IIT KHARAGPUR

NPTEL ONLINE
CERTIFICATION COURSES

CLOUD COMPUTING

Virtualization

PROF. SOUMYA K. GHOSH

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

IIT KHARAGPUR

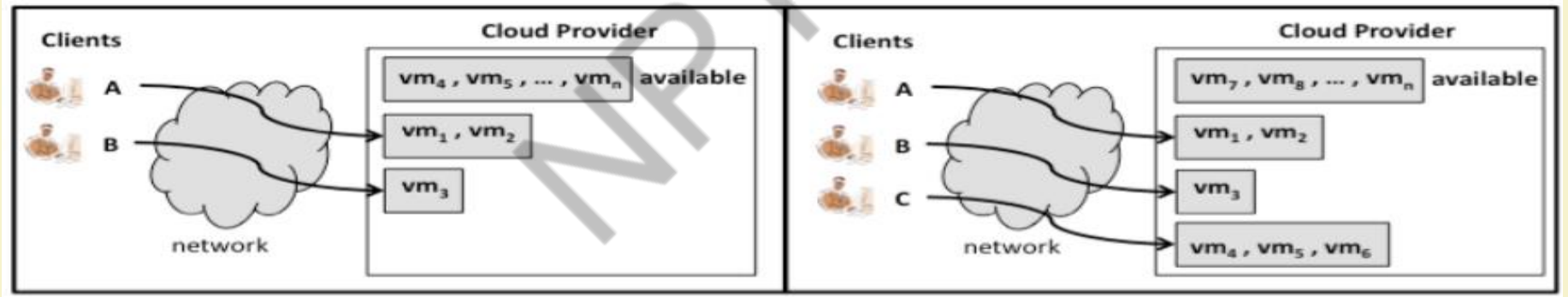
IaaS – Infrastructure as a Service

- What does a subscriber get?
  - Access to virtual computers, network-accessible storage, network infrastructure components such as firewalls, and configuration services.

- How are usage fees calculated?
  - Typically, per CPU hour, data GB stored per hour, network bandwidth consumed, network infrastructure used (e.g., IP addresses) per hour, value-added services used (e.g., monitoring, automatic scaling)

IaaS Provider/Subscriber Interaction Dynamics

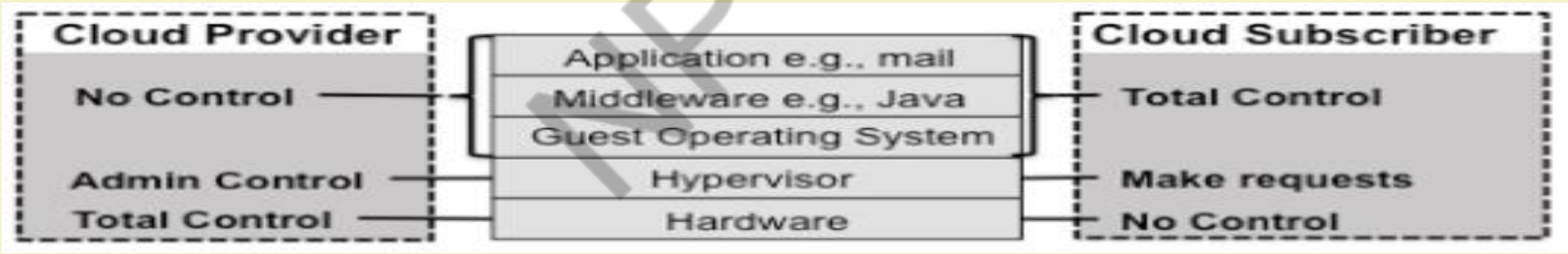
- The provider has a number of available virtual machines (vm's) that it can allocate to clients.
  - Client A has access to vm1 and vm2, Client B has access to vm3 and Client C has access to vm4, vm5 and vm6
  - Provider retains only vm7 through vmN



Source: LeeBadger, and Tim Grance "NIST DRAFT Cloud Computing Synopsis and Recommendations "

IaaS Component Stack and Scope of Control

- IaaS component stack comprises of hardware, operating system, middleware, and applications layers.
- Operating system layer is split into two layers.
  - Lower (and more privileged) layer is occupied by the Virtual Machine Monitor (VMM), which is also called the Hypervisor
  - Higher layer is occupied by an operating system running within a VM called a guest operating system



Source: LeeBadger, and Tim Grance "NIST DRAFT Cloud Computing Synopsis and Recommendations "

