Introduction and Recap

Prepared by:
Ms. Avita Katal
Assistant Professor (SG)
School of Computer Science
UPES, Dehradun

Container Orchestration and Infrastructure Automation CSVT3001

Link to Theory Course Plan

Link to Lab Course Plan

Recap

Q1: What is Virtualization and how it is a foundation technology to the concept of Cloud Computing?

Answer

Q2: What are the different types of Virtualizations?

Answer

Q3: What is Operating System Virtualization?

Answer

Virtualization

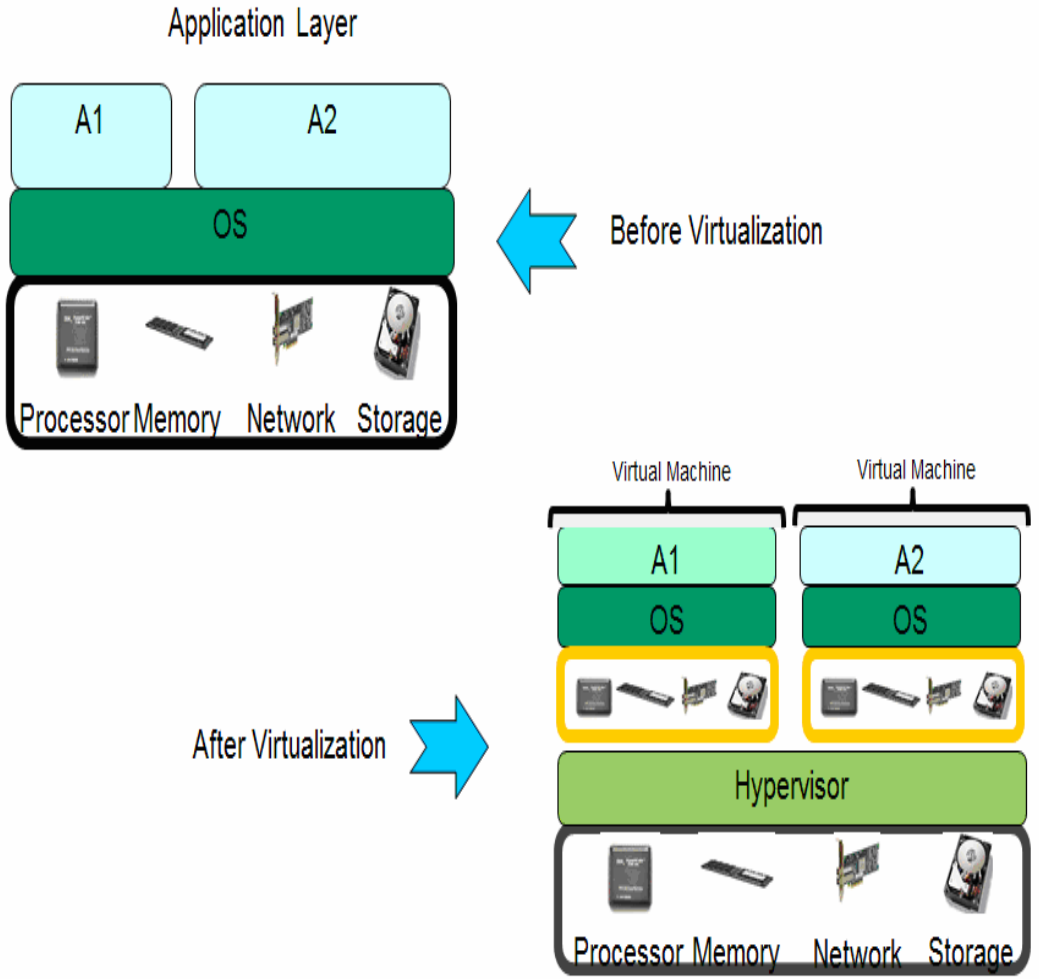
- Virtualization is creation of **an alternative to actual version of something**:
  - ➢virtual memory (more memory than physically available),
  - ➢virtual time (buffering provides virtual/effective download time that less than the actual time),
  - ➢Virtual hardware, desktop, disk, appliances, scenes,
  - ➢Virtual worlds
- In our context it is **realizing one or more complete computer systems as guests** on the base machine/operating system,

OR

Virtualization is way to run **multiple operating systems** and **user applications** on the same hardware

  - E.g., run both **Windows and Linux on the same laptop**

- This offers an excellent **conduit** for delivering the vastly **underutilized power of the multi-core and other resources such as storage and devices.**



Virtualization technologies have gained renewed interested recently due to the confluence of several phenomena:

1. **Increased performance and computing capacity.**

(supercomputers and desktops)

2. **Underutilized hardware and software resources.**

(administrative or IT automation tasks)

3. **Lack of space**

(server consolidation)

4. **Greening initiatives.**

(carbon footprint for cooling these servers)

5. **Rise of administrative costs.**

(administration tasks: hardware monitoring, defective hardware replacement, server set up and updates, server resources monitoring, and backups.)

The technologies of today allow profitable use of virtualization and make it possible to fully exploit the advantages that come with it. Such advantages have always been characteristics of virtualized solutions.

1. Increased security
2. Managed execution
    - Sharing
    - Aggregation:
    - Emulation
    - Isolation
    - Performance Tuning
3. Portability

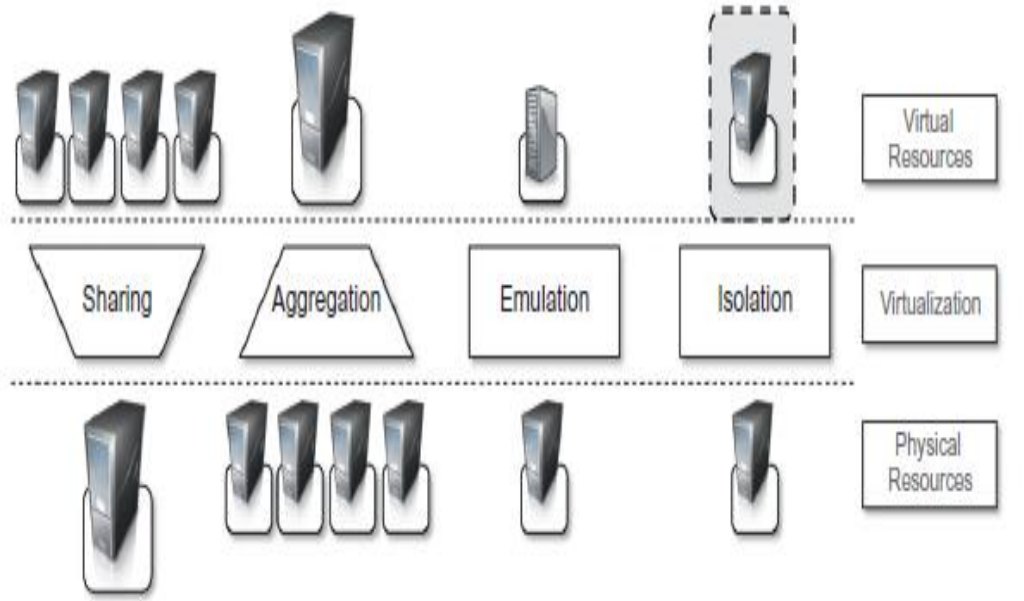


FIGURE 3.2

Functions enabled by managed execution.

