2.3 Technology Example: Vmware, Microsoft Hyper-V, KVM , Xen:

Write short note : **(i) Vmware (ii) Microsoft Hyper-V (iii) KVM (iv) Xen**

(i) Vmware:

- 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.
- Besides these two core solutions, VMware provides additional tools and software that simplify the use of virtualization technology either in a desktop environment, with tools enhancing the integration of virtual guests with the host, or in a server environment, with solutions for building and managing virtual computing infrastructures.

a) Full virtualization and binary translation:

- VMware is well known for the capability to virtualize x86 architectures, which runs unmodified on top of their hypervisors.
- With the new generation of hardware architectures and the introduction of hardware-assisted virtualization (Intel VT-x and AMD V) in 2006, full virtualization is made possible with hardware support, but before that date, the use of dynamic **binary translation** was the only solution that allowed running x86 guest operating systems unmodified in a virtualized environment.

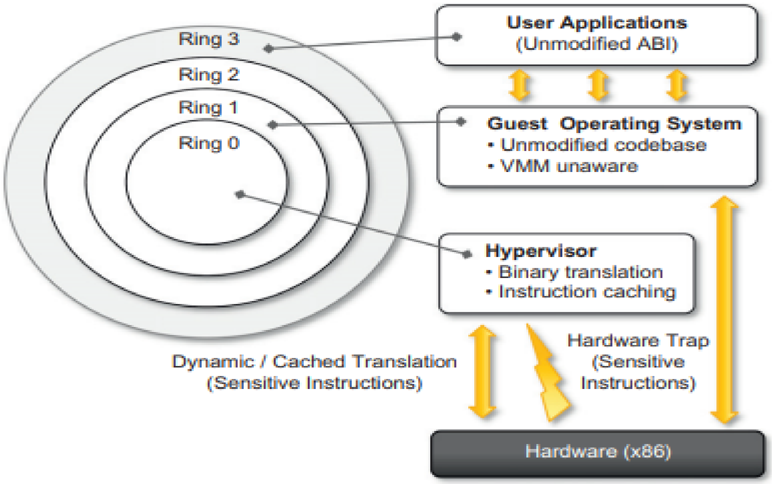

Figure: A full virtualization reference model

- The major advantage is that guests can run unmodified in a virtualized environment, which is a crucial feature for operating systems for which source code is not available.
- Binary translation is a more portable solution for full virtualization. On the other hand, translating instructions at runtime introduces an additional overhead that is not present in other approaches (paravirtualization or hardware-assisted virtualization).
- Finally, VMware also provides full virtualization of I/O devices such as network controllers and other peripherals such as keyboard, mouse, disks, and universal serial bus (USB) controllers.

b) Virtualization solutions

VMware is a pioneer in virtualization technology and offers a collection of virtualization solutions covering the entire range of the market, from desktop computing to enterprise computing and infrastructure virtualization.

(i)End-user (desktop) virtualization

- Figure provides an overview of the architecture of these systems.

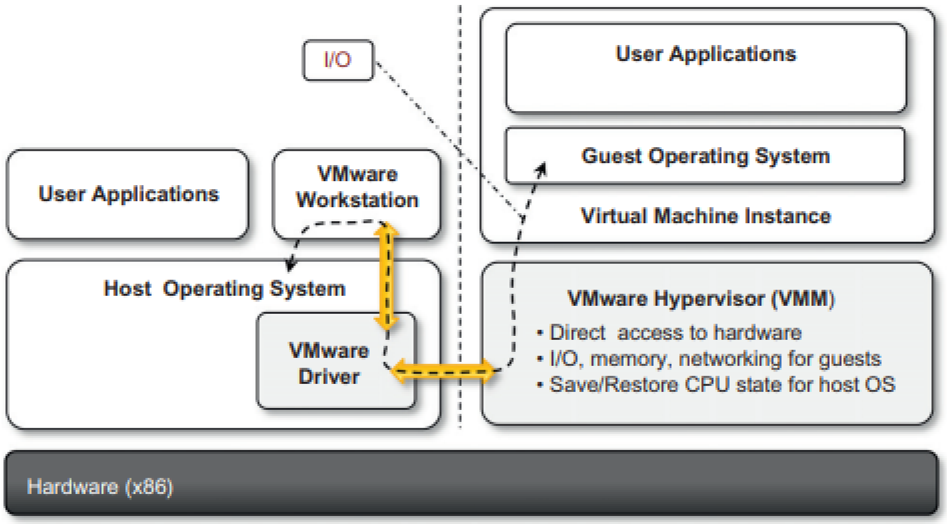

Figure: VMWare Workstation Architecture

- VMware supports virtualization of operating system environments and single applications on end user computers. The first option is the most popular and allows installing a different operating systems and applications in a completely isolated environment from the hosting 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.
- 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: • It deploys a virtual machine manager that can run in privileged mode.