IaaS Component Stack and Scope of Control

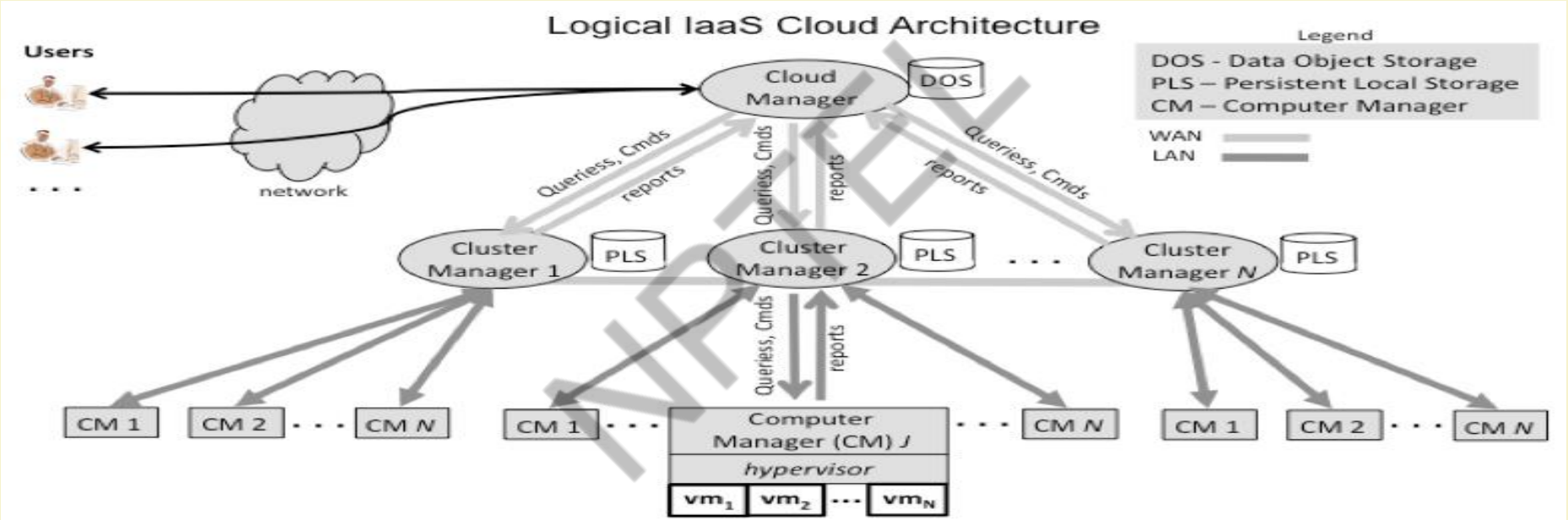
- In IaaS Cloud provider maintains total control over the physical hardware and administrative control over the hypervisor layer
- Subscriber controls the Guest OS, Middleware and Applications layers.
- Subscriber is free (using the provider's utilities) to load any supported operating system software desired into the VM.
- Subscriber typically maintains complete control over the operation of the guest operating system in each VM.

IaaS Component Stack and Scope of Control

- A hypervisor uses the hardware to synthesize one or more Virtual Machines (VMs); each VM is "an efficient, isolated duplicate of a real machine" .
- Subscriber rents access to a VM, the VM appears to the subscriber as actual computer hardware that can be administered (e.g., powered on/off, peripherals configured) via commands sent over a network to the provider.

IaaS Cloud Architecture

- Logical view of IaaS cloud structure and operation



Source: LeeBadger, and Tim Grance "NIST DRAFT Cloud Computing Synopsis and Recommendations "

IaaS Cloud Architecture

- Three-level hierarchy of components in IaaS cloud systems
  - *Top level* is responsible for *central control*
  - *Middle level* is responsible for *management of possibly large computer clusters* that may be *geographically distant* from one another
  - *Bottom level* is responsible for *running the host computer systems* on which virtual machines are created.
- Subscriber queries and commands generally flow into the system at the top and are forwarded down through the layers that either answer the queries or execute the commands

IaaS Cloud Architecture

- Cluster Manager can be geographically distributed
- Within a cluster manger computer manger is connected via high speed network.

Operation of the Cloud Manager

- Cloud Manager is the public access point to the cloud where subscribers sign up for accounts, manage the resources they rent from the cloud, and access data stored in the cloud.
- Cloud Manager has mechanism for:
  - Authenticating subscribers
  - Generating or validating access credentials that subscriber uses when communicating with VMs.
  - Top-level resource management.
- For a subscriber's request cloud manager determines if the cloud has enough free resources to satisfy the request

Data Object Storage (DOS)

- DOS generally stores the subscriber's metadata like user credentials, operating system images.
- DOS service is (usually) single for a cloud.

Operation of the Cluster Managers

- Each *Cluster Manager* is responsible for the operation of a collection of computers that are connected via high speed local area networks
- *Cluster Manager* receives resource allocation commands and queries from the *Cloud Manager*, and calculates whether part or all of a command can be satisfied using the resources of the computers in the cluster.
- *Cluster Manager* queries the *Computer Managers* for the computers in the cluster to determine resource availability, and returns messages to the *Cloud Manager*

Operation of the Cluster Managers

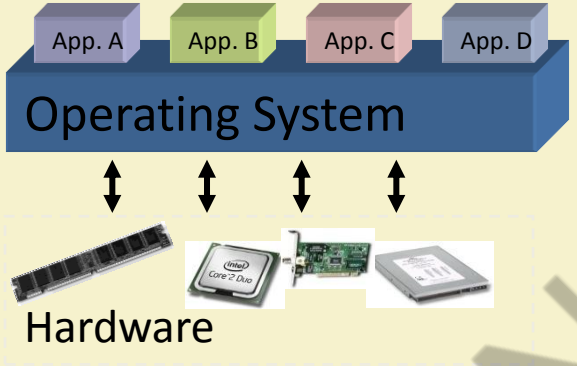
- Directed by the Cloud Manager, a Cluster Manager then instructs the Computer Managers to perform resource allocation, and reconfigures the virtual network infrastructure to give the subscriber uniform access.
- Each Cluster Manager is connected to Persistent Local Storage (PLS)
- PLS provide persistent disk-like storage to Virtual Machine

