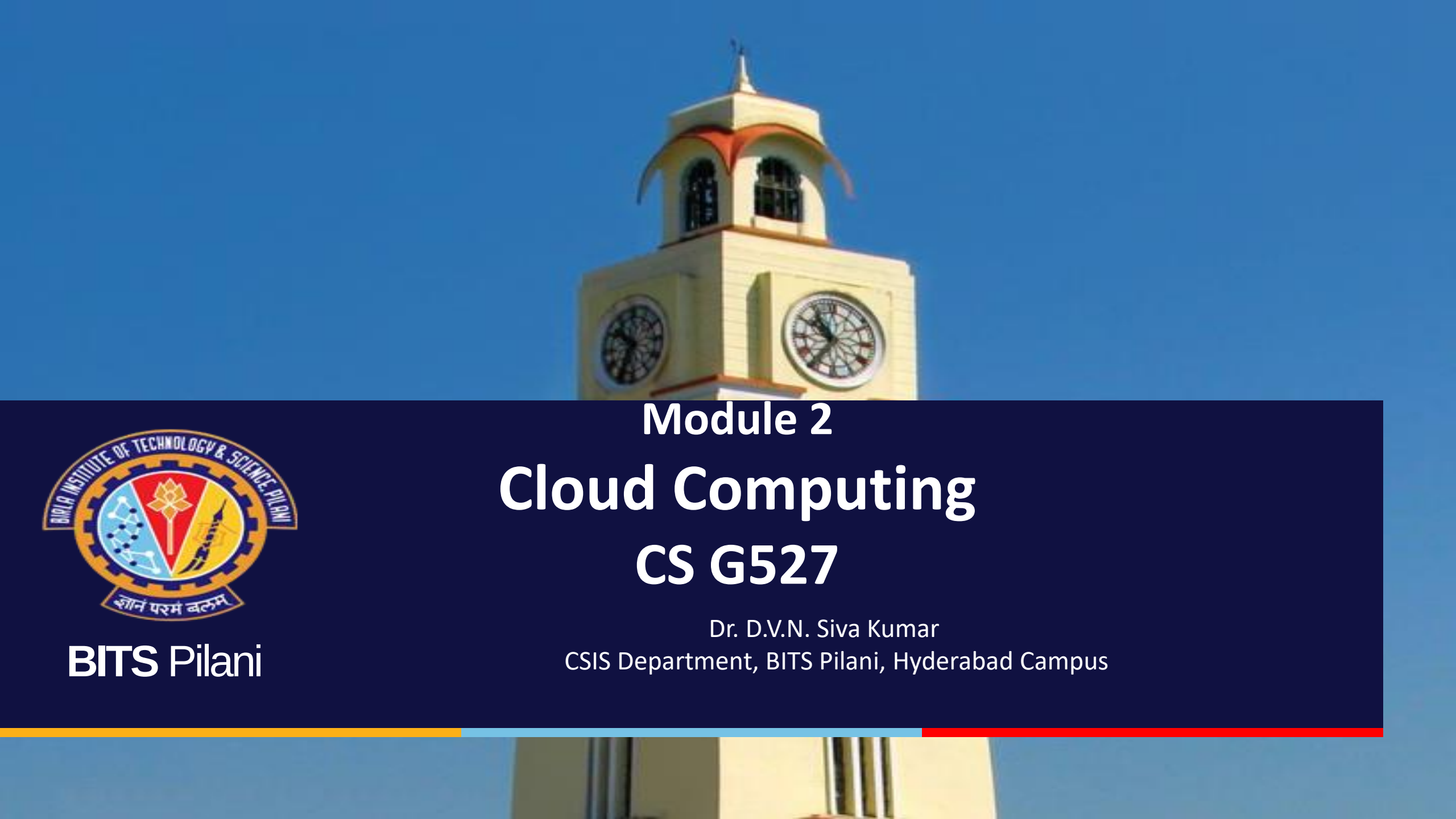


Module 2

Cloud Computing

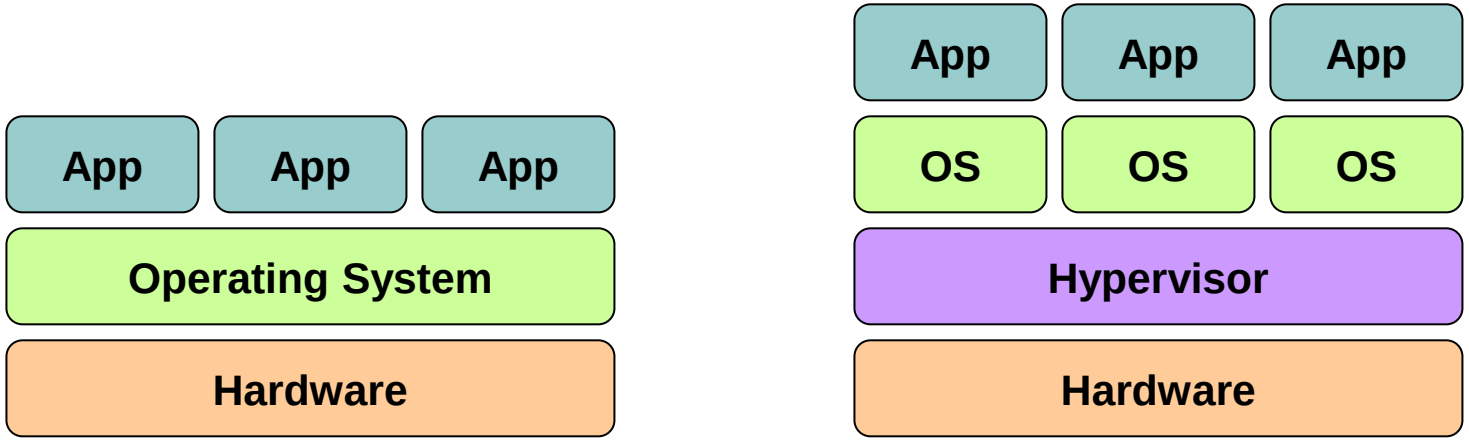
CS G527

Dr. D.V.N. Siva Kumar
CSIS Department, BITS Pilani, Hyderabad Campus

BITS Pilani

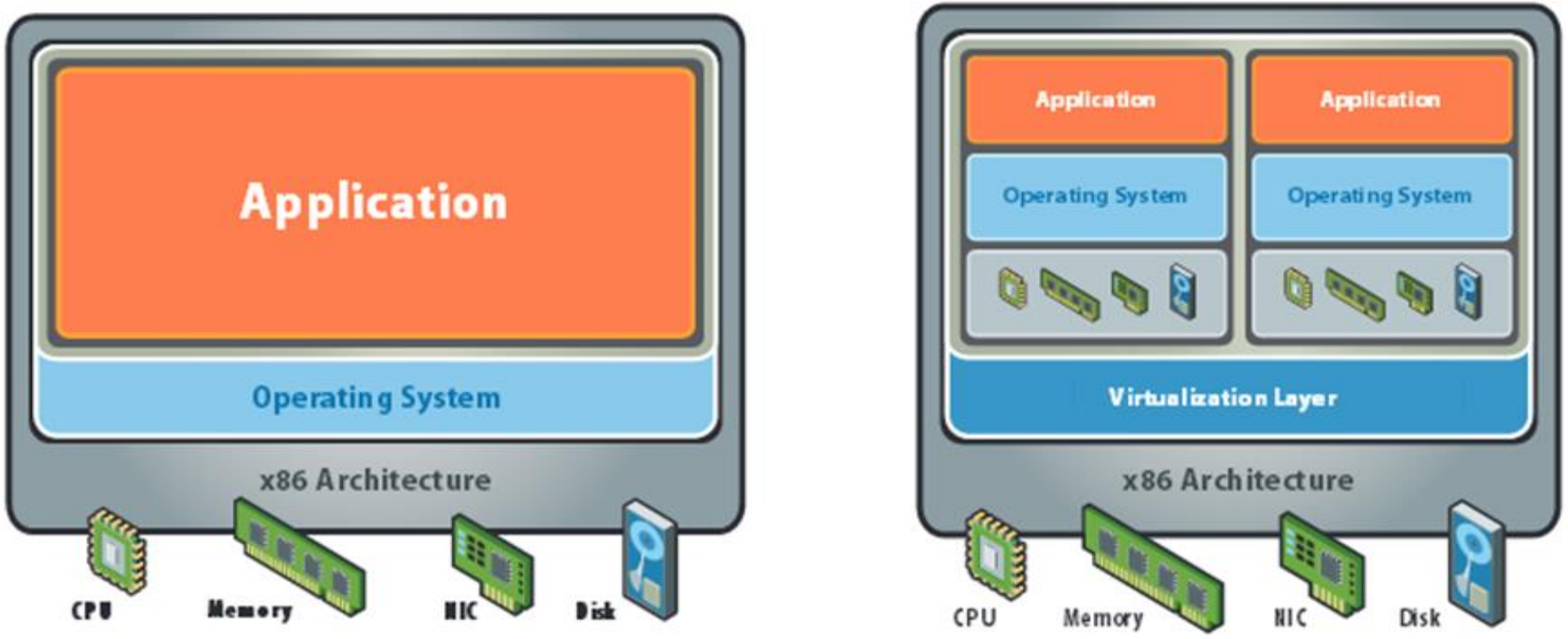
Cloud Infrastructures

Key Technology is Virtualization



Virtualization plays an important role as an enabling technology for datacentre implementation by abstracting compute, network, and storage service platforms from the underlying physical hardware

What is Virtualization?