  • 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.

(ii)Server virtualization:

- VMware provided solutions for server virtualization with different approaches over time. Initial support for server virtualization was provided by VMware GSX server, which replicates the approach used for end-user computers and introduces remote management and scripting capabilities.
- The architecture of VMware GSX Server is depicted in Figure.

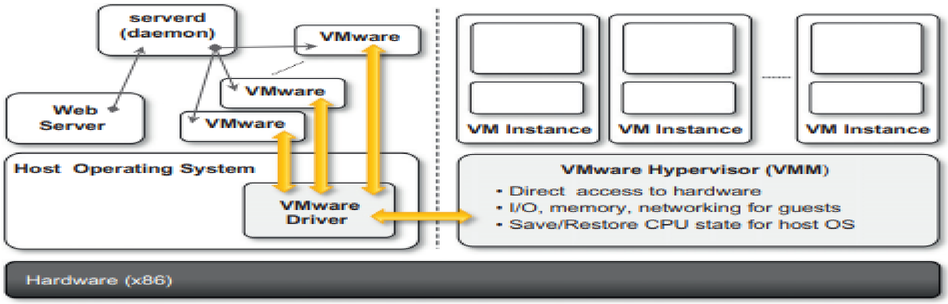

Figure: VMware GSX server architecture.

- The architecture is mostly designed to serve the virtualization of Web servers. A daemon process, called serverd, controls and manages VMware application processes. These applications are then connected to the virtual machine instances by means of the VMware driver installed on the host operating system. Virtual machine instances are managed by the VMM as described previously. User requests for virtual machine management and provisioning are routed from the Web server through the VMM by means of serverd.

- The architecture of VMware ESXi is displayed below **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.

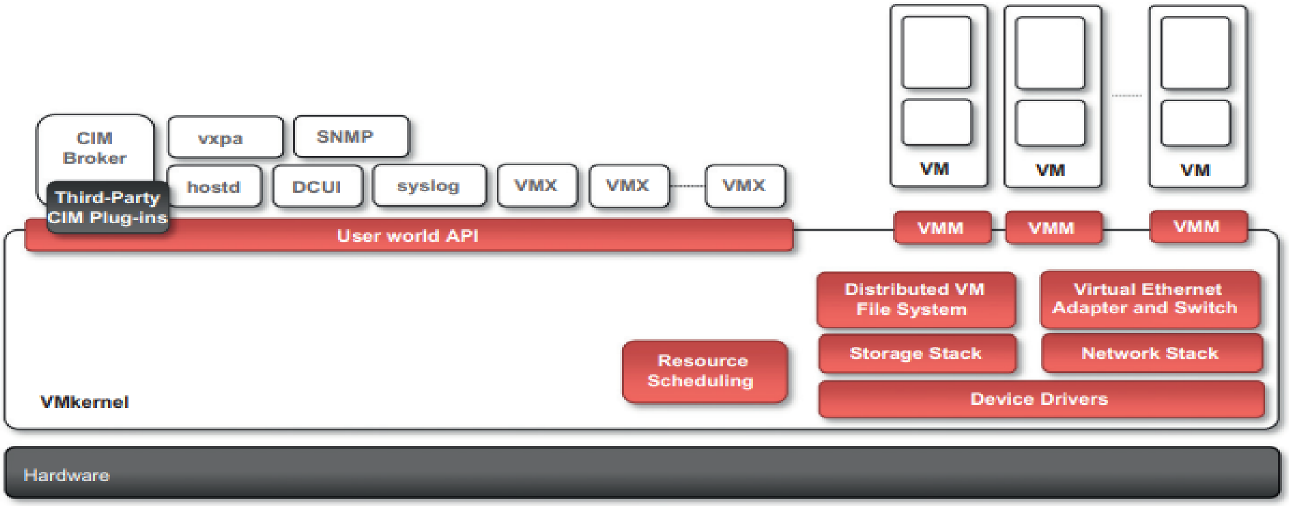

Figure: VMware ESXi server architecture.

