Implementation Levels of Virtualization

Y. V. Lokeswari AP/ CSE
SSN College of Engineering

Reference: Distributed and Cloud Computing
K. Hwang, G. Fox and J. Dongarra

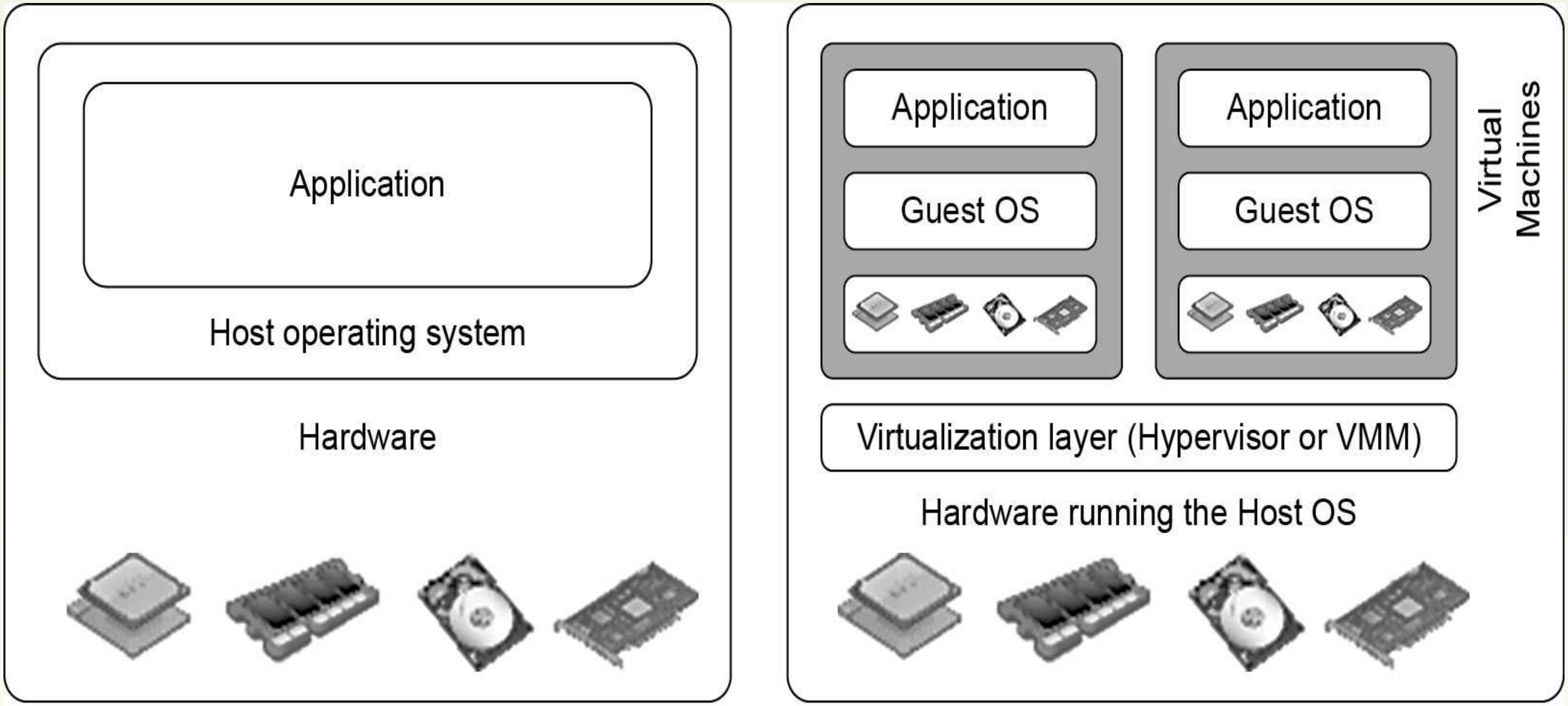
Overview

- Implementation Levels of Virtualization
  - ISA Level
  - Hardware Level
  - OS- Level
  - Library Level
  - Application Level
- VMM Design Requirements

Virtualization and VM

- Virtualization is a technology by which **multiple Virtual Machines** are multiplexed in the same hardware machine each VM running their own OS.
- With Virtualization, any computer platform can be installed in another host computer, even if they use processor with different ISA and distinct OS.
- Purpose of VM is to enhance the following
  - Resource Sharing by many users
  - Improved Computer Performance (Resource Utilization and application Flexibility)

Difference between Traditional Computer and Virtual machines



(a) Traditional computer

(b) After virtualization

(Courtesy of VMWare, 2008)

Virtual Machine, Guest Operating System, and VMM (Virtual Machine Monitor)

Virtual Machine

A representation of a real machine using software that provides an environment to host guest OS

Guest Operating System

An operating system running in a virtual machine.