Virtualization is the **"creation of a virtual (rather than actual) version of something,** such as **a server, a desktop, a storage device, an operating system or network resources".**

In other words, Virtualization is a technique, which allows to share a single physical instance of a resource or an application among multiple customers and organizations. It does by assigning a logical name to a physical storage and providing a pointer to that physical resource when demanded.

Key Properties of Virtualization

Partitioning
- Run multiple operating systems on one physical machine.
- Divide system resources between virtual machines.

Isolation
- Provide fault and security isolation at the hardware level.
- Preserve performance with advanced resource controls.

Encapsulation
- Save the entire state of a virtual machine to files.
- Move and copy virtual machines as easily as moving and copying files.

Hardware Independence
- Provision or migrate any virtual machine to any physical server.

Benefits of Virtualization

- More flexible and efficient allocation of resources.
- Improves Security
- Enhance development productivity.
- It lowers the cost of IT infrastructure.
- Remote access and rapid scalability.
- High availability and disaster recovery.
- Pay per use of the IT infrastructure on demand.
- Enables running multiple operating systems.

Goals of Virtualization

1. How to virtualize CPU?.
2. How to virtualize Memory?.
3. How to virtualize I/O?.

Virtualization Providers

- **Microsoft:** Virtual PC, Virtual Server 2005, Hyper-V
- **VMWare:** Vmware Workstation, Vmware Server
- **Oracle:** Oracle VM VirtualBox

Virtual Machine

- Virtual Machine is a result of Virtualization, which typically refers to the creation of virtual machine that can virtualize all of the hardware resources, including **processors, memory, storage, and network connectivity.**
- **VM can be treated as tightly isolated software container with an operating system and application inside**.
- Each self-contained VM is completely independent.
- Putting multiple VMs on a single computer enables several operating systems and applications to run on just one physical server, or “host.”

Hypervisor (Virtual Machine Monitor)

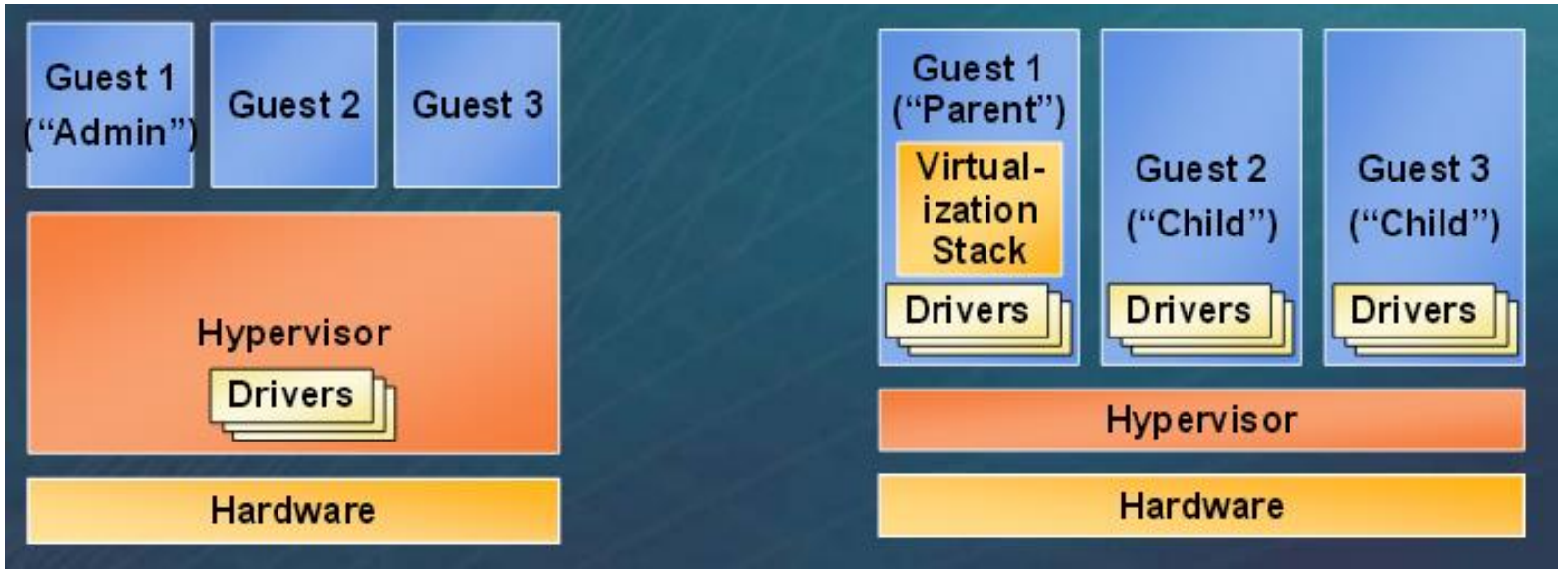
- The software responsible for system virtualization is called the **Virtual machine Monitor (VMM) or Hypervisor.**
- The software is used in two ways (three different structures):
  - **Bare-Metal or Native Hypervisors:** Run directly on the hardware. Examples are VMWare ESX server and KVM
  - **Hosted Hypervisors:** Run on top of existing OS and leverage the features of the underlying OS. Examples are VMWare ESX server,
  - **Hybrid Hypervisors**: Run directly on the hardware, but leverage the features of an existing OS running as a guest. Examples are Xen and Microsoft's Hyper-V

How does Virtualization work?

- Hypervisor separate the physical resources from the virtual environments—the things that need those resources.
- Hypervisors take our physical resources and divide them up so that virtual environments can use them.
- Resources are partitioned as needed from the physical environment to the many virtual environments.
- Users interact with and run computations within the virtual environment (typically called a guest machine or virtual machine).
- When the virtual environment is running and a user or program issues an instruction that requires additional resources from the physical environment, the hypervisor relays the request to the physical system and caches the changes—which all happens at close to native speed (subjected to the type of hypervisor)

Hypervisor
Monolithic versus Microkernelized

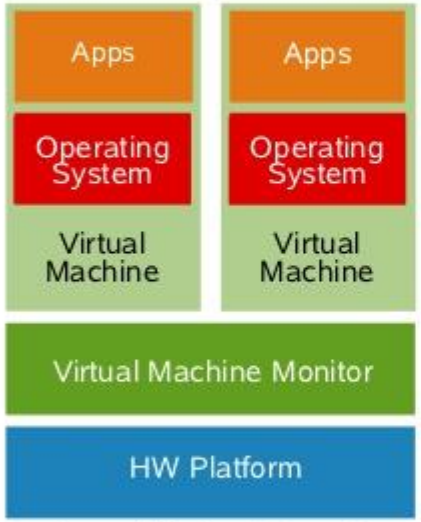
- **Monolithic hypervisor**
  - Simpler than a modern kernel, but still complex
  - Contains its own drivers model
- **Microkernelized hypervisor**
  - Simple partitioning functionality
  - Increase reliability and minimize lowest level of the TCB
  - No third-party code
  - Drivers run within guests



Two Popular Approaches for Server Virtualization

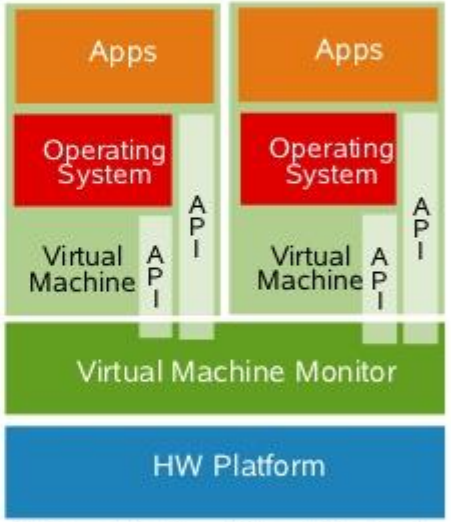
Full & Paravirtualization Overview

Full Virtualization



Runtime modification of Guest OS: VMM manages the conflict, then returns to OS

Paravirtualization



Static modification of Guest OS prior to runtime: Privileged instruction calls are exchanged with API functions provided by the VMM

- Almost no performance degradation
- Significant scalability

Full Virtualization

❑ **Full virtualization**

- In its basic form known as “full virtualization” the hypervisor provides a fully emulated machine in which an operating system can run. **VMWare** is a good example.
- Full virtualization is achieved by using a combination of **binary translation** and **direct execution**. In this type of virtualization, hypervisor translates all privileged instructions (from machine code of guest OS to machine code of host OS) on the fly and caches the results for future use, while user level instructions run unmodified at native speed.
- Guest OS doesn’t see that is used in an emulated environment.
- The biggest advantage to this approach is its flexibility: one could run a **RISC-based OS** as a guest on an Intel-based host.
- While this is an obvious approach, there are significant performance problems in trying to emulate a complete set of hardware in software.