Virtualization contd…

Hypervisors

- A fundamental element of hardware virtualization is the hypervisor, or virtual machine manager (VMM).
- It **recreates a hardware environment in which guest operating systems are installed**.
- There are two major types of hypervisor: ***Type I and Type II***

Type I

- Type I hypervisors run **directly on top of the hardware.**
- Therefore, they take the place of the operating systems and **interact directly with the ISA interface exposed by the underlying hardware,** and they emulate this interface in order to allow the management of guest operating systems.
- This type of hypervisor is also called a **Native Virtual Machine/Bare metal** since it runs natively on hardware.

Virtualization contd…

Type II

- Hypervisors require the ***support of an operating system*** to provide virtualization services.
- This means that they are programs managed by the operating system, **which *interact with it through the ABI and emulate the ISA of virtual hardware*** for guest operating systems.
- This type of hypervisor is also called a ***hosted virtual machine*** since it is hosted within an operating system.

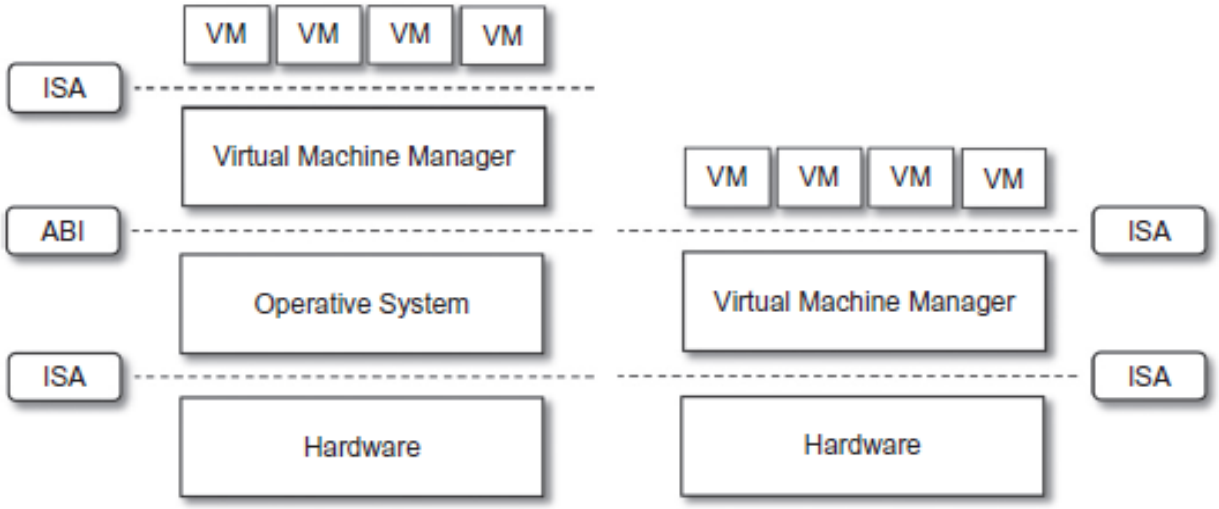


FIGURE 3.7

Hosted (left) and native (right) virtual machines. This figure provides a graphical representation of the two types of hypervisors.

Taxonomy of Virtualization Techniques

The first classification discriminates against **the service or entity that is being emulated**.

Virtualization is mainly used to emulate ***execution environments, storage, and networks.***

Among these categories, ***execution virtualization*** constitutes the oldest, most popular, and most developed area.

Virtualization techniques can be divided into **two major categories** by considering the **type of host they require.**

- ***Process-level techniques*** are implemented on top of an **existing operating system**, **which has full control of the hardware.**
- ***System-level techniques*** are implemented **directly on hardware** and do not require—or require a minimum of support from—an existing operating system.

