UNIT- III

What is Virtualization?

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

Virtualization Concept

- Creating a virtual machine over existing operating system and hardware is referred as Hardware Virtualization.
- Virtual Machines provide an environment that is logically separated from the underlying hardware.
- The machine on which the virtual machine is created is known as **host machine** and **virtual machine** is referred as a **guest machine**. This virtual machine is managed by a software or firmware, which is known as **hypervisor**.
- Virtualization technology is the ability of a computer program or a combination of software and hardware to emulate an executing environment separate from the one that hosts such programs. For example, we can run Windows OS on top of a virtual machine, which itself is running on Linux OS.
- The term virtualization is often synonymous with hardware virtualization, which plays a fundamental role in efficiently delivering Infrastructure-as-a-Service (IaaS) solutions for cloud computing.

BENEFITS OF THE VIRTUALIZATION

Managed Execution and Isolation:

- A virtual execution environment can be configured as a sandbox, thus preventing any harmful operation to cross the borders of the virtual host.
- Allocation of resources and their partitioning among different guests is simplified, being the virtual host controlled by a program.

Reduce Infrastructure Costs:

- Migrating resources into the cloud—particularly public cloud computing, where a hosting company provides shared or dedicated hardware—delivers a significant bottom-line reduction in capital expenditure.
- No need to buy as much computing hardware. You don't have to power, manage or maintain it on-premises. When you look at the total cost relative to computing power delivered, virtualization offers significant cost benefits as a result of shared resources and elastic scalability.
- These cost savings can be a boon to your budget, particularly in difficult economic conditions.

Elastic Scalability:

- Migrating temporary workloads to the cloud lets you benefit from the elastic scalability of virtualization. You can add additional capacity whenever you need it, and easily and cleanly decommission it when you don't.
- The majority of cloud solution providers charge on a pay-for-use basis, so this can also deliver a significant benefit to your bottom line by reducing your costs during lower-utilization times.

Redundancy and Reliability:

- Virtualization gives you an additional layer of flexibility to quickly respond to the challenges like server failure, disaster component failure etc.
- Critical business systems are self-contained within their virtualized environment, so you can easily replicate them to multiple physical locations or migrate them to new hardware any time. This gives you improved reliability by eliminating failure points.

- It also enhances your disaster recovery planning efforts.

Portability:

- Virtual machine instances are normally represented by one or more files that can be easily transported with respect to physical systems. Moreover, they also tend to be self-contained since they do not have other dependencies besides the virtual machine manager for their use.
- Portability and self-containment simplify their administration. Java programs are "compiled once and run everywhere"; they only require that the Java virtual machine be installed on the host. The same applies to hardware-level virtualization.

DISADVANTAGES OF VIRTUALIZATION

Performance degradation

- Performance is definitely one of the major concerns in using virtualization technology. Virtualization interposes an abstraction layer between the guest and the host, the guest can experience increased latencies.
- In the case of hardware virtualization, where the intermediate emulates a bare machine on top of which an entire system can be installed, the causes of performance degradation can be traced back to the overhead introduced by the following activities: Maintaining the status of virtual processors, Support of privileged instructions (trap and simulate privileged instructions), Support of paging within VM, Console functions

Inefficiency and degraded user experience

- Virtualization can sometime lead to an inefficient use of the host. In particular, some of the specific features of the host cannot be exposed by the abstraction layer and then become inaccessible.
- In the case of hardware virtualization, this could happen for device drivers: The virtual machine can sometime simply provide a default graphic card that maps only a subset of the features available in the host.
- In the case of programming-level virtual machines, some of the features of the underlying operating systems may become inaccessible unless specific libraries are used.

Security holes and new threats

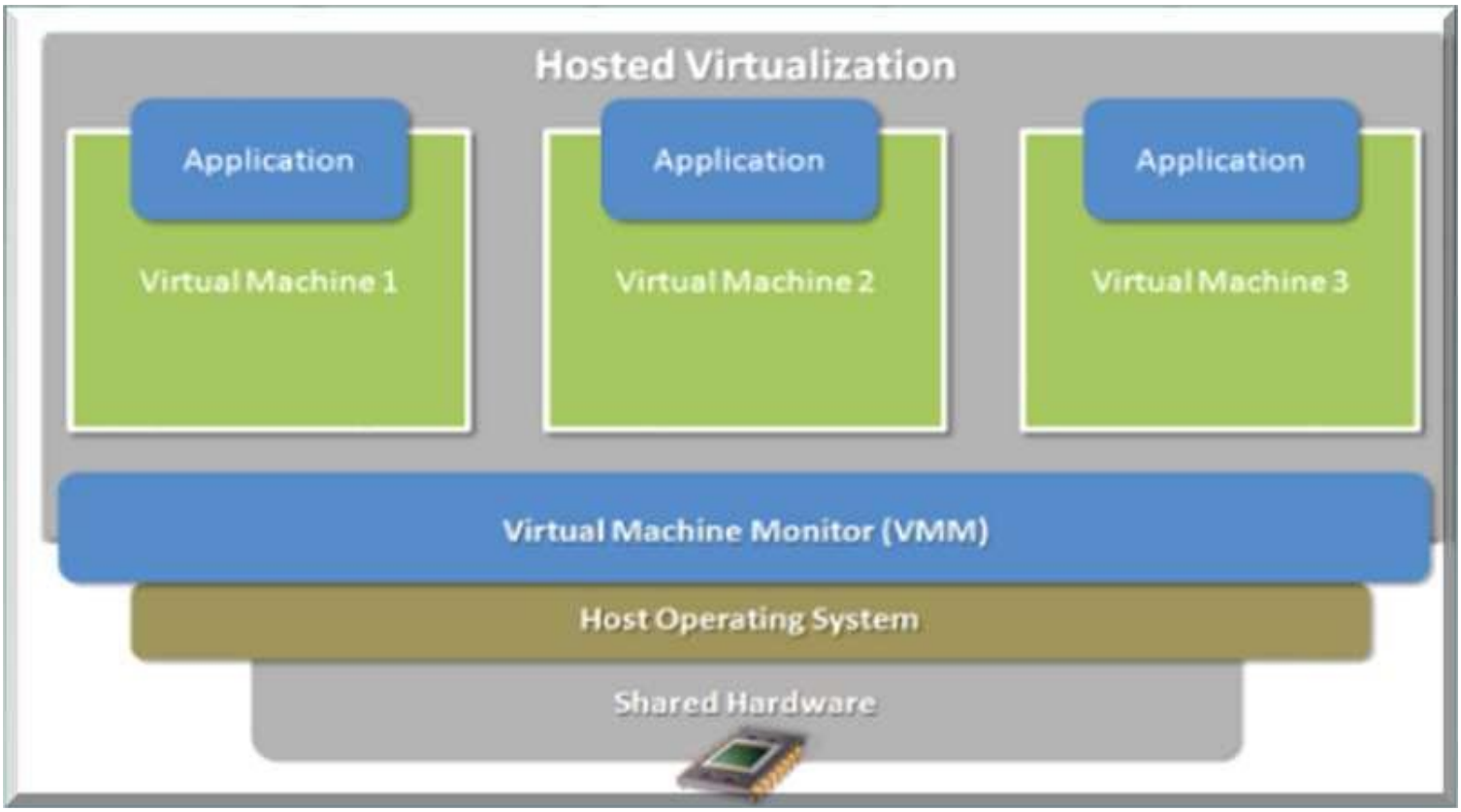
- Virtualization opens the door to a new and unexpected form of phishing. The capability of emulating a host in a completely transparent manner led the way to malicious programs that are designed to extract sensitive information from the guest.
- In the case of hardware virtualization, malicious programs can preload themselves before the operating system and act as a thin virtual machine manager toward it. The operating system is then controlled and can be manipulated to extract sensitive information of interest to third parties.

Virtualization Architecture

- Hosted Architecture.
- Bare-Metal Architecture.

Hosted Architecture

- In this architecture, host operating system is first installed.
- A piece of software called a hypervisor or virtual machine monitor (VMM) is installed on top of the host OS.
- It allows users to run various guest operating systems within their own application windows.
    - Eg. VMware Workstation, Oracle Virtual Box , Microsoft Virtual PC.



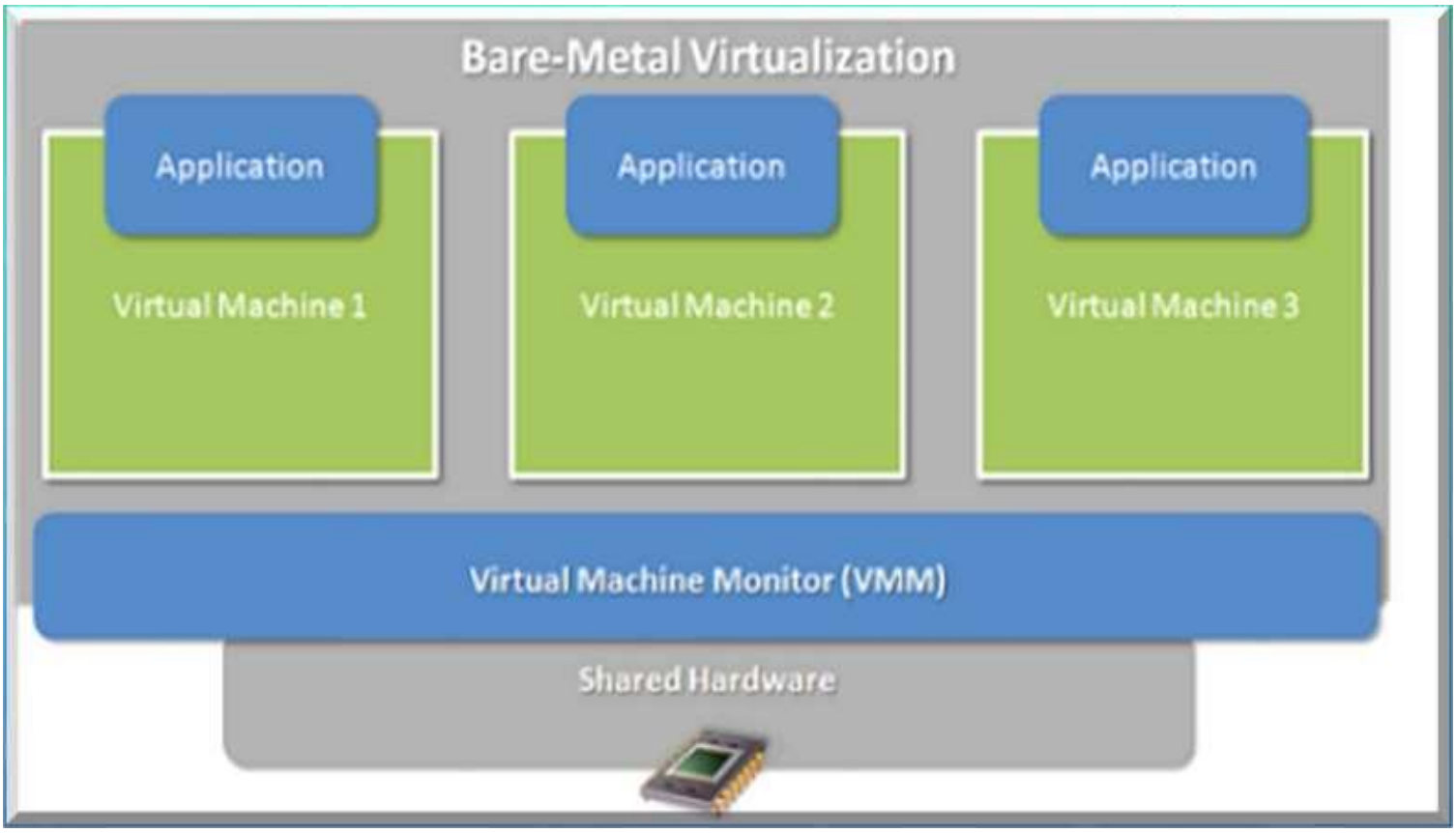
Hosted Virtual Machine Monitor is installed on top of host OS

Hosted Architecture(Pros & Cons)

- **Advantage**
  - ease of installation and configuration.
  - Unmodified Host OS & Guest OS.
  - run on a wide variety of pc.
  - Disadvantage
  - performance degradation.
  - lack of support for real-time operating systems.

