UNIT 4

Virtual Machine

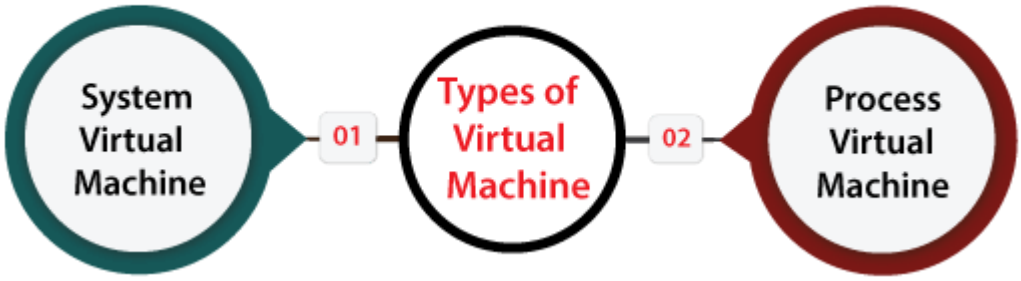
Virtual Machine can be defined as an **emulation of the computer systems** in computing. Virtual Machine is based on computer architectures. It also gives the functionality of physical computers. The implementation of VM may consider specialized software, hardware, or a combination of both.

History of Virtual Machine

- System Virtual Machines are notably implemented within the CTSS (Compatible Time-Sharing System). Time-Sharing permitted more than one user for using the computer concurrently. All the programs displayed to have complete access to a machine, but a single program can only run at a time. It was derived into virtual machines by the research system of IBM notably. The 44X/M44 are using partial virtualization, and SIMMON and CP-40 are using full virtualization. These are some examples of hypervisors.
- The first architecture of the virtual machine was **CMS/CP-67**. An important differentiation was among using many virtual machines on a single host for time-sharing as within the CP-40 and 44X/M44.
- Emulators date turn to the 360/IBM System in 1963 with hardware emulation of previous systems for compatibility.
- Originally, process virtual machines developed as abstract environments for any intermediate language applied as a program's intermediate representation by the compilers. The O-code machine was an example of early 1966. The O-code machine was a VM that runs object code (O-code) expanded by the BCPL compiler's front end. This abstraction permitted the compilers to be ported to any new architecture easily.
- The Euler language applied the same design with an intermediate language called portable (P). It was promoted by Pascal in 1970, notably within the Pascal-P system and Pascal-S compiler. They were known as **p-code** and the p-code machine as the resulting machine.
- It has been affecting, and VMs within this type of sense have been generally known as p-code machines often. In addition, Pascal p-code was run by an interpreter directly which is used for implementing VM.

Faculty: Dr Naveen Singh
+91-9953320298 drnaveenkrsingh@gmail.com

- Another example is SNOBOL (1967). It was specified in SIL (**SNOBOL Implementation Language**) that is an assembly language for VM. It was intended to physical machines through transpiling to the native assembler by a macro assembler.
- Process VM was a famous approach for implementing microcomputer software, containing adventure and Tiny BASIC games. It can be done by some Implementations like Pyramid 2000 to any general-purpose engine such as z-machine of Infocom.
- Significant advances illustrated in the Smalltalk-80 implementation (specifically the Schiffmann/Deutsch Implementations). They can push forward the JIT (Just In Time) compilation as the implementation approach which applies process VM. Notably, later Smalltalk virtual machines were Strongtalk, Squeak Virtual Machine, and VisualWorks.
- A complimentary language generated many VM innovation which pioneered generational garbage collection and adaptive optimization. Commercially, these methods were approved successfully within the HostSpot Java virtual machine in 1999.
- Other innovations contain the register-based VM to match various underlying hardware, instead of a stack-based VM that is closer for any programming language. It was pioneered in 1995 for the Limbo language by Dis VM. OpenJ9 is a substitute for HotSpot Java virtual machine inside the OpenJDK. Also, it is an open-source project requesting good startup and fewer resource consumption when compared to HotSpot.



Faculty: Dr Naveen Singh
+91-9953320298 drnaveenkrsingh@gmail.com

- System virtual machines: These types of virtual machines are also termed as full virtualization VMs. It facilitates a replacement for an actual machine. These VMs offers the functionality required for executing the whole operating system (OS). A hypervisor applies native execution for managing and sharing hardware. It permits for more than one environment that is separated from each other while exists on a similar physical machine. Novel hypervisor applies virtualization-specific hardware and hardware-assisted virtualization from various host CPUs primarily.
- Process virtual machines: These Virtual Machines are created for executing several programs of the computer within the platform-independent environment.