Taxonomy of Virtualization Techniques

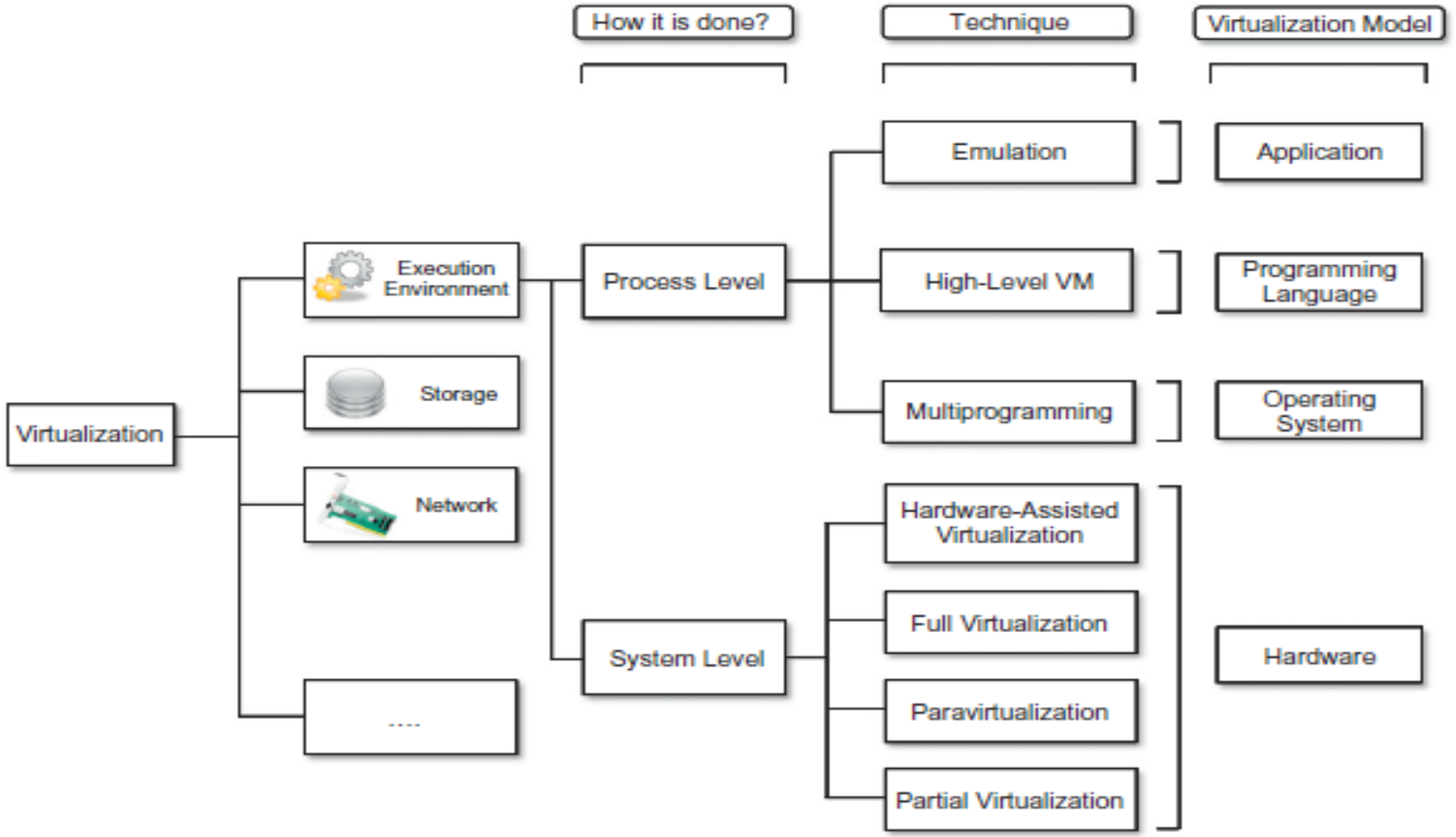


FIGURE 3.3

A taxonomy of virtualization techniques.

SYSTEM LEVEL VIRTUALIZATION

Hardware Virtualization Techniques

- Hardware-assisted virtualization.
- Full virtualization.
- Paravirtualization
- Partial virtualization.
- Hybrid Virtualization

- The **hardware provides architectural support** for building a virtual machine manager able to run a guest operating system in complete isolation.
- This technique was originally introduced in the **IBM System/370.**
- Examples of hardware-assisted virtualization are the extensions to the x86-64 bit architecture introduced with **Intel VT** (formerly known as **Vanderpool**) and **AMD V** (formerly known as **Pacifica).**
- Before the introduction of hardware-assisted virtualization**, software emulation of x86 hardware was significantly costly from the performance point of view**.
- The reason for this is that by design the x86 architecture **did not meet the formal requirements introduced by Popek and Goldberg,** and early products were using binary translation to trap some sensitive instructions and provide an emulated version.eg: VMWare Virtual Platform (1999).

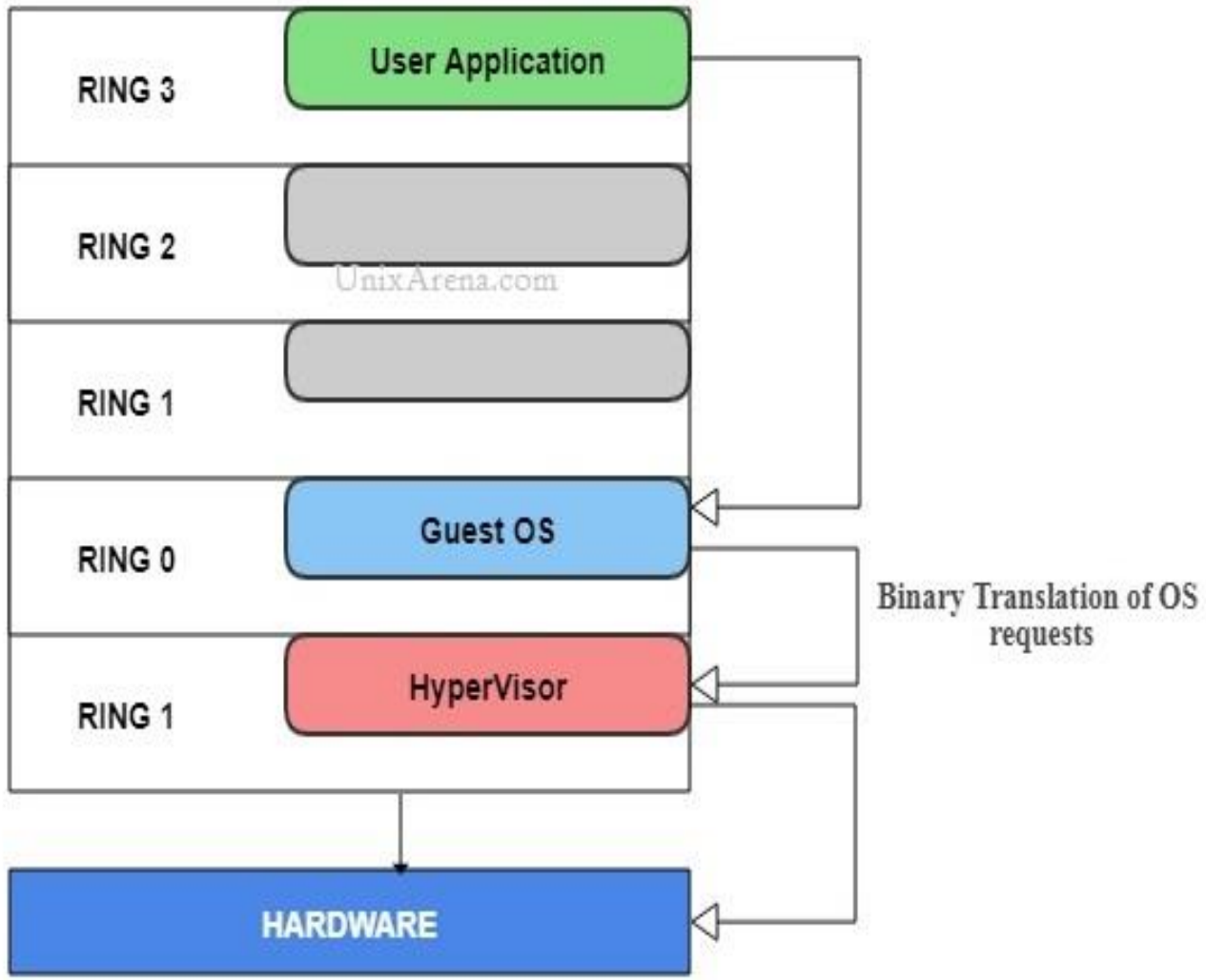
Full Virtualization (Hardware Assisted/ Binary Translation)

