UNIT IV – VIRTUALIZATION AND CONTAINER TECHNOLOGIES

Introduction to virtualization: VMs, Hypervisors (Type 1 and 2) , Docker architecture: images, containers, registries, Namespaces, cgroups, layered file systems , Resource limits and process isolation , Use case: Microservices, CI/CD containers, VirtualBox, Tools: Docker, KVM, VirtualBox.

4.1 INTRODUCTION TO VIRTUALIZATION

Virtualization in operating systems is the technique of creating a **virtual version** of computer resources such as hardware, storage, network, or the operating system itself.

It allows one physical system to run **multiple independent virtual machines (VMs)**, each acting like a separate computer with its own operating system and applications.

Definition:

Virtualization is the process that uses a special software (called a **hypervisor**) to divide a physical machine into multiple isolated environments, so that multiple operating systems can run on the same hardware simultaneously.

Need for Virtualization:

- Better utilization of hardware resources.
- Cost savings (reduced number of physical servers).
- Easy to test and develop applications.
- Scalability and flexibility for cloud computing.
- Improved isolation and security between applications.

Working of Virtualization:

Virtualizations uses special software known as hypervisor that create many virtual computers on one physical computer. The Virtual Machines behave like actual computers but use the same physical machine.

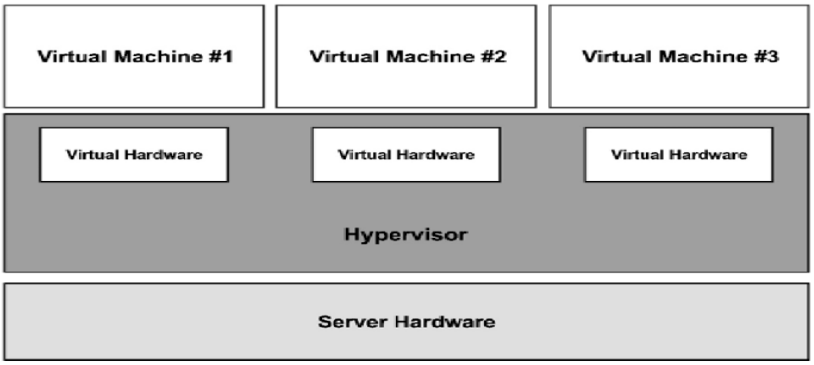
Types of Virtualization:

Virtualization is not limited to just virtual machines. Here are 5 major types:

1. Hardware Virtualization (Server Virtualization)

Hardware virtualization is the process of creating multiple virtual machines (VMs) that run on a single physical server using a **hypervisor**.

- A hypervisor (like VMware ESXi or Microsoft Hyper-V) directly interacts with the physical hardware.
- It allocates physical resources (CPU, memory, storage) to each VM.
- Each VM operates independently with its own OS and applications.



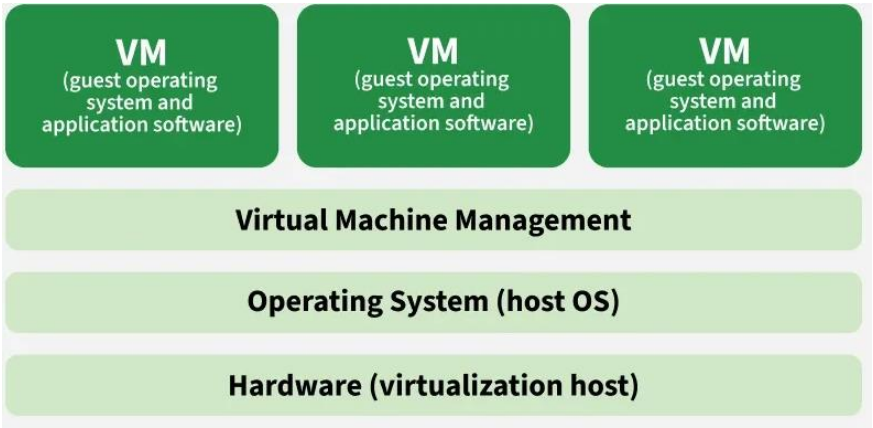
Benefits:

- Efficient hardware utilization.
- Cost savings (fewer physical servers needed).
- Scalability and easier deployment of services.
- Fault isolation: if one Virtual Machine crashes, others remain unaffected.