Bare-Metal Architecture

- In this architecture, type1 hypervisor or VMM is installed on the bare hardware.
- VMM communicates directly with system hardware rather than relying on a host operating system.
- A bare-metal hypervisor is virtualization software that has been installed directly onto the computing hardware.
- This type of hypervisor controls not only the hardware, but one or more guest operating systems (OSes).
  - E.g: VMWARE ESX, VMWARE ESXi, Microsoft Hyper-V.



Bare-metal virtual machine monitor is installed directly on system hardware

Bare-Metal Architecture (Pros & Cons)

- **Advantages**

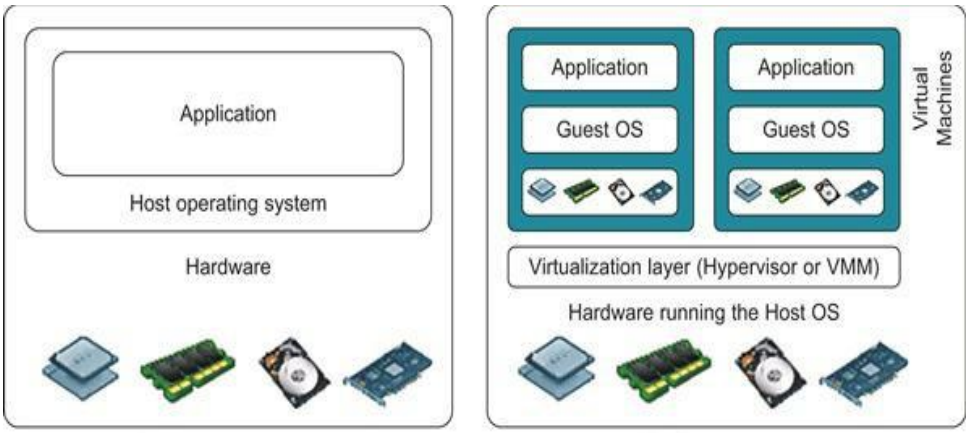
  - Improved I/O Performance.
  - Support Real Time OS.
  - Disadvantage
  - Difficult to install & Configure.
  - Depends upon hardware platform.

IMPLEMENTATION LEVELS OF VIRTUALIZATION

Virtualization is a computer architecture technology by which multiple virtual machines (VMs) are multiplexed in the same hardware machine. The purpose of a VM is to enhance resource sharing by many users and improve computer performance in terms of resource utilization and application flexibility. Hardware resources (CPU, memory, I/O devices, etc.) or software resources (operating system and software libraries) can be virtualized in various functional layers.

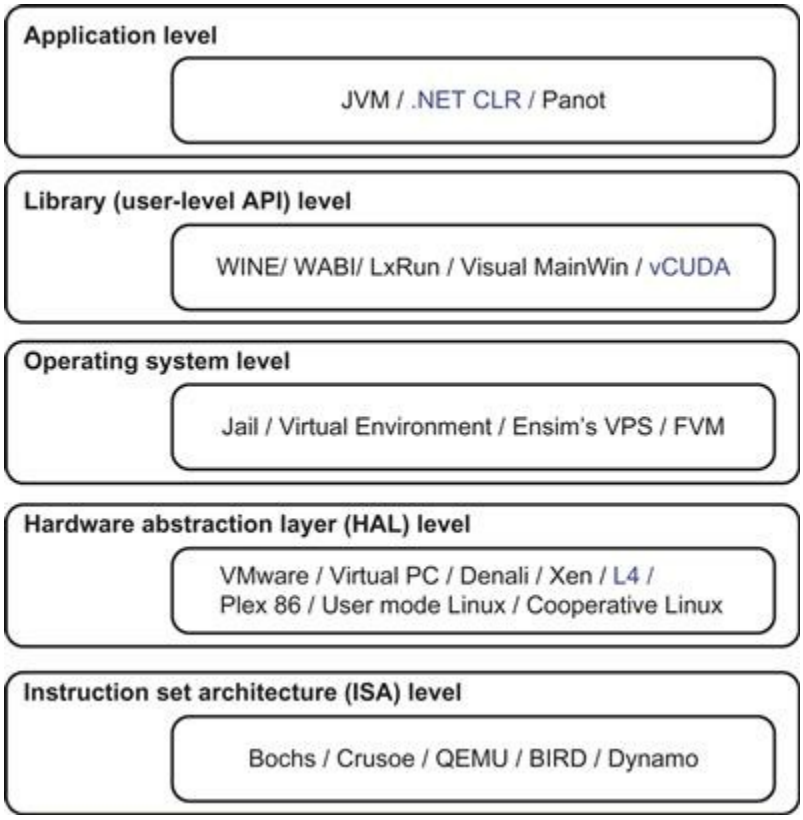
Levels of Virtualization Implementation

A traditional computer runs with a host operating system specially tailored for its hardware architecture, as shown in Figure 3.1(a). After virtualization, different user applications managed by their own operating systems (guest OS) can run on the same hardware, independent of the host OS. This is often done by adding additional software, called a virtualization layer as shown in Figure 3.1(b). This virtualization layer is known as hypervisor or virtual machine monitor (VMM). The VMs are shown in the upper boxes, where applications run with their own guest OS over the virtualized CPU, memory, and I /O resources.



(a) Traditional computer (b) After virtualization

The main function of the software layer for virtualization is to virtualize the physical hardware of a host machine into virtual resources to be used by the VMs, exclusively. This can be implemented at various operational levels. The virtualization software creates the abstraction of VMs by interposing a virtualization layer at various levels of a computer system. Common virtualization layers include the instruction set architecture (ISA) level, hardware level, operating system level, library support level, and application level.



Virtualization ranging from hardware to applications in five abstraction levels

Instruction Set Architecture Level

- At the ISA level, virtualization is performed by emulating a given ISA by the ISA of the host machine.
- In this technique every source instruction is interpreted by an emulator for executing native ISA instructions, leading to poor performance. Interpretation has a minimal startup cost but a huge overhead, since each instruction is emulated.

- The basic emulation method is through code interpretation. An interpreter program interprets the source instructions to target instructions one by one. One source instruction may require tens or hundreds of native target instructions to perform its function, this process is relatively slow.
- In binary translation technique every source instruction is converted to native instructions with equivalent functions. After a block of instructions is translated, it is cached and reused. Binary translation has a large initial overhead cost, but over time it is subject to better performance, since previously translated instruction blocks are directly executed.

Hardware Abstraction Level

- Hardware-level virtualization is performed right on top of the bare hardware. On the one hand, this approach generates a virtual hardware environment for a VM. On the other hand, the process manages the underlying hardware through virtualization.
- In this technique hardware provides architectural support for building a virtual machine manager able to run a guest operating system in complete isolation.
- The idea is to virtualize a computer's resources, such as its processors, memory, and I/O devices. The intention is to upgrade the hardware utilization rate by multiple users concurrently.

Operating System Level

- This refers to an abstraction layer between traditional OS and user applications. OS-level virtualization creates isolated containers on a single physical server and the OS instances to utilize the hardware and software in data centers.
- The containers behave like real servers. OS-level virtualization is commonly used in creating virtual hosting environments to allocate hardware resources among a large number of mutually distrusting users.
- It is also used, to a lesser extent, in consolidating server hardware by moving services on separate hosts into containers or VMs on one server.
- Differently from hardware virtualization, there is no virtual machine manager or hypervisor, and the virtualization is done within a single operating system, where the OS kernel allows for multiple isolated user space instances.

Library Support Level

- Most applications use API s exported by user-level libraries rather than using lengthy system calls by the OS.
- Virtualization with library interfaces is possible by controlling the communication link between applications and the rest of a system through API hooks.
- The software tool WINE has implemented this approach to support Windows applications on top of UNIX hosts.

User-Application Level

- Application-level virtualization is a technique allowing applications to be run in runtime environments that do not natively support all the features required by such applications.
- Virtualization at the application level virtualizes an application as a VM. On a traditional OS, an application often runs as a process. Therefore, application-level virtualization is also known as process-level virtualization.
- The most popular approach is to deploy high level language (HLL) VMs. In this scenario, the virtualization layer sits as an application program on top of the operating system, and the layer exports an abstraction of a VM that can run programs written and compiled to a particular abstract machine definition.
- Programs compiled into byte code can be executed on any operating system and platform for which a virtual machine able to execute that code has been provided.
- Any program written in the HLL and compiled for this VM will be able to run on it. The Microsoft .NET CLR and Java Virtual Machine (JVM) are two good examples of this class of VM.

VIRTUALIZATION STRUCTURES/TOOLS AND MECHANISMS

In general, there are three typical classes of VM architecture. Virtualization architecture figure showed the architectures of a machine before and after virtualization. Before virtualization, the operating system manages the hardware. After virtualization, a virtualization layer is inserted between the hardware and the operating system. I n such a case, the virtualization layer is responsible for converting portions of the real hardware into virtual hardware. Therefore, different operating systems such as Linux and Windows can run on the same physical machine, simultaneously.

BINARY TRANSLATION WITH FULL VIRTUALIZATION

Depending on implementation technologies, hardware virtualization can be classified into two categories: full virtualization and host-based virtualization. Full virtualization does not need to modify the host OS. It relies on binary translation to trap and to virtualize the execution of certain sensitive, nonvirtualizable instructions. The guest OSes and their applications consist of noncritical and critical instructions.

FULL VIRTUALIZATION

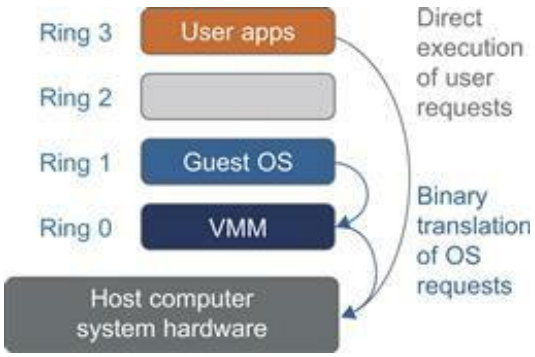
- Full virtualization refers to the ability to run a program, most likely an operating system, directly on top of a virtual machine and without any modification, as though it were run on the raw hardware. To make this possible, virtual machine managers are required to provide a complete emulation of the entire underlying hardware.
- The principal advantage of full virtualization is complete isolation, which leads to enhanced security, ease of emulation of different architectures, and coexistence of different systems on the same platform.
- With full virtualization, noncritical instructions run on the hardware directly while critical instructions are discovered and replaced with traps into the VMM to be emulated by software.