- Virtual machine **simulates hardware** to allow an **unmodified guest OS** to be run in isolation.
- There is two type of Full virtualizations in the enterprise market. On both full virtualization types, guest **operating system's source information will not be modified**.
    - Software-assisted full virtualization
    - Hardware-assisted full virtualization

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

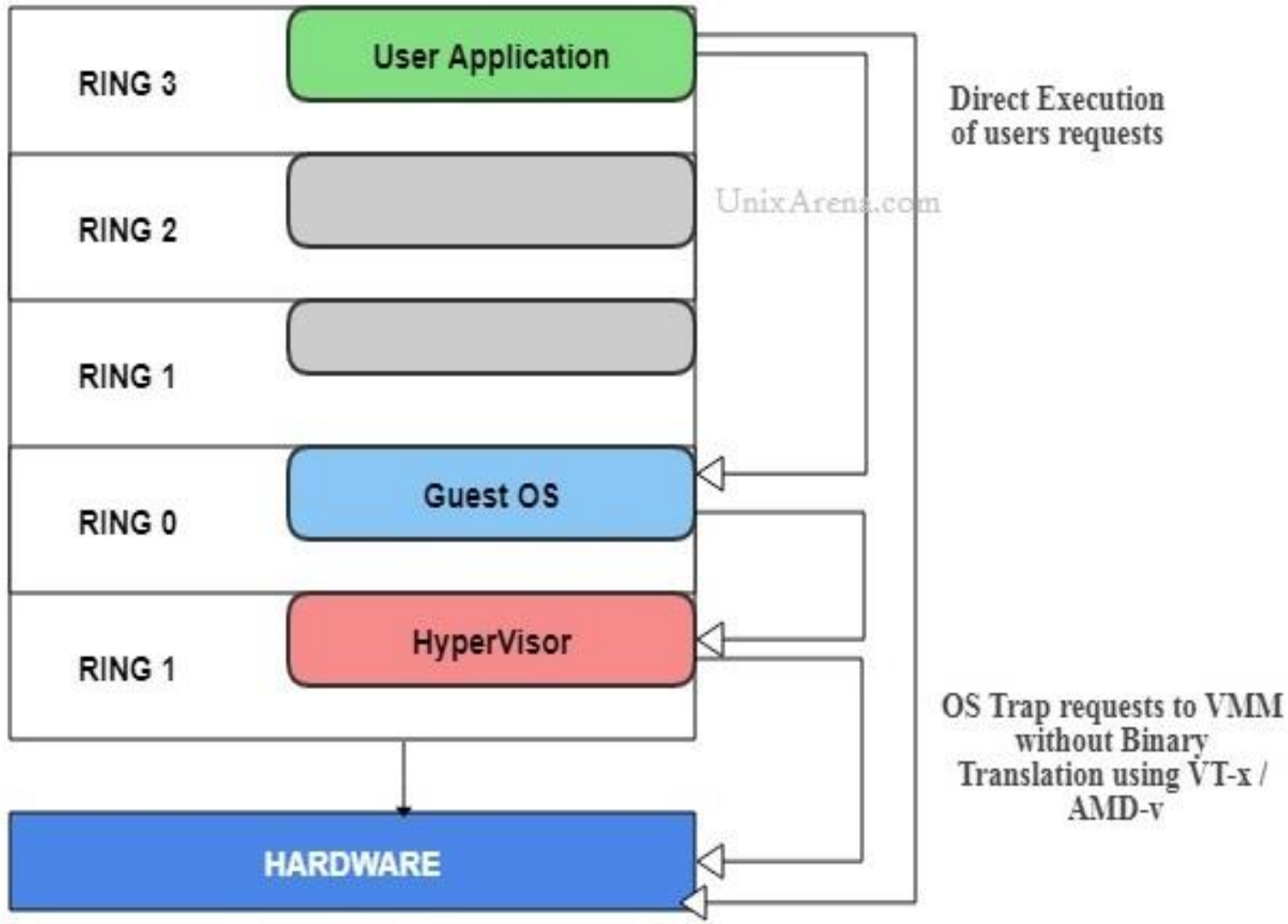
BINARY TRANSLATION - FULL VIRTUALIZATION