A few VMs are developed for emulating distinct architectures like QEMU. It permits the execution of operating system and software applications written for other architectures or CPU. Operating-system-level virtualization permits the computer resources to be categorized by the kernel.

What is System Virtual Machines?

Originally, a Virtual Machine was described by Goldberg and Popek as "an isolated and efficient duplicate of an actual computer machine." The latest use combines virtual machines that haven't any direct relation with actual hardware. Generally, the real world or physical hardware (executing the virtual machine) is termed as the "host" and the VM copied on the machine is generally termed as the "guest."

Working of System Virtual Machines

The host could emulate various guests, all of which could emulate distinct hardware platforms and operating systems.

A craving to execute more than one operating system was a starting objective of the virtual machines. It allows time-sharing between many individual tasking operating systems. A system VM can be could be considered the concept generalization of virtual memory that preceded it historically.

CMS/CP of IBM, the initial systems that permit full virtualization, equipped to be sharing by giving all users an individual-user OS (Operating System). The system VM designated the user for writing privileged instructions inside the code. This type of method has some advantages like including output/input devices not permitted by any standard system.

Memory over-commitment's new systems may be used for managing memory sharing between several VMs over a single computer OS. It is because technology expands VM for various virtualization purposes. It can be possible to distribute memory pages that include identical contents for many VMs that execute on a similar physical machine. As a result, mapping them to a similar physical page by a method called KSM (kernel-same page merging).

Faculty: Dr Naveen Singh
+91-9953320298 drnaveenkrsingh@gmail.com

It is useful especially for various read-only pages, like those containing code segments. It is a case for more than one VM executing the similar or same middleware components, web servers, software libraries, software, etc. A guest OS doesn't require to be compliant with any host hardware, hence making it feasible to execute distinct OS on a similar computer (such as an operating system's prior version, Linux, or Windows) for supporting future software.

Uses of System Virtual Machines

The virtual machine can be used for supporting isolated guest OS. It is popular regarding embedded systems. A common use might be to execute the real-time operating system with a preferred complicated operating system simultaneously such as Windows or Linux.

Other uses might be for unproven and novel software that is still in the stage of development, thus it executes in a sandbox. VMs have other aspects of OS development. It may contain faster reboots and developed debugging access.
More than one virtual machine running their guest OS is engaged for the consolidation of the server frequently.

What is Process Virtual Machines?

A process virtual machine is sometimes known as MRE (Manages Runtime Environment) or application virtual machine. It runs as a general application in the host operating system and supports an individual process. These are created if that process begins and destroyed if it exits.

The purpose of the process VM is to facilitate a programming environment that is **platform-independent**. It abstracts away all the information of the underlying operating system or hardware. It allows the programs to be executed on any platform in a similar way.

A process virtual machine gives the high-level abstraction of a high-level programming language. Process virtual machine can be implemented with the use of an interpreter. Its performance proportionate to the programming language (compiled) can be attained by using a just-in-time compilation.

The process virtual machine has become famous with the **Java programming** language. It can be implemented with the Java virtual machine. Another example includes the .**NET Framework** *and* **Parrot virtual machine** which executes on the virtual machine known as the **Common Language Runtime**. Each of them could be served as the abstraction layer for a computer language.

The process virtual machine has a special case for those systems that essence on the communication mechanisms of the (heterogeneous potentially) computer clusters. These types of virtual machines do not include any individual process, although one process/physical machine inside the cluster.

These clusters are created to mitigate the programming confluent applications task by enabling the programmers to concentrate on algorithms instead of the communication mechanisms given by the OS and interconnect.

Faculty: Dr Naveen Singh
+91-9953320298 drnaveenkrsingh@gmail.com

Full Virtualization

The virtual machine affects hardware to permit a guest operating system to be executed in separation in full virtualization. It was developed in 1966 using the IBM CP-67 and CP-40 which are the VM family's predecessors.

Some of the examples outside the field of mainframe include Egenera vBlade technology, Win4Lin Pro, Win4BSD, Mac-on Linux, Adeos, QEMU, VMware ESXi, VMware Server (also known as GSX Server), VMware Workstation, Hyper-V, Virtual Server, Virtual PC, Oracle VM, Virtual Iron, VirtualBox, Parallels Desktop for Mac, and Parallels Workstation.

Hardware-assisted virtualization

The hardware facilitates architectural support in hardware-assisted virtualization. This architectural support provides help for creating a monitor of the virtual machine and permits various guest operating systems to be executed in separation.

This type of virtualization was first defined in 1972 on the IBM System/370. It was introduced for applying with VM/370. The initial virtual machine OS provided by IBM was the official product.

