Lecture#3 –Virtualization

S. M. Ahsan Kazmi

Outline

- Introduction
  - Recap
- Virtualization Reference Models
- Types of virtualization

Virtualization

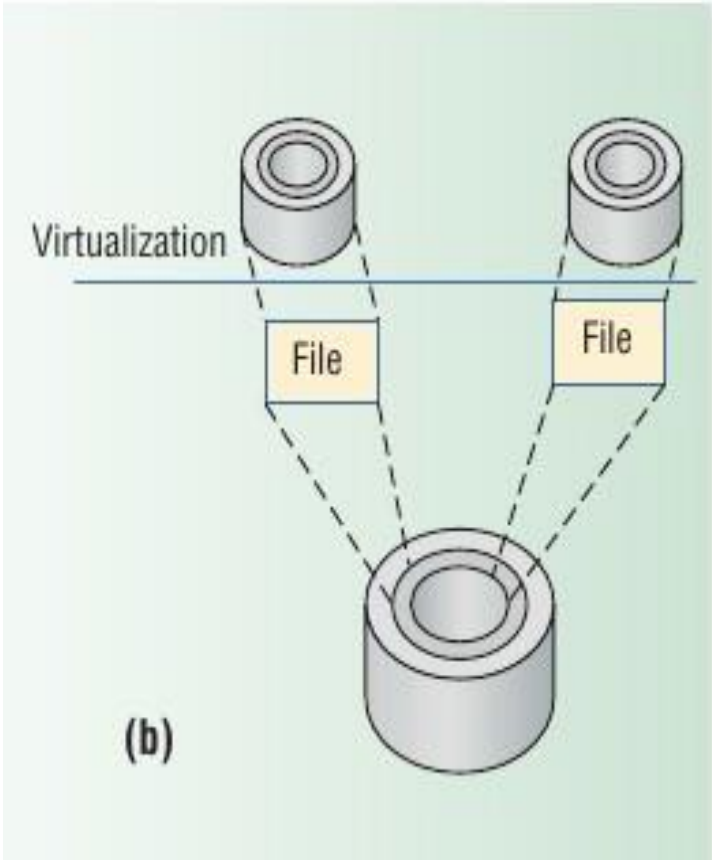
- What is Virtualization?
  - In computing, virtualization refers to the act of creating a virtual (rather than actual) version of something, including virtual hardware platforms, storage devices, and computer network resources.
  - "a technique for hiding the physical characteristics of computing resources from the way in which other systems, applications, or end users interact with those resources."
- Virtualization is the process of making things more abstract in order to make them easier to use.

Virtualization

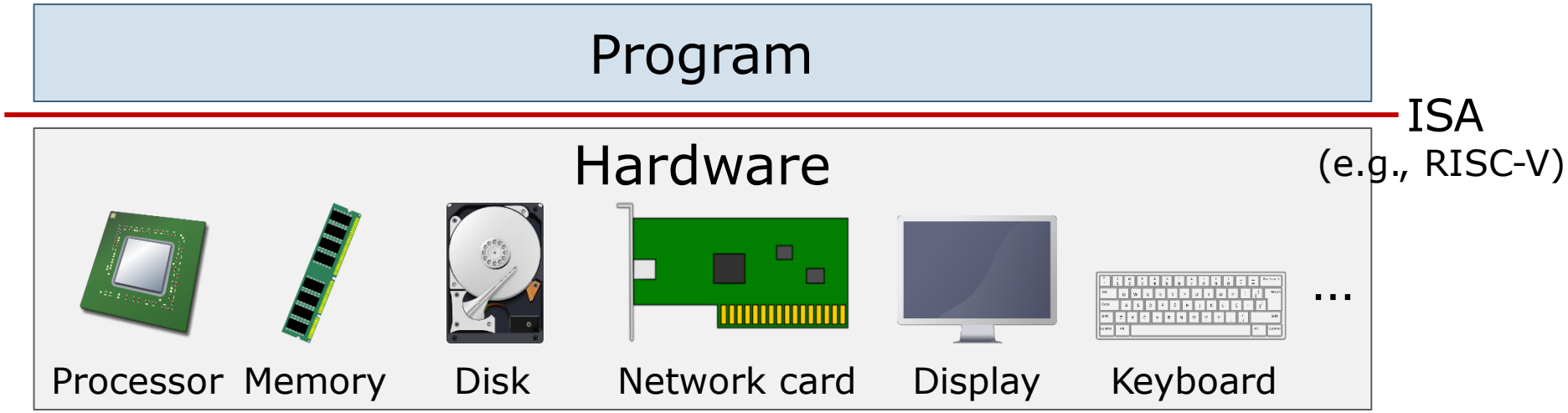
- Virtualization of systems or components like: processors, memory or an I/O device.