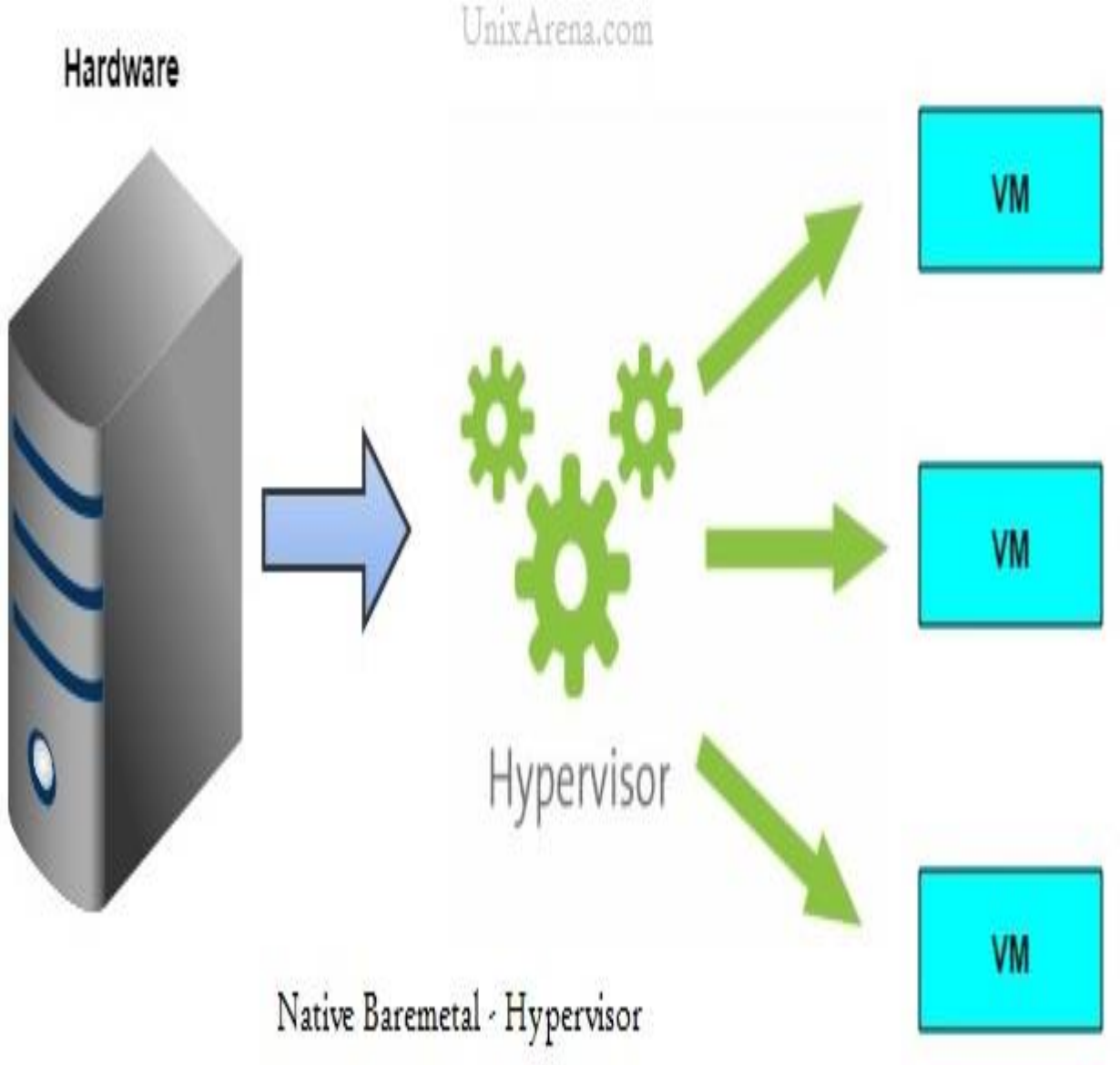


2.2.2 Hardware-Assisted – Full Virtualization (VT)

- 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**.
- A "pure" hardware-assisted virtualization approach, **using entirely unmodified guest operating systems, involves many VM traps**, and thus **high CPU overheads, limiting scalability and the efficiency of server consolidation**.
- This performance hit can be mitigated by the use of **paravirtualized drivers; the combination has been called "hybrid virtualization"**
- Hardware-assisted virtualization reduces the maintenance overhead of **paravirtualization as it reduces (ideally, eliminates) the changes needed in the guest operating system.** It is also considerably easier to obtain better performance

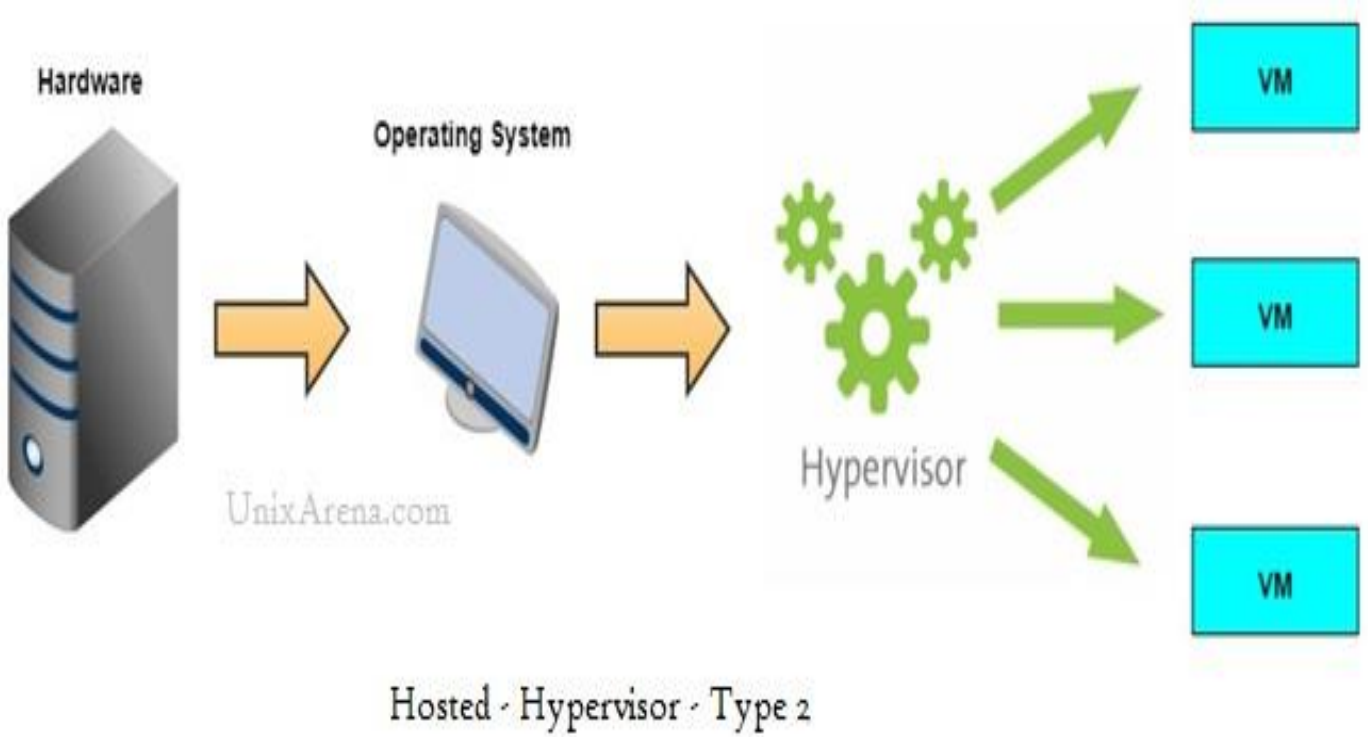
HARDWARE ASSISTED - FULL VIRTUALIZATION





Here is the list of enterprise software which supports **hardware-assisted – Full virtualization** which falls under **hypervisor type 1 (Bare metal)**

- VMware ESXi /ESX
- KVM
- Hyper-V
- Xen



Hosted - Hypervisor - Type 2

The following virtualization type of virtualization falls under hypervisor type 2 (Hosted).