AMD and Intel give additional hardware for supporting virtualization in 2006 and 2005. In 2005, Sun Microsystems (Oracle Corporation) have included similar aspects in the UltraSPARC T-Series processors. Virtualization platform's examples adapted to some hardware include Parallels Workstation, VirtualBox, Oracle VM Server for SPARC, Parallels Desktop for Mac, Xen, Windows Virtual PC, Hyper-V, VMware Fusion, VMware Workstations, and KVM.

First-generation 64-bit and 32-bit x86 hardware support have been detected to facilitate performance benefits on software virtualization in 2006.

Operating-system-level virtualization

A physical server can be virtualized on the OS level in operating-system-level virtualization. It allows more than one secure and isolated virtualized server for running on an individual physical server.

The environment of the guest operating system shares a similar running instance of an operating system as any host system. Hence, a similar operating system kernel is used for implementing guest environments. Also, various applications that are running within the provided guest environment consider it as the stand-alone system.

Faculty: Dr Naveen Singh
+91-9953320298 drnaveenkrsingh@gmail.com

Full Virtualization and Binary Translation –
VMware is widely used as it tends to virtualize x86 architectures, which executes unmodified on-top of their hypervisors. With the introduction full virtualization is possible to achieve by support of hardware. But earlier, x86 guest operating systems unmodified in a virtualized environment could be executed only with the use of dynamic binary translation. Since the set of sensitive instruction is not a subset of privileged instruction, x86 architecture design is not satisfy the first theorem of virtualization. Due to this different behaviour occurs while such instructions are not run in the Ring 0, which is normal in a virtualization environment where the guest OS is run in Ring 1. Basically, a trap is created, and the method in which it manages differentiation of the solution in which virtualization is applied for x86. In dynamic binary translation, the trap encounters the translation of interrupts or offending instructions into a corresponding set of instructions that establishes the same target without making exceptions. In addition, to expand performance, the corresponding set of instruction is cached, so the translation is not important anymore for further encounters of the same instructions. Below is the figure which demonstrates it.

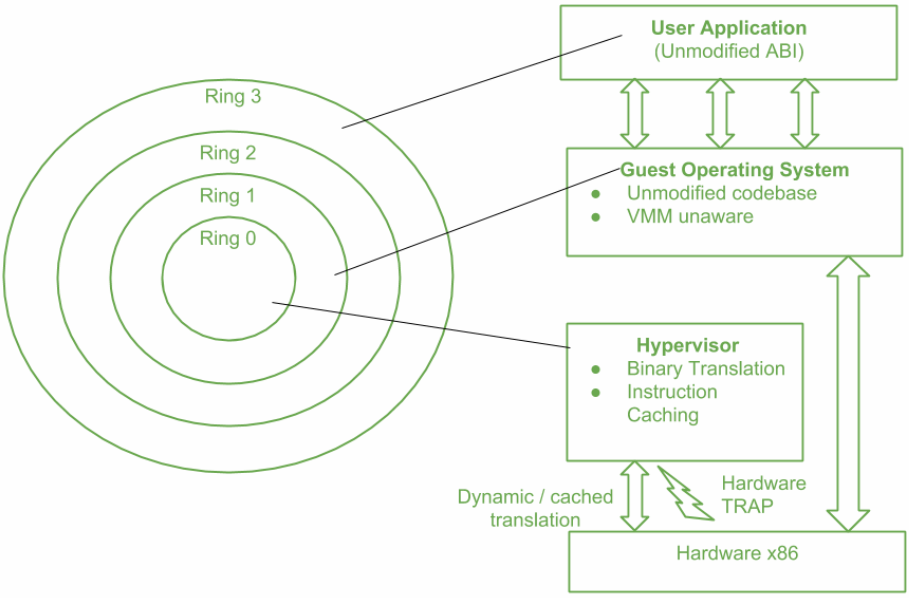


Figure – Full Virtualization Reference Model
The major benefit of this approach is that guests can run unmodified in a virtualized environment, which is an important feature for operating system whose source code does not existed. Binary translation is portable for full virtualization. As well as translation of instructions at runtime presents an additional overhead that is not existed in other methods like para virtualization or hardware-assisted virtualization. Contradict, binary translation is only implemented to a subset of the instruction set, while the others are managed through direct execution on the primary hardware. This depletes somehow the impact on performance of binary translation.

Faculty: Dr Naveen Singh
+91-9953320298 drnaveenkrsingh@gmail.com

Advantages of Binary Translation –

1. This kind of virtualization delivers the best isolation and security for Virtual Machine.
2. Truly isolated numerous guest OS can execute concurrently on the same hardware.
3. It is only implementation that needs no hardware assist or operating system assist to virtualize sensitive instruction as well as privileged instruction.