- 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)8 protocol.

Infrastructure virtualization and cloud computing solutions:

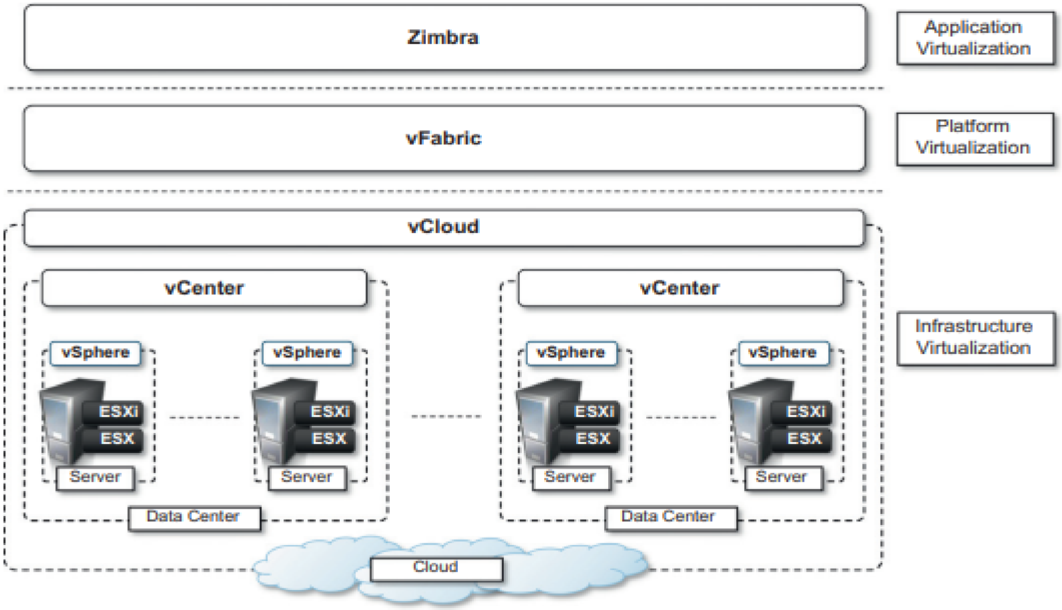

Figure: VMware Cloud Solution stack

- VMware provides a set of products covering the entire stack of cloud computing, from infrastructure management to Software-as-a-Service solutions hosted in the cloud. Figure 3.16 gives an overview of the different solutions offered and how they relate to each other.
- ESX and ESXi constitute the building blocks of the solution for virtual infrastructure management: A pool of virtualized servers is tied together and remotely managed as a whole by VMware vSphere.
- As a virtualization platform it provides a set of basic services besides virtual compute services: Virtual file system, virtual storage, and virtual network constitute the core of the infrastructure; application services, such as virtual machine migration, storage migration, data recovery and etc.

(ii) Microsoft Hyper-V

Hyper-V is an infrastructure virtualization solution developed by Microsoft for server virtualization. As the name recalls, it uses a hypervisor-based approach to hardware virtualization, which leverages several techniques to support a variety of guest operating systems. Hyper-V is currently shipped as a component of Windows Server 2008 R2 that installs the hypervisor as a role within the server.