Operation of the Computer Managers

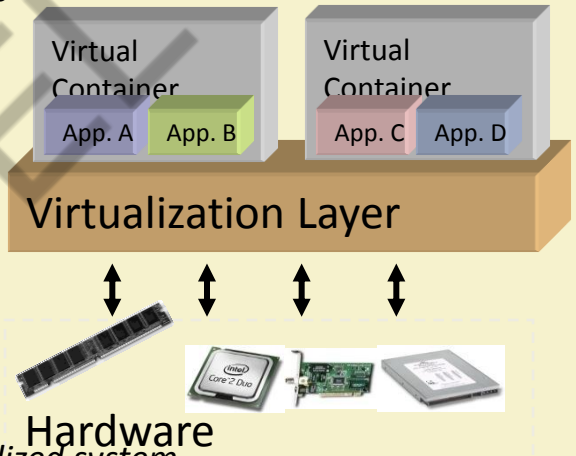
- At the lowest level in the hierarchy computer manger runs on each computer system and uses the concept of virtualization to provide Virtual Machines to subscribers
- Computer Manger maintains status information including how many virtual machines are running and how many can still be started
- Computer Manager uses the command interface of its hypervisor to start, stop, suspend, and reconfigure virtual machines

Virtualization

- Virtualization is a broad term (virtual memory, storage, network, etc)
- Focus: **Platform virtualization**
- Virtualization basically allows one computer to do the job of multiple computers, by sharing the resources of a single hardware across multiple environments



'Non-virtualized' system
A single OS controls all hardware platform resources

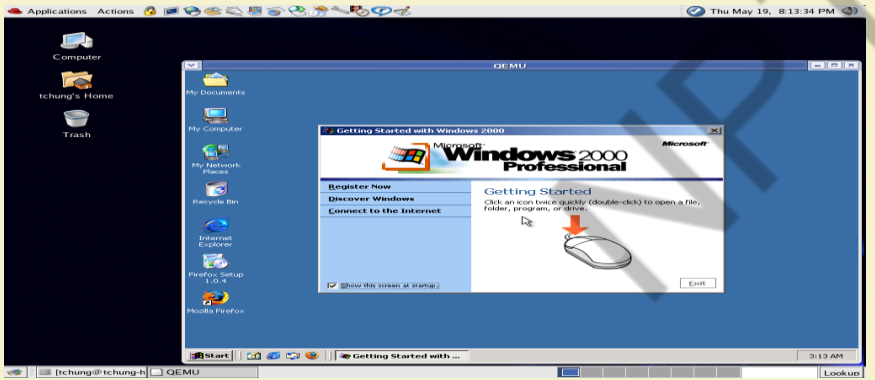


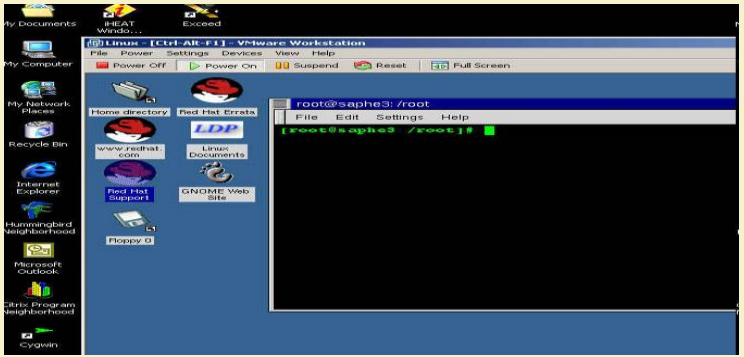
Virtualized system
It makes it possible to run multiple Virtual Containers on a single physical platform

Source: www.dc.uba.ar/events/eci/2008/courses/n2/Virtualization-Introduction.ppt

Virtualization

- Virtualization is way to run **multiple operating systems** and **user applications** on the same hardware
  - E.g., run both Windows and Linux on the same laptop
- How is it different from **dual-boot**?
  - Both OSes run **simultaneously**
- The OSes are completely **isolated** from each other





Hypervisor or Virtual Machine Monitor

Research Paper :Popek and Goldberg, "Formal requirements for virtualizable third generation architectures", CACM 1974 (http://portal.acm.org/citation.cfm?doid=361011.361073).

A **hypervisor** or **virtual machine monitor** runs the guest OS directly on the CPU. (This only works if the guest OS uses the same instruction set as the host OS.) Since the guest OS is running in user mode, privileged instructions must be intercepted or replaced. This further imposes restrictions on the instruction set for the CPU, as observed in a now-famous paper by Popek and Goldberg identify three goals for a virtual machine architecture:

- *Equivalence*: The VM should be indistinguishable from the underlying hardware.
- *Resource control*: The VM should be in complete control of any virtualized resources.
- *Efficiency*: Most VM instructions should be executed directly on the underlying CPU without involving the hypervisor.

