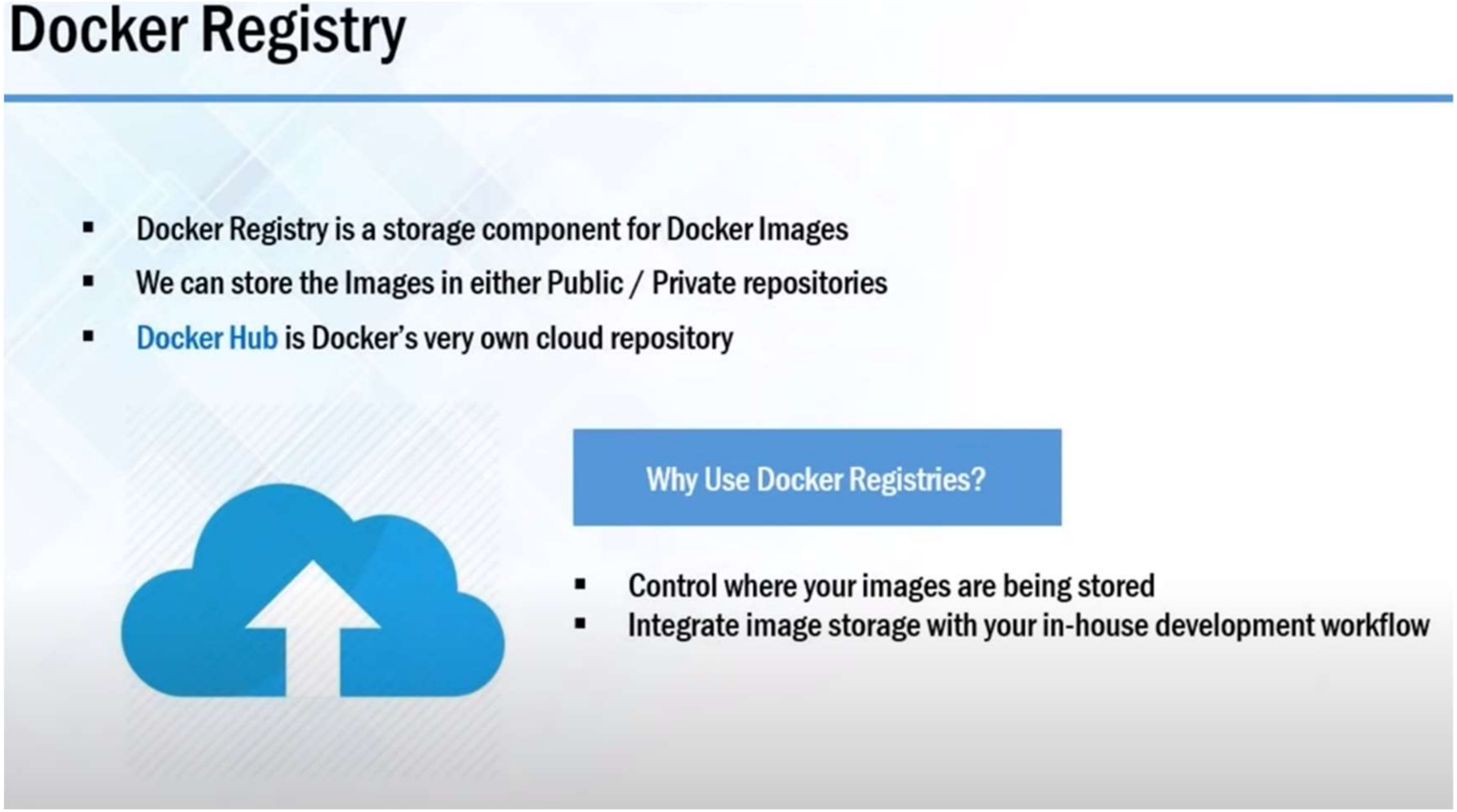
- Docker Hub is the official online repository where you can find other Docker Images that are available for use.
- It makes it easy to find, manage, and share container images with others.

CI/CD using Docker


Docker File
Git Repo
Jenkins Server
Testing
Staging
Production
Complex Requirements for a microservice are written in easy to write DockerFile
Push the code to Git Repo

- Create complex requirements for a microservice within an easy-to-write Dockerfile.
- Push the code up to the Git Repo.

- CI server pull it down and build the exact environment that will be used in production to run the test suite without needing to configure the CI server at all.
- Deploy it out to a staging environment for testers.
- Roll exactly what you had in development, testing, and staging into production



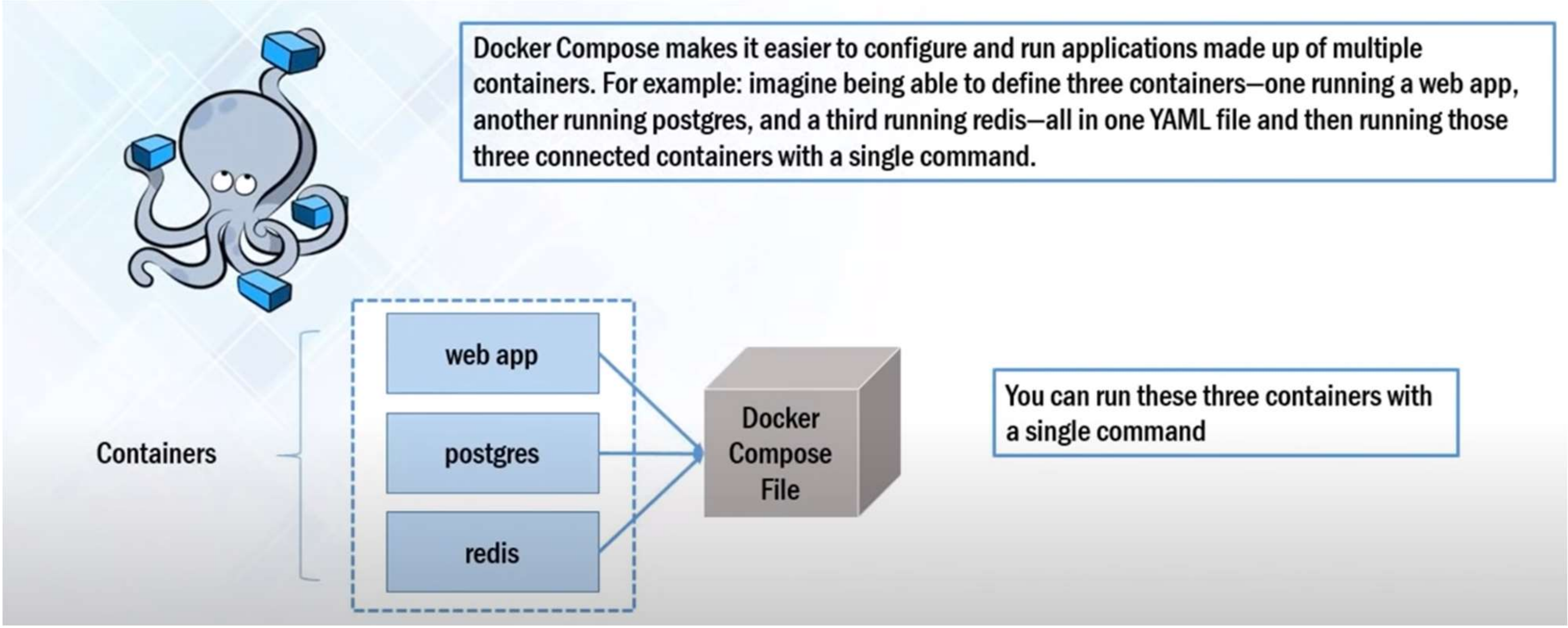
Docker Registry

- Docker Registry is a storage component for Docker Images
- We can store the Images in either Public / Private repositories
- Docker Hub is Docker's very own cloud repository

Why Use Docker Registries?