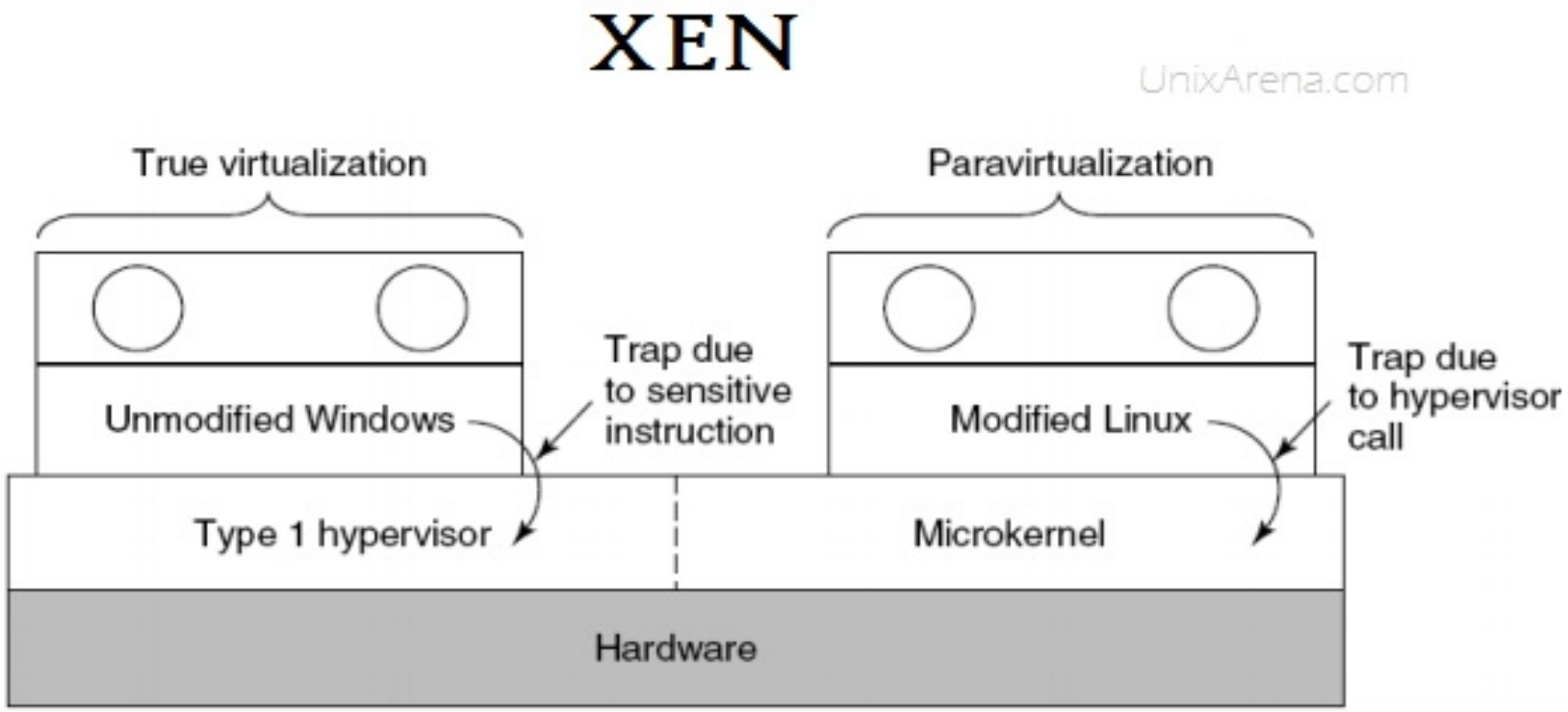
- VMware Workstation (64-bit guests only)
- Virtual Box (64-bit guests only)
- VMware Server (Retired)

Paravirtualization

- Paravirtualization works differently from the full virtualization. It **doesn’t need to simulate the hardware for the virtual machines.**
- The hypervisor is installed on a physical server (host) and a guest OS is installed into the environment.
- **Virtual guests aware that it has been virtualized**, unlike the full virtualization (where the guest doesn’t know that it has been virtualized) to take advantage of the functions.
- In this virtualization method, **guest source codes will be modified with sensitive information to communicate with the host.**
- Guest Operating systems **require extensions to make API calls to the hypervisor.**

- In full virtualization, **guests will issue a hardware calls** but in paravirtualization, **guests will directly communicate with the host (hypervisor) using the drivers.**
- Here is the list of products which supports paravirtualization.
  - Xen
  - IBM LPAR
  - Oracle VM for SPARC (LDOM)
  - Oracle VM for X86 (OVM)

- The below diagram might help you to understand how Xen supports both full virtualization and paravirtualization. Due to the architecture difference between windows and Linux based Xen hypervisor, **Windows operating system can't be para-virtualized**. But it does for **Linux guest by modifying the kernel**. But VMware ESXi doesn't modify the kernel for both Linux and Windows guests.



Partial virtualization

- Partial virtualization provides a **partial emulation of the underlying hardware**, thus **not allowing the complete execution of the guest operating system in complete isolation.**

- Partial virtualization allows **many applications to run transparently**, but not all **the features of the operating system can be supported**, as happens with full virtualization.

- An example of partial virtualization is **address space virtualization** used in time-sharing systems; this allows multiple applications and **users to run concurrently in a separate memory space, but they still share the same hardware resources** (disk, processor, and network).

- Historically, partial virtualization has been an important milestone for achieving full virtualization, and it was implemented on the experimental **IBM M44/44X.**

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

Operating system Virtualization

•**OS-level virtualization** is an operating system (OS) paradigm in which the kernel allows the existence of multiple isolated user space instances, called ***containers*** (LXC, Solaris containers, Docker, Podman), ***zones*** (Solaris containers), ***virtual private servers*** (OpenVZ), ***partitions***, ***virtual environments*** (VEs), ***virtual kernels*** (DragonFly BSD), or ***jails*** (FreeBSD jail or chroot jail).

• Such instances may look like **real computers from the point of view of programs** running in them.

•A computer program running on an ordinary operating system can see all resources (connected devices, files and folders, network shares, CPU power, quantifiable hardware capabilities) of that computer. However, **programs running inside of a container can only see the container's contents** and **devices assigned to the container.**

A Brief History of Container Technology

BEGINNING AGE

•Container technology was born in **1979** with **Unix version 7** and the **chroot** system.

