

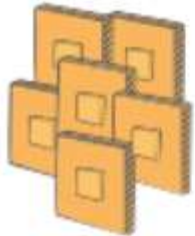
# CS351 - Cloud Computing

## Lecture #4



**Dr. Ferdous Ahmed Barbhuiya**  
Indian Institute of Information Technology Guwahati

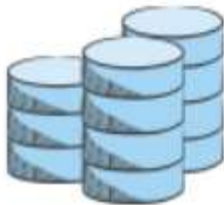
# Amazon Web Services (AWS)



**Compute**  
Amazon EC2



**Storage**  
Amazon S3



**Database**  
DynamoDB

- Elastic Cloud Compute (EC2)  
“Virtual Servers in the Cloud”
- Simple Storage Service (S3)  
“Scalable Storage in the Cloud”
- DynamoDB  
“Fast, Predictable, Highly-scalable  
NoSQL data store”
- Other services ...

<https://aws.amazon.com/>



# Virtualization

- In computing, virtualization is a broad term that refers to the **abstraction of computer resources**.
- Platform virtualization
- Resource virtualization

## Virtualization – What it provides?

- Decouple [OS, *service*] pair from hardware
- Multiplex lightly-used services on common host hardware
- Migrate services from host to host as needed
- Introduce new [OS, *service*] pairs as needed
  - Commissioning new services
  - Testing upgrades of existing services
  - Experimental usage

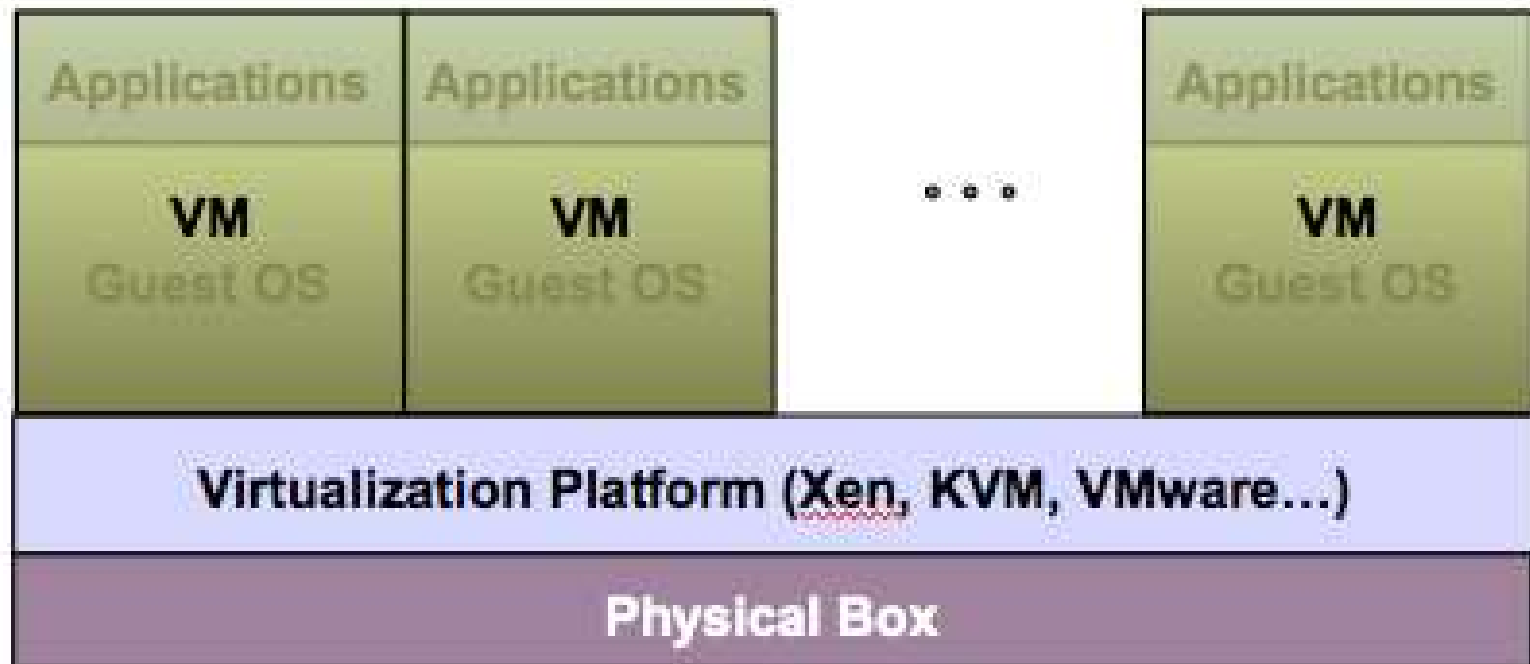
# Definition

- **Virtualization** is the ability to run multiple operating systems on a single physical system and share the underlying hardware resources\*
- It is the process by which one computer hosts the appearance of many computers.
- Virtualization is used to improve IT throughput and costs by using physical resources as a pool from which virtual resources can be allocated.