Source: www.dc.uba.ar/events/eci/2008/courses/n2/Virtualization-Introduction.ppt

Hypervisor or Virtual Machine Monitor

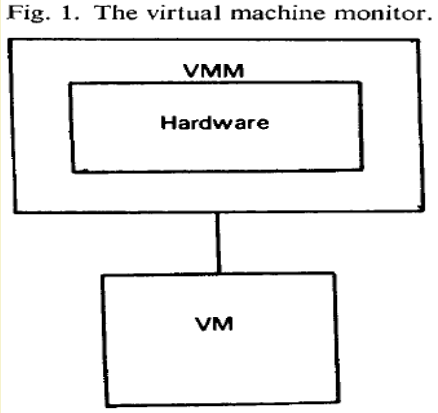
Popek and Goldberg describe (and give a formal proof of) the requirements for the CPU's instruction set to allow these properties. The main idea here is to classify instructions into

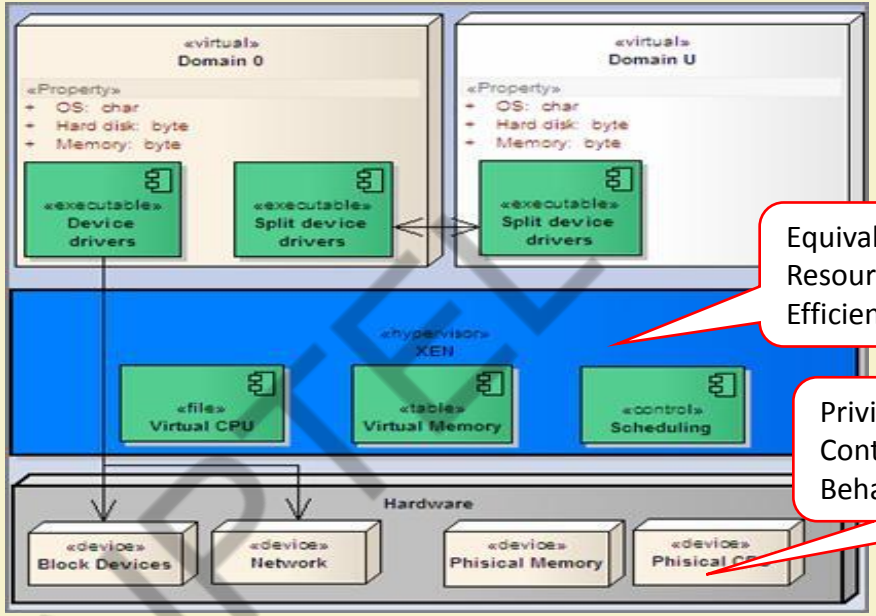
- **privileged** instructions, which cause a trap if executed in user mode, and
- **sensitive** instructions, which change the underlying resources (e.g. doing I/O or changing the page tables) or observe information that indicates the current privilege level (thus exposing the fact that the guest OS is not running on the bare hardware).
- The former class of sensitive instructions are called **control sensitive** and the latter **behavior sensitive** in the paper, but the distinction is not particularly important.

What Popek and Goldberg show is that we can only *run a virtual machine with all three desired properties if the sensitive instructions are a subset of the privileged instructions*. If this is the case, then we can run most instructions directly, and any sensitive instructions trap to the hypervisor which can then emulate them (hopefully without much slowdown).

Source: www.dc.uba.ar/events/eci/2008/courses/n2/Virtualization-Introduction.ppt

VMM and VM





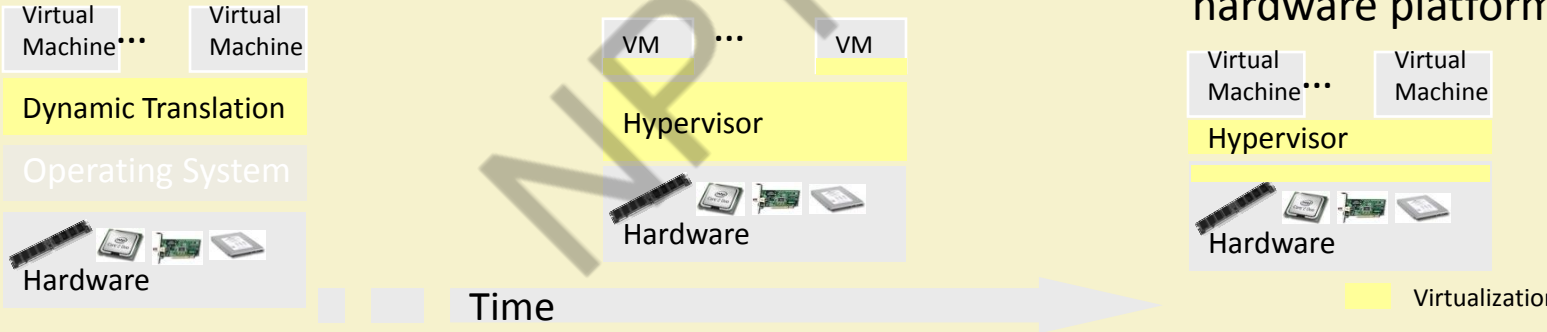
- For any conventional third generation computer, a VMM may be constructed if the set of sensitive instructions for that computer is a subset of the set of privileged instructions
- A conventional third generation computer is recursively virtualizable if it is virtualizable and a VMM without any timing dependencies can be constructed for it.