Advantages and Drawbacks of Full Virtualization

Advantages:

- Isolates VMs from each host OS and from each other.
- Controls individual VM access to system resources, preventing an unstable VM from impacting system performance.
- Total VM portability: VMs have the ability to transparently move between hosts with dissimilar hardware without any problems. A VM running on a Dell Server can be relocated to a HP Server.

Drawbacks:

- Performance due to binary translation of specific privileged instructions.

ParaVirtualization / OS Assisted Virtualization

❑**Paravirtualization**

- "Paravirtualization," found in the **XenSource**, open source Xen product.
- **Paravirtualization** uses slightly altered versions of the operating system which allows access to the hardware resources directly as managed by the hypervisor.
- This typically involves replacing any privileged operations that will only run in CPU with calls to the hypervisor **(known as *hypercalls*)**. The hypervisor in turn performs the task on behalf of the guest kernel and also provides hypercall interfaces for other critical kernel operations such as memory management, interrupt handling, etc.
- Here, the guest OS is modified version and thus run kernel level specific instructions.

ParaVirtualization (Cont…)

- In order to retain flexibility, the guest OS is not tied to its host OS. Drastically different operating systems can be running in a hypervisor at the same time, just as they can under full virtualization.
- Privileged instruction translation by the VMM (Hypervisor) is not necessary.
- Guest OS uses a specialized API to talk to the VMM and execute the privileged instructions.
- In this way, paravirtualization can be thought of as a low-overhead full virtualization.

Advantages and Drawbacks of ParaVirtualization

Advantages:

- Significant performance improvement.

Drawbacks:

- Requires the modified Guest OSes.
- Guest OS could expose the host to security threats due to the direct communication line.

Could you give me an example scenario where one application requires Windows OS?.

Could you give me an example scenario where one application requires Windows OS?.

Ans: An application that is developed using .NET requires Windows OS and

What are the few examples of Hypervisors that are available for use.?

Why should a separate computer system is required for hosting an application?

What are the few examples of Hypervisors that are available for use.?

Ans: Oracle Virtual Box and Hyper-V, etc.

Why should a separate computer system is required for hosting an application?

Ans:

- To ensure speed, scalability, and reliability.
- To identify the reasons for the problems encountered while accessing application, etc.

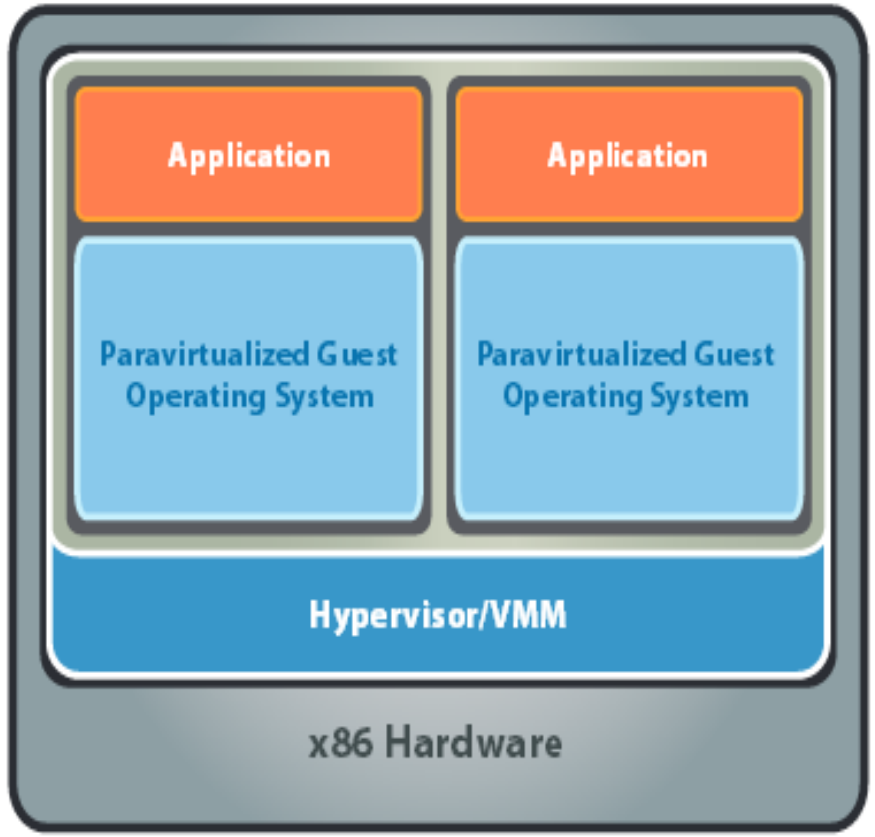
Virtualization Versus Cloud Computing
?

Virtualization is software that makes computing environments independent of physical infrastructure, while cloud computing is a service that delivers shared computing resources (software and/or data) on demand via the Internet.

Note: organizations begin by virtualizing their servers and then moving to cloud computing for even greater agility and self-service.

x86 Hardware Virtualization

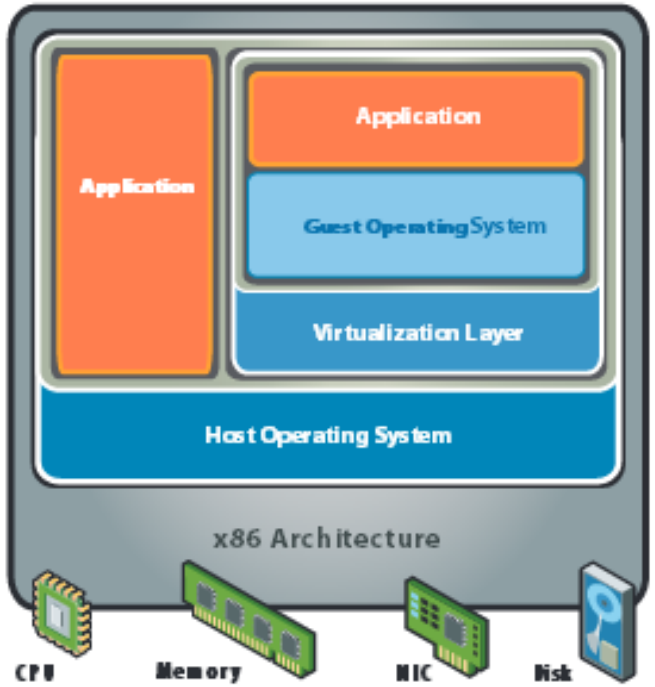
- The latest generation of x86-based systems feature processors with 64-bit extensions supporting very large memory capacities.
- This enhances their ability to host large, memory-intensive applications, as well as allowing many more virtual machines to be hosted by a physical server deployed within a virtual infra-structure.
- The continual decrease in memory costs will further accelerate this trend.



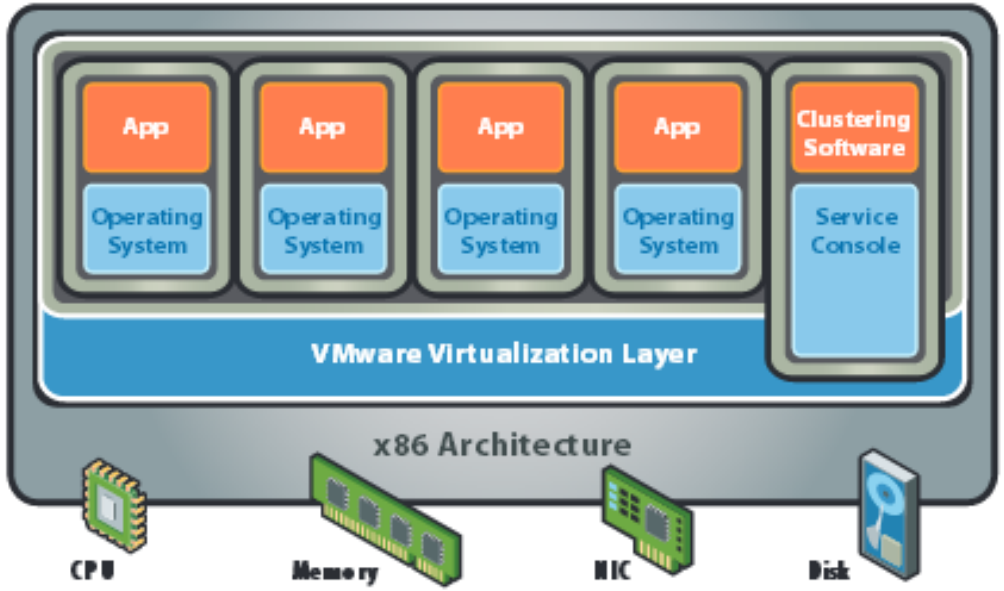
x86 Hardware Virtualization

- For Industry-standard x86 systems, the two approaches typically used with software-based partitioning are
  - Type 1 Hypervisor (also called **bare metal or native**)
  - Type 2 Hypervisor (also known **as hosted hypervisors**)

x86 Hardware Virtualization



Hosted Architecture



Bare-Metal (Hypervisor) Architecture

Type 1 Hypervisor (also called as Bare-metal hypervisor)

- A bare-metal hypervisor (Type 1) is a layer of software we install directly on top of a physical server and its underlying hardware.
- There is no software or any operating system in between, hence the name *bare-metal hypervisor*.
- A Type 1 hypervisor is proven in providing excellent performance and stability since it does not run inside Windows or any other operating system.
- Type 1 hypervisors are mainly found in enterprise environments.
- It offers high performance since it has direct access to the underlying hardware (and no other Operating Systems and device drivers to content with). Due to this, Type 1 Hypervisor is considered to be the best performing and most efficient.
- Two examples of Type 1 Vendors: VMware vSphere, KVM (Kernel-Based Virtual Machine), Citrix Hypervisor (formerly known as Xen Server)

Example Applications of Type 1 Hypervisor

- When High-performance computing is required, where any overhead should be avoided, and hardware components are selected and tuned for maximum performance: e.g., computing clusters for silicon chip design.
- Imagine building a medical imaging device that needs to process huge medical data in real-time and also simultaneously provide an interactive GUI to users. Both the real-time OS and general purpose can be run simultaneously.