Disadvantages of Binary Translation –

1. It is time consuming at run-time.
2. It acquires a large performance overhead.
3. It employs a code cache to stock the translated most used instructions to enhance the performance, but it increases memory utilization along with the hardware cost.
4. The performance of full virtualization on the x86 architecture is 80 to 95 percent that of the host machine.

History of Virtualization

(from "Modern Operating Systems" 4th Edition, p474 by Tanenbaum and Bos)

- **1960's, IBM: CP/CMS** control program: a virtual machine operating system for the IBM System/360 Model 67
- **2000, IBM: z-series** with 64-bit virtual address spaces and backward compatible with the System/360
- **1974: Popek and Golberg** from UCLA published *"Formal Requirements for Virtualizable Third Generation Architectures"* where they listed the conditions a computer architecture should satisfy to support virtualization efficiently. The popular x86 architecture that originated in the 1970s did not support these requirements for decades.
- **1990's, Stanford researchers, VMware:** Researchers developed a new hypervisor and founded VMware, the biggest virtualization company of today's. First virtualization solution was is 1999 for x86.
- Today many virtualization solutions: Xen from Cambridge, KVM, Hyper-V,
- IBM was the first to produce and sell virtualization for the mainframe. But, VMware popularised virtualization for the masses.

Faculty: Dr Naveen Singh
+91-9953320298 drnaveenkrsingh@gmail.com

Need for Virtualization

1. Enhanced Performance

Currently, the end user system i.e. PC is sufficiently powerful to fulfill all the basic computation requirements of the user, with various additional capabilities which are rarely used by the user. Most of their systems have sufficient resources which can host a virtual machine manager and can perform a virtual machine with acceptable performance so far.

2. Limited use of Hardware and Software Resources

The limited use of the resources leads to under-utilization of hardware and software resources. As all the PCs of the user are sufficiently capable to fulfill their regular computational needs that's why many of their computers are used often which can be used 24/7 continuously without any interruption. The efficiency of IT infrastructure could be increase by using these resources after hours for other purposes. This environment is possible to attain with the help of Virtualization.

3. SHORTAGE OF SPACE

The regular requirement for additional capacity, whether memory storage or compute power, leads data centers raise rapidly. Companies like Google, Microsoft and Amazon develop their infrastructure by building data centers as per their needs. Mostly, enterprises unable to pay to build any other data center to accommodate additional resource capacity. This heads to the diffusion of a technique which is known as server consolidation.

4. ECO-FRIENDLY INITIATIVES-

At this time, corporations are actively seeking for various methods to minimize their expenditures on power which is consumed by their systems. Data centers are main power consumers and maintaining a data center operations needs a continuous power supply as well as a good amount of energy is needed to keep them cool for well-functioning. Therefore, server consolidation drops the power consumed and cooling impact by having a fall in number of servers. Virtualization can provide a sophisticated method of **server consolidation.**

Faculty: Dr Naveen Singh
+91-9953320298 drnaveenkrsingh@gmail.com

5. ADMINISTRATIVE COSTS

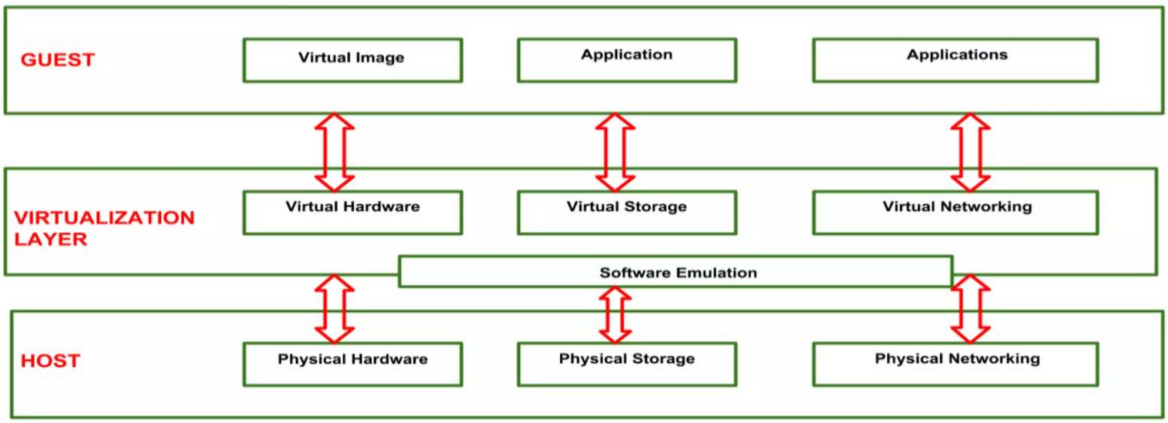
Furthermore, the rise in demand for capacity surplus, that convert into more servers in a data center, accountable for a significant increase in administrative costs. Hardware monitoring, server setup and updates, defective hardware replacement, server resources monitoring, and backups are included in common system administration tasks. These are personnel-intensive operations. The administrative costs is increased as per the number of servers. Virtualization decreases number of required servers for a given workload, hence reduces the cost of administrative employees.