Source: www.dc.uba.ar/events/eci/2008/courses/n2/Virtualization-Introduction.ppt

Approaches to Server Virtualization

Evolution of Software Solutions

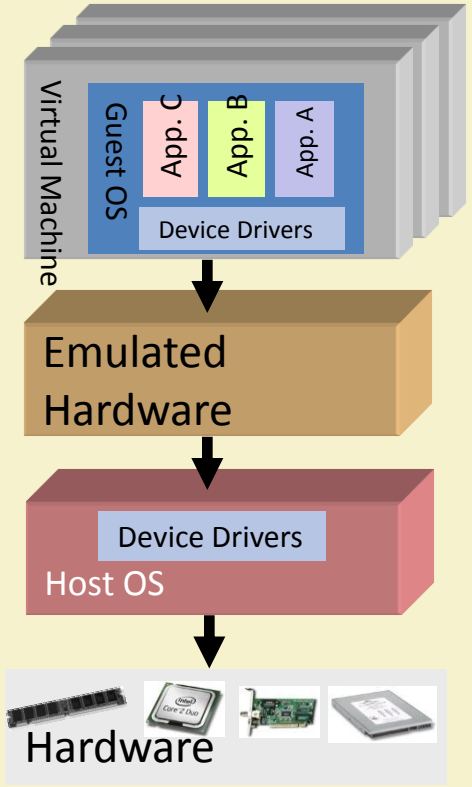
- 1st Generation: Full virtualization (Binary rewriting)
  - Software Based
  - VMware and Microsoft
- 2nd Generation: Para-virtualization
  - Cooperative virtualization
  - Modified guest
  - VMware, Xen
- 3rd Generation: Silicon-based (Hardware-assisted) virtualization
  - Unmodified guest
  - VMware and Xen on virtualization-aware hardware platforms



Source: www.dc.uba.ar/events/eci/2008/courses/n2/Virtualization-Introduction.ppt

Full Virtualization

- 1st Generation offering of x86/x64 server virtualization
- Dynamic binary translation
  - Emulation layer talks to an operating system which talks to the computer hardware
  - Guest OS doesn't see that it is used in an emulated environment
- All of the hardware is emulated including the CPU
- Two popular open source emulators are QEMU and Bochs



Source: www.dc.uba.ar/events/eci/2008/courses/n2/Virtualization-Introduction.ppt

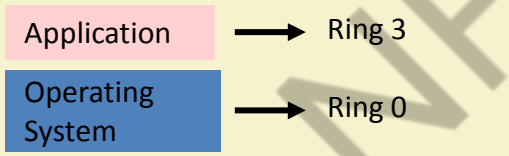
Full Virtualization - Advantages

- Emulation layer
  - Isolates VMs from the host OS and from each other
  - Controls individual VM access to system resources, preventing an unstable VM from impacting system performance
- Total VM portability
  - By emulating a consistent set of system hardware, VMs have the ability to transparently move between hosts with dissimilar hardware without any problems
    - It is possible to run an operating system that was developed for another architecture on your own architecture
    - A VM running on a Dell server can be relocated to a Hewlett-Packard server

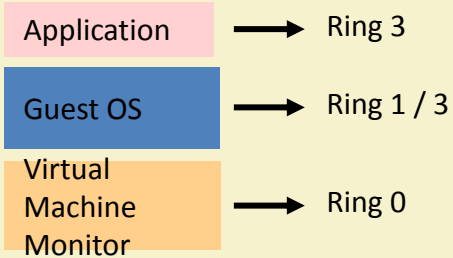
Source: www.dc.uba.ar/events/eci/2008/courses/n2/Virtualization-Introduction.ppt

Full Virtualization - Drawbacks

- Hardware emulation comes with a performance price
- In traditional x86 architectures, OS kernels expect to run privileged code in Ring 0
  - However, because Ring 0 is controlled by the host OS, VMs are forced to execute at Ring 1/3, which requires the VMM to trap and emulate instructions
- Due to these performance limitations, para-virtualization and hardware-assisted virtualization were developed