Type 2 Hypervisor (also called as Hosted Hypervisors)

- This type of hypervisor runs inside of an operating system of a physical host machine.
- A **hosted approach** provides partitioning services on top of a standard operating system and supports the broadest range of hardware configurations.
- We call type 2 hypervisors – ***hosted hypervisors*****.** As opposed to type 1 hypervisors that run directly on the hardware, **hosted hypervisors have one software layer underneath**. In this case we have:

  -A physical machine.

  -An operating system installed on the hardware (Windows, Linux, macOS).

  -A type 2 hypervisor software within that operating system.

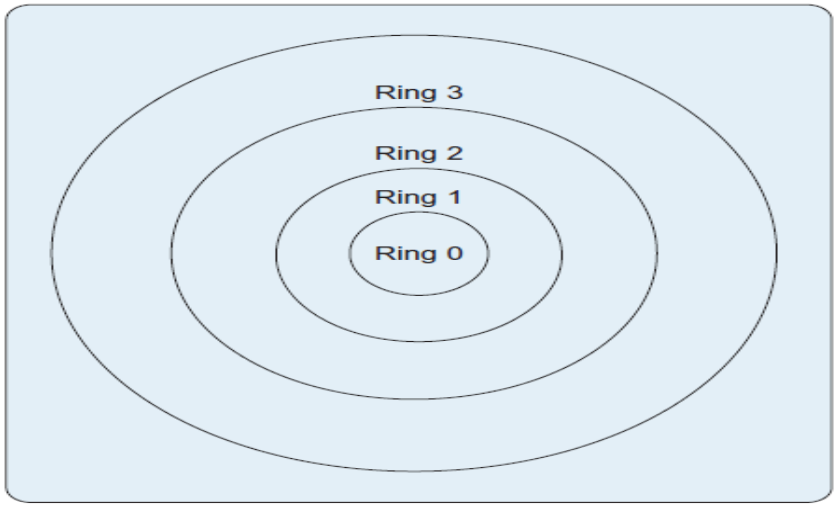
  -The actual instances of guest virtual machines.
- Type 2 hypervisors are typically found in environments with a small number of servers.
- Type 2 hypervisors are less efficient than Type 1 since they cannot directly communicate with the hardware due to which they are less efficient than the type 1 hypervisors.

Example Applications of Type 2 Hypervisor

- It is used during the development process. It could be used for testing alpha and beta software as each individual VM is isolated from each other. If one VM (may be a VM running beta software) corrupts the operating system, it will not affect any other VM OS and the host OS.
- This type of hypervisors enables running applications written for several different operating systems on one computer. One VM may be used for running Windows applications and another VM for running Linux applications. In all such cases, type-2 hypervisors are more suitable.

Virtualization Techniques

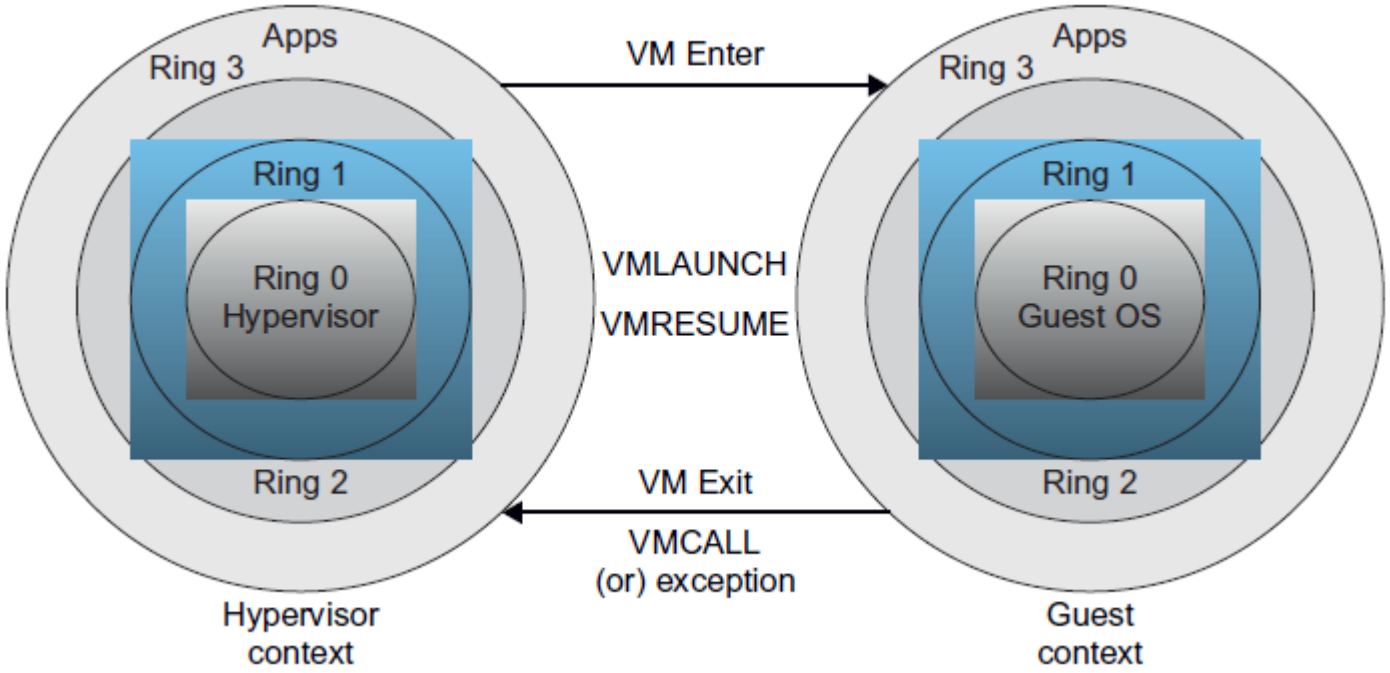
1. Trap and Emulate
2. Binary Translation
3. Paravirtualization
4. Hardware Assisted (Intel and AMD created new processor extensions to support virtualization in the hardware.)



X86 Protection Rings

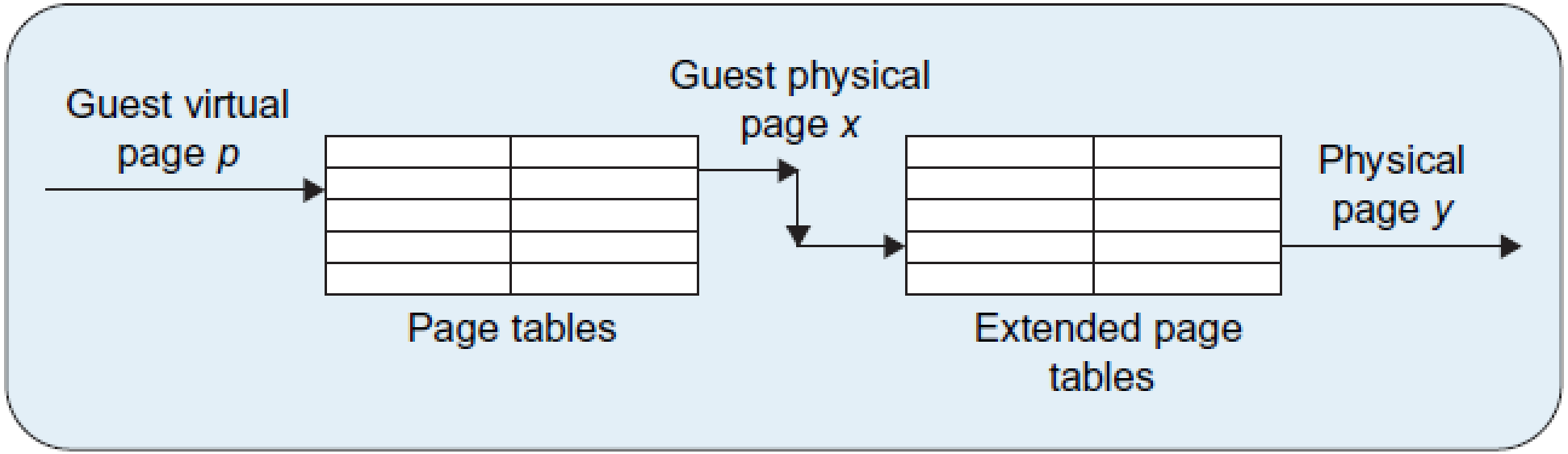
Hardware Support for Processor Virtualization

- VT-x, called VanderPool, represents Intel's technology for virtualization on x86 processors.
- **Two modes** for processor execution: VMX root operation and VMX non-root operation



Hardware Support for Memory Virtualization

Extended page tables.