•This system was the first glimpse of an isolated process, and it was soon adopted and added into **BSD OS in 1982**; however, container technology would unfortunately not progress over the next two decades and remain dormant.

A Brief History of Container Technology

The Adolescent Years

•Container technology finally picked up steam in the **2000s** with the introduction of **Free BSD Jails**. “Jails” are partitions of a computer, where there could be multiple jails/partitions on the same system.

•This jail system was further refined in **2001** with **Linux VServer** with the **partitioning of resources**, which was later linked to the **linux kernel** with **OpenVZ in 2005**, and jails were combined with boundary separation to create in **Solaris Containers in 2004.**

•After jails, container tech further progressed with the introduction of **control groups in 2006.** They were soon used and built upon in **LXC (linuX Containers) in 2008**, and this was the most complete and stable version of container technology at the time, as it worked on the Linux kernel without any patches.

•Due to LXC’s reliability and stability, many other technologies built on LXC, the first of which to be **Warden in 2011** and more importantly **Docker in 2013**.

A Brief History of Container Technology

UPES
UNIVERSITY OF TOMORROW

The Golden Age

- Even though container technology had remarkable improvements from 2000-2011, the introduction of **Docker** changed everything and single-handedly led to a **Renaissance in container technology**.

- Docker built its foundation on two systems, **LXC and libcontainers**. Libcontainers came from LMCTFY(open source Google container stack), which was an open source container stack where applications created and managed their own subcontainers.

- In addition to building on previous software, Docker had an easy to use GUI and was capable of having multiple applications with different OS requirements run on a single OS.

A Brief History of Container Technology

UPES
UNIVERSITY OF TOMORROW

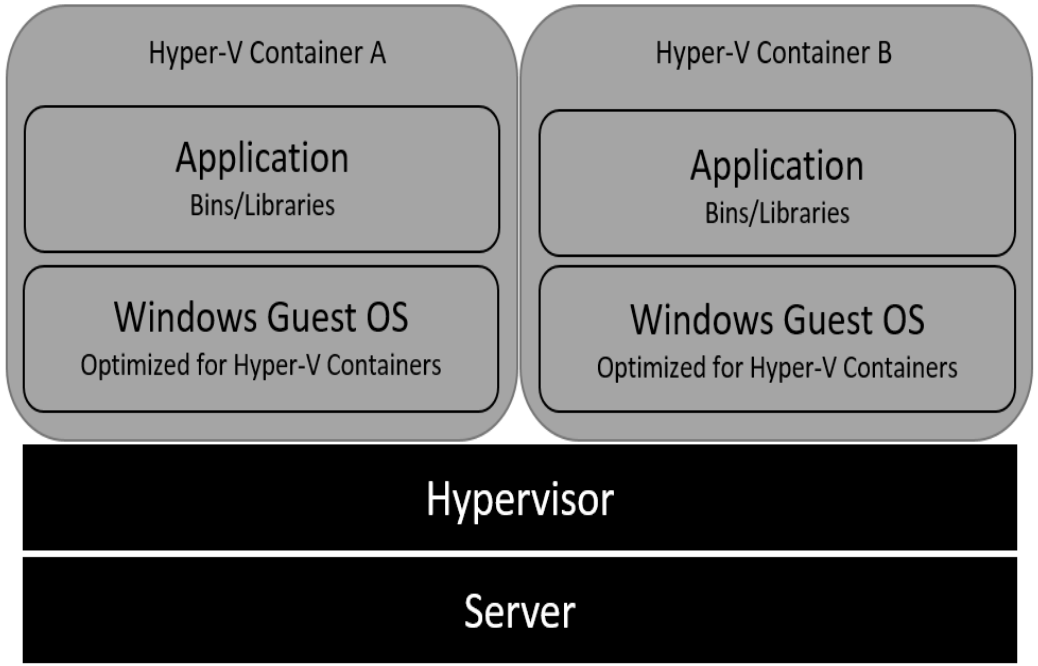
The Golden Age

- Docker blew up, leading to **100 million downloads within a year**, and after Docker's success, another technology, **rkt (pronounced Rocket),** tried to fix some of Docker's problems by incorporating more rigorous security and production requirements.

- Previously, containers were mostly **linux based technology**, but this decision lead to Microsoft becoming a major influence in container technology, seen by both **Process and Hyper-V containers on Windows computers**.

- Container technology's momentum continued in **2017** through the introduction of the Kubernetes, which was a highly effective container orchestration technology.

How Hyper V containers are different from the rest?



- Hyper-V Containers are a special type of container with a higher degree of isolation and security.
- Unlike Windows Server Containers, which share the kernel, Hyper-V Containers do not share kernels and instead each container runs its own kernel, which makes them special VMs.

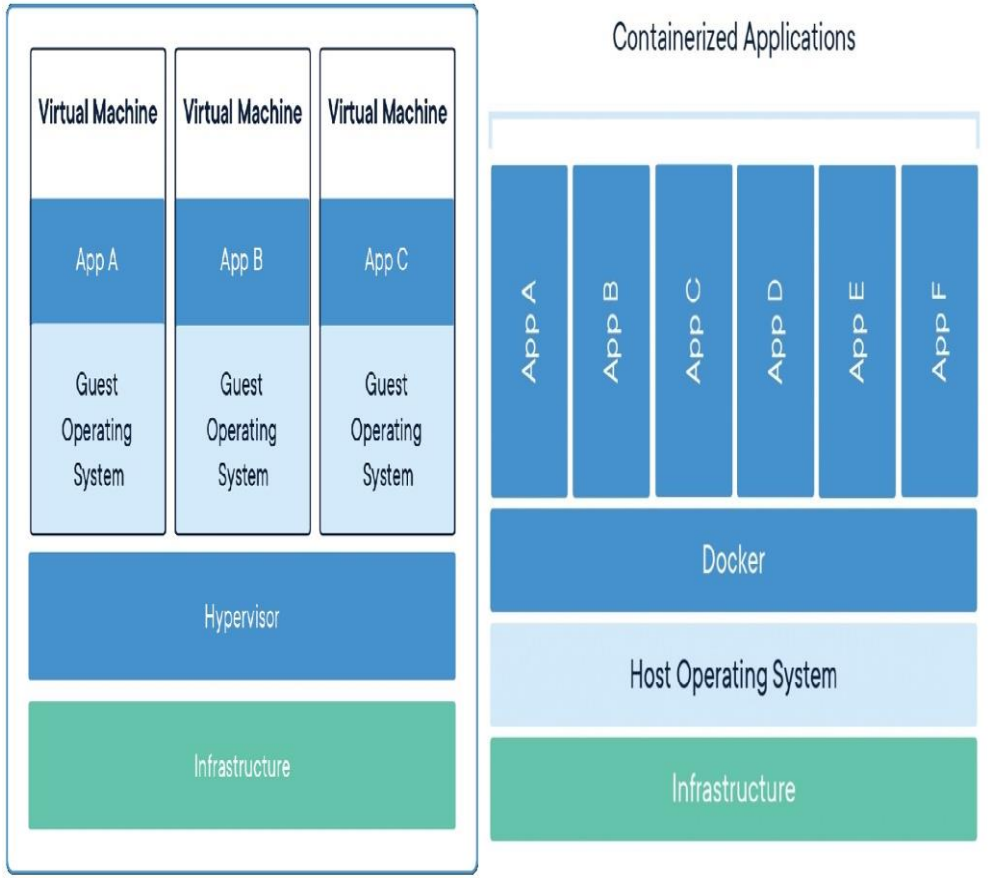
The following are some of the other container types:

- **Windows Server Containers** run as isolated containers on a shared kernel. In a single tenant environment or private clouds this is not a problem, since the containers run in a trusted environment. But Windows Containers are not ideal for a multitenant environment. There could be security or performance related issues such as noisy neighbors or intentional attacks on neighboring containers.
- Since Windows Container shares the host OS, patching the host OS disrupts the normal functioning of applications hosted in the OS.

This is where Hyper-V Containers make perfect sense.

Comparing Virtual Machines and Containers

What's Diff?	VMs	Containers
Size	Heavyweight (GB)	Lightweight(MB)
Boot Time	Startup time in minutes	Startup time in seconds
Performance	Limited performance	Native performance
OS	Each VM runs in its own OS	All containers share the host OS
Runs on	Hardware-level virtualization (Type1)	OS virtualization
Memory	Allocates required memory	Requires less memory space
Isolation	Fully isolated and hence more secure	Process-level isolation, possibly less secure



Comparing Virtual Machines and Containers

Virtual machines:

•A virtual machine is the emulated equivalent of a physical computer system with their virtual CPU, memory, storage, and operating system.

•A program known as a hypervisor creates and runs virtual machines. The physical computer running a hypervisor is called the host system, while the virtual machines are called guest systems.

•The hypervisor treats resources — like the CPU, memory, and storage — as a pool that can be easily reallocated between the existing guest virtual machines.

There are two types of hypervisors:

Type 1 Hypervisor (VMware vSphere, KVM, Microsoft Hyper-V).

Type 2 Hypervisor (Oracle VM VirtualBox, VMware Workstation Pro/VMware Fusion).

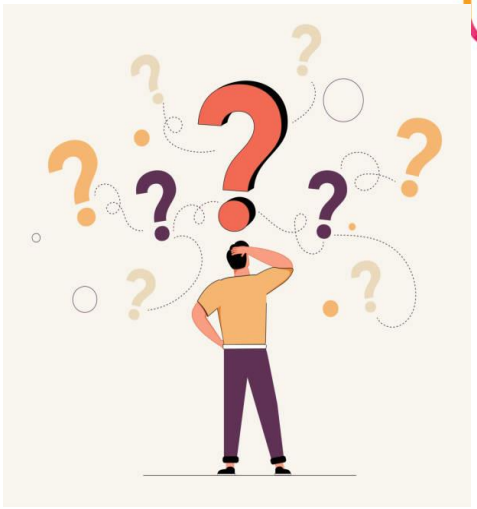
Containers:

•A container is an abstraction **at the application layer that packages code and dependencies together.**

•Instead of virtualizing the entire physical machine, **containers virtualize the host operating system only.**

•Containers sit on top of the physical machine and its operating system. Each container shares the host operating **system kernel and, usually, the binaries and libraries, as well**.

What is Containerization and Docker???



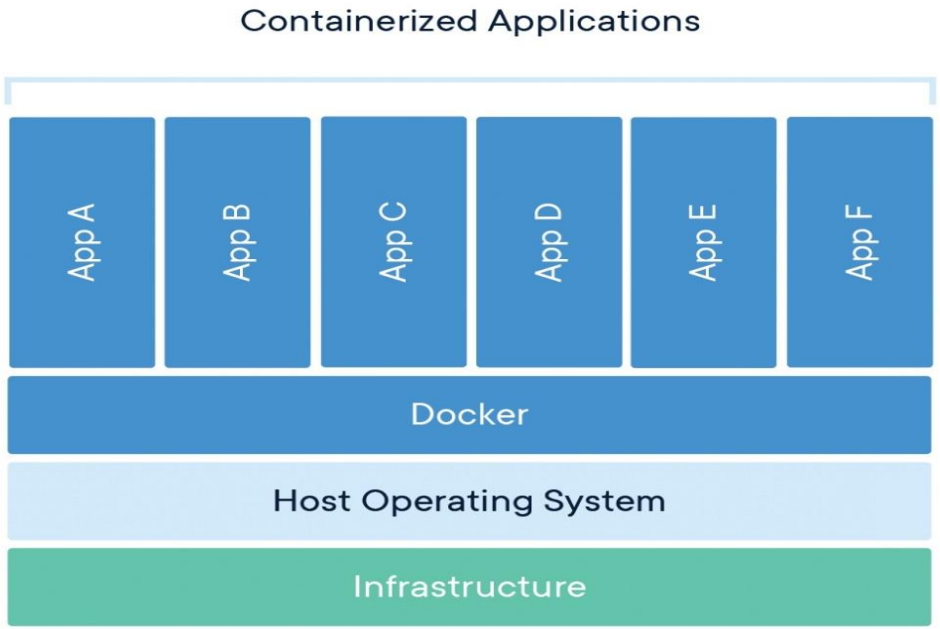
Containerization

Containerization is the process of **encapsulating software code along with all of its dependencies inside a single package** so that it can be run consistently anywhere.

Docker

Docker is an open source containerization platform. It provides the **ability to run applications in an isolated environment known as a container.**

Containers are like very lightweight virtual machines that can run directly on our host operating system's kernel without the need of a hypervisor. As a result we can run multiple containers simultaneously.



Thank You