Why are only critical instructions trapped into the VMM?

This is because binary translation can incur a large performance overhead. Noncritical instructions do not control hardware or threaten the security of the system, but critical instructions do. Therefore, running noncritical instructions on hardware not only can promote efficiency, but also can ensure system security.

Binary Translation of Guest OS Requests Using a VMM

This approach was implemented by VMware and many other as shown in Figure, VMware puts the VMM at Ring 0 and the guest OS at Ring 1. The VMM scans the instruction stream and identifies the privileged, control- and behavior sensitive instructions. When these instructions are identified, they are trapped into the VMM, which emulates the behavior of these instructions. The method used in this emulation is called binary translation. Therefore, full virtualization combines binary translation and direct execution. The guest OS is completely decoupled from the underlying hardware. Consequently, the guest OS is unaware that it is being virtualized.



The performance of full virtualization may not be ideal, because it involves binary translation which is rather time-consuming. In particular, the full virtualization of I/O intensive applications is a really a big

challenge. Binary translation employs a code cache to store translated hot instructions to improve performance, but it increases the cost of memory usage.

Host-Based Virtualization

An alternative VM architecture is to install a virtualization layer on top of the host OS. This host OS is still responsible for managing the hardware. The guest OSes are installed and run on top of the virtualization layer. Dedicated applications may run on the VMs. Certainly, some other applications can also run with the host OS directly. This host-based architecture has some distinct advantages.

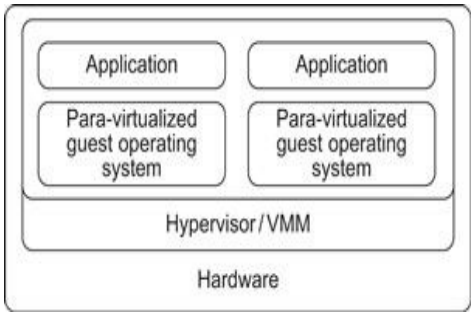
1) The user can install this VM architecture without modifying the host OS. The virtualizing software can rely on the host OS to provide device drivers and other low-level services. This will simplify the VM design and ease its deployment.
2) The host-based approach appeals to many host machine configurations. Compared to the hypervisor/VMM architecture, the performance of the host-based architecture may also be low.

When an application requests hardware access, it involves four layers of mapping which downgrades performance significantly. When the ISA of a guest OS is different from the ISA of the underlying hardware, binary translation must be adopted. Although the host-based architecture has flexibility, the performance is too low to be useful in practice.

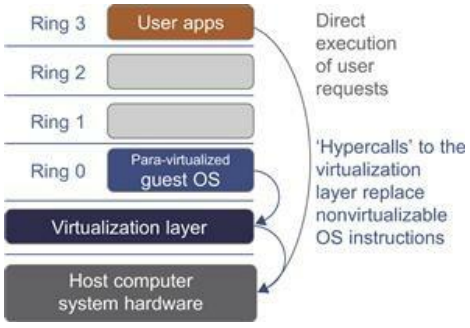
PARA-VIRTUALIZATION

- Para-virtualization needs to modify the guest operating systems. A para-virtualized VM provides special APIs requiring substantial OS modifications in user applications.
- Performance degradation is a critical issue of a virtualized system. No one wants to use a VM if it is much slower than using a physical machine. The virtualization layer can be inserted at different positions in a machine software stack. However, para-virtualization attempts to reduce the virtualization overhead, and thus improve performance by modifying only the guest OS kernel.

Figure (a) illustrates the concept of a para-virtualized VM architecture. The guest operating systems are para-virtualized. They are assisted by an intelligent compiler to replace the nonvirtualizable OS instructions by hypercalls as illustrated in Figure (b). The traditional x86 processor offers four instruction execution rings: Rings 0, 1, 2, and 3. The lower the ring number, the higher the privilege of instruction being executed. The OS is responsible for managing the hardware and the privileged instructions to execute at Ring 0, while user-level applications run at Ring 3. The best example of para-virtualization is the KVM.

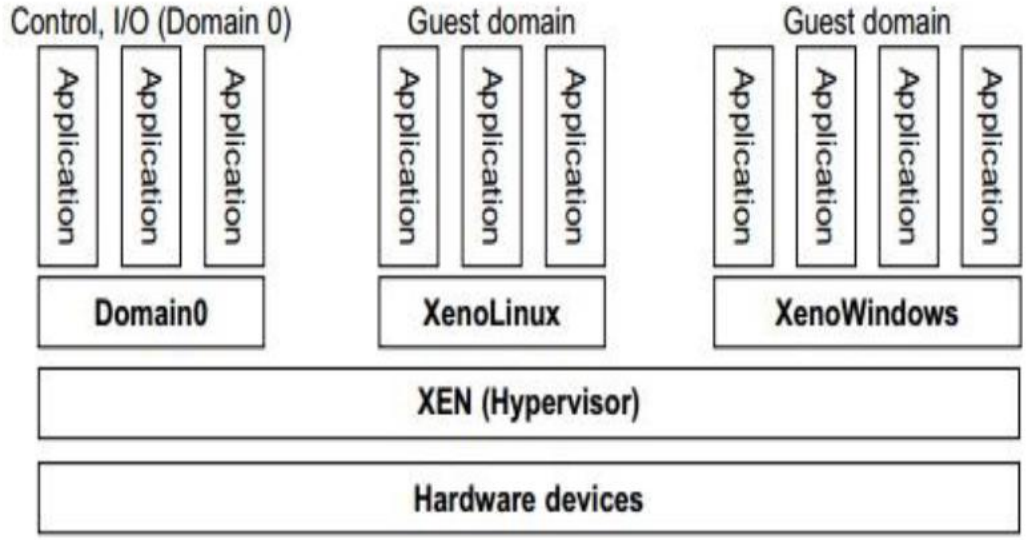


a)Para-virtualized VM Architecture



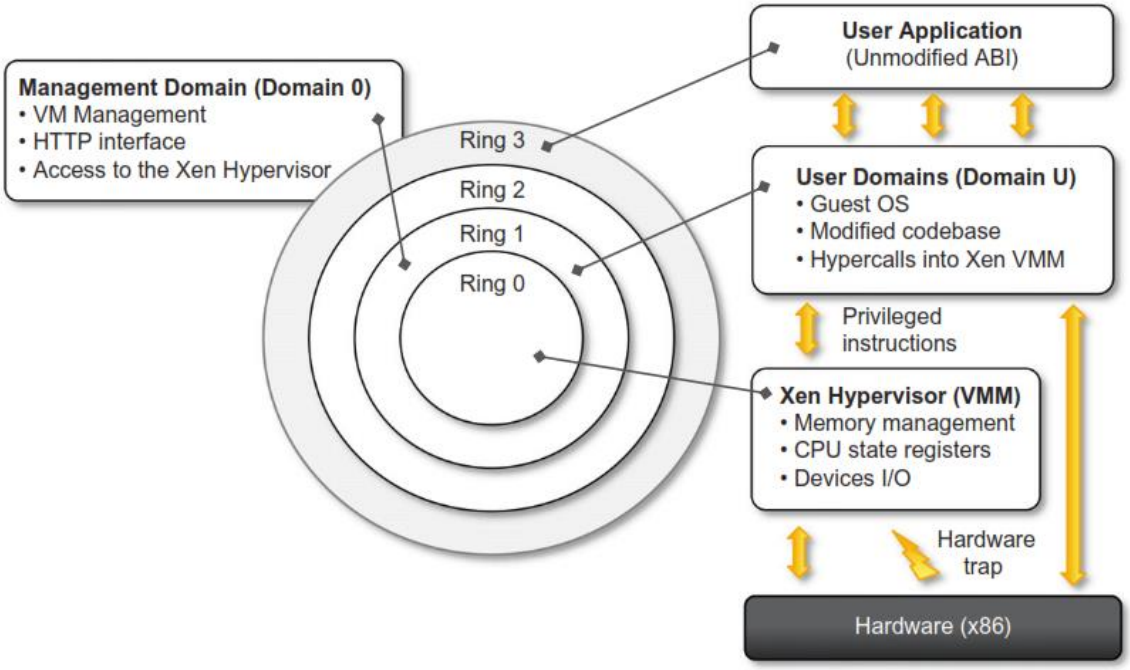
b) The use of a para-virtualized guest OS assisted by an intelligent Compiler to replace nonvirtualizable OS instructions by hypercalls

Xen Hypervisor Architecture (Para virtualization)



The Xen architecture's special domain 0 for control and I/O and several guest domain for user application

- Xen is an open-source initiative implementing a virtualization platform based on paravirtualization. Xen now has a large open-source community backing it. Xen-based technology is used for either desktop virtualization or server virtualization, and recently it has also been used to provide cloud computing solutions by means of Xen Cloud Platform (XCP).
- Xen is the most popular implementation of paravirtualization, which, in contrast with full virtualization, allows high-performance execution of guest operating systems. This is done by modifying portions of the guest operating systems run by Xen with reference to the execution of such instructions.
- Figure describes the architecture of Xen and its mapping onto a classic x86 privilege model. A Xen-based system is managed by the Xen hypervisor, which runs in the highest privileged mode and controls the access of guest operating system to the underlying hardware. Guest operating systems are executed within domains, which represent virtual machine instances.
- Specific control software, which has privileged access to the host and controls all the other guest operating systems, is executed in a special domain called Domain 0. This is the first one that is loaded once the virtual machine manager has completely booted, and it hosts a HyperText Transfer Protocol (HTTP) server that serves requests for virtual machine creation, configuration, and termination.
- Many of the x86 implementations support four different security levels, called rings, where Ring 0 represent the level with the highest privileges and Ring 3 the level with the lowest ones. Almost all the most popular operating systems, except OS/2, utilize only two levels: Ring 0 for the kernel code, and Ring 3 for user application and nonprivileged OS code.
- This provides the opportunity for Xen to implement virtualization by executing the hypervisor in Ring 0, Domain 0, and all the other domains running guest operating systems—generally referred to as Domain U—in Ring 1, while the user applications are run in Ring 3. Because of the structure of the x86 instruction set, some instructions allow code executing in Ring 3 to jump into Ring 0 (kernel mode).



- The guest OS, which has control ability, is called Domain 0, and the others are called Domain U. Domain 0 is a privileged guest OS of Xen. It is first loaded when Xen boots without any file system drivers being available. Domain 0 is designed to access hardware directly and manage devices.
- Therefore, one of the responsibilities of Domain 0 is to allocate and map hardware resources for the guest domains (the Domain U domains). If Domain 0 is compromised, the hacker can control the entire system. Security policies are needed to improve the security of Domain 0. Domain 0, behaving as a VMM, allows users to create, copy, save, read, modify, share, migrate, and roll back VMs.