Traditional x86 Architecture

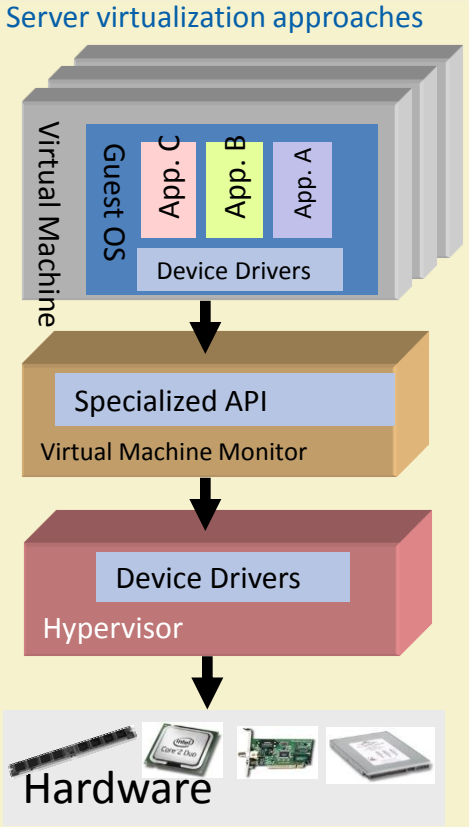


Full Virtualization

Source: www.dc.uba.ar/events/eci/2008/courses/n2/Virtualization-Introduction.ppt

Para-Virtualization

- Guest OS is modified and thus run kernel-level operations at Ring 1 (or 3)
  - Guest is fully aware of how to process privileged instructions
  - Privileged instruction translation by the VMM is no longer necessary
  - Guest operating system uses a specialized API to talk to the VMM and, in this way, execute the privileged instructions
- VMM is responsible for handling the virtualization requests and putting them to the hardware



Source: www.dc.uba.ar/events/eci/2008/courses/n2/Virtualization-Introduction.ppt

Para-Virtualization

- Today, VM guest operating systems are para-virtualized using two different approaches:

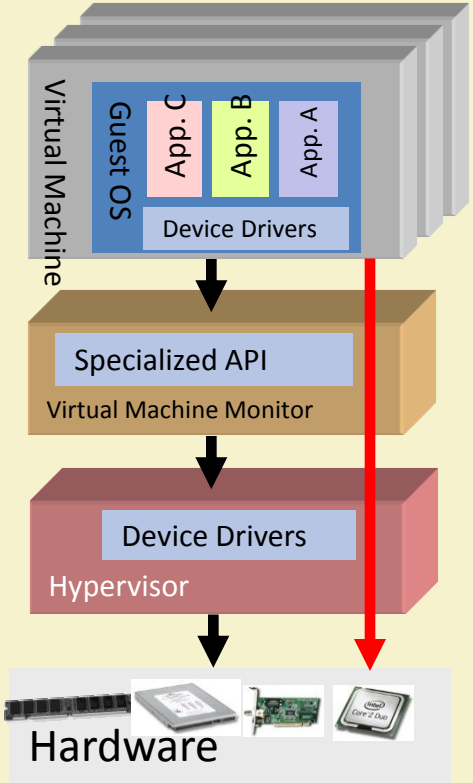
– ***Recompiling the OS kernel***
  - Para-virtualization drivers and APIs must reside in the guest operating system kernel
  - You do need a modified operating system that includes this specific API, requiring a compiling operating systems to be virtualization aware
    – Some vendors (such as Novell) have embraced para-virtualization and have provided para-virtualized OS builds, while other vendors (such as Microsoft) have not

– ***Installing para-virtualized drivers***
  - In some operating systems it is not possible to use complete para-virtualization, as it requires a specialized version of the operating system
  - To ensure good performance in such environments, para-virtualization can be applied for individual devices
  - For example, the instructions generated by network boards or graphical interface cards can be modified before they leave the virtualized machine by using para-virtualized drivers

Source: www.dc.uba.ar/events/eci/2008/courses/n2/Virtualization-Introduction.ppt

Hardware-assisted virtualization

- Guest OS runs at ring 0
- VMM uses processor extensions (such as Intel®-VT or AMD-V) to intercept and emulate privileged operations in the guest
- Hardware-assisted virtualization removes many of the problems that make writing a VMM a challenge
- VMM runs in a more privileged ring than 0, a *Virtual-1* ring is created