2. Software Virtualization

Software virtualization allows an operating system or application to run in an environment that is different from its native platform.

- A software layer mimics the required hardware or operating system.
- Example: Running Android OS on a Windows PC using an emulator like BlueStacks.



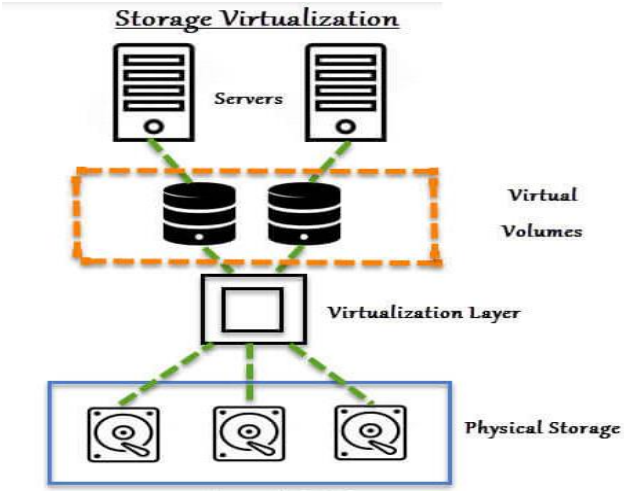
Benefits:

- It support Cross-platform compatibility.
- It help to testing apps on different OS versions.
- It reduces software conflicts.

3. Storage Virtualization

Storage virtualization combines multiple physical storage devices into a single **logical storage pool**, making management easier and more flexible.

- Software abstracts the physical storage from multiple devices.
- The system presents the unified storage as a single resource to users and applications.



Benefits:

- It has a centralized storage management.
- Better performance through load balancing.

- Easier scaling and disaster recovery.

4. Network Virtualization

Network virtualization abstracts network resources, allowing multiple virtual networks to be created on a single physical infrastructure.

- It combines hardware (switches, routers) and software resources into a virtual network.
- Divides bandwidth into independent channels for better resource allocation.

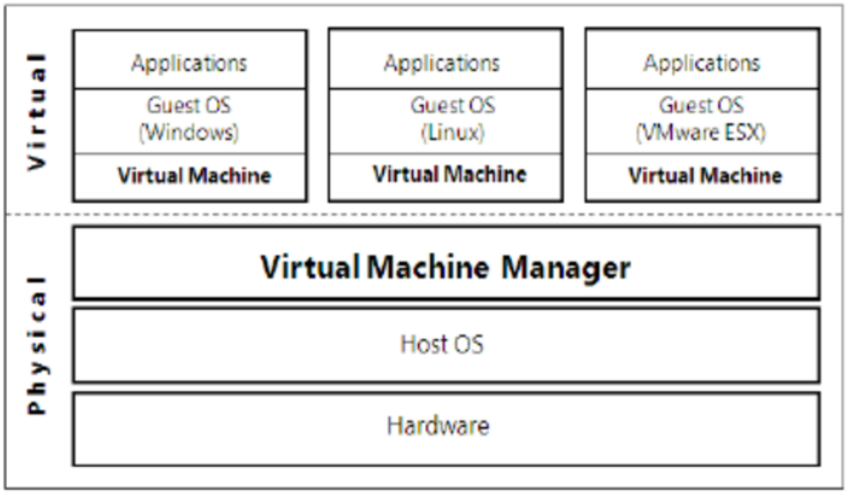
Benefits:

- It improves network efficiency and security.
- It is easy manage and easy to scaling.
- It provides dynamic provisioning of network resources.

5. Desktop Virtualization

Desktop virtualization separates the desktop environment from the physical machine and stores it on a remote server.

- Users access their virtual desktop over a network using any device (PC, tablet, thin client).
- All processing and data remain on the central server.



Benefits:

- It has Centralized control and security.
- It allows easy software updates and patches.
- It is flexibility for remote work.

4.1.1 VIRTUAL MACHINES

Definition:

A Virtual Machine (VM) is a software-based computer that runs within a physical computer (known as the host). It acts as the behaviour of a physical machine and can run its own operating system (guest OS) and applications independently just like a real computer.