The Virtualization layer is the middleware between the underlying hardware and virtual machines represented in the system, also known as ***virtual machine monitor*** **(VMM) or** ***hypervisor*.**

Virtualization Ranging from Hardware to Applications in Five Abstraction Levels

Application level

JVM / .NET CLR / Panot

Library (user-level API) level

WINE/ WABI/ LxRun / Visual MainWin / vCUDA

Operating system level

Jail / Virtual Environment / Ensim's VPS / FVM

Hardware abstraction layer (HAL) level

VMware / Virtual PC / Denali / Xen / L4 /
Plex 86 / User mode Linux / Cooperative Linux

Instruction set architecture (ISA) level

Bochs / Crusoe / QEMU / BIRD / Dynamo

Virtualization at ISA (Instruction Set Architecture) level

- Emulating a given ISA by the ISA of the host machine.
- e.g, MIPS binary code can run on an x-86-based host machine with the help of ISA emulation.
  - **Dynamic Binary Translation** : Translates basic blocks of **source instruction** to **target instruction**.
  - Typical systems: Bochs, Crusoe, Quemu, BIRD, Dynamo
- **Advantage**:
  - It can run a large amount of legacy binary codes written for **various processors** on any given **new hardware** host machines
  - Best application flexibility
- **Shortcoming & limitation**:
  - One source instruction may require tens or hundreds of native target instructions to perform its function, which is relatively slow.
  - V-ISA requires adding a processor-specific software translation layer in the complier.

Virtualization at Hardware Abstraction level

- Virtualization is performed right on top of the hardware.
- It generates virtual hardware environments for VMs (CPU, Memory, I/O devices)
- Manages the underlying hardware through virtualization.
- Typical systems: VMware, Virtual PC, Denali, Xen
- **Xen** hypervisor is applied to virtualize x-86 based machines to run Linux or other guest OS.
- Advantage:
  - Has higher performance and good application isolation
- Shortcoming & limitation:
  - Very expensive to implement (complexity)

Virtualization at Operating System (OS) level

- It is an abstraction layer between traditional OS and user applications.
- This virtualization creates isolated containers or Virtual Execution Environment or Virtual Private System (VPS) on a single physical server and the OS-instance to utilize the hardware and software in datacenters.
- Typical systems: Jail / Virtual Environment / Ensim's VPS / FVM