Source: www.dc.uba.ar/events/eci/2008/courses/n2/Virtualization-Introduction.ppt

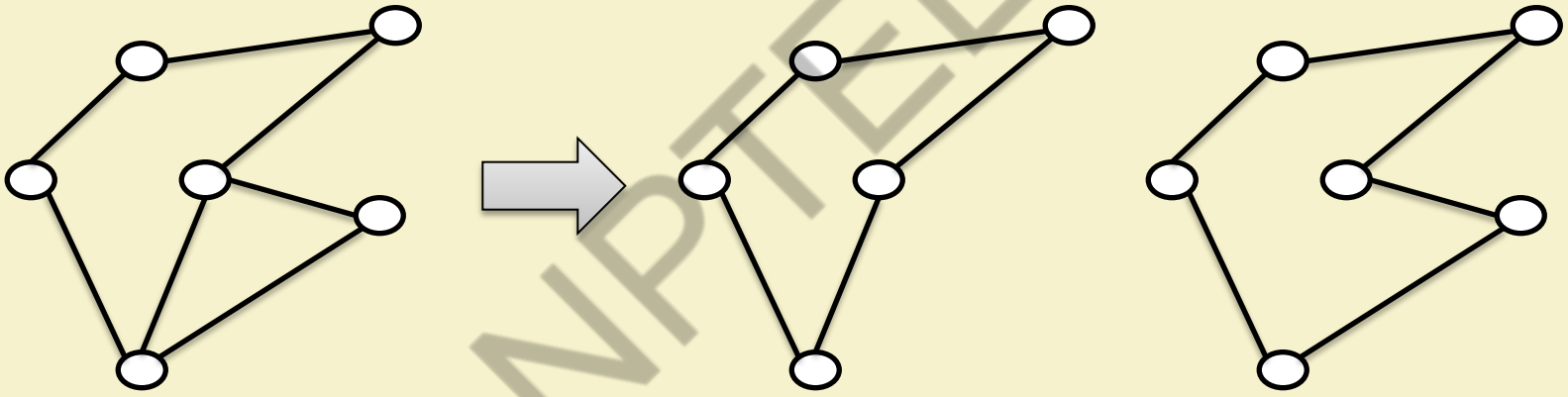
Hardware-assisted virtualization

- **Pros**
  - It allows to run unmodified OSs (so legacy OS can be run without problems)
- **Cons**
  - Speed and Flexibility
    - An unmodified OS does not know it is running in a virtualized environment and so, it can't take advantage of any of the virtualization features
      - It can be resolved using para-virtualization partially

Source: www.dc.uba.ar/events/eci/2008/courses/n2/Virtualization-Introduction.ppt

Network Virtualization

Making a physical network appear as multiple logical ones



Physical Network

Virtualized Network - 1

Virtualized Network - 2

Why Virtualize ?

- Internet is *almost* "paralyzed"
  - Lots of makeshift solutions (e.g. overlays)
  - A new architecture (aka clean-slate) is needed
- Hard to come up with a *one-size-fits-all* architecture
  - Almost impossible to predict what future might unleash
- Why not create an *all-sizes-fit-into-one* instead!
  - Open and expandable architecture
- Testbed for future networking architectures and protocols

Related Concepts

- Virtual Private Networks (VPN)
  - Virtual network connecting distributed sites
  - Not customizable enough
- Active and Programmable Networks
  - Customized network functionalities
  - Programmable interfaces and active codes
- Overlay Networks
  - Application layer virtual networks
  - Not flexible enough

Network Virtualization Model

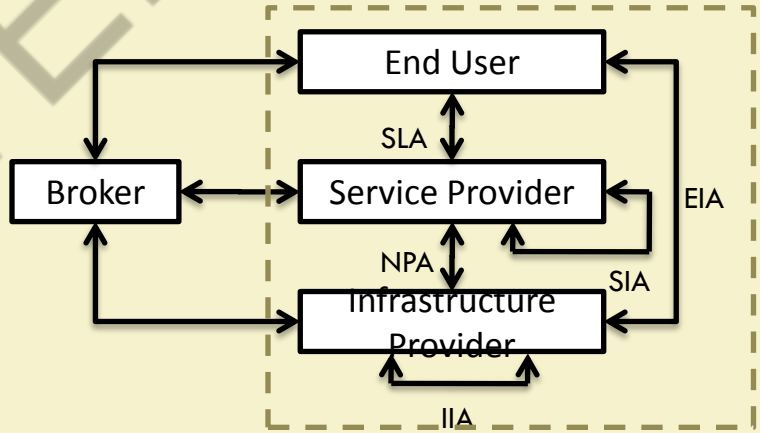
- Business Model
- Architecture
- Design Principles
- Design Goals