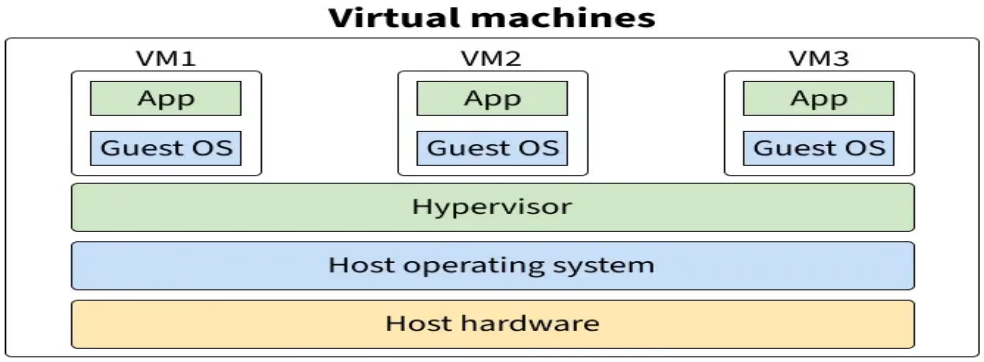
For example, you can run a Linux VM inside a Windows PC using tools like VirtualBox or VMware.

- A VM is a "**fake computer**" made by software that **behaves like a real one**.
- A **Virtual Machine (VM)** is like a computer inside your computer.
- After installing virtualization software, you can create one or more virtual machines on your computer.
- Virtual machines (VMs) behave like regular applications on your system.
- The real physical computer is called the **Host**, while the virtual machines are called **Guests**.
- A single host can run multiple guest virtual machines.
- Each guest can have its own operating system, which may be the same or different from the host OS.
- Every virtual machine functions like a standalone computer, with its own settings, programs, and configuration.

- VMs access system resources such as **CPU, RAM, and storage**, but they work as if they are using their own hardware.
- Two virtual machines don't interrupt in each other's working and functioning nor they can access each other's space which gives an illusion that we are using totally different hardware system.

Each VM behaves like a separate computer, even though it's running on shared hardware. It includes:

- Virtual CPU
- Virtual memory
- Virtual disk (storage)
- Network interface card (NIC)



Example: When you run multiple processes on a regular OS, they appear to have separate CPUs and memory (thanks to CPU scheduling and virtual memory). A VM extends this illusion to the OS level itself.

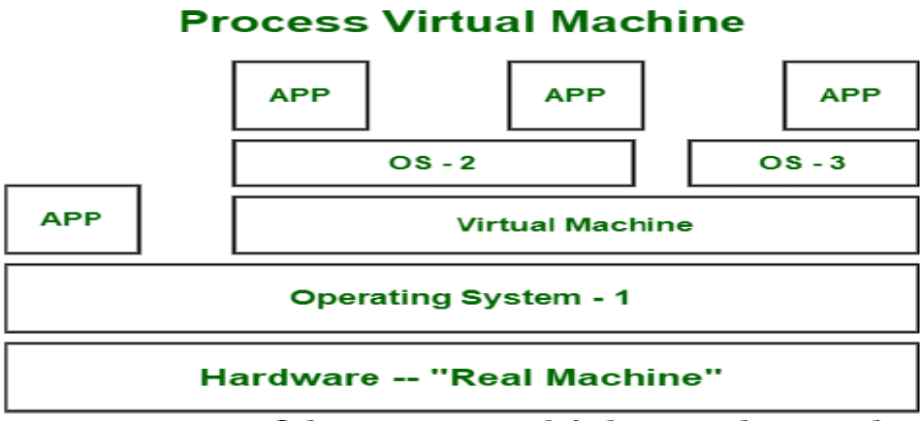
Types of Virtual Machines

There are **two main types** of VMs:

1. Process Virtual Machine

Unlike System Virtual Machine, Process Virtual Machine does not provide us with the facility to install the virtual operating system completely. Rather it creates

virtual environment of that OS while using some app or program and this environment will be destroyed as soon as we exit from that app. Like in below image, there are some apps running on main OS as well some virtual machines are created to run other apps. This shows that as those programs required different OS, process virtual machine provided them with that for the time being those programs are running. **Example –** Wine software in Linux helps to run Windows applications.