\*VMWare white paper, *Virtualization Overview*

# Virtualization Architecture

- A Virtual machine (VM) is an isolated runtime environment (guest OS and applications)
- Multiple virtual systems (VMs) can run on a single physical system



# Hypervisor

- A **hypervisor**, a.k.a. a virtual machine manager/monitor (VMM), or virtualization manager, is a program that allows multiple operating systems to share a single hardware host.
- Each guest operating system appears to have the host's processor, memory, and other resources all to itself. However, the hypervisor is actually controlling the host processor and resources, allocating what is needed to each operating system in turn and making sure that the guest operating systems (called virtual machines) cannot disrupt each other.

# Benefits of Virtualization

- Sharing of resources helps cost reduction
- Isolation: Virtual machines are isolated from each other as if they are physically separated
- Encapsulation: Virtual machines encapsulate a complete computing environment
- Hardware Independence: Virtual machines run independently of underlying hardware
- Portability: Virtual machines can be migrated between different hosts.



# Virtualization in Cloud Computing

Cloud computing takes virtualization one step further:

- You don't need to own the hardware
- Resources are rented as needed from a cloud
- Various providers allow creating virtual servers:
  - Choose the OS and software each instance will have
  - The chosen OS will run on a large server farm
  - Can instantiate more virtual servers or shut down existing ones within minutes
- You get billed only for what you used

# Definitions

- *Host Operating System:*
  - The operating system actually running on the hardware
  - Together with *virtualization layer*, it simulates environment for ...
- *Guest Operating System:*
  - The operating system running in the simulated environment
  - I.e., the one we are trying to isolate

## Virtual Machines (continued)

- Virtual-machine concept provides complete protection of system resources
  - Each *virtual machine* is isolated from all other virtual machines.
  - However, limited sharing of resources
- Virtual-machine system is a good vehicle for operating-systems research and development.
  - System development is done on the virtual machine does not disrupt normal operation
  - Multiple concurrent developers can work at same time
- The virtual machine concept is difficult to implement due to the effort required to provide an *exact duplicate* to the simulated machine

## Example – Page tables

- Suppose *guest OS* has its own page tables  
Then *virtualization layer* must
  - Copy those tables to its own
  - Trap every reference or update to tables and simulate it
- During page fault
  - *Virtualization layer* must decide whether fault belongs to *guest OS* or self
  - If *guest OS*, must simulate a page fault
- Likewise, *virtualization layer* must trap and simulate *every* privileged instruction in machine!

## Virtual Machines (continued)

- Some hardware architectures or features are impossible to *virtualize*
  - Certain registers or state not exposed
  - Unusual devices and device control
  - Clocks, time, and real-time behavior
- Solution – drivers or tools in guest OS
  - *VMware Tools*
  - *Xen* configuration options in Linux build