Architecture

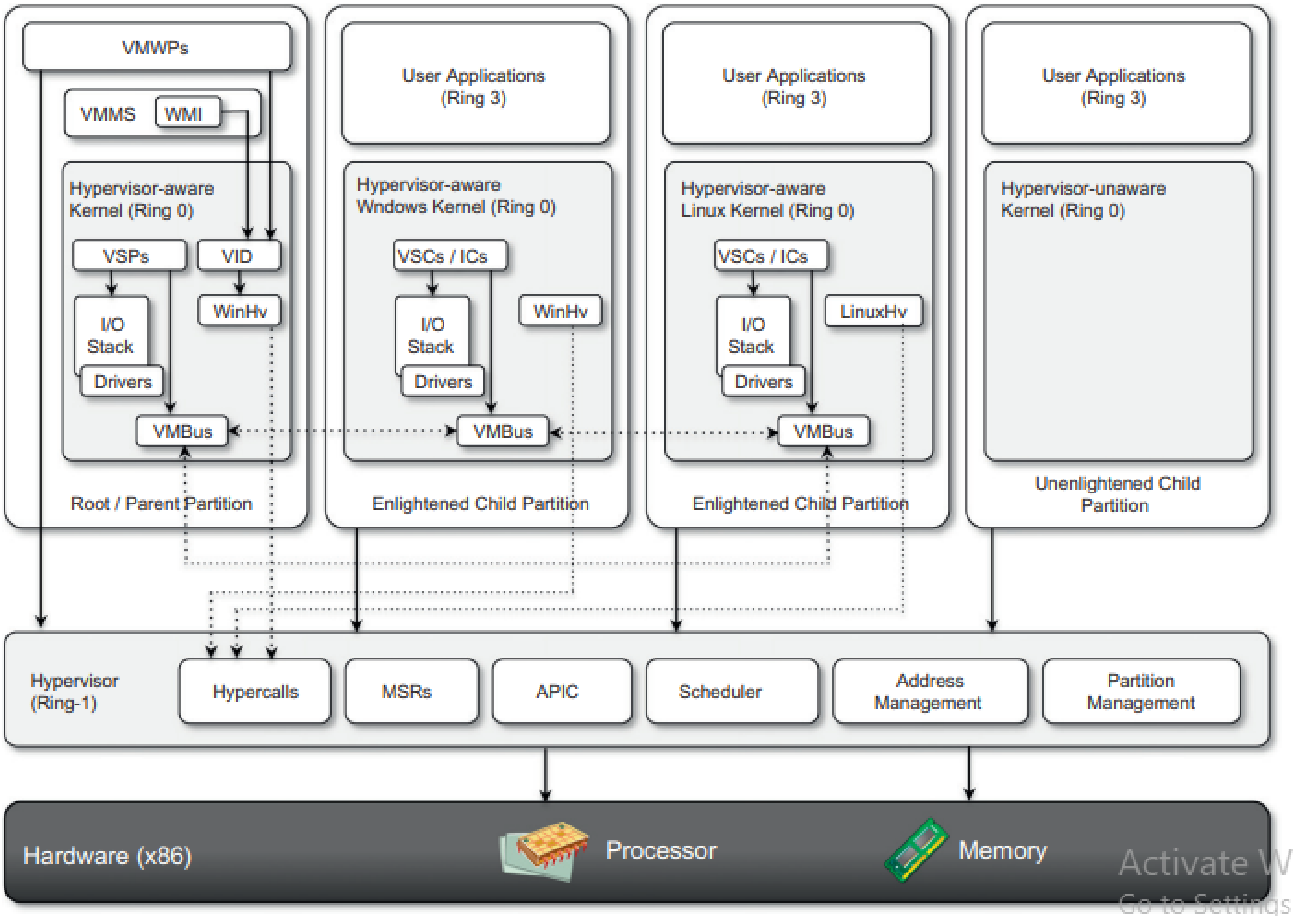
VMWPs
VMMS
WMI
Hypervisor-aware Kernel (Ring 0)
VSPs
VID
I/O Stack
Drivers
WinHv
VMBus
Root / Parent Partition
User Applications (Ring 3)
Hypervisor-aware Wndows Kernel (Ring 0)
VSCs / ICs
I/O Stack
Drivers
WinHv
VMBus
Enlightened Child Partition
User Applications (Ring 3)
Hypervisor-aware Linux Kernel (Ring 0)
VSCs / ICs
I/O Stack
Drivers
LinuxHv
VMBus
Enlightened Child Partition
User Applications (Ring 3)
Hypervisor-unaware Kernel (Ring 0)
Unenlightened Child Partition
Hypervisor (Ring-1)
Hypercalls
MSRs
APIC
Scheduler
Address Management
Partition Management
Hardware (x86)
Processor
Memory

