Cloud Computing

Virtualization

Implementation Levels of Virtualization

Virtualization Architectures

Virtualization

- Virtualization: It is a technology used to enhance the utilization of computing resources.
- A single hardware machine is multiplexed among multiple virtual machines (VMs).

Virtualization

- A software based virtual machine monitor/manager (VMM) or hypervisor is a program that manages the hardware resources for the VMs and also keeps each VM from disrupting other VMs.

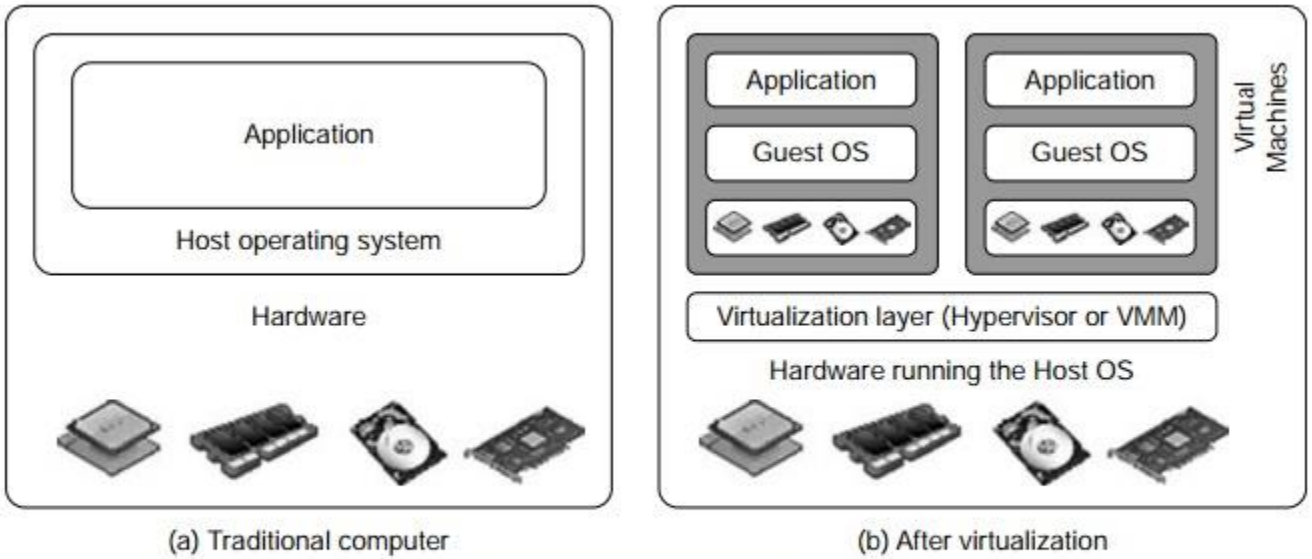


FIGURE 3.1

The architecture of a computer system before and after virtualization, where VMM stands for virtual machine monitor.

Virtualization Levels

- Virtualization implementation levels:
  1. Instruction Set Architecture (ISA) level
  2. Hardware Abstraction Layer (HAL) level
  3. Operating System level
  4. Library support (user-level API) level
  5. Application level

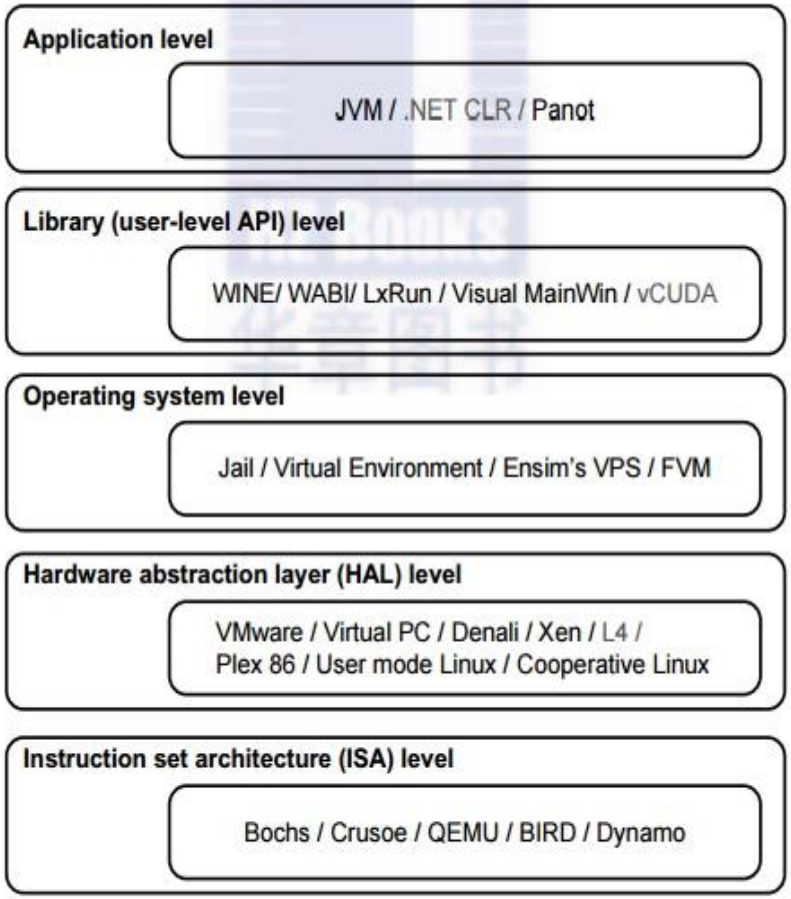


FIGURE 3.2

Virtualization ranging from hardware to applications in five abstraction levels.

Virtualization Levels

- Virtualization implementation levels:
  1. Instruction Set Architecture (ISA) level: Executing legacy code over new machines using ISA emulator tool such as an *interpreter* which translate one instruction of source code into corresponding instruction of the target machine.