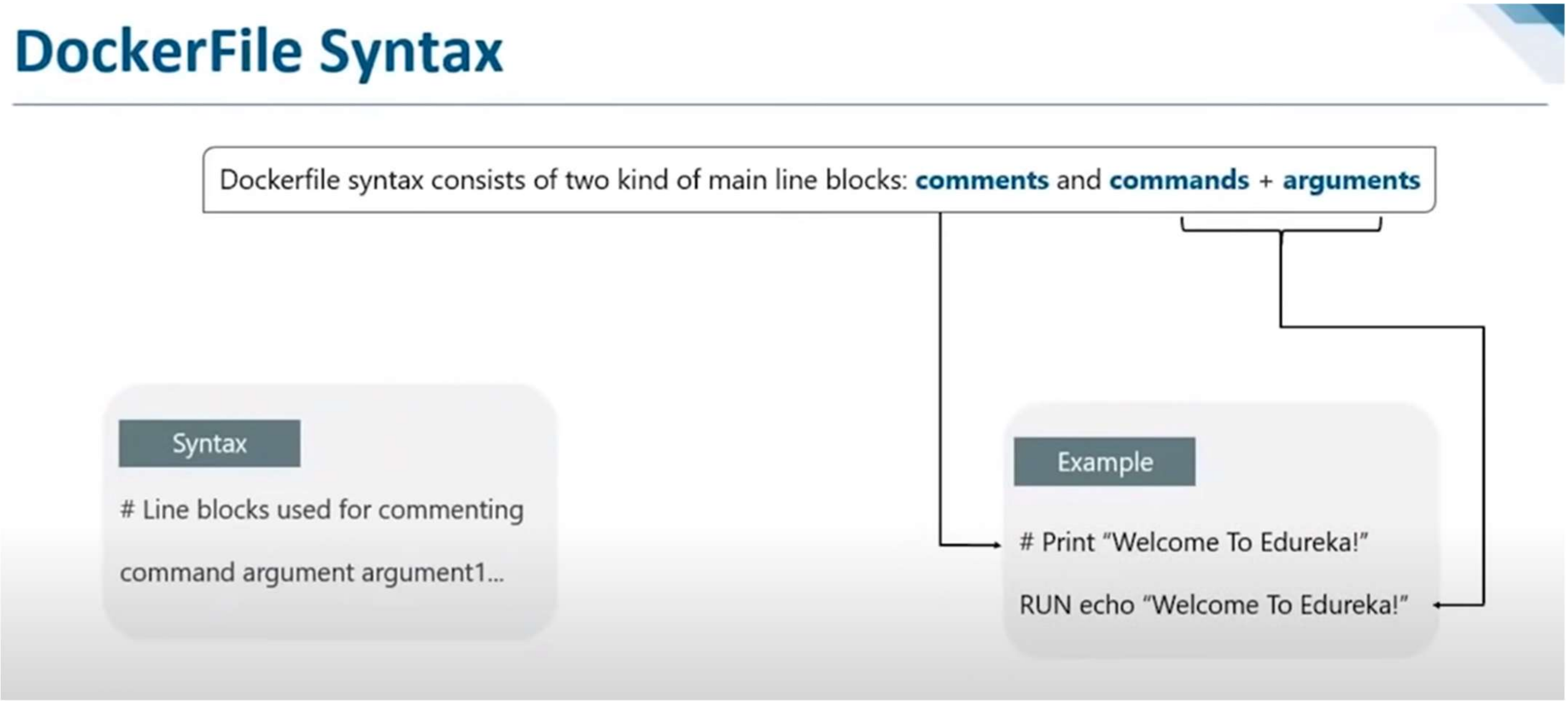
- Control where your images are being stored
- Integrate image storage with your in-house development workflow

Docker Compose



Docker Compose makes it easier to configure and run applications made up of multiple containers. For example: imagine being able to define three containers—one running a web app, another running postgres, and a third running redis—all in one YAML file and then running those three connected containers with a single command.

You can run these three containers with a single command



DockerFile Syntax

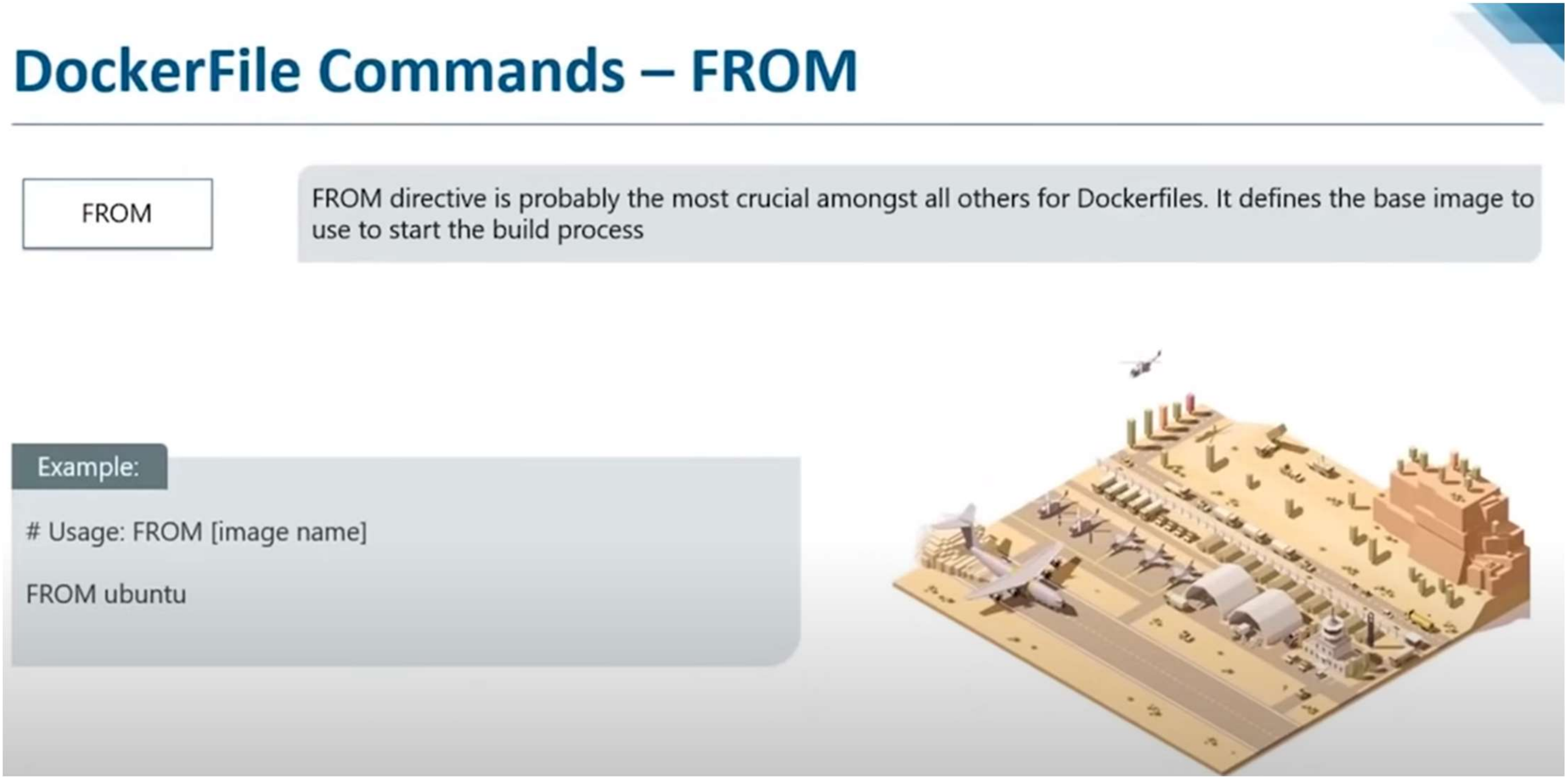
Dockerfile syntax consists of two kind of main line blocks: **comments** and **commands** + **arguments**