Figure: Microsoft Hyper-V Architecture

- Hyper-V supports multiple and concurrent execution of guest operating systems by means of partitions. A partition is a completely isolated environment in which an operating system is installed and run. Figure provides an overview of the architecture of Hyper-V. Despite its straightforward installation as a component of the host operating system, Hyper-V takes control of the hardware, and the host operating system becomes a virtual machine instance with special privileges, called the parent partition.
- The parent partition (also called the root partition) is the only one that has direct access to the hardware. It runs the virtualization stack, hosts all the drivers required to configure guest operating systems, and creates child partitions through the hypervisor. Child partitions are used to host guest operating systems and do not have access to the underlying hardware, but their interaction with it is controlled by either the parent partition or the hypervisor itself.
- **Hypervisor :**The hypervisor is the component that directly manages the underlying hardware (processors and memory). It is logically defined by the following components:
  • **Hypercalls interface**: This is the entry point for all the partitions for the execution of sensitive instructions. This is an implementation of the paravirtualization approach already discussed with Xen. This interface is used by drivers in the partitioned operating system to contact the hypervisor using the standard Windows calling convention. The parent partition also uses this interface to create child partitions.
  • **Memory service routines (MSRs):** These are the set of functionalities that control the memory and its access from partitions. By leveraging hardware-assisted virtualization, the hypervisor uses the Input/Output Memory Management Unit (I/O MMU or IOMMU) to fast-track access to devices from partitions by translating virtual memory addresses.

- **Advanced programmable interrupt controller (APIC):** This component represents the interrupt controller, which manages the signals coming from the underlying hardware when some event occurs (timer expired, I/O ready, exceptions and traps). Each virtual processor is equipped with a synthetic interrupt controller (SynIC), which constitutes an extension of the local APIC. The hypervisor is responsible of dispatching, when appropriate, the physical interrupts to the synthetic interrupt controllers.
- **Scheduler:** This component schedules the virtual processors to run on available physical processors. The scheduling is controlled by policies that are set by the parent partition.
- **Address manager:** This component is used to manage the virtual network addresses that are allocated to each guest operating system.
- **Partition manager:** This component is in charge of performing partition creation, finalization, destruction, enumeration, and configurations. Its services are available through the hypercalls interface API previously discussed

- Enlightened I/O and synthetic devices Enlightened I/O provides an optimized way to perform I/O operations, allowing guest operating systems to leverage an interpartition communication channel rather than traversing the hardware emulation stack provided by the hypervisor. This option is only available to guest operating systems that are hypervisor aware.
- **Parent partition**

  The parent partition executes the host operating system and implements the virtualization stack that complements the activity of the hypervisor in running guest operating systems. This partition always hosts an instance of the Windows Server 2008 R2, which manages the virtualization stack made available to the child partitions. This partition is the only one that directly accesses device drivers and mediates the access to them by child partitions by hosting the VSPs.
- **Child partitions**

  Child partitions are used to execute guest operating systems. These are isolated environments that allow secure and controlled execution of guests. Two types of child partition exist, they differ on whether the guest operating system is supported by Hyper-V or not. These are called Enlightened and Unenlightened partitions, respectively. The first ones can benefit from Enlightened I/O; the other ones are executed by leveraging hardware emulation from the hypervisor.

Cloud computing and infrastructure management :

- Hyper-V constitutes the basic building block of Microsoft virtualization infrastructure. Other components contribute to creating a fully featured platform for server virtualization. To increase the performance of virtualized environments, a new version of Windows Server 2008, called Windows Server Core, has been released. This is a specific version of the operating system with a reduced set of features and a smaller footprint.
- Another component that provides advanced management of virtual machines is System Center Virtual Machine Manager (SCVMM) 2008. This is a component of the Microsoft System Center suite, which brings into the suite the virtual infrastructure management capabilities from an IT lifecycle point of view. Essentially, SCVMM complements the basic features offered by Hyper-V with management capabilities, including:

  - Management portal for the creation and management of virtual instances
  - Virtual to Virtual (V2V) and Physical to Virtual (P2V) conversions
  - Delegated administration
  - Library functionality and deep PowerShell integration
  - Intelligent placement of virtual machines in the managed environment
  - Host capacity management
- SCVMM has also been designed to work with other virtualization platforms such as VMware vSphere (ESX servers) but benefits most from the virtual infrastructure management implemented with Hyper-V.