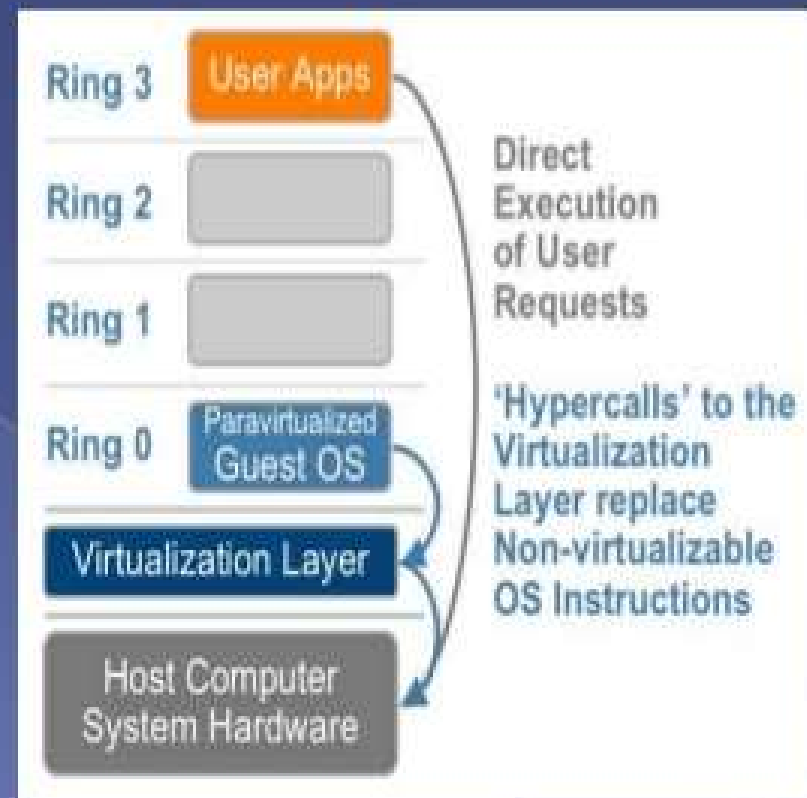
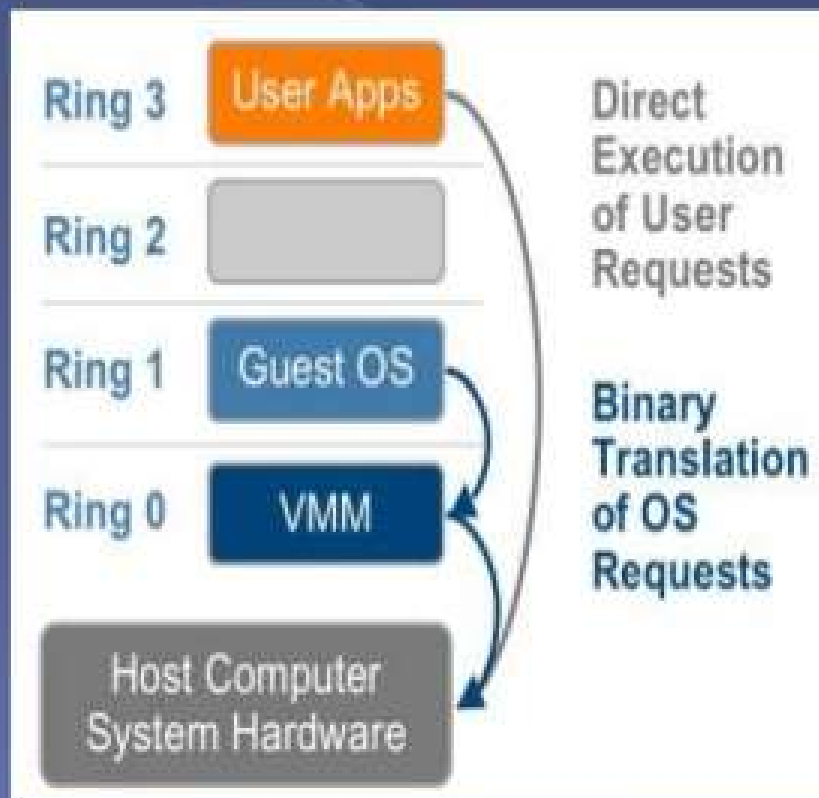
# Snapshots & Migration

- *Snapshot*: freeze a copy of virtual machine
  - Identify all pages in disk files, VM memory
  - Use copy-on-write for any subsequent modifications
  - To revert, throw away the copy-on-write pages
- *Migration*: move a VM to another host
  - Take snapshot (fast)
  - Copy all pages of snapshot (not so fast)
  - Copy modified pages (fast)
  - Freeze virtual machine and copy VM memory
    - Very fast, fractions of a second

# Cloning

- *Simple clone:*
  - Freeze virtual machine
  - Copy all files implementing it
  - Use copy-on-write to speed up
- *Linked clone:*
  - Take snapshot
  - Original and each clone is a copy-on-write version of snapshot

# Full vs. paravirtualization



Unlike Full virtualization, Paravirtualization involves modifying the OS kernel to replace nonvirtualizable instructions with hypercalls that communicate directly with the virtualization layer hypervisor.



# Application Virtualization

- typically for the purpose allowing application binaries to be portably run on many different computer architectures and operating systems.
- [http://en.wikipedia.org/wiki/Comparison\\_of\\_Application\\_Virtual\\_Machines](http://en.wikipedia.org/wiki/Comparison_of_Application_Virtual_Machines)
- Examples:
  - .NET CLR
  - JVM
  - Script Languages:Python,Ruby,Javascript...

# Resource Virtualization

- ◆ RAID
- ◆ SAN
- ◆ Channel bondings
- ◆ VPN/NAT
- ◆ Multiprocessor and multi-core
- ◆ Cluster and Grid computing
- ◆ Partitioning

# Virtualization Under Linux(1)

- ◆ UML (User Mode Linux)



- <http://user-mode-linux.sourceforge.net/>

- ◆ KVM (Kernal-based Virtual Machine)

- From Linux-2.6.20

- <http://kvm.qumranet.com/kvmwiki>



- ◆ XEN

- <http://xen.xensource.com/>



## Virtualization Under Linux(2)

- QEMU
  - <http://fabrice.bellard.free.fr/qemu/>
- QEMU Accelerators
  - KQEMU
  - QVM86
  - VirtualBox (released in January 2007)
  - KVM with QEMU