Syntax

```
# Line blocks used for commenting
command argument argument1...
```

Example

```
# Print "Welcome To Edureka!"
RUN echo "Welcome To Edureka!"
```



DockerFile Commands – FROM

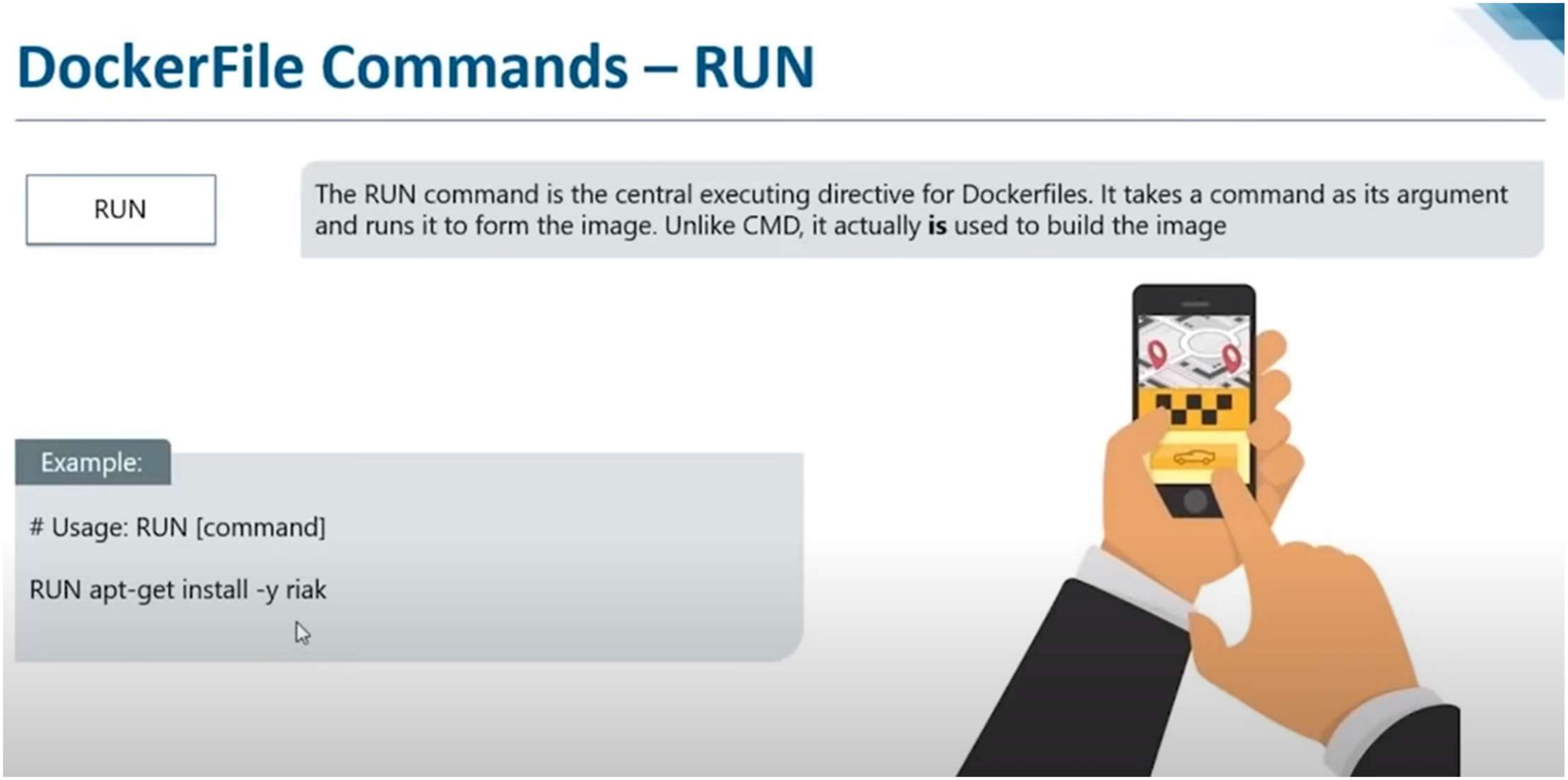
FROM

FROM directive is probably the most crucial amongst all others for Dockerfiles. It defines the base image to use to start the build process