Hardware Support for IO Virtualization

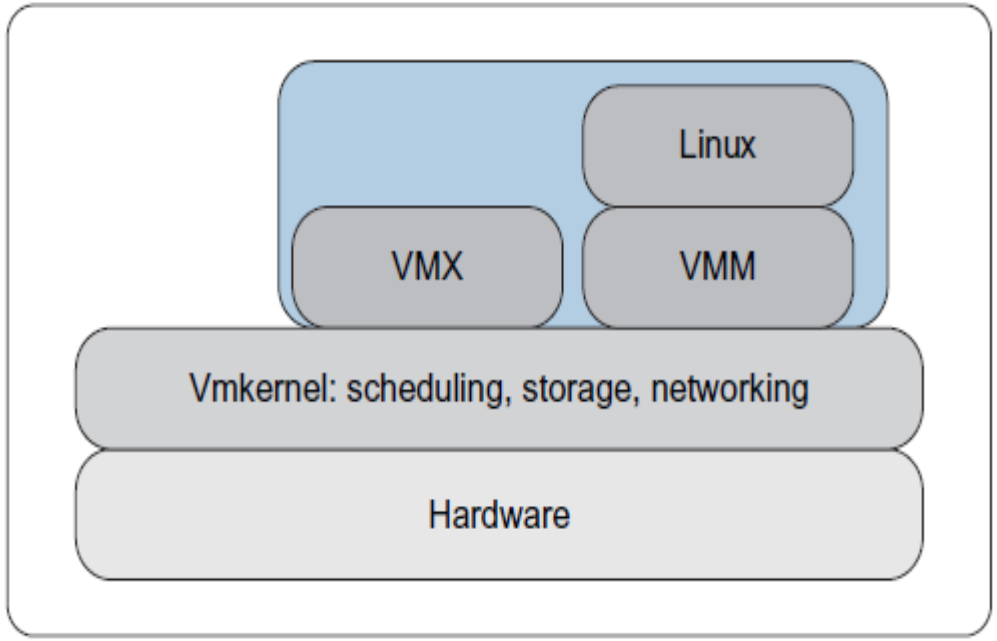
- Intel's VT-d technology can reduce the overhead of I/O virtualization
- It has **two components**, Interrupt Remapping and DMA Remapping
- **DMA remapping** is targeted at eliminating the need for hypervisors to translate guest virtual addresses in I/O commands.
- **Interrupt remapping** is a technology that allows the hypervisor to ensure that interrupts from I/O devices can be delivered directly to the appropriate guest.

Two Popular Hypervisors

1. Vmware
2. Xen

Note: In general, all hypervisors need some operating system-level components—such as a memory manager, process scheduler, input/output (I/O) stack, device drivers, security manager, a network stack, and more—to run VMs.

VMware

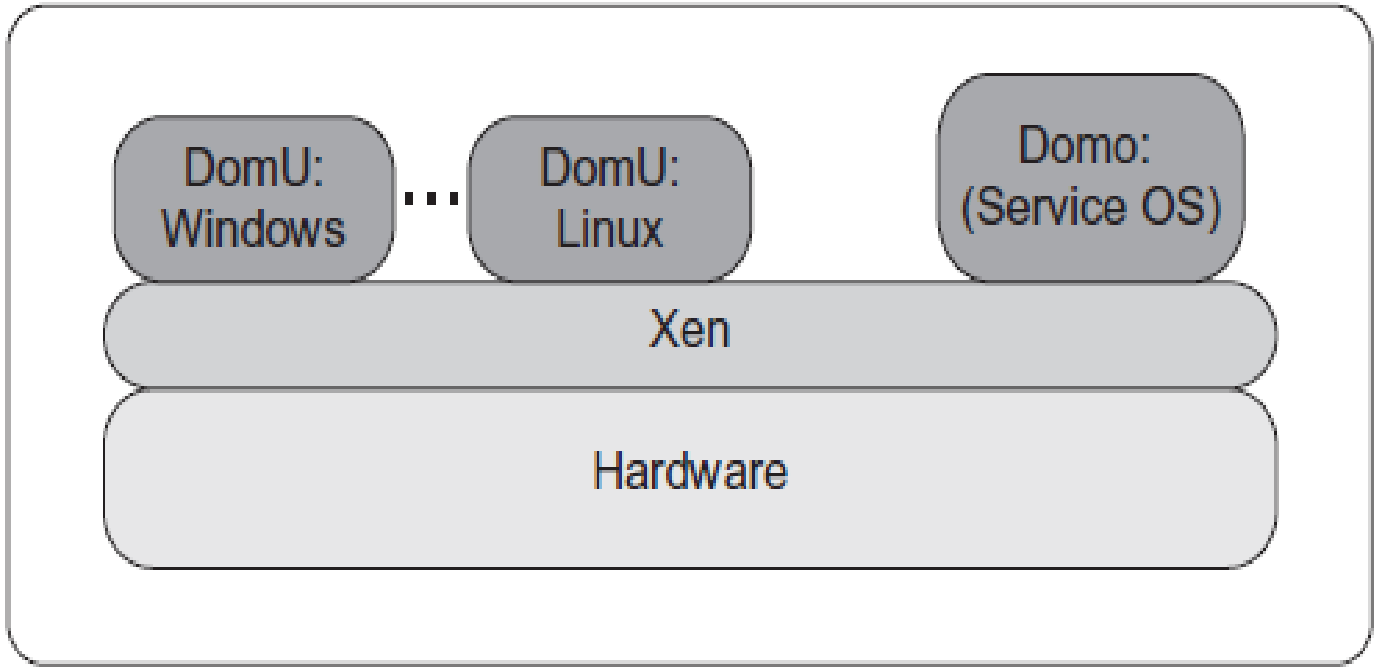


High-level architecture of VMware ESX 3i server

VM virtualization:

- For CPU virtualization, the VMM uses a combination of binary transalation and VT-x.
- VMM uses a paravirtualization for I/O virtulization.

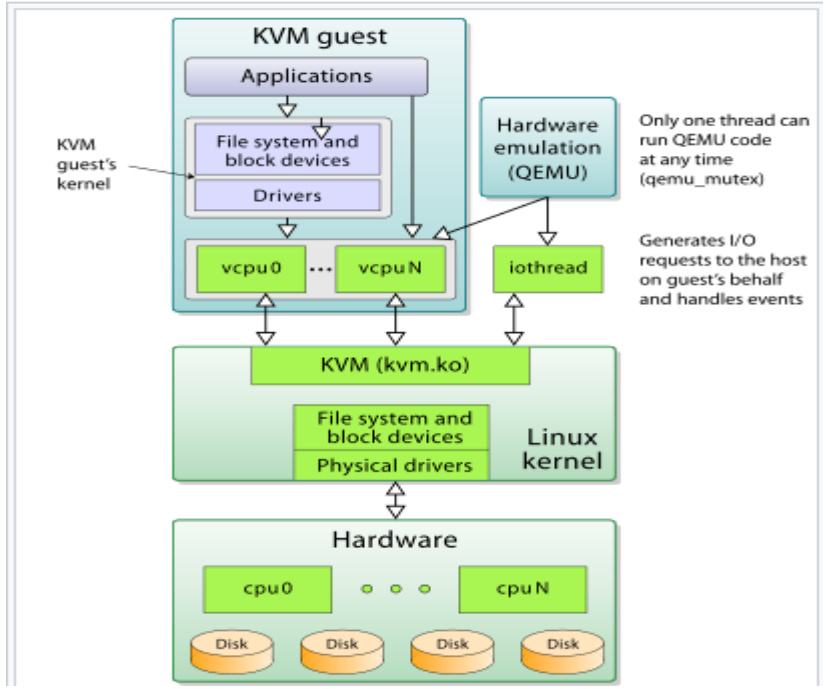
Xenserver Architecture



- Guest VMs are called Domains.
- Domain 0 is a specialy privileged VM that access to the physical hardware.
- It makes use of paravirtulization for I/O virtulization; (I/O requsts from any other non-domain 0 VM (called as DomU) are sent to **Dom0.**

KVM (Kernel-based Virtual Machine)

- KVM is an open source virtualization technology built into Linux
- KVM turns Linux into a **hypervisor** that allows a host machine to run multiple, isolated virtual environments called guests or virtual machines (VMs).
- KVM provides hardware-assisted virtualization for a wide variety of guest operating systems (Linux, Solaris, Windows, macOS, etc.)