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.

Faculty: Dr Naveen Singh
+91-9953320298 drnaveenkrsingh@gmail.com

Virtualization Reference Model



1. GUEST

The guest represents the system component that interacts with the virtualization layer rather than with the host, as would normally happen. Guests usually consist of one or more virtual disk files, and a VM definition file. Virtual Machines are centrally managed by a host application that sees and manages each virtual machine as a different application.

2. HOST

The host represents the original environment where the guest is supposed to be managed. Each guest runs on the host using shared resources donated to it by the host. The operating system, works as the host and manages the physical resource management, and the device support.

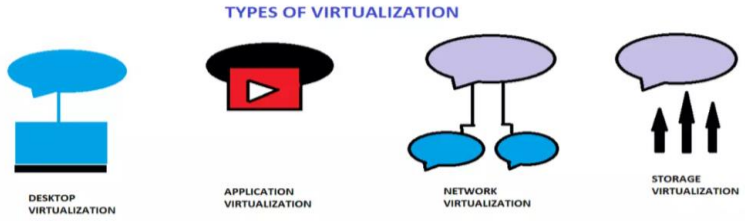
3. VIRTUALIZATION LAYER

The virtualization layer is responsible for recreating the same or a different environment where the guest will operate. It is an additional abstraction layer between a network and storage hardware, computing, and the application running on it. Usually it helps to run a single operating system per machine which can be very inflexible compared to the usage of virtualization.

Faculty: Dr Naveen Singh
+91-9953320298 drnaveenkrsingh@gmail.com

Types of Virtualization:

1. Application Virtualization
2. Network Virtualization
3. Desktop Virtualization
4. Storage Virtualization
5. Server Virtualization
6. Data Virtualization



1. 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.

2. 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.

Faculty: Dr Naveen Singh
+91-9953320298 drnaveenkrsingh@gmail.com

3. 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.

4. 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.

5. Server Virtualization

This is a kind of virtualization in which masking of server resources takes place. Here, the central-server(physical server) is divided into multiple different virtual servers by changing the identity number, processors. So, 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.

6. 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.

Faculty: Dr Naveen Singh
+91-9953320298 drnaveenkrsingh@gmail.com

System Virtual Machine

- A **System Virtual Machine** is also called as Hardware Virtual Machine. It is the **software emulation** of a computer system. It mimics the entire computer.
- In computing, **an emulator is hardware or software that enables one computer system** (called the host) **to behave like another computer system** (called the guest). An emulator typically enables the host system to run software or use a peripheral device designed for the guest system.
- It is an environment that allows multiple instances of the operating system (virtual machines) to run on a host system, sharing the physical resources.
- System Virtual Machine provides a platform for the execution of a complete operating system. It will create a number of different isolated identical execution environments in a single computer by partitioning computer memory to install and execute the different operating systems at the time.
- It allows us to install applications in each operating system, run the application in this operating system as if we work in real work on a real computer. For example, we can install Windows XP/7/8 or Linux Ubuntu/Kali in Windows 10 operating system with the help of VM.
- **Examples of System VMs software** - VMware, VirtualBox, Windows Virtual PC, Parallels, QEMU, Citrix Xen

System VM & Process VM

A **System Virtual Machine (System VM)** provides **a complete system platform which supports the execution of** a complete operating system (OS).

In contrast, a **Process Virtual Machine (Process VM)** is designed to run a single program, which means that it supports a single process.