Example:

```
# Usage: FROM [image name]

FROM ubuntu
```



DockerFile Commands – RUN

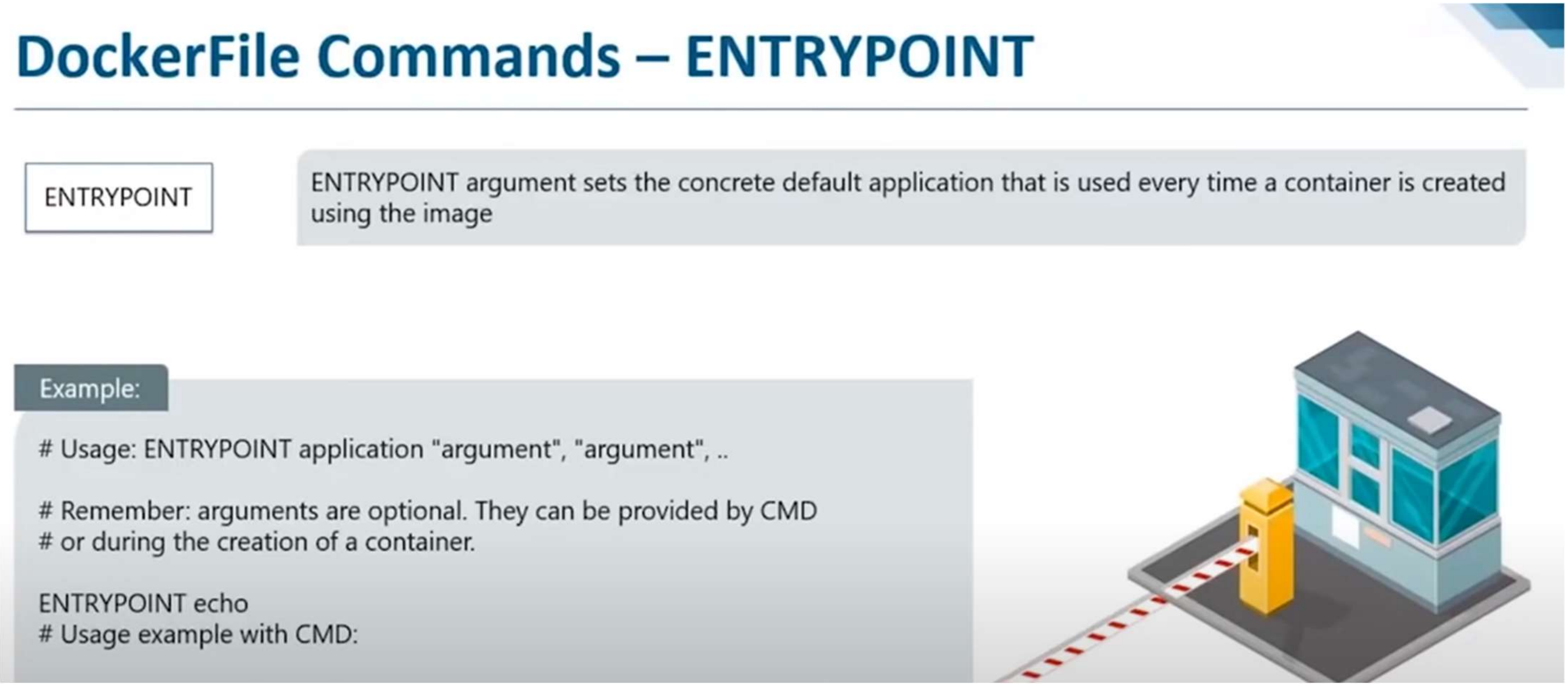
RUN

The RUN command is the central executing directive for Dockerfiles. It takes a command as its argument and runs it to form the image. Unlike CMD, it actually **is** used to build the image

Example:

```
# Usage: RUN [command]

RUN apt-get install -y riak
```



DockerFile Commands – ENTRYPOINT

ENTRYPOINT

ENTRYPOINT argument sets the concrete default application that is used every time a container is created using the image

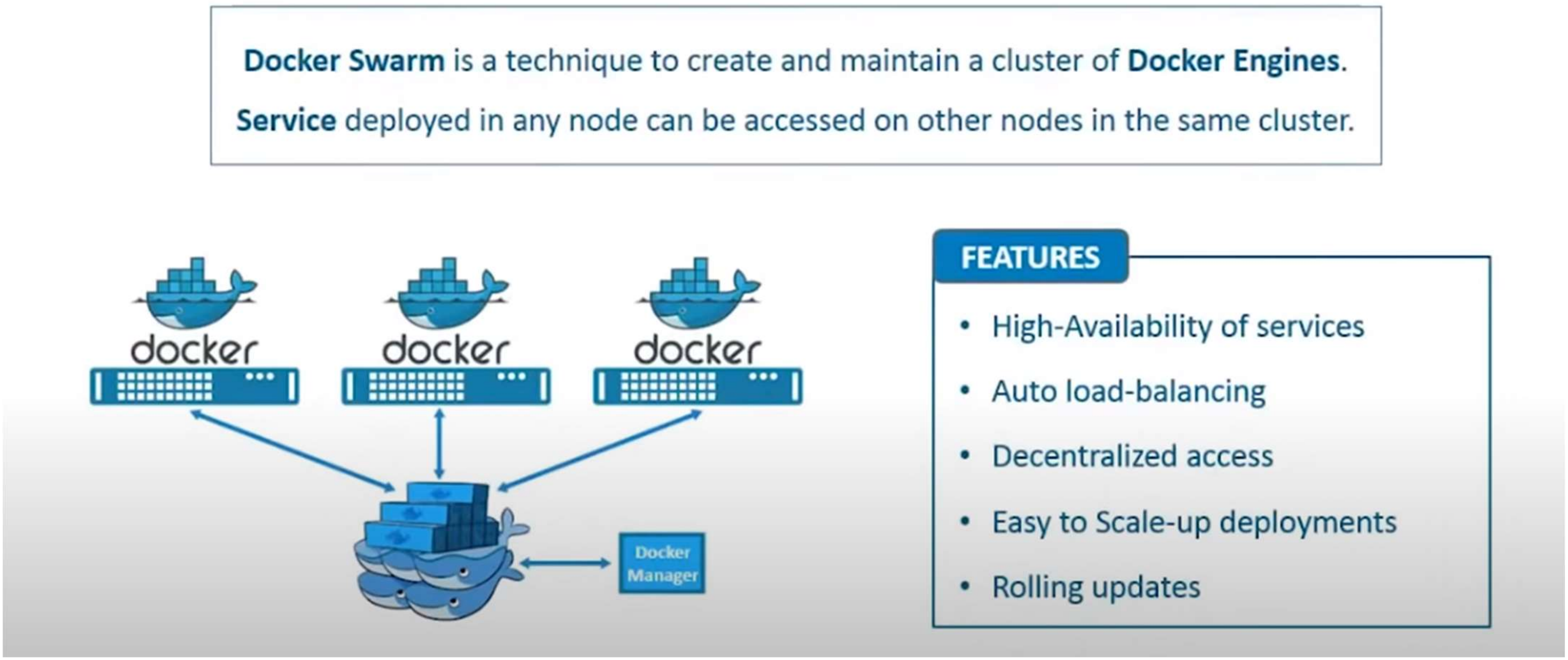
Example:

```
# Usage: ENTRYPOINT application "argument", "argument", ..

# Remember: arguments are optional. They can be provided by CMD
# or during the creation of a container.

ENTRYPOINT echo
# Usage example with CMD:
```

What Is Docker Swarm?



Docker Swarm is a technique to create and maintain a cluster of **Docker Engines**.

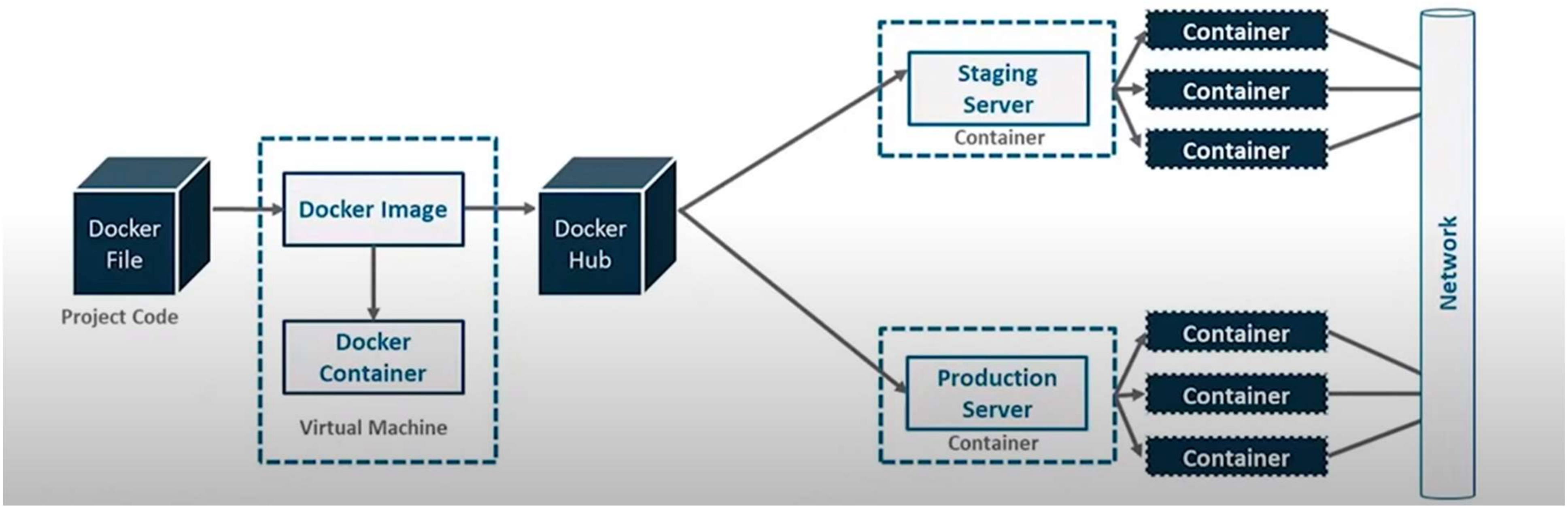
Service deployed in any node can be accessed on other nodes in the same cluster.