# Virtualization Under Linux(3)

- Bochs (GPLed, very slow)
  - A portable x86 and AMD64 PCs emulator mostly written in C++ and distributed as free software under GPL.
  - <http://bochs.sourceforge.net/>
- VirtualBox(commercial&open source, fast)
  - <http://www.virtualbox.org/>
- VMWare (Workstation,Server,Player)



# Virtualization Under Linux(4)

- SWSOFT Virtualizations

- <http://www.swsoft.com>



- <http://www.parallels.com/>



- <http://openvz.org/>



- Linux-VServer



- <http://linux-vserver.org/>

- Compare with:

- FreeBSD Jail
  - Solaris Containers (Zones)

# Linux Virtualization in Windows(1)

- VMWare
- Virtual PC
- VirtualBox
- Bochs
- QEMU

# Linux Virtualization in Windows(2)

- CoLinux Cooperative Lin

<http://www.colinux.org/>





- Topologilinux

<http://www.topologilinux.com>





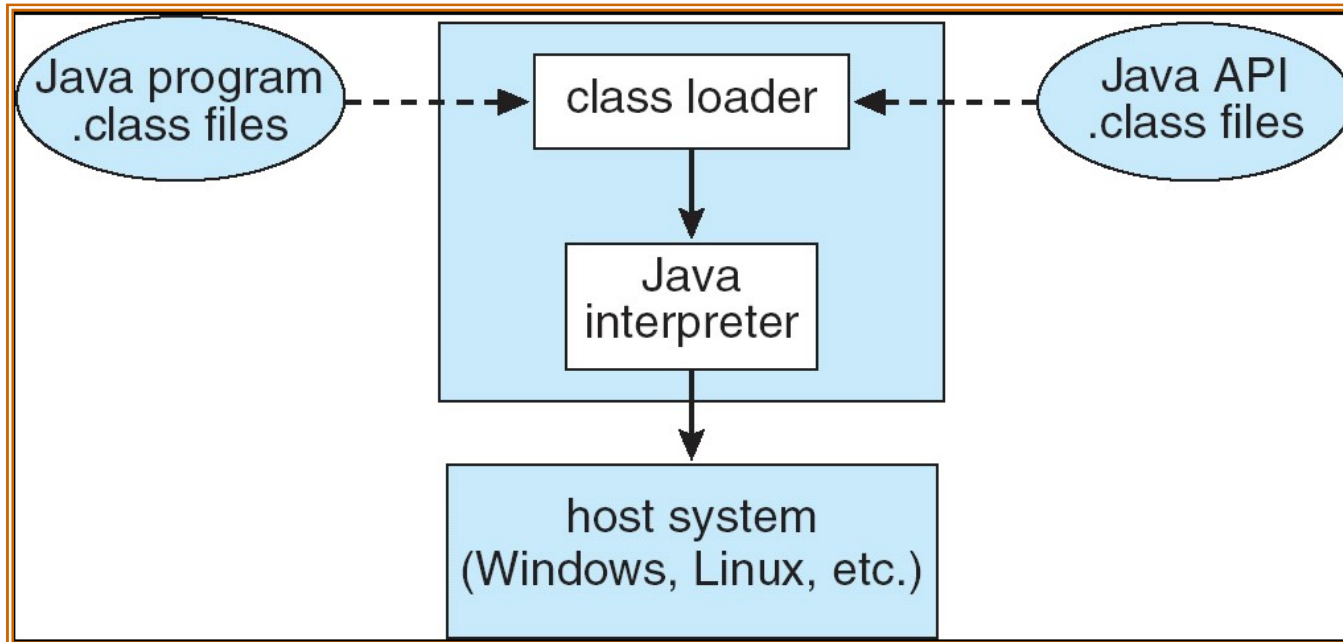
# Linux Virtualization in Windows(3)

- MinGW Minimalist GNU for Windows 
- Cygwin 
- [GNUWIN32](#)
- [GNUWINII](#)
- [UnxUtils](#)
- [UWIN](#)
- SFU  
Microsoft Windows Services for UNIX, aka Interix

## Must virtual machine be replica of host machine?

- No, *virtualization layer* can simulate *any* architecture
  - Typically used for debugging specialized systems
  - Real-time systems, niche products, etc.
- Guest architecture does not even have to be real hardware!

# The Java Virtual Machine



- Own idealized architecture
- Stylized machine language
  - *Byte codes*
- Readily available interpreter

# Comparison of virtual machines

- [http://en.wikipedia.org/wiki/Comparison\\_of\\_virtual\\_machines](http://en.wikipedia.org/wiki/Comparison_of_virtual_machines)

# References

- Wikipedia for any virtualization Terminologies
- Virtual Linux:An overview of virtualization methods, architectures, and implementations <http://www-128.ibm.com/developerworks/linux/library/l-linuxvirt/>