Faculty: Dr Naveen Singh
+91-9953320298 drnaveenkrsingh@gmail.com

Process Virtual Machine

- A Process Virtual Machine is also called a **Language Virtual Machine** or an **Application Virtual Machine** or **Managed Runtime Environment**.
- Process VM is a software ***simulation*** of a computer system. It provides a runtime environment to execute a single program and supports a single process.
- The purpose of a process virtual machine is to provide a platform-independent programming environment that abstracts the details of the underlying hardware or operating system and allows a program to execute in the same way on any platform.
- Process virtual machines are implemented using an interpreter; for improving performance these virtual machines will use just-in-time compilers internally.
- Examples of Process VMs - **JVM** (Java Virtual Machine) is used for the Java language **PVM** (Parrot Virtual Machine) is used for PERL Language, **CLR** (Common Language Runtime) is used for .NET Framework

Faculty: Dr Naveen Singh
+91-9953320298 drnaveenkrsingh@gmail.com

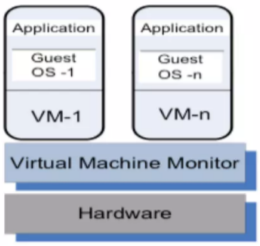
Virtual Machine Monitor (VMM)

- A **Virtual Machine Monitor (VMM)** is a software program that enables the creation, management and governance of virtual machines (VM) and manages the operation of a virtualized environment on top of a physical host machine.
- VMM is also known as **Virtual Machine Manager** and **Hypervisor**. However, the provided architectural implementation and services differ by vendor product.
- VMM is the primary software behind virtualization environments and implementations. When installed over a host machine, VMM facilitates the creation of VMs, each with separate operating systems (OS) and applications. VMM manages the backend operation of these VMs by allocating the necessary computing, memory, storage and other input/output (I/O) resources.
- VMM also provides a centralized interface for managing the entire operation, status and availability of VMs that are installed over a single host or spread across different and interconnected hosts.

Virtual Machine Monitor (VMM / Hypervisor)

A **virtual machine monitor (VMM/hypervisor)** partitions the resources of computer system into one or more **virtual machines (VMs)**. Allows several operating systems to run concurrently on a single hardware platform.

- A VM is an execution environment that runs an OS
- VM – an isolated environment that appears to be a whole computer, but actually only has access to a portion of the computer resources



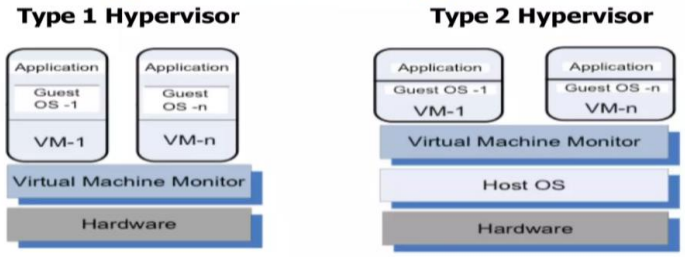
- **A VMM allows:**
  - Multiple services to share the same platform
  - Live migration - the movement of a server from one platform to another
  - System modification while maintaining backward compatibility with the original system
  - Enforces isolation among the systems, thus security
- A **guest operating system** is an OS that runs in a VM under the control of the VMM.

Faculty: Dr Naveen Singh
+91-9953320298 drnaveenkrsingh@gmail.com

VMM Virtualizes the CPU and the Memory

- **A VMM (also hypervisor)**
  - Traps the privileged instructions executed by a guest OS and enforces the correctness and safety of the operation
  - Traps interrupts and dispatches them to the individual guest operating systems
  - Controls the virtual memory management
  - Maintains a shadow page table for each guest OS and replicates any modification made by the guest OS in its own shadow page table. This shadow page table points to the actual page frame and it is used by the Memory Management Unit (MMU) for dynamic address translation.
  - Monitors the system performance and takes corrective actions to avoid performance degradation. For example, the VMM may swap out a VM to avoid thrashing.

Type 1 and 2 Hypervisors



- **Taxonomy of VMMs:**
  1. Type 1 Hypervisor (bare metal, native): supports multiple virtual machines and runs directly on the hardware (e.g., VMware ESX , Xen, Denali)
  2. Type 2 Hypervisor (hosted) VM - runs under a host operating system (e.g., user-mode Linux)

Faculty: Dr Naveen Singh
+91-9953320298 drnaveenkrsingh@gmail.com

Virtual Machine Properties

Being able to use apps and operating systems without the need for hardware presents users with some advantages over a traditional computer. The benefits of virtual machines include:

1. Compatibility
Virtual machines host their own guest operating systems and applications, using all the components found in a physical computer (motherboard, VGA card, network card controller, etc). This allows VMs to be fully compatible with all standard x86 operating systems, applications and device drivers. You can therefore run all the same software that you would usually use on a standard x86 computer.

2. Isolation
VMs share the physical resources of a computer, yet remain isolated from one another. This separation is the core reason why virtual machines create a more secure environment for running applications when compared to a non-virtual system. If, for example, you're running four VMs on a server and one of them crashes, the remaining three will remain unaffected and will still be operational.

3. Encapsulation
A virtual machine acts as a single software package that encapsulates a complete set of hardware resources, an operating system, and all its applications. This makes VMs incredibly portable and easy to manage. You can move and copy a VM from one location to another like any other software file, or save it on any storage medium — from storage area networks (SANs) to a common USB flash drive.