FEATURES

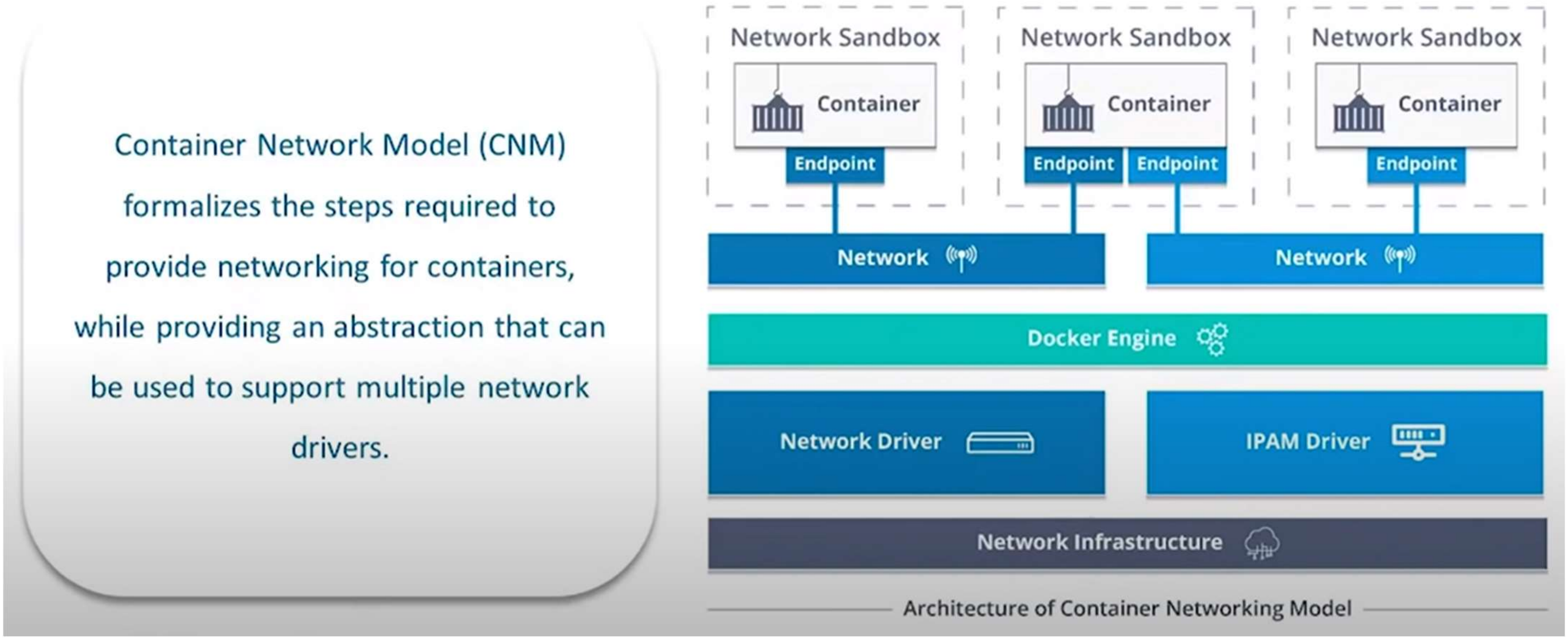
- High-Availability of services
- Auto load-balancing
- Decentralized access
- Easy to Scale-up deployments
- Rolling updates

Docker Networking

Docker containers and services do not need to be aware that they are deployed on Docker, or whether their peers are also Docker workloads or not, and this introduces the concept of Docker Networking.

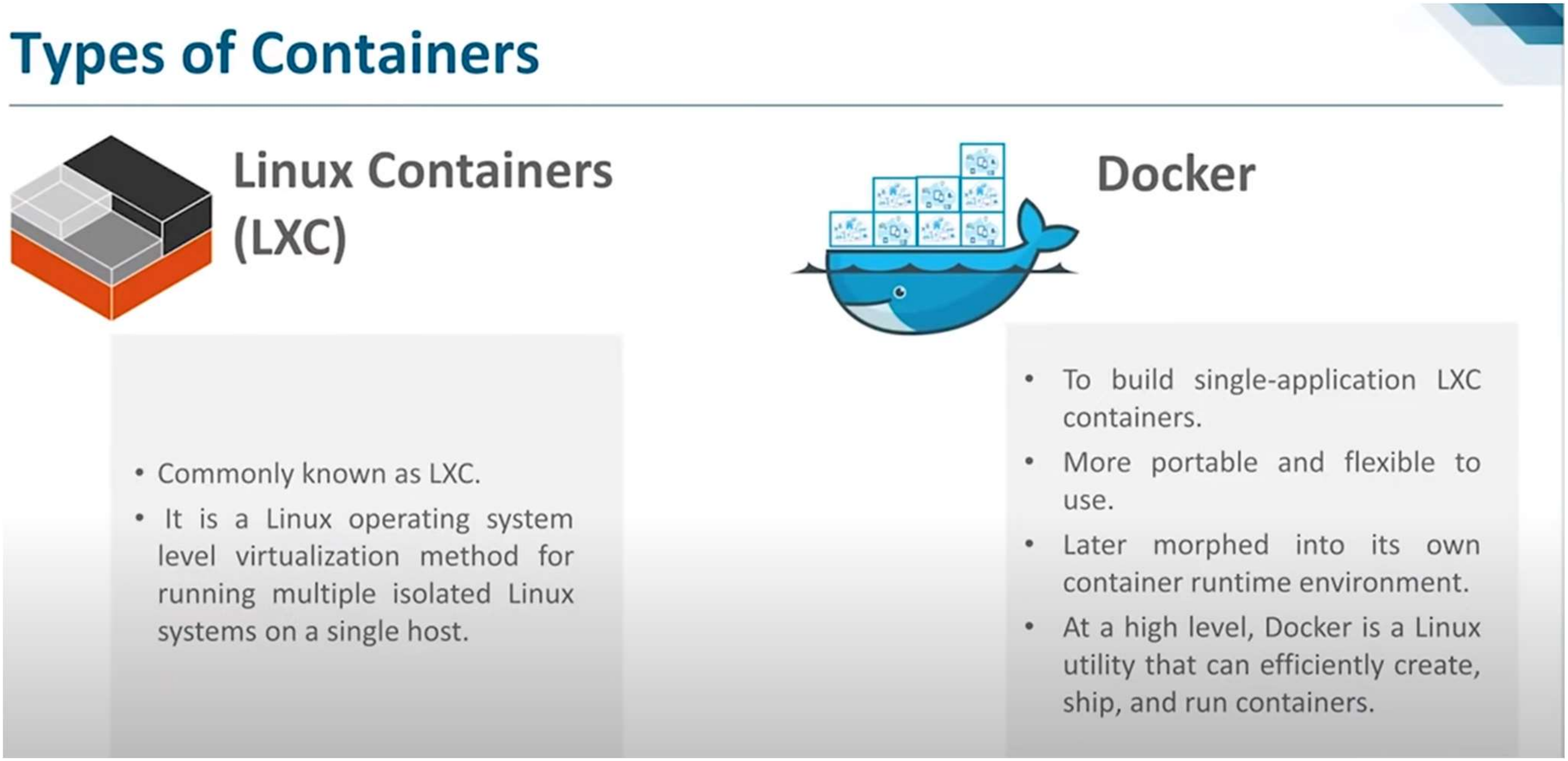


Container Network Model



Container Network Model (CNM) formalizes the steps required to provide networking for containers, while providing an abstraction that can be used to support multiple network drivers.