VMware Hypervisor (Full virtualization)

VMware's technology is based on the concept of full virtualization, where the underlying hardware is replicated and made available to the guest operating system, which runs unaware of such abstraction layers and does not need to be modified. VMware implements full virtualization either in the desktop environment, by means of Type II hypervisors, or in the server environment, by means of Type I hypervisors. In both cases, full virtualization is made possible by means of direct execution (for nonsensitive instructions) and binary translation (for sensitive instructions), thus allowing the virtualization of architecture such as x86.

VMware ESX and VMware ESXi are both bare metal hypervisor that install directly on the server hardware.

KVM (Kernel-Based VM)

This is a Linux para-virtualization system—a part of the Linux version 2.6.20 kernel. Memory management and scheduling activities are carried out by the existing Linux kernel. The KVM does the rest, which makes it simpler than the hypervisor that controls the entire machine. KVM is a hardware-assisted para-virtualization tool, which improves Performance and supports unmodified guest OSes such as Windows, Linux, Solaris, and other UNIX variants.

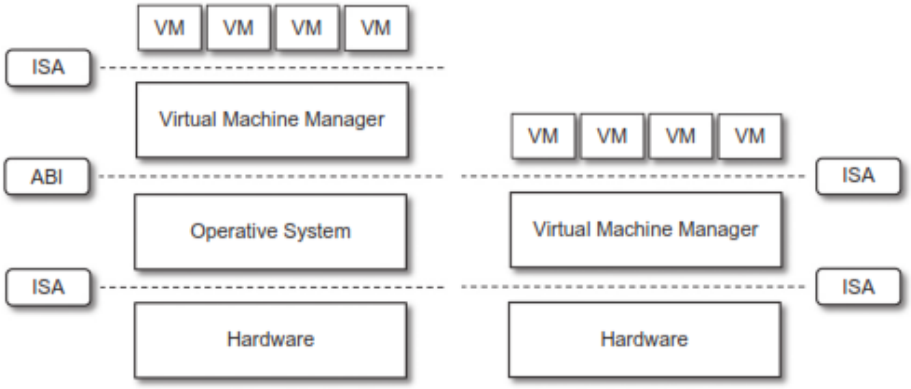
Using KVM, one can run multiple virtual machines running unmodified Linux or windows images. Each virtual machine has private virtualized hardware: a network card, hard disk, graphics adapter etc.

KVM can be managed either via a graphical management tool , similar to VMware products or virtual box.The most popular GUI is called Virtual machine manager (VMM) developed by Red Hat.

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. The hypervisor provides hypercalls for the guest OSes and applications. Depending on the functionality, a hypervisor can assume a microkernel architecture like the Microsoft Hyper-V. Or it can assume a monolithic hypervisor architecture like the VMware ESX for server virtualization.

There are two major types of hypervisor: Type I and Type II



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 since it runs natively on hardware or bare metal hypervisor.
- 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.

Three properties have to be satisfied:

- **Equivalence.** A guest running under the control of a virtual machine manager should exhibit the same behavior as when it is executed directly on the physical host.
- **Resource control.** The virtual machine manager should be in complete control of virtualized resources.
- **Efficiency.** A statistically dominant fraction of the machine instructions should be executed without intervention from the virtual machine manager.

VIRTUALIZATION OF CPU, MEMORY, AND I/O DEVICES

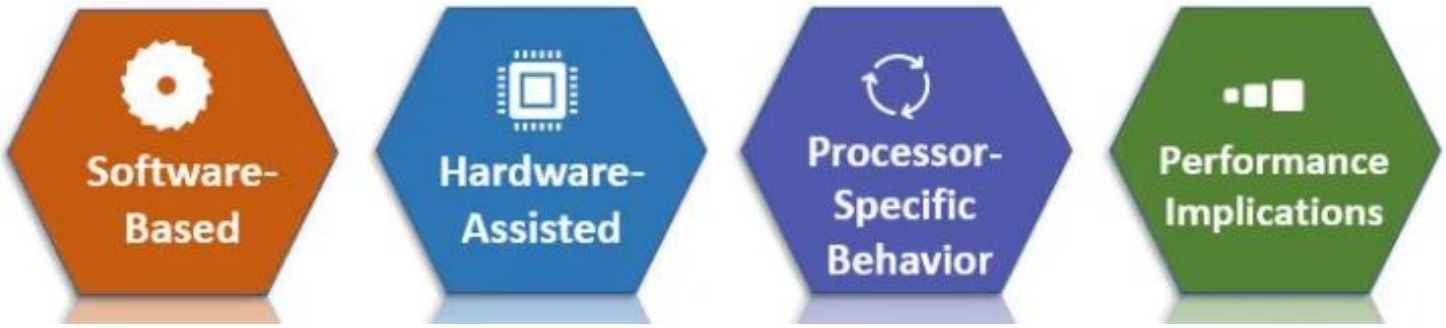
CPU VIRTUALIZATION

CPU Virtualization emphasizes running programs and instructions through a virtual machine, giving the feeling of working on a physical workstation. All the operations are handled by an emulator that controls software to run according to it. Nevertheless, CPU Virtualization does not act as an emulator. The emulator performs the same way as a normal computer machine does. It replicates the same copy or data and generates the same output just like a physical machine does. The emulation function offers great portability and facilitates working on a single platform, acting like working on multiple platforms.

With CPU Virtualization, all the virtual machines act as physical machines and distribute their hosting resources like having various virtual processors. Sharing of physical resources takes place to each virtual

machine when all hosting services get the request. Finally, the virtual machines get a share of the single CPU allocated to them, being a single-processor acting as a dual-processor.

Types of CPU Virtualization
The various types of CPU virtualization available are as follows

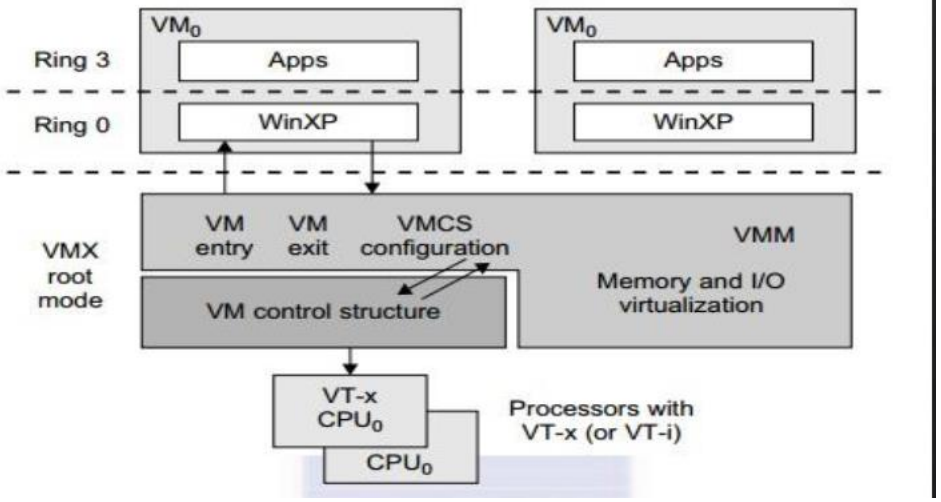


1. Software-Based CPU Virtualization

This CPU Virtualization is software-based where with the help of it, application code gets executed on the processor and the privileged code gets translated first, and that translated code gets executed directly on the processor. This translation is purely known as Binary Translation (BT). The code that gets translated is very large in size and also slow at the same time on execution. The guest programs that are based on privileged coding runs very smooth and fast. The code programs or the applications that are based on privileged code components that are significant such as system calls, run at a slower rate in the virtual environment.

2. Hardware-Assisted CPU Virtualization

There is hardware that gets assistance to support CPU Virtualization from certain processors. Here, the guest user uses a different version of code and mode of execution known as a guest mode. The guest code mainly runs on guest mode. The best part in hardware-assisted CPU Virtualization is that there is no requirement for translation while using it for hardware assistance. For this, the system calls runs faster than expected. Workloads that require the updation of page tables get a chance of exiting from guest mode to root mode that eventually slows down the program's performance and efficiency.



Intel hardware assisted CPU virtualization

3. Virtualization and Processor-Specific Behavior

Despite having specific software behavior of the CPU model, the virtual machine still helps in detecting the processor model on which the system runs. The processor model is different based on the CPU and the wide variety of features it offers, whereas the applications that produce the output generally utilize such features. In such cases, vMotion cannot be used to migrate the virtual machines that are running on feature-rich processors. Enhanced vMotion Compatibility easily handles this feature.

4. Performance Implications of CPU Virtualization

CPU Virtualization adds the amount of overhead based on the workloads and virtualization used. Any application depends mainly on the CPU power waiting for the instructions to get executed first. Such applications require the use of CPU Virtualization that gets the command or executions that are needed to be executed first. This overhead takes the overall processing time and results in an overall degradation in performance and CPU virtualization execution.

Advantages of CPU Virtualization:

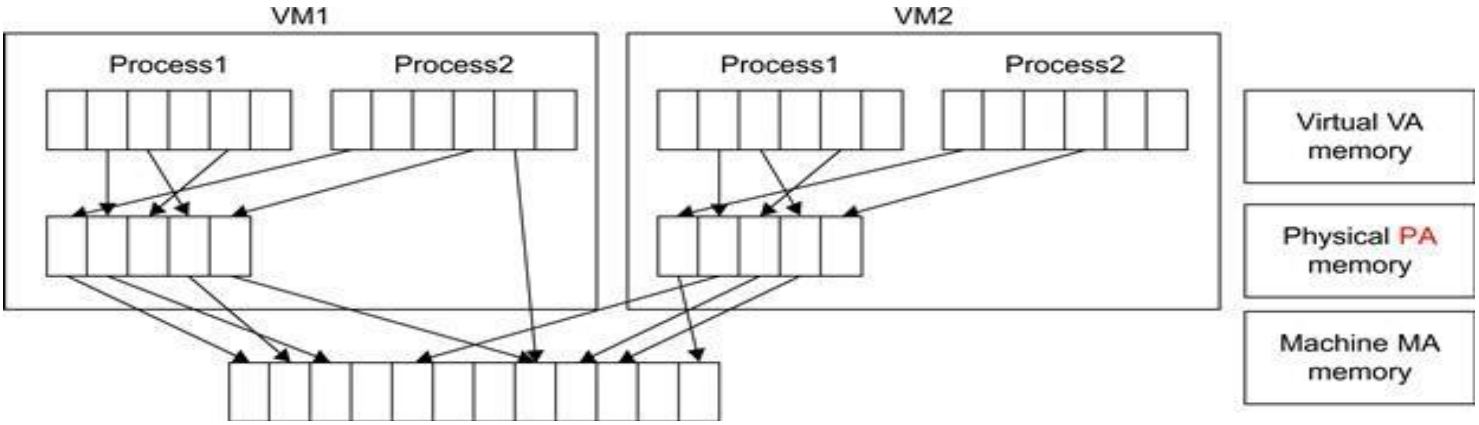
- Using CPU Virtualization, the overall performance and efficiency are improved to a great extent because it usually takes virtual machines to work on a single CPU, sharing resources acting like using multiple processors at the same time. This saves cost and money.
- As CPU Virtualization uses virtual machines to work on separate operating systems on a single sharing system, security is also maintained by it. The machines are also kept separate from each other. Because of that, any cyber-attack or software glitch is unable to damage the system, as a single machine cannot affect another machine.
- It purely works on virtual machines and hardware resources. It consists of a single server where all the computing resources are stored, and processing is done based on the CPU's instructions that are shared among all the systems involved. Since the hardware requirement is less and the physical machine usage is absent, that is why the cost is very less, and timing is saved.
- It provides the best backup of computing resources since the data is stored and shared from a single system. It provides reliability to users dependent on a single system and provides greater retrieval options of data for the user to make them happy.
- It also offers great and fast deployment procedure options so that it reaches the client without any hassle, and also it maintains the atomicity. Virtualization ensures the desired data reach the

desired clients through the medium and checks any constraints are there, and are also fast to remove it.