Note: KVM can only be used on a processor with hardware virtualisation extensions such as Intel-VT or AMD-V

Ref: https://ubuntu.com/blog/kvm-hyphervisor
https://en.wikipedia.org/wiki/Kernel-based_Virtual_Machine

Advantages of Virtualization

- Security
- Reliability and Availability
- Cost
- Adoptability to workload variations
- Load balancing
- Support for legacy applications

Issues with Virtualization

- Software licensing
- IT training
- Hardware investment
- Interoperability between various vendors

Can we have a single VM on multiple systems?

Can you run multiple virtual machines at once?

Can we have a single VM on multiple systems?

Ans: No, Virtual machines run on a single OS.

Can you run multiple virtual machines at once?

Ans: Yes, you can run multiple virtual machines at once.

Few Questions

Could you give me an example of Hosted Architecture based Virtualization?

Could you give me an example of Bare-Metal (Hypervisor) Architecture based virtualization?

Few Questions

Could you give me an example of Hosted Architecture based virtualization?

Ans: Vmware Workstation and Oracle Virtual Box.

Could you give me an example of Bare-Metal (Hypervisor) Architecture?

Ans: VMware ESX Server employs a hypervisor architecture on certified hardware for data center class performance. **Openstack** also supports.

Which hypervisor does Google Compute Engine (GCP) use for creating VMs?.

Ans: KVM

Which hypervisor does AWS use for creating VMs?.

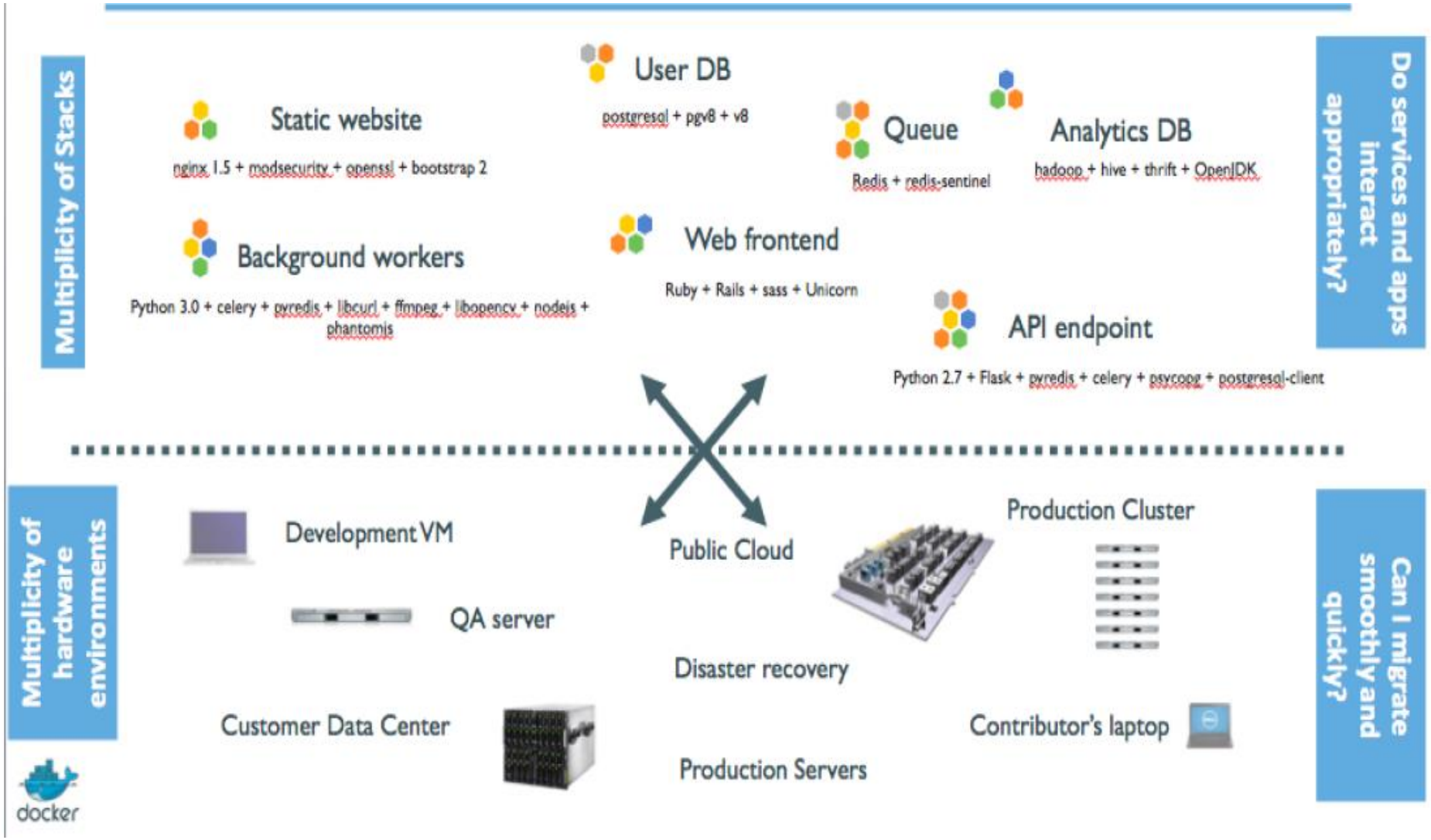
Ans: Modified verisions of KVM (Nitro)

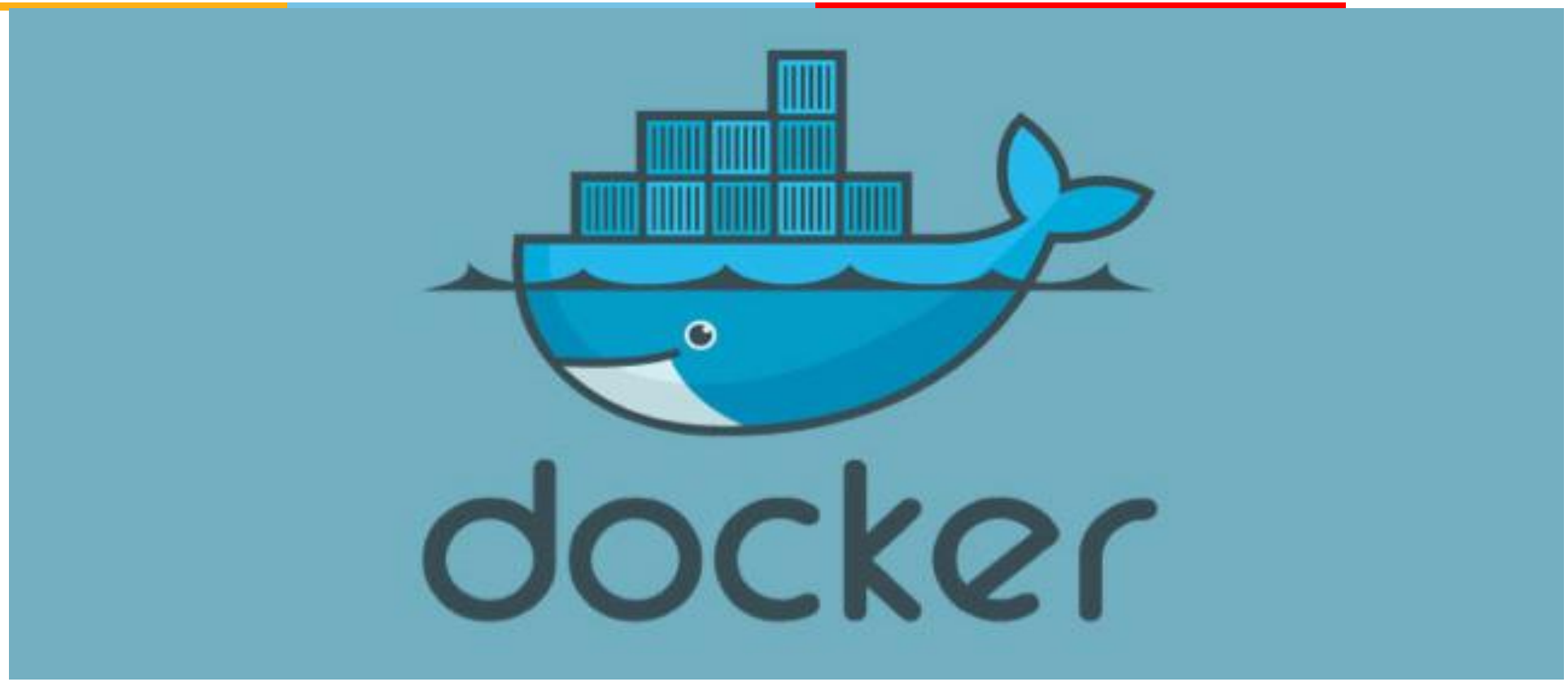
What problems we face with the current virtualization techniques?

Ans:

- Portability of one application from one environment to another across several devices.
- Challenges in Software build up (redeployment or reconfiguration or recompilation of the software whenever we want to ship one package from one environment to other.)
- More booting time for virtual machines to start.
- Possibility of more resource wastage.