Architecture of Container Networking Model



Types of Containers

Linux Containers (LXC)

- Commonly known as LXC.
- It is a Linux operating system level virtualization method for running multiple isolated Linux systems on a single host.

Docker

- To build single-application LXC containers.
- More portable and flexible to use.
- Later morphed into its own container runtime environment.
- At a high level, Docker is a Linux utility that can efficiently create, ship, and run containers.

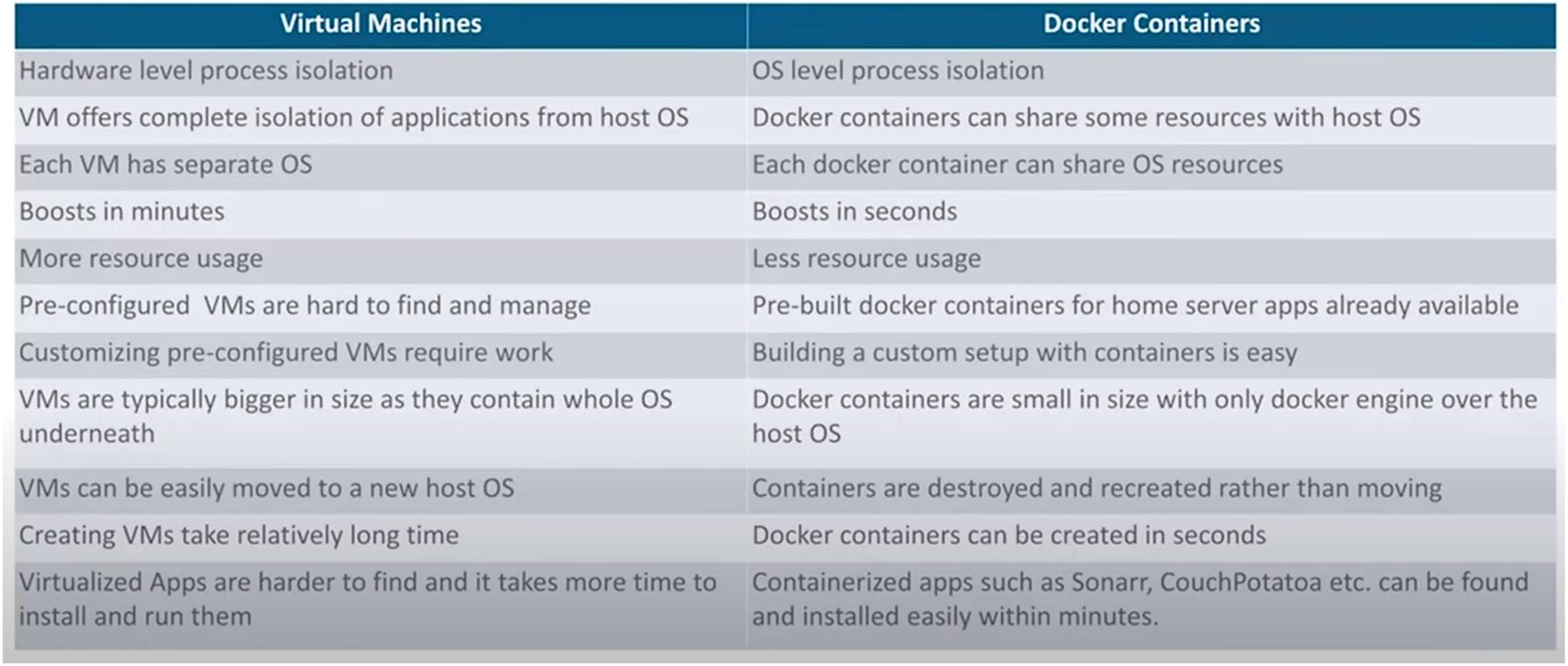
Difference between Docker Containers and Virtual Machines

1. Docker Containers

- Docker Containers contain binaries, libraries, and configuration files along with the application itself.
- They don’t contain a guest OS for each container and rely on the underlying OS kernel, which makes the containers lightweight.
- Containers share resources with other containers in the same host OS and provide OS-level process isolation.

2. Virtual Machines

- Virtual Machines (VMs) run on Hypervisors, which allow multiple Virtual Machines to run on a single machine along with its own operating system.
- Each VM has its own copy of an operating system along with the application and necessary binaries, which makes it significantly larger and it requires more resources.
- They provide Hardware-level process isolation and are slow to boot.



Virtual Machines	Docker Containers
Hardware level process isolation	OS level process isolation
VM offers complete isolation of applications from host OS	Docker containers can share some resources with host OS
Each VM has separate OS	Each docker container can share OS resources
Boosts in minutes	Boosts in seconds
More resource usage	Less resource usage
Pre-configured VMs are hard to find and manage	Pre-built docker containers for home server apps already available
Customizing pre-configured VMs require work	Building a custom setup with containers is easy
VMs are typically bigger in size as they contain whole OS underneath	Docker containers are small in size with only docker engine over the host OS
VMs can be easily moved to a new host OS	Containers are destroyed and recreated rather than moving
Creating VMs take relatively long time	Docker containers can be created in seconds
Virtualized Apps are harder to find and it takes more time to install and run them	Containerized apps such as Sonarr, CouchPotatoa etc. can be found and installed easily within minutes.

Hypervisor vs. Docker

- Functioning Mechanism
  - The most significant difference between hypervisors and Dockers is the way they boot up and consume resources.
  - Hypervisors are of two types – the bare metal works directly on the hardware while type two hypervisor works on top of the operating system.
  - Docker, on the other hand, works on the host kernel itself. Hence, it does not allow the user to create multiple instances of operating systems.
  - Instead, they create containers that act as virtual application environments for the user to work on.

Hypervisor vs. Docker

- Number of Application Instances Supported
  - A hypervisor allows the users to generate multiple instances of complete operating systems.
  - Dockers can run multiple applications or multiple instances of a single application. It does this with containers.
- Memory Requirement
  - Hypervisors enable users to run multiple instances of complete operating systems. This makes them resource hungry.
  - They need dedicated resources for any particular instance among the shared hardware which the hypervisor allocates during boot.
  - Dockers, however, do not have any such requirements. One can create as many containers as needed.
  - Based on the application requirement and availability of processing power, the Docker provides it to the containers.

Hypervisor vs. Docker

- Boot-Time
  - As Dockers do not require such resource allocations for creating containers, they can be created quickly to get started.
  - One of the primary reasons why the use of Dockers and containers is gaining traction is their capability to get started in seconds.
  - A hypervisor might consume up to a minute to boot the OS and get up and running.
  - Docker can create containers in seconds, and users can get started in no time.