MEMORY VIRTUALIZATION

Virtual memory virtualization is similar to the virtual memory support provided by modern operating systems. In a traditional execution environment, the operating system maintains mappings of virtual memory to machine memory using page tables, which is a one-stage mapping from virtual memory to machine memory.

- In modern x86 CPUs include a memory management unit (MMU) and a translation look a side buffer (TLB) to optimize virtual memory performance. However, in a virtual execution environment, virtual memory virtualization involves sharing the physical system memory in RAM and dynamically allocating it to the physical memory of the VMs.
- A two-stage mapping process should be maintained by the guest OS and the VMM, respectively: virtual memory to physical memory and physical memory to machine memory.
- MMU virtualization should be supported, which is transparent to the guest OS. The guest OS continues to control the mapping of virtual addresses to the physical memory addresses of VMs. But the guest OS cannot directly access the actual machine memory.
- The VMM is responsible for mapping the guest physical memory to the actual machine memory. Figure shows the two-level memory mapping procedure.



Two level memory mapping procedure

- Since each page table of the guest OSes has a separate page table in the VMM corresponding to it, the VMM page table is called the shadow page table. Nested page tables add another layer of indirection to virtual memory.
- The MMU handles virtual-to-physical translations as defined by the OS. Then the physical memory addresses are translated to machine addresses using another set of page tables defined by the hypervisor.
- Since modern operating systems maintain a set of page tables for every process, the shadow page tables will get flooded. Consequently, the performance overhead and cost of memory will be very high.
- VMware uses shadow page tables to perform virtual-memory-to-machine-memory address translation. Processors use TLB hardware to map the virtual memory directly to the machine memory to avoid the two levels of translation on every access. When the guest OS changes the virtual memory to a physical memory mapping, the VMM updates the shadow page tables to enable a direct lookup.

Benefits to use memory virtualization:

1. Higher memory utilization by sharing contents and consolidating more virtual machines on a physical host.

2. Ensuring some memory space exists before halting services until memory frees up.

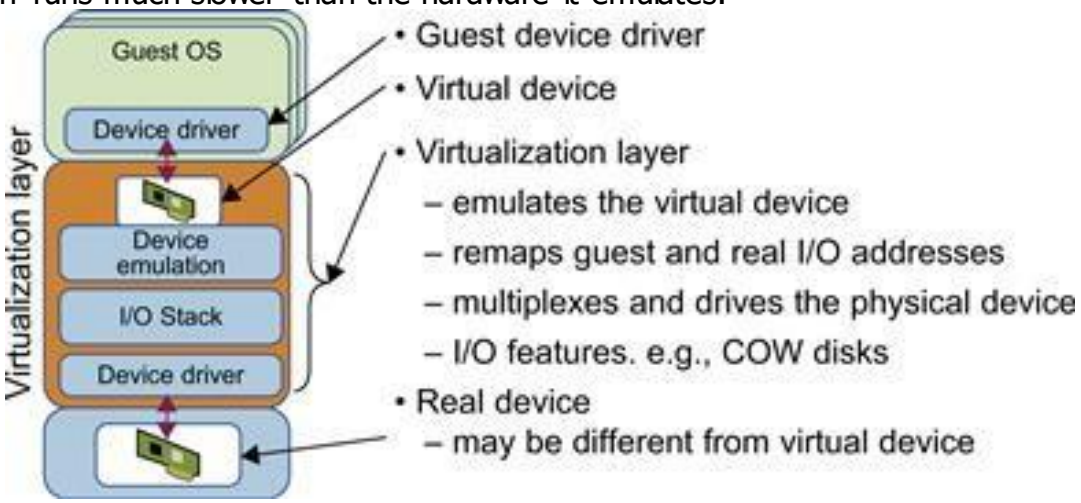
3. Access to more memory than the chassis can physically allow.

4. Advanced server virtualization functions, like live migrations.

I/O VIRTUALIZATION

I/O virtualization involves managing the routing of I/O requests between virtual devices and the shared physical hardware. At the time of this writing, there are three ways to implement I/O virtualization: full device emulation, para-virtualization, and direct I/O. Full device emulation is the first approach for I/O virtualization. Generally, this approach emulates well-known, real-world devices. All the functions of a device or bus infrastructure, such as device enumeration, identification, interrupts, and DMA, are replicated in software. This software is located in the VMM and acts as a virtual device.

- The I/O access requests of the guest OS are trapped in the VMM which interacts with the I/O devices. The full device emulation approach is shown in Figure.
- A single hardware device can be shared by multiple VMs that run concurrently. However, software emulation runs much slower than the hardware it emulates.



- The para-virtualization method of I/O virtualization is typically used in Xen. It is also known as the split driver model consisting of a frontend driver and a backend driver. The frontend driver is running in Domain U and the backend driver is running in Domain 0. They interact with each other via a block of shared memory. The frontend driver manages the I/O requests of the guest OSes and the backend driver is responsible for managing the real I/O devices and multiplexing the I/O data of different VMs.
- Para-I/O virtualization achieves better device performance than full device emulation, it comes with a higher CPU overhead.
- Direct I/O virtualization lets the VM access devices directly. It can achieve close-to-native performance without high CPU costs.

However, current direct I/O virtualization implementations focus on networking for mainframes. Since software-based I/O virtualization requires a very high overhead of device emulation, hardware-assisted I/O virtualization is critical.

VIRTUALIZATION IN MULTI-CORE PROCESSORS

Virtualizing a multi-core processor is relatively more complicated than virtualizing a unicore processor. Though multicore processors are claimed to have higher performance by integrating multiple processor cores in a single chip, muti-core virtualiuzation has raised some new challenges to computer architects, compiler constructors, system designers, and application programmers. There are mainly two difficulties: Application programs must be parallelized to use all cores fully, and software must explicitly assign tasks to the cores, which is a very complex problem.

Concerning the first challenge, new programming models, languages, and libraries are needed to make parallel programming easier. The second challenge has spawned research involving scheduling algorithms and resource management policies. Yet these efforts cannot balance well among performance, complexity, and other issues. What is worse, as technology scales, a new challenge called dynamic heterogeneity is emerging to mix the fat CPU core and thin GPU cores on the same chip, which further complicates the multi-core or many-core resource management. The dynamic heterogeneity of hardware infrastructure mainly comes from less reliable transistors and increased complexity in using the transistors.

VIRTUAL CLUSTERS AND RESOURCE MANAGEMENT

A physical cluster is a collection of servers (physical machines) interconnected by a physical network such as a LAN. When a traditional VM is initialized, the administrator needs to manually write configuration information or specify the configuration sources. When more VMs join a network, an inefficient configuration always causes problems with overloading or underutilization. Most virtualization platforms, including XenServer and VMware ESX Server, support a bridging mode which allows all domains to appear on the network as individual hosts. By using this mode, VMs can communicate with one another freely through the virtual network interface card and configure the network automatically.

Physical versus Virtual Clusters

Virtual clusters are built with VMs installed at distributed servers from one or more physical clusters. The VMs in a virtual cluster are interconnected logically by a virtual network across several physical networks. Figure illustrates the concepts of virtual clusters and physical clusters. Each virtual cluster is formed with physical machines or a VM hosted by multiple physical clusters. The virtual cluster boundaries are shown as distinct boundaries.

The provisioning of VMs to a virtual cluster is done dynamically to have the following interesting properties:

- The virtual cluster nodes can be either physical or virtual machines. Multiple VMs running with different OSes can be deployed on the same physical node.
- A VM runs with a guest OS, which is often different from the host OS, that manages the resources in the physical machine, where the VM is implemented.
- The purpose of using VMs is to consolidate multiple functionalities on the same server. This will greatly enhance server utilization and application flexibility.
- VMs can be colonized (replicated) in multiple servers for the purpose of promoting distributed parallelism, fault tolerance, and disaster recovery.

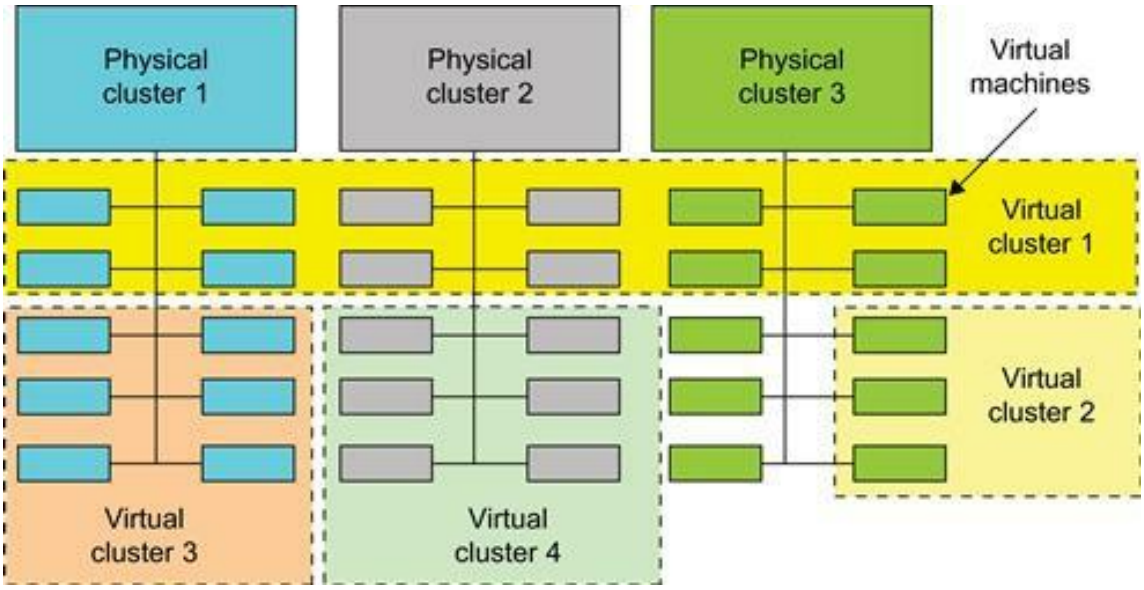


Fig. (a)

- The size (number of nodes) of a virtual cluster can grow or shrink dynamically, similar to the way an overlay network varies in size in a peer-to-peer (P2P) network.
- The failure of any physical nodes may disable some VMs installed on the failing nodes. But the failure of VMs will not pull down the host system.