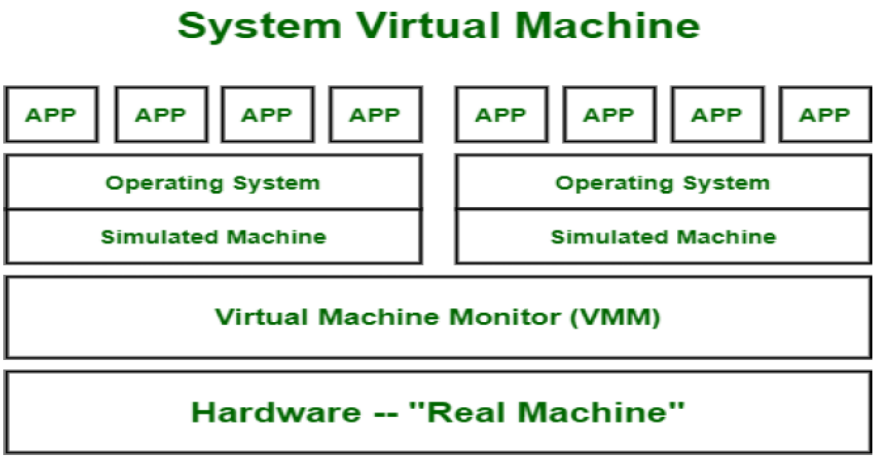


- Designed to run a single application (or process).
- Example: Java Virtual Machine (JVM), which allows Java programs to run on any platform.

2. System Virtual Machine

These types of virtual machines give us complete system platform and gives the execution of the complete virtual operating system. Just like virtual box, system virtual machine is providing an environment for an OS to be installed completely. From the below image, our hardware of Real Machine is being distributed between two simulated operating systems by Virtual machine monitor. And some programs, processes are going on in that distributed hardware of simulated machines separately.

- It acts as an entire physical machine.
- It can run a full OS like Linux, Windows, or macOS.
- Example: VMs created using VirtualBox, VMware, or Hyper-V.



Steps to Set Up a Virtual Machine

Setting up a VM involves the following steps (example for installing a Linux VM):

1. Create a new virtual machine
2. Allocate virtual disk space
3. Attach a virtual network adapter
4. Install the guest operating system (Linux)
5. Register and activate the VM (if needed)
6. Install necessary drivers or guest additions

Most platforms like VirtualBox or VMware offer wizards to simplify this setup.

Uses:

- Running multiple OS's on a single machine.
- Testing new software or OS safely (sandboxed)
- Running legacy applications
- Server virtualization (hosting multiple VMs on a single server)
- Secure development and debugging (isolate risky tasks)

4.1.2 HYPERVISORS

A **hypervisor** (or Virtual Machine Monitor, VMM) is **software that enables virtualization**. It acts as an intermediary between the physical computer (host) and the virtual machines (VMs). The hypervisor manages hardware resources such as CPU, memory, storage, and network, and allocates them to multiple virtual machines without interference.

This allows multiple operating systems to run simultaneously on a single physical machine, improving hardware utilization, reducing costs, and providing flexibility in cloud and server environments.

How It Works

The hypervisor can run directly on physical hardware (Type 1) or on a host operating system (Type 2). It creates and manages VMs, each with its own virtual CPU, memory, storage, and network. When a guest OS in a VM requests resources, the hypervisor intercepts the request and translates it to the physical hardware, ensuring **isolation, security, and stability** among all virtual machines. Host is the real computer which has its own hardware and OS.

Example: VMware ESXi, Microsoft Hyper-V**, KVM, Xen, VirtualBox**, Parallels Desktop ,VMware Workstation/Fusion.

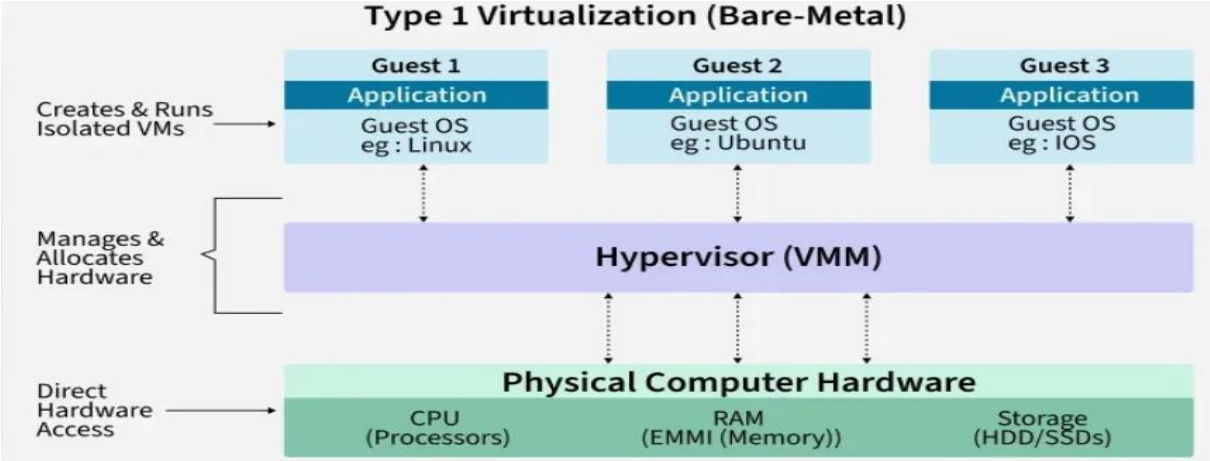
There exist two categories of hypervisors:

Type 1 Hypervisor (Bare-Metal Hypervisor):

- The hypervisor is installed directly onto the computer hardware, without an operating system sitting in between.
- A **Type 1 Hypervisor**, also called a **bare-metal hypervisor**, runs **directly on the physical hardware** of a host machine, without needing a host operating

system. It provides virtualization by creating and managing virtual machines (VMs) directly on the hardware.

- It provides high performance and efficiency.
- Common used in **enterprise servers and data centers**.
- It is highly efficient as it has a direct access to the resources of the computer.



Key Features:

- **High Performance:** Since it communicates directly with hardware, it provides near-native performance for VMs.
- **Resource Efficiency:** Manages CPU, memory, storage, and network resources efficiently.
- **Isolation & Security:** Each VM is isolated from others, and the hypervisor has a small attack surface, making it more secure.
- **Enterprise Use:** Commonly used in data centre's and cloud environments.

Examples: VMware ESXi, Microsoft Hyper-V, KVM (Kernel-based Virtual Machine), and Xen.

Advantage:

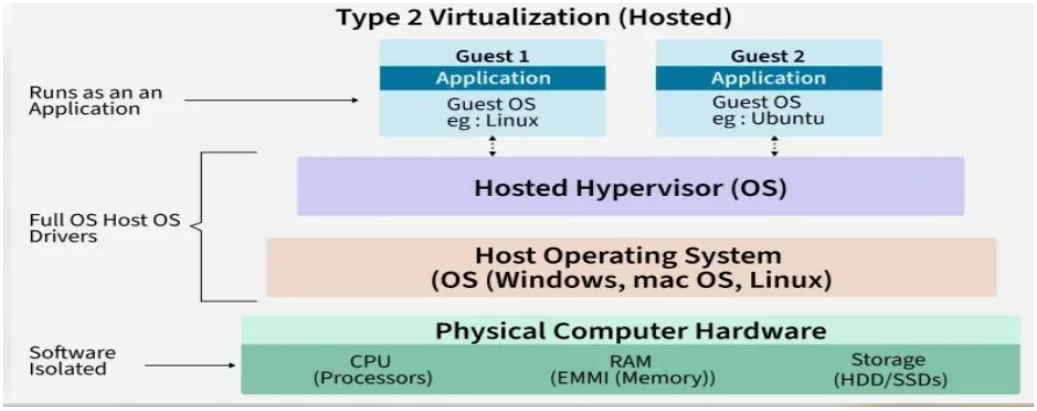
- It gives high performance (direct hardware access).
- Strong security (no intermediate OS layer).
- It is suitable for large-scale cloud and enterprise workloads.

Disadvantage:

- It requires dedicated hardware.
- Setup and management are complex compared to Type-2.

Type 2 Hypervisor:

- It is run over an installed operating system (such as Windows or macOS). ie., Runs **on top of a host operating system**.



- It relies on the host OS to manage hardware resources to communicate and creates virtual machines (VMs) that operate within this environment. It's used when you need to execute more than one operating system on one machine.
- Suitable for **desktop/laptop use, testing, and development**.

Examples: Oracle VM VirtualBox, VMware Workstation, and Parallels Desktop.

Advantage:

- It is Easy to install and use.
- It is useful for development, testing, and malware analysis.
- It provides good host–guest integration features.
- It works on personal computers

Disadvantage:

- **Slower** performance (no direct hardware access).

- **Security** depends on the host OS; compromise of host may affect guests.