Hypervisor vs. Docker

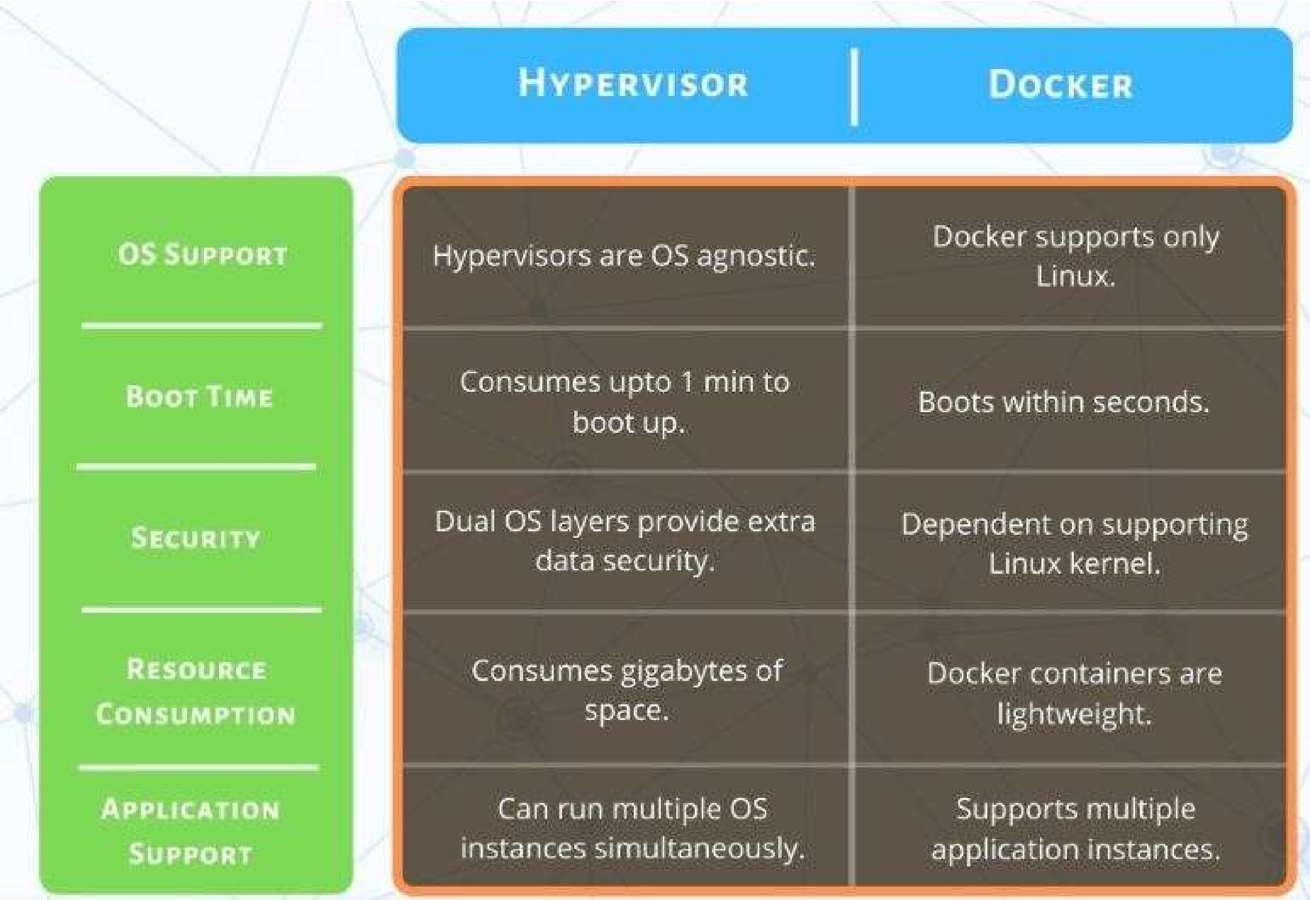
- OS Support
  - Hypervisors are OS agnostic. They can run across Windows, Mac, and Linux.
  - Dockers, on the other hand, are limited to Linux only. That, however, is not a deterrent for Dockers since Linux is a strong eco-system. Many major players are entering into the Dockers’ fray

- Security
  - Hypervisors are much more secure since the additional layer helps keep data safe.
  - One of the major differences between the two is the capability to run operating systems or rather run on operating systems.

Hypervisor vs. Docker

- Architecture Structure
  - If we consider both hypervisor and Docker’s architecture, we can notice that the Docker engine sits right on top of the host OS.
  - It only creates instances of the application and libraries.
  - Hypervisor though, has the host OS and then also has the guest OS further. This creates two layers of the OS that are running on the hardware.
  - If you are to run a portable program and want to run multiple instances of it, then containers are the best way to go. Hence you can benefit significantly with a Docker.
  - Dockers help with the agile way of working. Within each container, different sections of the program can be developed and tested.
  - In the end, all containers can be combined into a single program. Hypervisors do not provide such capability.

Hypervisor vs. Docker



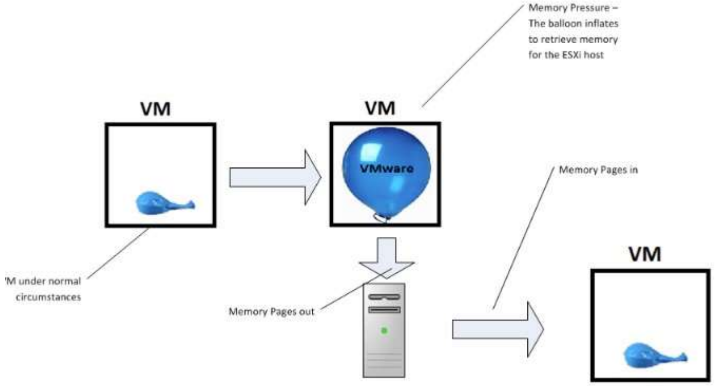
	HYPERVISOR	DOCKER
OS SUPPORT	Hypervisors are OS agnostic.	Docker supports only Linux.
BOOT TIME	Consumes upto 1 min to boot up.	Boots within seconds.
SECURITY	Dual OS layers provide extra data security.	Dependent on supporting Linux kernel.
RESOURCE CONSUMPTION	Consumes gigabytes of space.	Docker containers are lightweight.
APPLICATION SUPPORT	Can run multiple OS instances simultaneously.	Supports multiple application instances.

Hypervisor Memory Optimization

- Memory management strategies, such as memory ballooning, are essential components of a properly functioning system. Without them, admins risk resource contention and inadequate system performance. But admins should optimize memory cautiously because some tactics have issues, such as prolonged downtime.
- To optimize memory resources, admins can either select the exact amount of memory required to run a VM or enable system features to improve memory use for VMs with fluctuating resource requirements. Too little memory and the system uses page files, which can harm performance. Admins risk resource contention if they allocate too much memory.
- Admins can use memory ballooning, dynamic allocation, paging, overcommitment, mirroring, swapping and transparent page sharing (TPS) to govern and distribute resources.

Hypervisor Memory Optimization

- **Essential memory management strategies to optimize virtual memory**
- **Memory ballooning**
- Memory ballooning lets a physical host temporarily assign unused memory from one guest VM to another with a balloon driver. A balloon driver resides within each VM and locks any unused memory. The balloon driver then communicates with the hypervisor, which allocates that memory to other VMs. But memory ballooning can introduce performance issues if VMs no longer have the amount of memory they require.
- **Dynamic memory allocation**
- Dynamic memory allocation lets admins automatically assign memory to VMs. VMs claim the exact amount of memory they require from a physical resource pool. The hypervisor then monitors and redistributes memory according to fluctuating workload demands. Admins can overprovision memory with dynamic allocation, which enables the hypervisor to use paging or swapping. But overprovisioning can lead to low physical memory and inefficient performance.