Current Problem the Industry is facing





Reference: Getting Started with Docker by Author: Christopher M. Judd (For detailed information about dockers)

Why Dockers are required?

Why Dockers are required? (Cont…)

Docker is a software platform that allows you to build, test, and deploy applications quickly.

It is mainly for:

- Application Level Virtualization
- Build once, deploy anywhere and run any where.
- Better collaboration while development of applications.
- A single host can run several applications for proper utilization of resources.

Note1 : Docker enables developers to easily **pack, ship, and run any application as a lightweight, portable, self-sufficient container, which can run virtually anywhere.**

Note 2: The whole idea of Docker is for developers to easily develop applications, ship them into containers which can then be deployed anywhere.

Dockers

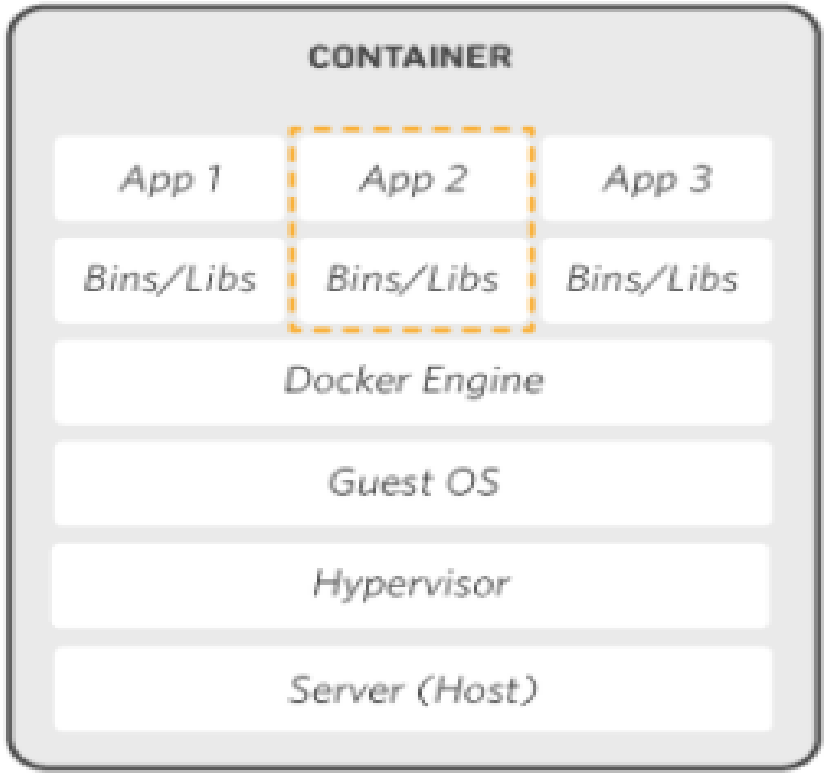
- It is an open-source project that automates the deployment of applications inside software containers.
- All applications have their own dependencies, which include both software and hardware resources.
- **Docker is a mechanism that helps in isolating the dependencies per each application by packing them into containers.**
- In terms of technology, it provides portability by running the same applications in different environments.
- Containers are scalable and safer to use and deploy as compared to regular approaches.

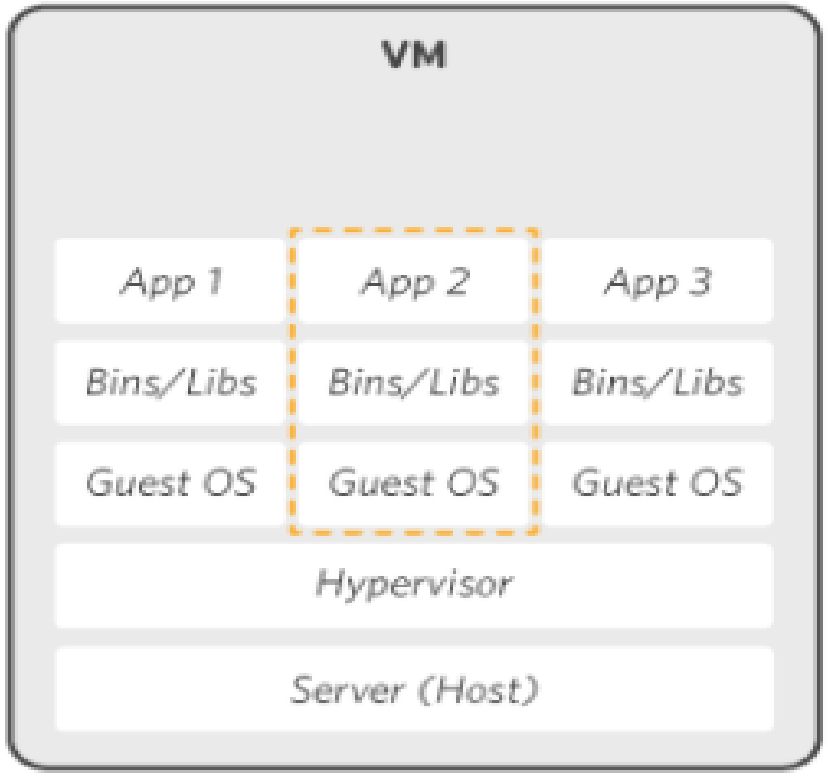
Note: The Dockers can also be called as container-based virtualization or Lightweight virtualization.

How are Docker Containers different from a Virtual Machine?

- Virtual machines have a full OS with its own memory management installed with the associated overhead of virtual device drivers.
- Docker containers are executed with the Docker engine rather than the hypervisor.
- Containers are therefore smaller than Virtual Machines and enable faster start up with better performance, less isolation and greater compatibility possible due to sharing of the host's kernel.

How are Docker Containers different from a Virtual Machine?





Ref: https://aws.amazon.com/docker/

Docker Architecture

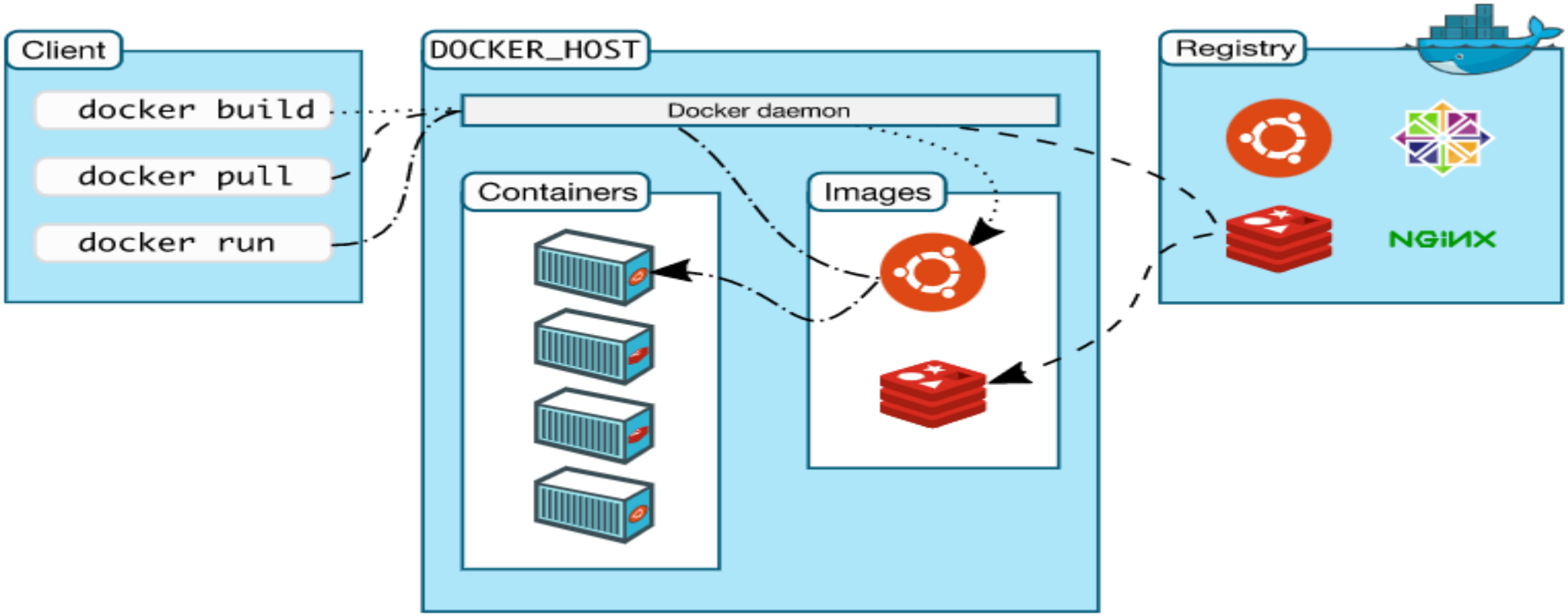


Image and Container

Image: An *image* is a read-only template with instructions for creating a Docker container. For example, build an image which is based on the ubuntu image, but install the Apache web server and our application, as well as the configuration details needed to make our application run.

Note: You might create your own images or you might only use those created by others and published in a registry.

Container:

- Docker provides the ability to package and run an application in a loosely isolated environment called a container.