Since system virtualization has been widely used, it is necessary to effectively manage VMs running on a mass of physical computing nodes (also called virtual clusters) and consequently build a high-performance virtualized computing environment. This involves virtual cluster deployment, monitoring and management over large-scale clusters, as well as resource scheduling, load balancing, server consolidation, fault tolerance, and other techniques.
In a virtual cluster system, it is quite important to store the large number of VM images efficiently.

Figure (b) shows the concept of a virtual cluster based on application partitioning or customization. The different colors in the figure represent the nodes in different virtual clusters. There are common installations for most users or applications, such as operating systems or user-level programming libraries. These software packages can be preinstalled as templates (called template VMs). With these templates, users can build their own software stacks. New OS instances can be copied from the template VM. User-specific components such as programming libraries and applications can be installed to those instances.

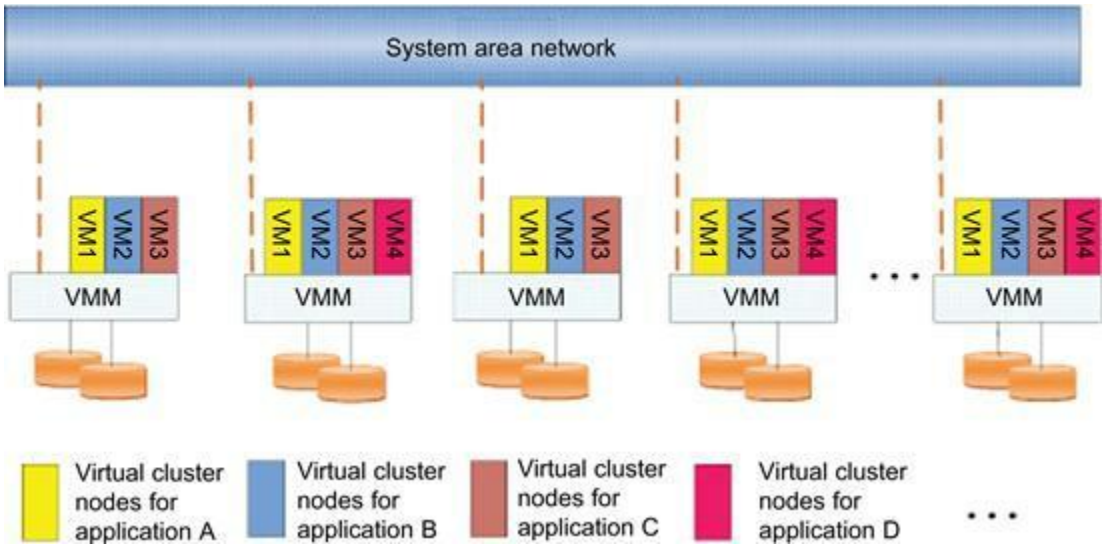


Fig. (b)

Three physical clusters are shown on the left side of Figure (a). Four virtual clusters are created on the right, over the physical clusters. The physical machines are also called host systems. In contrast, the VMs are guest systems. The host and guest systems may run with different operating systems. Each VM can be installed on a remote server or replicated on multiple servers belonging to the same or different physical clusters. The boundary of a virtual cluster can change as VM nodes are added, removed, or migrated dynamically over time.

Four ways to manage a virtual cluster

1) Using a guest-based manager, by which the cluster manager resides on a guest system. In this case, multiple VMs form a virtual cluster. For example, openMosix is an open source Linux cluster running different guest systems on top of the Xen hypervisor.
2) Build a cluster manager on the host systems. The host-based manager supervises the guest systems and can restart the guest system on another physical machine. A good example is the VMware HA system that can restart a guest system after failure.
3) Another technique to manage a virtual cluster is to use an independent cluster manager on both the host and guest systems. This will make infrastructure management more complex.
4) Technique to manage a virtual cluster is to use an integrated cluster on the guest and host systems.

Various cluster management schemes can be greatly enhanced when VM live migration is enabled with minimal overhead. VMs can be live-migrated from one physical machine to another; in case of failure, one VM can be replaced by another VM. Virtual clusters can be applied in computational grids, cloud platforms, and high-performance computing (HPC) systems. The major attraction of this scenario is that virtual clustering provides dynamic resources that can be quickly put together upon user demand or after a node failure.

- The motivation is to design a live VM migration scheme with negligible downtime, the lowest network bandwidth consumption possible, and a reasonable total migration time.
- It ensure that the migration will not disrupt other active services residing in the same host through resource contention (e.g., CPU, network bandwidth).

A VM can be in one of the following four states.

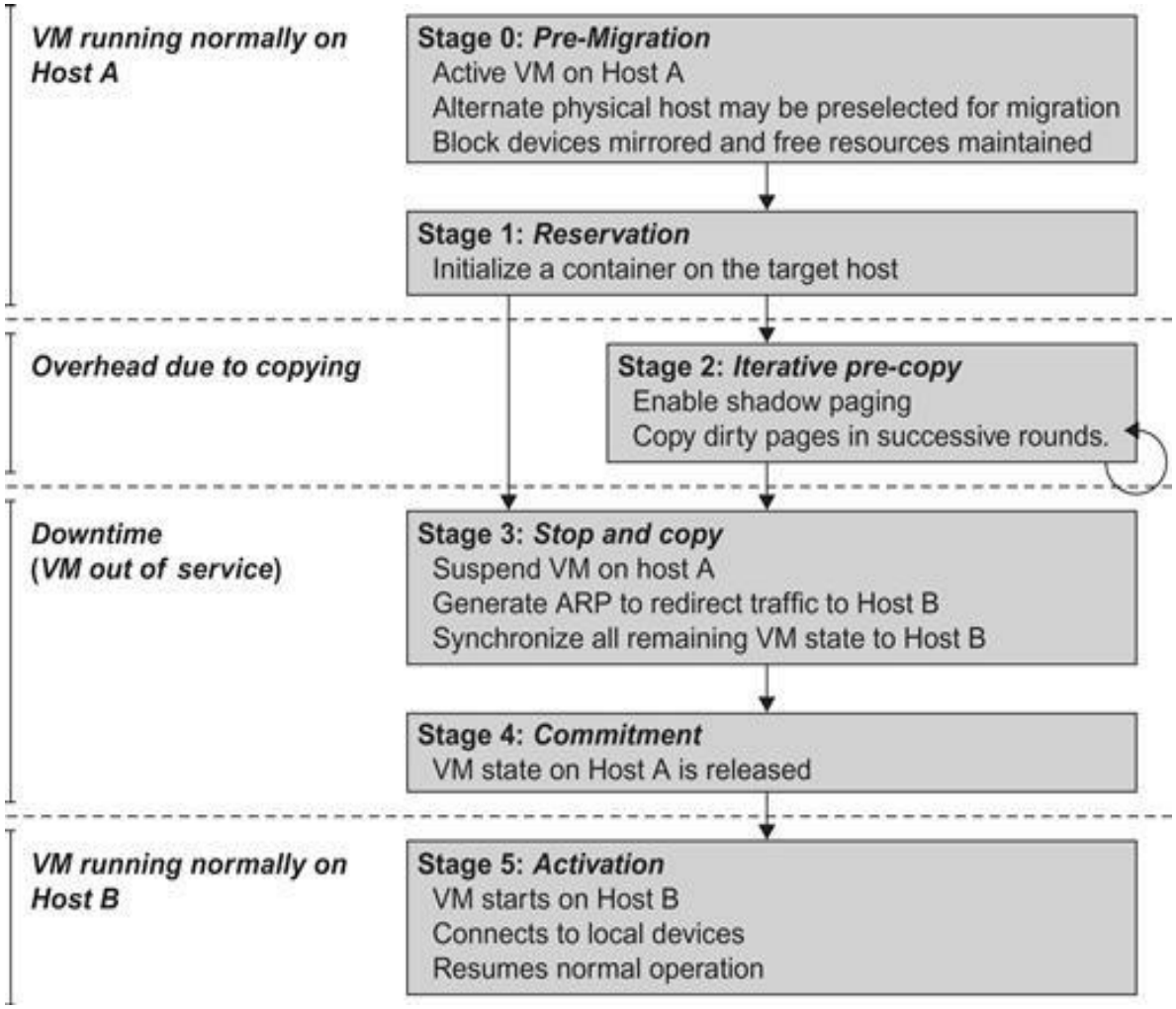
1. An inactive state is defined by the virtualization platform, under which the VM is not enabled.
2. An active state refers to a VM that has been instantiated at the virtualization platform to perform a real task.
3. A paused state corresponds to a VM that has been instantiated but disabled to process a task or paused in a waiting state.
4. A VM enters the suspended state if its machine file and virtual resources are stored back to the disk.

Live VM Migration

Steps and Performance Effects

In a cluster built with mixed nodes of host and guest systems, the normal method of operation is to run everything on the physical machine. When a VM fails, its role could be replaced by another VM on a different node, as long as they both run with the same guest OS. In other words, a physical node can fail over to a VM on another host. This is different from physical-to-physical failover in a traditional physical cluster. The potential drawback is that a VM must stop playing its role if its residing host node fails. However, this problem can be mitigated with VM life migration. Figure shows the process of life migration

of a VM from host A to host B. The migration copies the VM state file from the storage area to the host machine.



Live migration of VM consists of the following six steps:

Steps 0 and 1: Start migration. This step makes preparations for the migration, including determining the migrating VM and the destination host. Although users could manually make a VM migrate to an appointed host, in most circumstances, the migration is automatically started by strategies such as load balancing and server consolidation.

Steps 2: Transfer memory. Since the whole execution state of the VM is stored in memory, sending the VM's memory to the destination node ensures continuity of the service provided by the VM. All of the memory data is transferred in the first round, and then the migration controller recopies the memory data which is changed in the last round. These steps keep iterating until the dirty portion of the memory is small enough to handle the final copy. Although precopying memory is performed iteratively, the execution of programs is not obviously interrupted.

Step 3: Suspend the VM and copy the last portion of the data. The migrating VM's execution is suspended when the last round's memory data is transferred. Other nonmemory data such as CPU and network states should be sent as well. During this step, the VM is stopped and its applications will no longer run. This "service unavailable" time is called the "downtime" of migration, which should be as short as possible so that it can be negligible to users.

Steps 4 and 5: Commit and activate the new host. After all the needed data is copied, on the destination host, the VM reloads the states and recovers the execution of programs in it, and the service provided by this VM continues. Then the network connection is redirected to the new VM and the dependency to the source host is cleared. The whole migration process finishes by removing the original VM from the source host.