Memory Pressure – The balloon inflates to retrieve memory for the ESXi host
VM
VM
VMware
Memory Pages in
VM
'M under normal circumstances
Memory Pages out

Hypervisor Memory Optimization

- **Essential memory management strategies to optimize virtual memory**
- **Memory paging**
- Memory paging occurs when VM memory runs low and the hypervisor temporarily transfers data from the host's memory to a hard disk or solid-state drive (SSD). Sections of the hard disk or SSD known as page files serve as an extension of the main memory. Paging ensures VMs don't run out of memory and crash. But paging can harm application performance; use it only when other memory management strategies aren't available.
- **Memory overcommitment**
- Admins can use memory overcommitment as both a management technique and an operating condition. As a technique, it allocates more memory to VMs than the total physical memory available. As an operating condition, overcommitment actively consumes more physical memory than what's available. Overcommitment can improve memory usage, but it can also cause the hypervisor to swap files with the storage device to reallocate memory, which can harm VM performance.
- **Memory mirroring**
- Memory mirroring divides a host's physical memory into two separate channels. The host then copies one of the channels to the other to create a copy, which provides fault tolerance.
- **Transparent page sharing**
- TPS condenses identical pages into a single page. Multiple VMs that run the same OS often have identical memory pages. TPS calculates hash values for each page to verify which pages are identical and consolidates them into a single page. TPS helps consolidate memory, but it also takes longer to execute and introduces security vulnerabilities such as unauthorized access to sensitive data.

Hardening the virtualization layers

- In computer security, hardening is usually the process of securing a system by reducing its surface of vulnerability, which is larger when a system performs more functions; in principle a single-function system is more secure than a multipurpose one.
- Reducing available ways of attack typically includes changing default passwords, the removal of unnecessary software, unnecessary usernames or logins, and the disabling or removal of unnecessary services.
- There are two aspects of hardening the virtualization: one involves providing physical hardware to virtual machines securely, while the other involves providing virtual hardware securely.

Hardening the virtualization layers

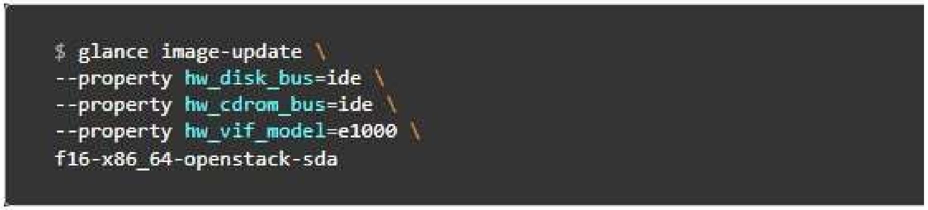
- Physical hardware – PCI passthrough
- One of the most common scenarios is the need to access from virtual machines to video cards and GPUs for high performance Compute Unified Device Architecture (CUDA). A lot of hypervisors give you this kind of capability, but it brings two possible security risks.
- Direct Memory Access (DMA) is a feature that allows many hardware devices to access the machine RAM directly and without any control. This feature allows the device to have a huge latency reduction in the read and write operations, so many low-latency devices, such as video cards
- However, an instance should not be given arbitrary physical memory access because this would give it full view of both the host system and other instances running on the same node. Hardware vendors use an input/output memory management unit (IOMMU) to manage DMA access in these situations. You should confirm that the hypervisor is configured to use this hardware feature.
- A hardware infection occurs when an instance makes a malicious modification to the firmware or some other part of a device. As this device is used by other instances or the host OS, the malicious code can spread into those systems. The end result is that one instance can run code outside of its security zone. This is a significant breach as it is harder to reset the state of physical hardware than virtual hardware, and can lead to additional exposure such as access to the management network.
- Due to the risk and complexities associated with PCI passthrough, it should be disabled by default. If enabled for a specific need, you will need to have appropriate processes in place to help ensure the hardware is clean before reuse.

Hardening the virtualization layers

- Virtual hardware (QEMU)
- When running a virtual machine, virtual hardware is a software layer that provides the hardware interface for the virtual machine. Instances use this functionality to provide network, storage, video, and other devices that may be needed.
- The major open source hypervisors use QEMU for this functionality. While QEMU fills an important need for
- virtualization platforms, it has proven to be a very challenging software project to write and maintain. Much of the functionality in QEMU is implemented with low-level code that is difficult for most developers to comprehend.
- It is important to take proactive steps to harden QEMU. We recommend three specific steps:
- Minimizing the code base.
- Using compiler hardening.
- Using mandatory access controls such as sVirt, SELinux, or AppArmor.

Hardening the virtualization layers

- Minimizing the QEMU code base
- We recommend minimizing the QEMU code base by removing unused components from the system. QEMU provides support for many different virtual hardware devices, however only a small number of devices are needed for a given instance. The most common hardware devices are the virtio devices. Some legacy instances will need access to specific hardware, which can be specified using glance metadata:



```
$ glance image-update \
--property hw_disk_bus=ide \
--property hw_cdrom_bus=ide \
--property hw_vif_model=e1000 \
f16-x86_64-openstack-sda
```

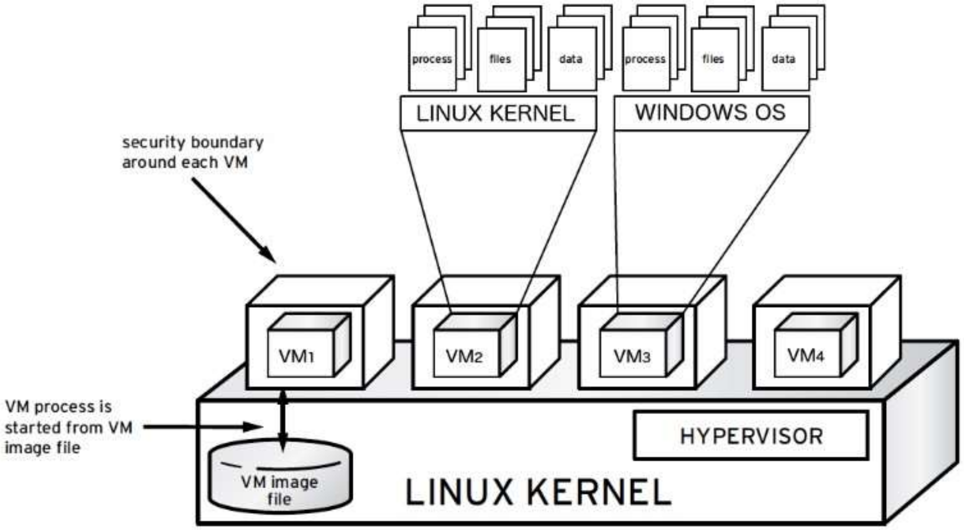
Hardening the virtualization layers

- Secure Encrypted Virtualization¶
- Secure Encrypted Virtualization (SEV) is a technology from AMD which enables the memory for a VM to be encrypted with a key unique to the VM. SEV is available in the Train release as a technical preview with KVM guests on certain AMD-based machines for the purpose of evaluating the technology.
- Mandatory Access Control

Hardening the virtualization layers

sVirt: SELinux and virtualization

With unique kernel-level architecture and National Security Agency (NSA) developed security mechanisms, KVM provides foundational isolation technologies for multi-tenancy. With developmental origins dating back to 2002, the Secure Virtualization (sVirt) technology is the application of SELinux against modern day virtualization. SELinux, which was designed to apply separation control based upon labels, has been extended to provide isolation between virtual machine processes, devices, data files and system processes acting upon their behalf.


process
files
data
process
files
data
LINUX KERNEL
WINDOWS OS
security boundary around each VM
VM1
VM2
VM3
VM4
VM process is started from VM image file
HYPERVISOR
VM image file
LINUX KERNEL