Virtualization at OS Level

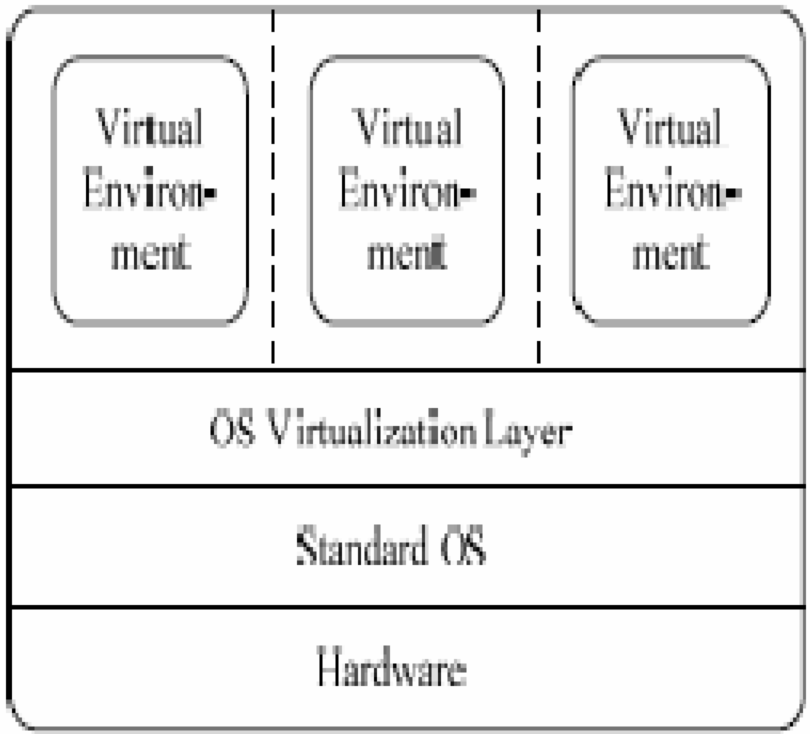
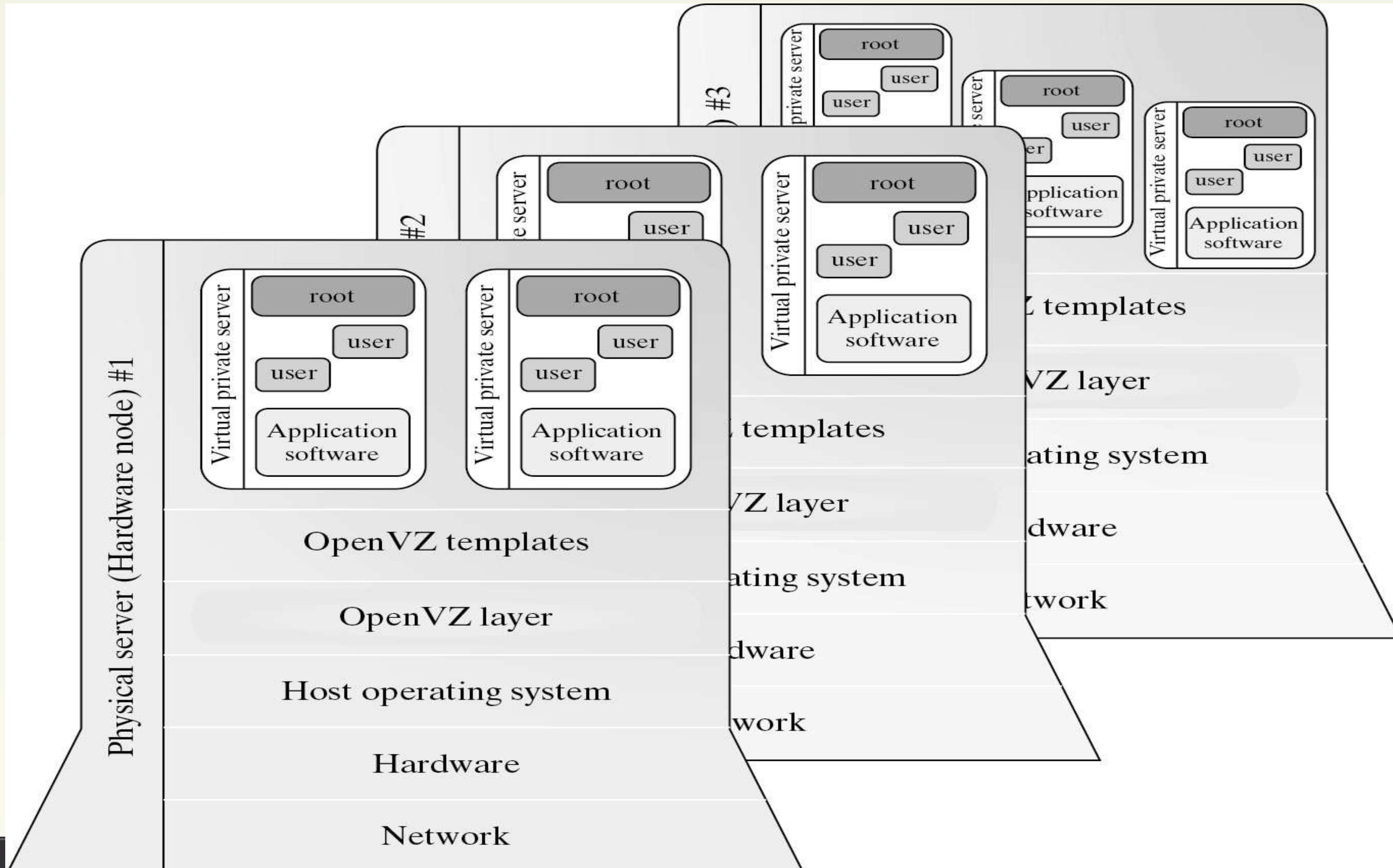


Figure 6.3 The virtualization layer is inserted inside an OS to partition the hardware resources for multiple VMs to run their applications in virtual environments

Virtualization for Linux and Windows NT Platforms



Virtualization for Linux and Windows NT Platforms

- So far, most reported OS- Level Virtualizations are Linux based.
- Virtualization for Windows-based platform is still in research stage.
- Linux Kernel offers an abstraction layer, to allow software process to work with and operate on resources without knowing the hardware details.