4. Hardware independence
Virtual machines can be configured with virtual components that are completely independent of the physical components of the underlying hardware. VMs that reside on the same server can even run different types of operating systems. Hardware independence allows you to move virtual machines from one x86 computer to another without needing to make any changes to the device drivers, operating system or applications.

Faculty: Dr Naveen Singh
+91-9953320298 drnaveenkrsingh@gmail.com

Interpretation and Binary Translation

- **Interpretation in Cloud Computing,** In simple terms, the behavior of the hardware is produced by a software program. Emulation process involves **only those hardware components so that user or virtual machines does not understand the underlying environment**. ***This process is also termed as interpretation.***
- **Binary Translation** is one specific approach to implementing full virtualization that does not require hardware virtualization features.
- It involves examining the **executable code of** the virtual guest for "unsafe" instructions, translating these into "safe" equivalents, and then executing the translated code.
- **VMware** is an example of virtualization using binary translation (VMware, n.d.). Hypervisors can also be distinguished by their relation to the host-operating system.

HLL VM

A static compiler is probably the best solution when performance is paramount, portability is not a great concern, destinations of calls are known at compile time and programs bind to external symbols before running. Thus, most third generation languages like C and FORTRAN are implemented this way. However, **if the language is object-oriented, binds to external references late, and must run on many platforms, it may be advantageous to implement a compiler that targets a fictitious *high-level language virtual machine* (HLL VM) instead.**

In Smith's taxonomy, an HLL VM is a system that provides a process with an execution environment that does not correspond to any particular hardware platform. The interface offered to the high-level language application process is usually designed to hide differences between the platforms to which the VM will eventually be ported. For instance, UCSD Pascal p-code and Java bytecode both express virtual instructions as stack operations that take no register arguments. Gosling, one of the designers of the Java virtual machine, has said that he based the design of the JVM on the p-code machine. Smalltalk, Self and many other systems have taken a similar approach. A VM may also provide virtual instructions that support peculiar or challenging features of the language. For instance, a Java virtual machine has specialized virtual instructions

Faculty: Dr Naveen Singh
+91-9953320298 drnaveenkrsingh@gmail.com

Supervisors

A **supervisory program** or **supervisor** is a computer program, usually part of an operating system, that controls the execution of other routines and regulates work scheduling, input/output operations, error actions, and similar functions and regulates the flow of work in a data processing system. It is thus capable of executing both input/output operations and privileged operations. The operating system of a computer usually operates in this mode.

Supervisor mode is "an **execution mode on some processors which enables execution of all instructions, including privileged instructions**. It may also give access to a different address space, to memory management hardware and to other peripherals. This is the mode in which the operating system usually runs."

It can also refer to a program that allocates computer component space and schedules computer events by task queuing and system interrupts. Control of the system is returned to the supervisory program frequently enough to ensure that demands on the system are met.

Historically, this term was essentially associated with IBM's line of mainframe operating systems starting with OS/360. In other operating systems, the supervisor is generally called the kernel. In the 1970s, IBM further abstracted the supervisor state from the hardware, resulting in a hypervisor that enabled full virtualization, i.e. the capacity to run multiple operating systems on the same machine totally independently from each other. Hence the first such system was called *Virtual Machine* or *VM*.

Xen

- **Xen** (pronounced /ˈzɛn/) is a **type-1 hypervisor**, providing services that allow multiple computer operating systems to execute on the same computer hardware concurrently.
- It was originally developed by the University of Cambridge Computer Laboratory and is now being developed by the Linux Foundation with support from Intel, Citrix, Arm Ltd, Huawei, AWS, Alibaba Cloud, AMD, Bitdefender and epam.
- The Xen Project community develops and maintains Xen Project as free and open-source software, subject to the requirements of the GNU General Public License (GPL), version 2. Xen Project is currently available for the IA-32, x86-64 and ARM instruction sets.

Faculty: Dr Naveen Singh
+91-9953320298 drnaveenkrsingh@gmail.com

KVM

- **Xen** (pronounced /ˈzɛn/) is a **type-1 hypervisor**, providing services that allow multiple computer operating systems to execute on the same computer hardware concurrently.
- It was originally developed by the University of Cambridge Computer Laboratory and is now being developed by the Linux Foundation with support from Intel, Citrix, Arm Ltd, Huawei, AWS, Alibaba Cloud, AMD, Bitdefender and epam.
- The Xen Project community develops and maintains Xen Project as free and open-source software, subject to the requirements of the GNU General Public License (GPL), version 2. Xen Project is currently available for the IA-32, x86-64 and ARM instruction sets.