  - Enables emulation of the instruction set of one processor on a different processor.
  - Allows the running or emulating of the instruction set architectures of different processors among each other.
  - 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. Obviously, this process is relatively slow.

Virtualization Levels

- Virtualization implementation levels:

  2. Hardware Abstraction level:

  The hardware components (CPU, RAM, Disk, NIC) of a physical system are virtualized and shared among virtual machines using Virtual Machine Monitor (VMM) tool or *hypervisor* which performs as abstraction layer.

  - 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.
  - 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.

Virtualization Levels

- Virtualization implementation levels:

  3. Operating System Level:

  The OS running over a server accommodates multiple *containers* or VMs. The host operating system acts as the abstraction layer between hardware and the containers.

  - 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 hard-ware and software in data centers.
  - The containers behave like real servers.

Virtualization Levels

- Virtualization implementation levels:

  4. Library support level:

  The API calls for hardware acceleration such as vCUDA stubs for graphic processing units (GPUs) are available at VM level.

  - Most applications use APIs exported by user-level libraries rather than using lengthy system calls by the OS. Since most systems provide well-documented APIs, such an interface becomes another candidate for virtualization.
  - 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.

Virtualization Levels

- Virtualization implementation levels:

  5. Application level:

  An application acts as a VM through wrapping of application in an abstraction layer which isolates it from OS and other applications. Another type is using virtualization layer as programming environment e.g; Java Virtual Machine (JVM).

  - 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.
  - 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.

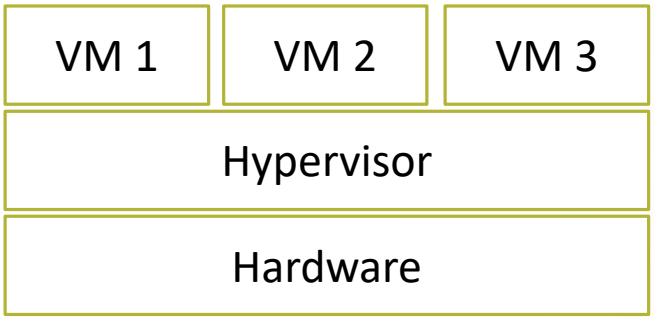
VM Architectures

- We know that the *virtualization layer* transforms the physical hardware into virtual hardware. There are three classes of VM architectures.
- Hypervisor transforms the physical hardware into virtual hardware.

VM Architectures

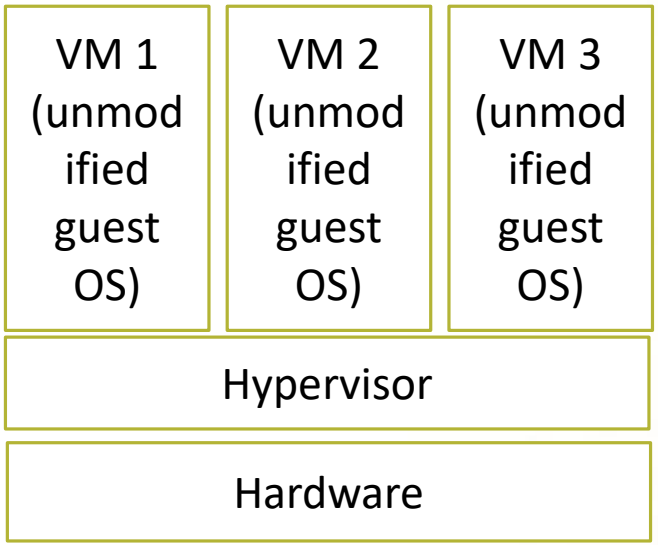
1. Hypervisor Architecture
2. Full-virtualization Architecture
3. Para-virtualization Architecture

VM Architectures



1. Hypervisor Architecture
   - It is the hardware level virtualization. Also called the bare-metal virtualization
   - The hypervisor sits between the hardware and the VMs and manages the VMs.
   - Example: Xen, VMware

VM Architectures



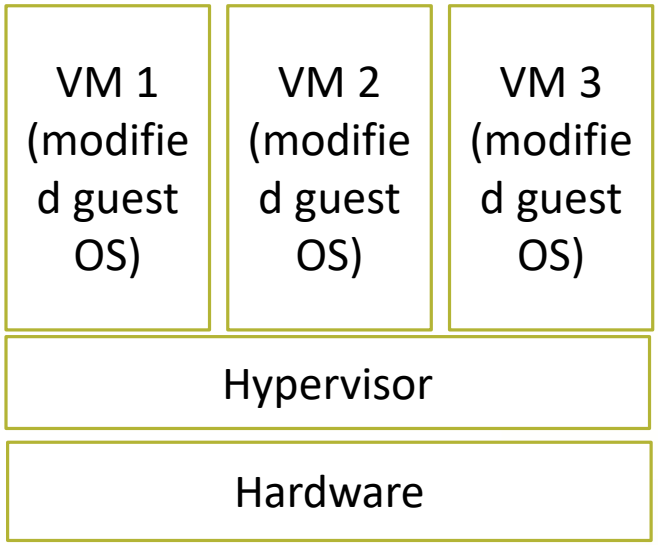
2. Full Virtualization: The guest operating system (OS) or the VM's OS does not know that it is installed on a VM.

The Virtualization layer manages the hardware acceleration. For example VMware

The virtualization layer can be installed on hardware or on host's OS.

Some of the instructions of a gust VM are directly run on hardware to enhance the performance.

VM Architectures



3. Para-virtualization Architecture: The guest OS is modified to comply with virtualization layer. All calls for hardware acceleration are handled by virtualization layer.

For example: KVM

end