Virtualization for Linux and Windows NT Platforms

Table 3.3 Virtualization Support for Linux and Windows NT Platforms

Virtualization Support and Source of Information	Brief Introduction on Functionality and Application Platforms
Linux vServer for Linux platforms (http://linux-vserver.org/)	Extends Linux kernels to implement a security mechanism to help build VMs by setting resource limits and file attributes and changing the root environment for VM isolation
OpenVZ for Linux platforms [65]; http://ftp.openvz.org/doc/OpenVZ-Users-Guide.pdf)	Supports virtualization by creating *virtual private servers (VPSes)*; the VPS has its own files, users, process tree, and virtual devices, which can be isolated from other VPSes, and checkpointing and live migration are supported
FVM (Feather-Weight Virtual Machines) for virtualizing the Windows NT platforms [78])	Uses system call interfaces to create VMs at the NY kernel space; multiple VMs are supported by virtualized namespace and copy-on-write

Virtualization at OS Level

• Advantages of OS Extension for Virtualization

1. VMs at OS level has minimum startup/shutdown costs, low resource requirement and high scalability
2. OS-level VM can easily synchronize with its environment

• Disadvantage of OS Extension for Virtualization

1. All VMs in the same OS container must have the same or similar guest OS, which restrict application flexibility & isolation of different VMs on the same physical machine.

Library Support level

- It creates execution environments for running alien programs on a platform rather than creating VM to run the entire operating system.
- Also known as **user-level Application Binary Interface (ABI)**
- It is done by API call interception and remapping.
- Typical systems: Wine, WABI, LxRun , VisualMainWin and vCUDA
- Advantage:
  - It has very low implementation effort
- Shortcoming & limitation:
  - Poor application flexibility and isolation

Virtualization with Middleware/Library Support

Table 3.4 Middleware and Library Support for Virtualization

Middleware or Runtime Library and References or Web Link	Brief Introduction and Application Platforms
WABI (http://docs.sun.com/app/docs/doc/802-6306)	Middleware that converts Windows system calls running on x86 PCs to Solaris system calls running on SPARC workstations
Lxrun (Linux Run) (http://www.ugcs.caltech.edu/~steven/lxrun/)	A system call emulator that enables Linux applications written for x86 hosts to run on UNIX systems such as the SCO OpenServer
WINE (http://www.winehq.org/)	A library support system for virtualizing x86 processors to run Windows applications under Linux, FreeBSD, and Solaris
Visual MainWin (http://www.mainsoft.com/)	A compiler support system to develop Windows applications using Visual Studio to run on Solaris, Linux, and AIX hosts
vCUDA (Example 3.2) (IEEE *IPDPS* 2009 [57])	Virtualization support for using general-purpose GPUs to run data-intensive applications under a special guest OS