Business Model

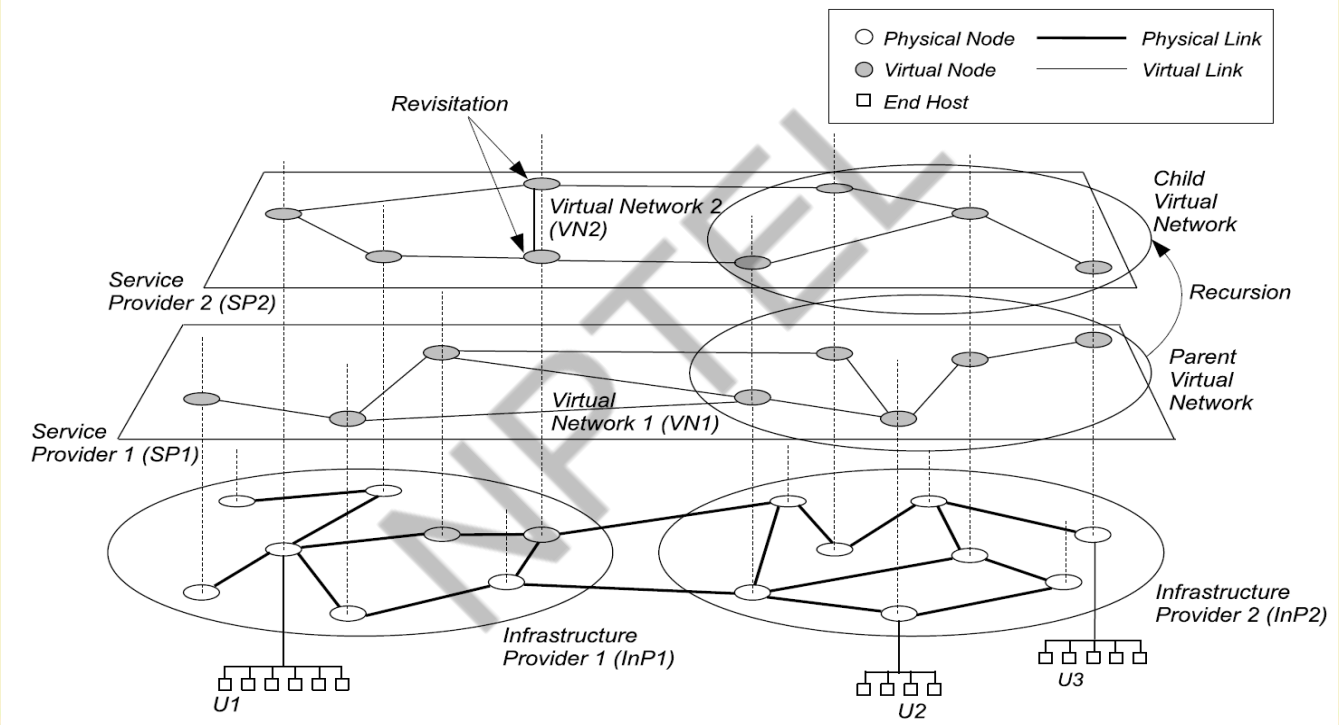
Players

- Infrastructure Providers (*InPs*)
  - Manage underlying physical networks
- Service Providers (*SPs*)
  - Create and manage virtual networks
  - Deploy customized end-to-end services
- End Users
  - Buy and use services from different service providers
- Brokers
  - Mediators/Arbiters

Relationships



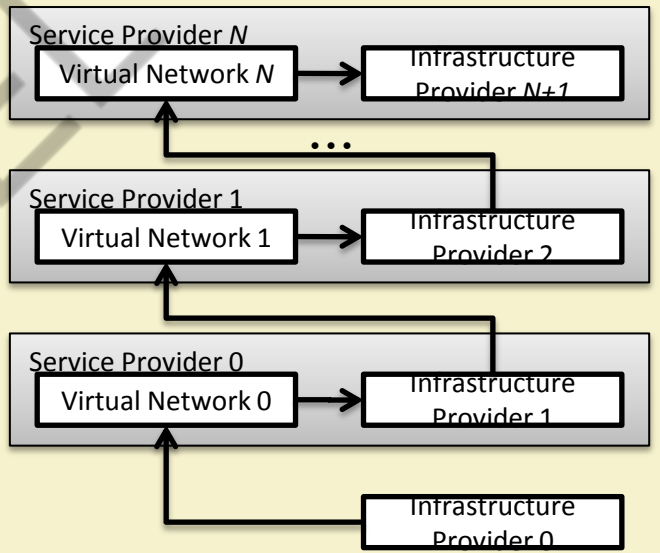
Architecture



Design Principles

- *Concurrence* of multiple heterogeneous virtual networks
  - Introduces diversity
- *Recursion* of virtual networks
  - Opens the door for network virtualization economics
- *Inheritance* of architectural attributes
  - Promotes value-addition
- *Revisitation* of virtual nodes
  - Simplifies network operation and management

Hierarchy of Roles



Design Goals (1)

- Flexibility
  - Service providers can choose
    - arbitrary network topology,
    - routing and forwarding functionalities,
    - customized control and data planes
  - No need for co-ordination with others
    - IPv6 fiasco should never happen again

- Manageability
  - Clear separation of policy from mechanism
  - Defined *accountability* of infrastructure and service providers
  - Modular management

Design Goals (2)

- Scalability
  - Maximize the number of co-existing virtual networks
  - Increase resource utilization and amortize CAPEX and OPEX

- Security, Privacy, and Isolation
  - Complete isolation between virtual networks
    - *Logical* and *resource*
  - Isolate faults, bugs, and misconfigurations
    - Secured and private

Design Goals (3)

- Programmability
  - Of network elements e.g. routers
  - Answer *"How much"* and *"how"*
  - Easy and effective without being vulnerable to threats

- Heterogeneity
  - Networking technologies
    - Optical, sensor, wireless etc.
  - Virtual networks

Design Goals (4)

- Experimental and Deployment Facility
  - PlanetLab, GENI, VINI
  - Directly deploy services in real world from the testing phase

- Legacy Support
  - Consider the existing Internet as a member of the collection of multiple virtual Internets
  - Very *important* to keep all concerned parties satisfied

Definition

Network virtualization is a networking environment that allows multiple service providers to dynamically compose multiple heterogeneous virtual networks that co-exist together in isolation from each other, and to deploy customized end-to-end services on-the-fly as well as manage them on those virtual networks for the end-users by effectively sharing and utilizing underlying network resources leased from multiple infrastructure providers.

Typical Approach

- Networking technology
  - IP, ATM
- Layer of virtualization

- Architectural domain
  - Network resource management, Spawning networks
- Level of virtualization
  - Node virtualization, Full virtualization

Thank You!