VMware

- **Xen** (pronounced /ˈzɛn/) is a **type-1 hypervisor**, providing services that allow multiple computer operating systems to execute on the same computer hardware concurrently.
- It was originally developed by the University of Cambridge Computer Laboratory and is now being developed by the Linux Foundation with support from Intel, Citrix, Arm Ltd, Huawei, AWS, Alibaba Cloud, AMD, Bitdefender and epam.
- The Xen Project community develops and maintains Xen Project as free and open-source software, subject to the requirements of the GNU General Public License (GPL), version 2. Xen Project is currently available for the IA-32, x86-64 and ARM instruction sets.

Faculty: Dr Naveen Singh
+91-9953320298 drnaveenkrsingh@gmail.com

VirtualBox

- VirtualBox is a powerful x86 and AMD64/Intel64 virtualization product for enterprise as well as home use. Not only is VirtualBox an extremely feature rich, high performance product for enterprise customers,
- it is also the only professional solution that is freely available as Open Source Software under the terms of the GNU General Public License (GPL) version 2.
- Presently, VirtualBox runs on Windows, Linux, Macintosh, and Solaris hosts and supports a large number of guest operating systems including but not limited to Windows (NT 4.0, 2000, XP, Server 2003, Vista, Windows 7, Windows 8, Windows 10), DOS/Windows 3.x, Linux (2.4, 2.6, 3.x and 4.x), Solaris and OpenSolaris, OS/2, and OpenBSD.
- VirtualBox is being actively developed with frequent releases and has an ever growing list of features, supported guest operating systems and platforms it runs on.
- VirtualBox is a community effort backed by a dedicated company: everyone is encouraged to contribute while Oracle ensures the product always meets professional quality criteria.

Hyper-V

Contd...

- **Microsoft Hyper-V (Type-1)**, codenamed Viridian, and briefly known before its release as Windows Server Virtualization, is a **native hypervisor**; it can create virtual machines on x86-64 systems running Windows.
- A Type 1 hypervisor **runs directly on the underlying computer's physical hardware**, interacting directly with its CPU, memory, and physical storage. For this reason, Type 1 hypervisors are also referred to as bare-metal hypervisors. A Type 1 hypervisor takes the place of the host operating system.
- A Type 2 hypervisor, also called a hosted hypervisor, is **a virtual machine (VM) manager that is installed as a software application on an existing operating system (OS)**. This makes it easy for an end user to run a VM on a personal computing (PC) device.
- The main difference between Type 1 vs. Type 2 hypervisors is that **Type 1 runs on bare metal and Type 2 runs on top of an operating system**.
- The key difference between Hyper-V and a Type 2 hypervisor is **that Hyper-V uses hardware-assisted virtualization**. This allows Hyper-V virtual machines to communicate directly with the server hardware, allowing virtual machines to perform far better than a Type 2 hypervisor would allow.

Faculty: Dr Naveen Singh
+91-9953320298 drnaveenkrsingh@gmail.com

- **Xen** provides a form of virtualization known as **Paravirtualization**, in which guests run a modified operating system.
- The guests are modified to use a special hypercall ABI, instead of certain architectural features.
- Through **Paravirtualization**, Xen can achieve high performance even on its host architecture (x86) which has a reputation for non-cooperation with traditional virtualization techniques.
- Xen can run paravirtualized guests ("PV guests" in Xen terminology) even on CPUs without any explicit support for virtualization.
- Paravirtualization avoids the need to emulate a full set of hardware and firmware services, which makes a PV system simpler to manage and reduces the attack surface exposed to potentially malicious guests. **On 32-bit x86**, the Xen host kernel code runs in Ring 0, while the hosted domains run in *Ring 1* (kernel) and *Ring 3* (applications).

- This approach has benefits for the users as well. For instance, applications can be distributed in a platform neutral format. In the case of the Java class libraries or UCSD Pascal programs, the amount of virtual software far exceeds the size of the VM.
- The advantage is that the relatively small amount of effort required to port the VM to a new platform enables a large body of virtual applications to run on the new platform also.
- There are various approaches a HLL VM can take to actually execute a virtual program. An interpreter fetches, decodes, then emulates each virtual instruction in turn. Hence, interpreters are slow but can be very portable.
-
- Faster, but less portable, a dynamic compiler can translate to native code and dispatch regions of the virtual application. A dynamic compiler can exploit runtime knowledge of program values so it can sometimes do a better job of optimizing the program than a static compiler

Faculty: Dr Naveen Singh
+91-9953320298 drnaveenkrsingh@gmail.com