Kernel Based Virtual Machine(KVM):

- KVM (for Kernel-based Virtual Machine) is a full virtualization solution for Linux on x86 hardware containing virtualization extensions (Intel VT or AMD-V).
- It consists of a loadable kernel module, kvm.ko, that provides the core virtualization infrastructure and a processor specific module, kvm-intel.ko or kvm-amd.ko.
- KVM also requires a modified QEMU although work is underway to get the required changes upstream.
- Using KVM, one can run multiple virtual machines running unmodified Linux or Windows images. Each virtual machine has private virtualized hardware: a network card, disk, graphics adapter, etc.
- The kernel component of KVM is included in mainline Linux, as of 2.6.20. KVM is open source software.

KVM Architecture:

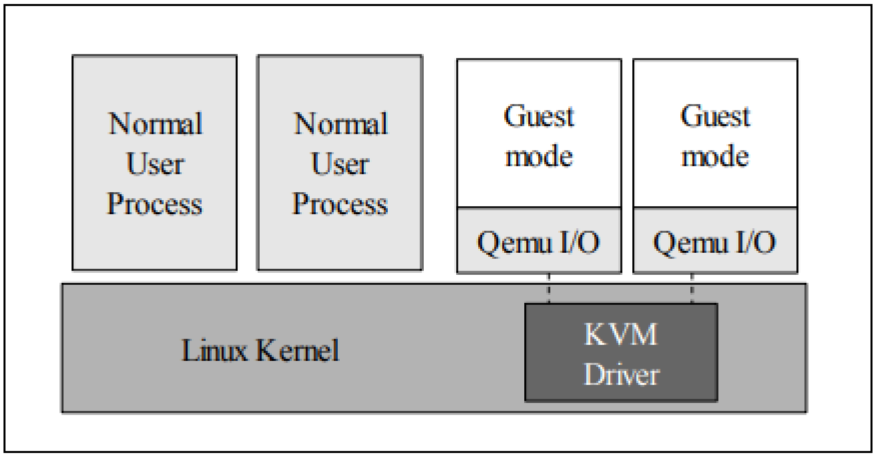

Figure: KVM Based Architecture

- A normal Linux process has two modes of execution: kernel and user. Kvm adds a third mode: guest mode (which has its own kernel and user modes, but these do not interest the hypervisor at all).
- The division of labor among the different modes is:
  - Guest mode: execute non-I/O guest code
  - Kernel mode: switch into guest mode, and handle any exits from guest mode due to I/O or special instructions.
  - User mode: perform I/O on behalf of the guest. By integrating into the kernel, the kvm 'hypervisor' automatically tracks the latest hardware and scalability features without additional effort.

kvm Components

- The simplicity of kvm is exemplified by its structure; there are two components:
  - A device driver for managing the virtualization hardware; this driver exposes its capabilities via a character device /dev/kvm
  - A user-space component for emulating PC hardware; this is a lightly modified qemu process
- The modified qemu process mmap()s the guest's physical memory and calls the kernel mode driver to execute in guest mode. The I/O model is directly derived from qemu's, with support for copy-on-write disk images and other qemu features.

How does KVM work?

Kernel-based Virtual Machine (KVM) requires a Linux kernel installation on a computer powered by a CPU that supports virtualization extensions. Specifically, KVM supports all x86 CPUs, a family of computer chips capable of processing the Intel x86 instruction language.

- Linux kernel

Linux kernel is the core of the open-source operating system. A kernel is a low-level program that interacts with computer hardware. It also ensures that software applications running on the operating system receive the required computing resources. Linux distributions, such as Red Hat Enterprise Linux, Fedora, and Ubuntu, pack the Linux kernel and additional programs into a user-friendly commercial operating system.

-How to enable KVM

(i) Once you have installed the Linux kernel, you need to install the following additional software components on the Linux machine:

- A host kernel module
- A processor-specific module
- An emulator
- A range of other Linux packages for expanding KVM's capabilities and performance

(ii) Once loaded, the server administrator creates a virtual machine via the command line tool or graphical user interface. KVM then launches the virtual machine as an individual Linux process. The hypervisor allocates every virtual machine with virtual memory, storage, network, CPU, and resources.