Migration of Memory, Files, and Network Resources

Memory Migration

- This is one of the most important aspects of VM migration. Moving the memory instance of a VM from one physical host to another can be approached in any number of ways. The techniques employed for this purpose depend upon the characteristics of application/workloads supported by the guest OS.
- Memory migration can be in a range of hundreds of megabytes to a few gigabytes in a typical system today, and it needs to be done in an efficient manner. The Internet SuspendResume (ISR) technique exploits temporal locality as memory states are likely to have considerable overlap in the suspended and the resumed instances of a VM.
- Temporal locality refers to the fact that the memory states differ only by the amount of work done since a VM was last suspended before being initiated for migration. To exploit temporal locality, each file in the file system is represented as a tree of small subfiles. A copy of this tree exists in both the suspended and resumed VM instances. The ISR technique deals with situations where the migration of live machines is not a necessity.

File System Migration

- To support VM migration, a system must provide each VM with a consistent, location independent view of the file system that is available on all hosts. A simple way to achieve this is to provide each VM with its own virtual disk which the file system is mapped to and transport the contents of this virtual disk along with the other states of the VM.
- Due to the current trend of high-capacity disks, migration of the contents of an entire disk over a network is not a viable solution. Another way is to have a global file system across all machines where a VM could be located. This way removes the need to copy files from one machine to another because all files are network-accessible.
- A distributed file system is used in ISR serving as a transport mechanism for propagating a suspended VM state. The actual file systems themselves are not mapped onto the distributed file system. Instead, the VMM only accesses its local file system. The relevant VM files are explicitly copied into the local file system for a resume operation and taken out of the local file system for a suspend operation.

Network Migration

- A migrating VM should maintain all open network connections without relying on forwarding mechanisms on the original host or on support from mobility or redirection mechanisms. To enable remote systems to locate and communicate with a VM, each VM must be assigned a virtual IP address known to other entities.
- This address can be distinct from the IP address of the host machine where the VM is currently located. Each VM can also have its own distinct virtual MAC address. The VMM maintains a mapping of the virtual IP and MAC addresses to their corresponding VMs. In general, a migrating VM includes all the protocol states and carries its IP address with it.
- If the source and destination machines of a VM migration are typically connected to a single switched LAN, an unsolicited ARP reply from the migrating host is provided advertising that the IP has moved to a new location. This solves the open network connection problem by reconfiguring

all the peers to send future packets to a new location. Although a few packets that have already been transmitted might be lost, there are no other problems with this mechanism. Alternatively, on a switched network, the migrating OS can keep its original Ethernet MAC address and rely on the network switch to detect its move to a new port.

- Only memory and CPU status needs to be transferred from the source node to the target node.
- Live migration techniques mainly use the precopy approach, which first transfers all memory pages, and then only copies modified pages during the last round iteratively. These issues with the precopy approach are caused by the large amount of transferred data during the whole migration process.
- Another strategy of postcopy is introduced for live migration of VMs. Here, all memory pages are transferred only once during the whole migration process and the baseline total migration time is reduced. But the downtime is much higher than that of precopy due to the latency of fetching pages from the source node before the VM can be resumed on the target.

Desktop Virtualization

VMware supports virtualization of operating system environments and single applications on enduser computers. It allows installing a different operating systems and applications in a completely isolated environment from the osting operating system.

Specific VMware software—VMware Workstation, for Windows operating systems, and VMware Fusion, for Mac OS X environments—is installed in the host operating system to create virtual machines and manage their execution. Besides the creation of an isolated computing environment, the two products allow a guest operating system to leverage the resources of the host machine (USB devices, folder sharing, and integration with the graphical user interface (GUI) of the host Operating system).

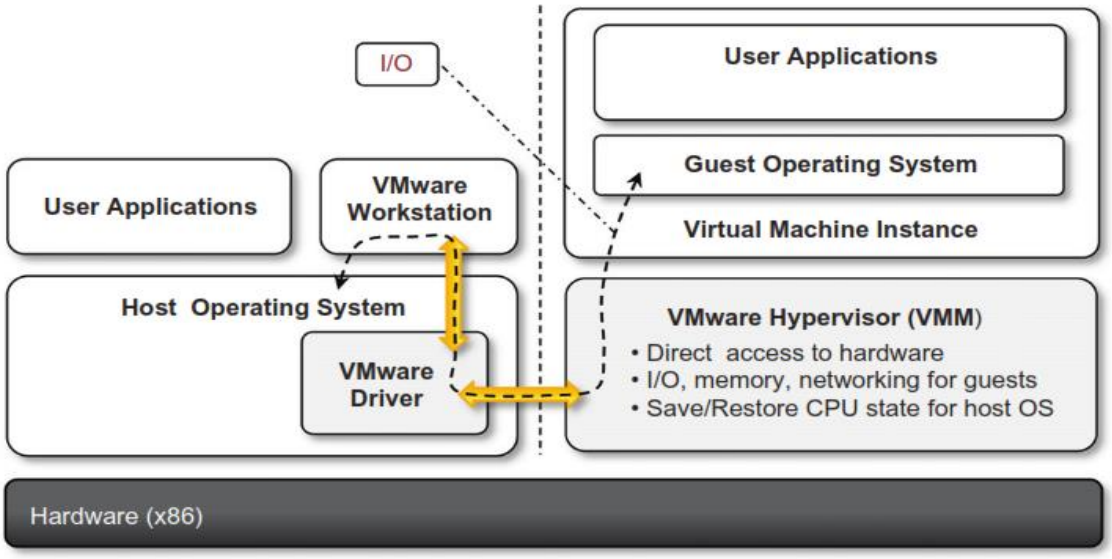


Figure provides an overview of the architecture of these systems.

- The virtualization environment is created by an application installed in guest operating systems, which provides those operating systems with full hardware virtualization of the underlying hardware. This is done by installing a specific driver in the host operating system that provides two main services:

1. It deploys a virtual machine manager that can run in privileged mode.
2. It provides hooks for the VMware application to process specific I/O requests eventually by relaying such requests to the host operating system via system calls.

- Using this architecture—also called Hosted Virtual Machine Architecture—it is possible to both isolate virtual machine instances within the memory space of a single application and provide reasonable performance, since the intervention of the VMware application is required only for instructions, such as device I/O, that require binary translation.

- Instructions that can be directly executed are managed by the virtual machine manager, which takes control of the CPU and the MMU and alternates its activity with the host OS. Virtual machine images are saved in a collection of files on the host file system, and both VMware Workstation and VMware Fusion allow creation of new images, pause their execution, create snapshots, and undo operations by rolling back to a previous state of the virtual machine.

Remote Desktop Virtualization

Remote desktop virtualization implementations operate as a client/server computing environment. Application execution takes place on a remote operating system with only display, keyboard, and mouse. Information communicated with the local client device, which may be a conventional PC/laptop, a thin client device, a tablet or even a smart phone. A common implementation of this approach is to host multiple desktop operating system instances on a server hardware platform running a hypervisor. This is generally referred to as "Virtual Desktop Infrastructure" or "VDI". Remote desktop virtualization is also used as a mean of resource sharing, to provide low cost desktop computing services in environments where providing every user with a dedicated desktop PC.

Remote desktop virtualization is frequently used in the following scenarios:

- In distributed environments with high availability requirements and where desk side technical support is not readily available, such as branch office and retail environments.
- In environments where high network latency degrades the performance of conventional client/server applications.
- In environments where remote access and data security requirements create conflicting requirements that can be addressed by retaining all (application) data within the data center with only display, keyboard and mouse information communicated with the remote client.

Desktop as a Service

Remote desktop virtualization can also be provided via a cloud computing similar to that provided using a software as a service model. This approach is usually referred to as Cloud Hosted virtual desktops. Cloud Hosted virtual desktops are divided into two technologies.

- Managed VDI, which is based on VDI technology provided as an out sourced managed service
- And desktop as a service which provides a higher level of automation and real multi-tenancy reducing the cost of the technology. The DaaS provider typically takes full responsibility for hosting and maintaining the compute, storage and access infrastructure, as well as applications and application software license needed to provide the desktop service in return for a fixed monthly fee. For example, VMware's Horizon DaaS, based on VMware's acquisition of Desktop, is a monthly fixed rate DaaS service.

Cloud hosted virtual desktops can be implemented using both VDI and remote desktop service based system and can be provided through public cloud, private cloud infrastructure, and hybrid cloud platforms. Private cloud implementations are commonly referred to as "managed VDI". Public cloud offering tend to be based on desktop as a service technology.

Network Virtualization

- Network virtualization is the process of combining hardware and software network resources and network functionality into a single software based administrative entity, a virtual network.
- Network virtualization is categorized as entire external virtualization, combining many networks or parts of networks into a virtual unit, or internal virtualization, providing network like functionality to software containers on a single network server.
- In software testing, software developers use network virtualization to test software under

development in a simulation of the network environment in which the software intended to operate.

- As a component of application performance engineering, network virtualization enables developers to emulates connection between applications, services, dependencies and end users in a test environment without having to physically test the software on all possible hardware or system software.

Components in Network Virtualization

Various equipment and software vendors offer network virtualization by combining any of the following:

- Network hardware, such as switches and network adapters, also known as Network Interface Cards (NICs).
- Network elements, such as firewalls and load balancers.
- Networks, such as Virtual LAN (VLANs) and containers such as virtual machines ' (VMs).
- Network storage devices.
- Network machine to machines elements, such as telecommunication devices.
- Network mobile elements, such as laptop computers, tablet computers, and smart phones.
- Network media, such as Ethernet and fibre channel.

External & Internal Network Virtualization

- External virtualization combines or subdivides one or more local area networks (LANs) into virtual networks to improve a large network's or data center's efficiency. A virtual local area network (VLAN) and network switch comprise the key components. Using this technology, a system administrator can configure system physically attached to the same local network into separate virtual networks. Conversely, an administrator can combine systems on separate local area networks (LANs) into a single VLAN spanning segment of a large network.

- Internal virtualization configures a single system with software containers, such as Xen hypervisor control programs, or pseudo-interfaces, such as s VNIC, to emulate a physical network with software. This can improve a single system's efficiency by isolating applications to separate containers or pseudo-interfaces.

1. Physical Network

Physical components: Network adapters, switches, bridges, repeaters, routers and hubs.
Grants connectivity among physical servers running a hypervisor, between physical servers and storage systems and between physical servers and clients.

2. VM Network

Consists of virtual switches.
Provides connectivity to hypervisor kernel.
Connects to the physical network.
Resides inside the physical server.