- A container is a runnable instance of an image. You can create, start, stop, move, or delete a container using the Docker API or CLI.

-We can connect a container to one or more networks, attach storage to it, or even create a new image based on its current state.

Docker Components

Docker for Mac − To run Docker containers on the Mac OS.

Docker for Linux − To run Docker containers on the Linux OS.

Docker for Windows − To run Docker containers on the Windows OS.

Docker Engine − For building Docker images and creating Docker containers.

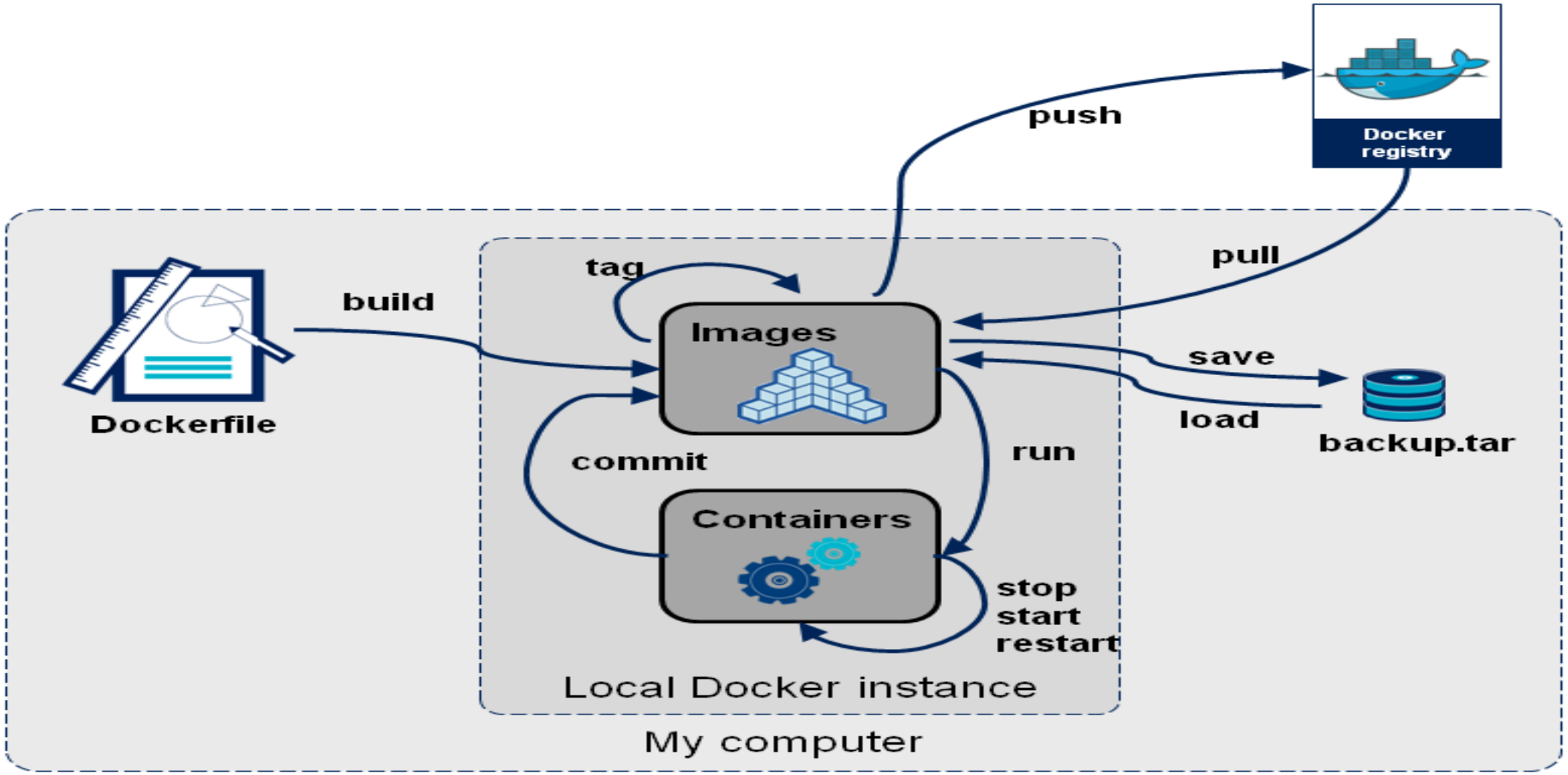
Docker Hub − It is the registry which is used to host various Docker images.

Docker Compose − This is used to define applications using multiple Docker containers.

Docker Features

- It has the ability to reduce the size of development by providing a smaller footprint of the operating system via containers.
- Can be deployed anywhere, on any physical machine (Mobile, System, Server), virtual machine and even in cloud.
- Higher scalability since containers are lightweight.

Docker Container Lifecycle ……



Docker Container Lifecycle ……

- The Life of a Container
  - Conception
    - **BUILD** an Image from a Dockerfile
  - Birth
    - **RUN** (create+start) a container
  - Reproduction
    - **COMMIT** (persist) a container to a new image
    - **RUN** a new container from an image
  - Sleep
    - **KILL** a running container
  - Wake
    - **START** a stopped container
  - Death
    - **RM** (delete) a stopped container
- Extinction
  - **RMI** a container image (delete image)

Dockerfile ……

- Like a Makefile (shell script with keywords)
- Extends from a Base Image
- Results in a new Docker Image
- Imperative, not Declarative
- A Docker file lists the steps needed to build an images
- docker build is used to run a Docker file
- Can define default command for docker run, ports to expose, etc

Example of a Dockerfile

file | 15 lines (11 sloc) | 0.475 kb | Open | Edit | Raw | Blame | History | Delete

```
1  FROM ubuntu:12.04
2
3  RUN apt-get update
4
5  # Make it easy to install PPA sources
6  RUN apt-get install -y python-software-properties
7
8  # Install Oracle's Java (Recommended for Hadoop)
9  # Auto-accept the license
10 RUN add-apt-repository -y ppa:webupd8team/java
11 RUN apt-get update
12 RUN echo oracle-java7-installer shared/accepted-oracle-license-v1-1 select true | sudo /usr/bin/debconf-set-selections
13 RUN apt-get -y install oracle-java7-installer
14 ENV JAVA_HOME /usr/lib/jvm/java-7-oracle
```

Differences between Virtual Machines and Docker Containers

	Virtual Machines	Docker Containers
Isolation Process Level	Hardware	Operating System
Operating System	Separated	Shared
Boot up time	Long	Short
Resources usage	More	Less
Pre-built Images	Hard to find and manage	Already available for home server
Customised preconfigured images	Hard to build	Easy to build
Size	Bigger because they contain whole OS underneath	Smaller with only docker engines over the host OS
Mobility	Easy to move to a new host OS	Destroyed and recreated instead of moving.
Creation time	Longer	Within seconds

Security concerns with Dockers

- Kernel exploitations
- Denial-of-service attacks
- Container breakouts
- Poisoned images

Reference: https://www.oreilly.com/content/five-security-concerns-when-using-docker/

When to use Virtualization?

When to use Containization?

Video Lectures of Demonstrating Hypervisors

- https://www.youtube.com/embed/phCWC7AgxYM?modestbranding=1&showsearch=0&autohide=1&showinfo=0&rel=0&frameborder=0 **(Demonstration of installing Ubuntu OS using hypervisor called "Oracle Virtual Box")**
- https://www.youtube.com/embed/EYqwMulUKWY?modestbranding=1&showsearch=0&autohide=1&showinfo=0&rel=0&frameborder=0 **(Demonstration of installing Ubuntu OS using another hypervisor called "Microsoft Hyper-V)**
- https://www.princeton.edu/~rblee/ELE572Papers/Fall04Readings/secureOS/popek_virtualizable.pdf (Formal Requirements for Virtualizable Third Generation Architectures)

References

- https://blackberry.qnx.com/content/dam/qnx/whitepapers/2017/what-is-a-hypervisor-and-how-does-it-work-pt1.pdf
- https://www.citefactor.org/journal/pdf/A-Virtualization-Approach-in-Smart-phone-Using-Cloud-computing-for-machine-to-machine-Communication.pdf
- http://dsc.soic.indiana.edu/publications/virtualization.pdf
- https://www.vmware.com/in/solutions/virtualization.html
- https://courses.cs.washington.edu/courses/cse451/18sp/readings/virtualization.pdf
- **Getting Started with Docker by Author: Christopher M. Judd (For detailed information about Docker architecture)**

- https://reader.elsevier.com/reader/sd/pii/S014036641730095 6?token=1BD05F31679AFAE6D696EDCF93DA910B51D2CDB3CDEA5A1B88AB87B6E50DD62AA0DB365B7AB9CD83A29F0C9568058F34&originRegion=eu-west-1&originCreation=20220205092527 (Dockers)
- https://reader.elsevier.com/reader/sd/pii/S1877050920311315?token=12D4662BF758D895621AEC6D7BFA5017249B01073B19B65D815E5DA9BF802399AEC3308F55520B7B4B19FFD993C62B46&originRegion=eu-west-1&originCreation=20220205092613 (Dockers)