XEN:

- Xen is an open-source initiative implementing a virtualization platform based on paravirtualization. Initially developed by a group of researchers at the University of Cambridge in the United Kingdom, 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).
- At the basis of all these solutions is the Xen Hypervisor, which constitutes the core technology of Xen. Recently Xen has been advanced to support full virtualization using hardware-assisted virtualization.
- Xen is the most popular implementation of paravirtualization, which, in contrast with full virtualization, allows high-performance execution of guest operating systems.
- This is made possible by eliminating the performance loss while executing instructions that require special management. This is done by modifying portions of the guest operating systems run by Xen with reference to the execution of such instructions. Therefore it is not a transparent solution for implementing virtualization. This is particularly true for x86, which is the most popular architecture on commodity machines and servers.

Architecture:

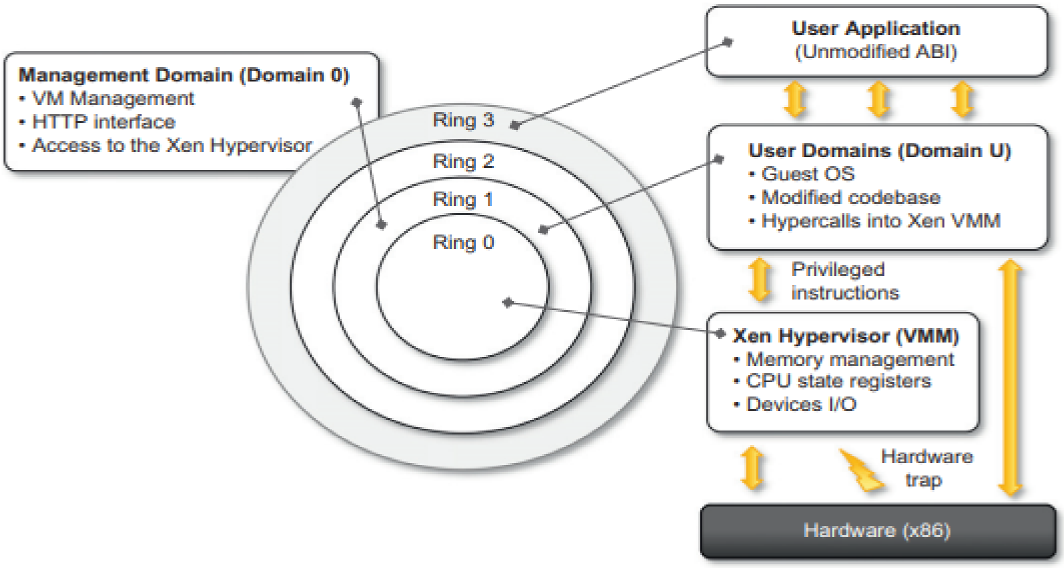

Figure: Xen architecture and guest OS management

Figure describes the architecture of Xen and its mapping onto a classic x86 privilege model.

(a)XEN Hypervisor(VMM):

- **The XEN Hypervisor is a free and open-source Type 1 VMM. It constructs and manages paravirtualized and hardware-assisted virtual machines, allowing for optimum resource utilization. It include:**
  1. **Memory Management:**
  2. **CPU state register:**
  3. **Devices I/O:**

(b)USER Domains(Domain U): User domains, also known as Domain U or DomU, are a key component of the XEN Hypervisor virtualization technology. They refer to isolated virtual machine instances that run on top of the XEN Hypervisor. It includes-

1. **Guest OS:**
2. **Modified Codebase:**
3. **Hypercalls into XEN VMM:**

(c)Management Doamin(Domain 0):

- **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. Moreover, 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. It includes:**
  1. **VM Management:** This is the first one that is loaded once the virtual machine manager has completely booted.
  2. **HTTP interface:** It hosts a HyperText Transfer Protocol (HTTP) server that serves requests for virtual machine creation, configuration, and termination.
  3. **Access to the XEN Hypervisor**:This component constitutes the embryonic version of a distributed virtual machine manager, which is an essential component of cloud computing systems providing Infrastructure-as-a-Service (IaaS) solutions.

(d)User Application: User applications typically run within the guest domains, which are also referred to as Domain U (DomU). These guest domains are isolated virtual environments created and managed by the XEN Hypervisor. It consist of -

1. **Unmodfied ABI: This allows Xen to maintain the ABI unchanged, thus allowing an easy switch to Xen-virtualized solutions from an application point of view.**