The vCUDA for Virtualization of GPGPU

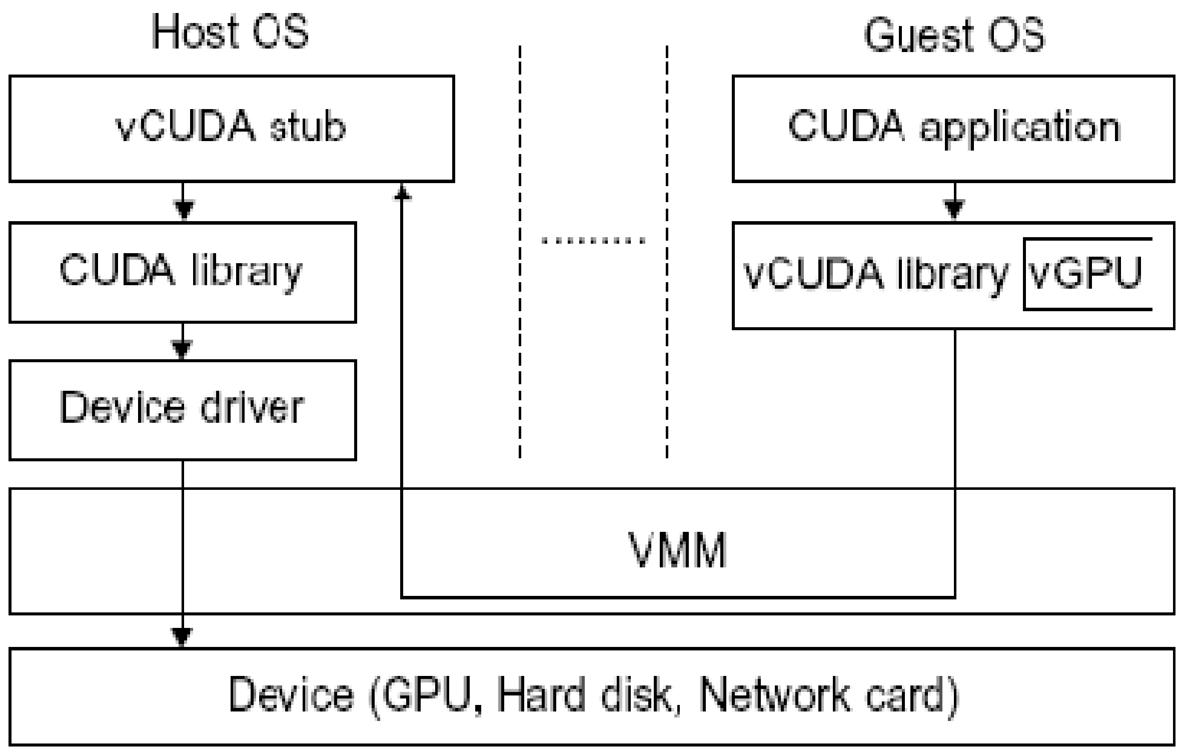


FIGURE 3.4

Basic concept of the vCUDA architecture.

(Courtesy of Lin Shi, et al. [57])

The vCUDA for Virtualization of GPGPU

- CUDA is programming model and library for GP- GPUs
- It provides GPUs to run compute intensive applications on host OS.
- It is difficult to run CUDA application on Hardware-level VMs directly.
- vCUDA virtualizes CUDA library.
- When CUDA applications run on guest OS and issues call to CUDA API, vCUDA intercepts and directs to CUDA API running on host OS.
- vCUDA employs client server model to implement CUDA virtualization.
- vGPU returns local virtual address to application and notify remote stub to allocate real device memory and execution context for APIs calls from guest OS.

User-Application level

- It virtualizes an application as a virtual machine.
- High Level Language VMs: This layer sits as an application program on top of an operating system and exports an abstraction of a VM that can run programs written and compiled to a particular abstract machine definition.
- Application-level or Process-Level Virtualization
- Typical systems: JVM , .NET CLI , Panot
- Advantage:
  - Best application isolation
- Shortcoming & limitation:
  - Low performance, low application flexibility and high implementation complexity.

Summary

Table 3.1 Relative Merits of Virtualization at Various Levels

Level of Implementation	Higher Performance	Application Flexibility	Implementation Complexity	Application Isolation
ISA	X	XXXXX	XXX	XXX
Hardware-level virtualization	XXXXX	XXX	XXXXX	XXXX
OS-level virtualization	XXXXX	XX	XXX	XX
Runtime library support	XXX	XX	XX	XX
User application level	XX	XX	XXXXX	XXXXX

More Xs mean higher merit

VMM Design Requirements

- First, a VMM should provide an environment for programs which is essentially identical to the original machine.
- Second, programs run in this environment should show, at worst, only minor decreases in speed.
- Third, a VMM should be in complete control of the system resources
  - The VMM is responsible for allocating hardware resources for programs.
  - It is not possible for a program to access any resource not explicitly allocated to it.
  - It is possible under certain circumstances for a VMM to regain control of resources already allocated.

Not all processors satisfy these requirements for a VMM. A VMM is tightly related to the architectures of processors. It is difficult to implement a VMM for some types of processors, such as the x86.

Summary

- Implementation Levels of Virtualization
  - ISA Level
  - Hardware Level
  - OS- Level
  - Library Level
  - Application Level
- VMM Design Requirements

THANK YOU