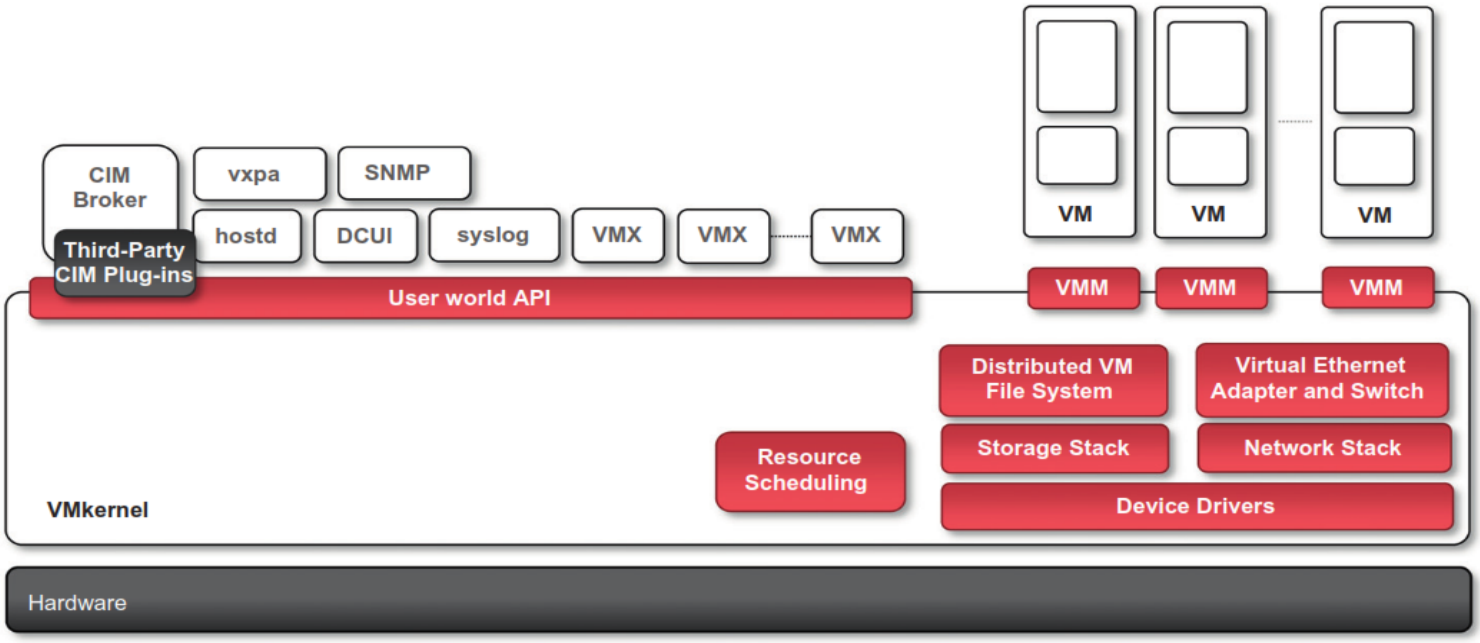
Server Virtualization

VMware provided solutions for server virtualization with different approaches over time. VMware ESX Server and its enhanced version, VMWare ESXi Server, are examples of the hypervisor-based approach. Both can be installed on bare metal servers and provide services for virtual machine management. The two solutions provide the same services but differ in the internal architecture, more specifically in the organization of the hypervisor kernel. VMware ESX embeds a modified version of a Linux operating

system, which provides access through a service console to hypervisor. VMware ESXi implements a very thin OS layer and replaces the service console with interfaces and services for remote management, thus considerably reducing the hypervisor code size and memory footprint.

The architecture of VMware ESXi is displayed in Figure.

- The base of the infrastructure is the VMkernel, which is a thin Portable Operating System Interface (POSIX) compliant operating system that provides the minimal functionality for processes and thread management, file system, I/O stacks, and resource scheduling.
- The kernel is accessible through specific APIs called User world API. These APIs are utilized by all the agents that provide supporting activities for the management of virtual machines. Remote management of an ESXi server is provided by the CIM Broker, a system agent that acts as a gateway to the VMkernel for clients by using the Common Information Model (CIM) protocol.



- The ESXi installation can also be managed locally by a Direct Client User Interface (DCUI), which provides a BIOS-like interface for the management of local users.

There are three popular approaches to server virtualization

1. The virtual machine model
2. The paravirtual machines model
3. The OS level model

The virtual Machine model (Full virtualization)

Virtual machines are based on the host/guest paradigm each guest runs on a virtual imitation of the hardware layer. This approach allows the guest operating system to run without modifications. It also allows the administrator to create guests that use different operating systems. The guest has no knowledge of the host's operating system because it is not aware that it is not running on real hardware. It does, however, require real computing resources from the host, so it uses a hypervisor to coordinate instructions to the CPU. The hypervisor is called a Virtual Machine Monitor (VMM). It validates all the guest issued CPU instructions and manages any executed code that requires additional privileges. VMware and Microsoft virtual server both use the virtual machine model.

The Paravirtual machine model

The Paravirtual Machine (PVM) model is also based on the host/guest paradigm, and it use a Virtual Machine Monitor too. In the paravirtual machine model, however, the VM actually modifies the guest

operating system's code. This modification is called porting. Porting supports the VMM so it can utilize privileged systems calls sparingly. Like virtual machines, paravirtual machines are capable of running multiple operating systems. Xen uses the paravirtual machine model.

The OS level Model

In the OS level model, the host runs a single OS kernel as its core and exports operating system functionality to each of the guests. Guests must use the same operating system as the host, although different distribution of the same system are allowed. This distributed architecture eliminates system calls between layers, which reduces CPU usage overhead. It also requires that each partition remains strictly isolated from its neighbours so that a failure or security breach in one partition isn't able to affect any of the other partitions. In this model, common binaries and libraries on the same physical machine can be shared, allowing an OS level virtual server to host thousands of guests at the same time. Solaris zones uses OS-level virtualization.

Advantages of Server Virtualization

There are the following advantages of Server Virtualization -

1. Independent Restart

In Server Virtualization, each server can be restart independently and does not affect the working of other virtual servers.

2. Low Cost

Server Virtualization can divide a single server into multiple virtual private servers, so it reduces the cost of hardware components.

3. Disaster Recovery

Disaster Recovery is one of the best advantages of Server Virtualization. In Server Virtualization, data can easily and quickly move from one server to another and these data can be stored and retrieved from anywhere.

4. Faster deployment of resources

Server virtualization allows us to deploy our resources in a simpler and faster way.

5. Security

It allows uses to store their sensitive data inside the data centers.

Disadvantages of Server Virtualization

There are the following disadvantages of Server Virtualization –

1. The biggest disadvantage of server virtualization is that when the server goes offline, all the websites that are hosted by the server will also go down.
2. 2. There is no way to measure the performance of virtualized environments.
3. 3. It requires a huge amount of RAM consumption.
4. 4. It is difficult to set up and maintain.
5. 5. Some core applications and databases are not supported virtualization.
6. 6. It requires extra hardware resources.

Uses of Server Virtualization

A list of uses of server virtualization is given below –

- Server Virtualization is used in the testing and development environment.
- It improves the availability of servers.
- It allows organizations to make efficient use of resources.
- It reduces redundancy without purchasing additional hardware components.

Virtualization for Data-Center Automation

Data centers have grown rapidly in recent years, and all major IT companies are pouring their resources into building new data centers. In addition, Google, Yahoo!, Amazon, Microsoft, HP, Apple, and I BM are all in the game. All these companies have invested billions of dollars in data-center construction and automation. Data-center automation means that huge volumes of hardware, software, and database resources in these data centers can be allocated dynamically to millions of Internet users simultaneously, with guaranteed QoS and cost-effectiveness.
This automation process is triggered by the growth of virtualization products and cloud computing services. Virtualization is moving towards enhancing mobility, reducing planned downtime (for maintenance), and increasing the number of virtual clients. The latest virtualization development highlights high availability (HA), backup services, workload balancing, and further increases in client bases.

Server Consolidation in Data Centers

In data centers, a large number of heterogeneous workloads can run on servers at various times. These heterogeneous workloads can be roughly divided into two categories: chatty workloads and noninteractive workloads. Chatty workloads may burst at some point and return to a silent state at some other point. A web video service is an example of this, whereby a lot of people use it at night and few people use it during the day. Non-interactive workloads do not require people's efforts to make progress after they are submitted. High-performance computing is a typical example of this.
At various stages, the requirements for resources of these workloads are dramatically different. However, to guarantee that a workload will always be able to cope with all demand levels, the workload is statically allocated enough resources so that peak demand is satisfied.
Server consolidation is an approach to improve the low utility ratio of hardware resources by reducing the number of physical servers. Among several server consolidation techniques such as centralized and physical consolidation, virtualization-based server consolidation is the most powerful. Data centers need to optimize their resource management.

Virtual Storage Management

The term "storage virtualization" was widely used before the renaissance of system virtualization. Yet the term has a different meaning in a system virtualization environment. Previously, storage virtualization was largely used to describe the aggregation and repartitioning of disks at very coarse time scales for use by physical machines. In system virtualization, virtual storage includes the storage managed by VMMs and guest OSes. Generally, the data stored in this environment can be classified into two categories: VM images and application data. The VM images are special to the virtual environment, while application data includes all other data which is the same as the data in traditional OS environments.
The most important aspects of system virtualization are encapsulation and isolation. Traditional operating systems and applications running on them can be encapsulated in VMs. Only one operating system runs in a virtualization while many applications run in the operating system.

Cloud OS for Virtualized Data Centers

Data centers must be virtualized to serve as cloud providers. Table summarizes four virtual infrastructure (VI) managers and OSes. These VI managers and OSes are specially tailored for virtualizing data centers which often own a large number of servers in clusters. Nimbus, Eucalyptus, and OpenNebula are all open source software available to the general public. Only vSphere 4 is a proprietary OS for cloud resource virtualization and management over data centers.

Manager/ OS, Platforms, License	Resources Being Virtualized, Web Link	Client API, Language	Hypervisors Used	Public Cloud Interface	Special Features
Nimbus Linux, Apache v2	VM creation, virtual cluster, www .nimbusproject.org/	EC2 WS, WSRF, CLI	Xen, KVM	EC2	Virtual networks
Eucalyptus Linux, BSD	Virtual networking (Example 3.12 and [41]), www .eucalyptus.com/	EC2 WS, CLI	Xen, KVM	EC2	Virtual networks
OpenNebula Linux, Apache v2	Management of VM, host, virtual network, and scheduling tools, www.opennebula.org/	XML-RPC, CLI, Java	Xen, KVM	EC2, Elastic Host	Virtual networks, dynamic provisioning
vSphere 4 Linux, Windows, proprietary	Virtualizing OS for data centers (Example 3.13), www .vmware.com/ products/vsphere/ [66]	CLI, GUI, Portal, WS	VMware ESX, ESXi	VMware vCloud partners	Data protection, vStorage, VMFS, DRM, HA

Trust Management in Virtualized Data Centers

A VMM changes the computer architecture. It provides a layer of software between the operating systems and system hardware to create one or more VMs on a single physical platform. A VM entirely encapsulates the state of the guest operating system running inside it. Encapsulated machine state can be copied and shared over the network and removed like a normal file, which proposes a challenge to VM security. In general, a VMM can provide secure isolation and a VM accesses hardware resources through the control of the VMM, so the VMM is the base of the security of a virtual system. Normally, one VM is taken as a management VM to have some privileges such as creating, suspending, resuming, or deleting a VM.

Once a hacker successfully enters the VMM or management VM, the whole system is in danger. A subtler problem arises in protocols that rely on the "freshness" of their random number source for generating session keys. Considering a VM, rolling back to a point after a random number has been chosen, but before it has been used, resumes execution; the random number, which must be "fresh" for security purposes, is reused.

With a stream cipher, two different plaintexts could be encrypted under the same key stream, which could, in turn, expose both plaintexts if the plaintexts have sufficient redundancy. Non cryptographic protocols that rely on freshness are also at risk. For example, the reuse of TCP initial sequence numbers can raise TCP hijacking attacks.