- It ***transforms*** a entire system or components of the system

- Ex. disk storage

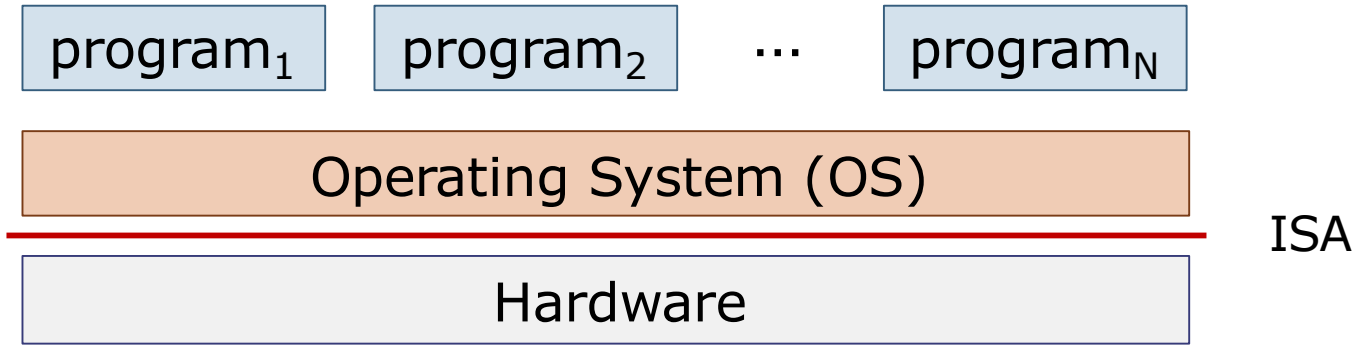


Single-User Machines



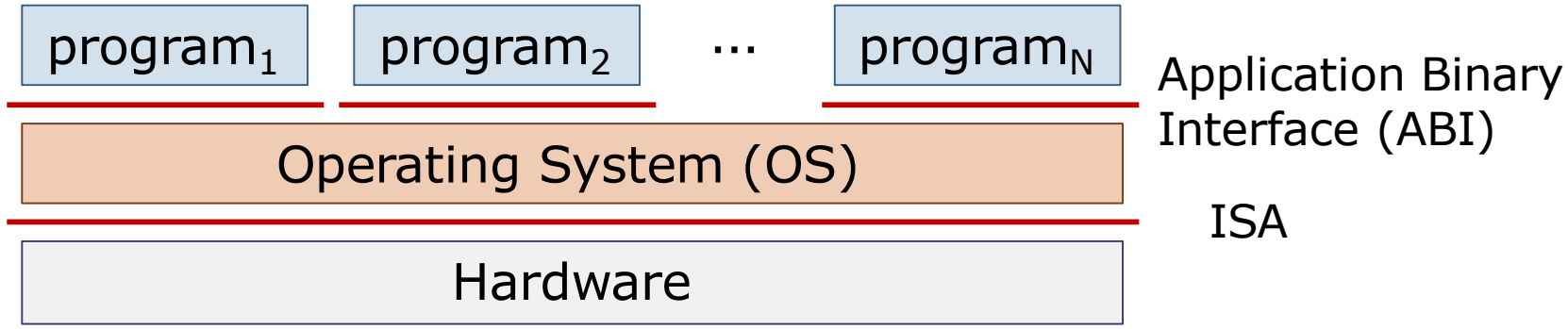
- Hardware executes a single program
- This program has direct and complete access to all hardware resources in the machine
- The instruction set architecture (ISA) is the interface between software and hardware

Operating Systems



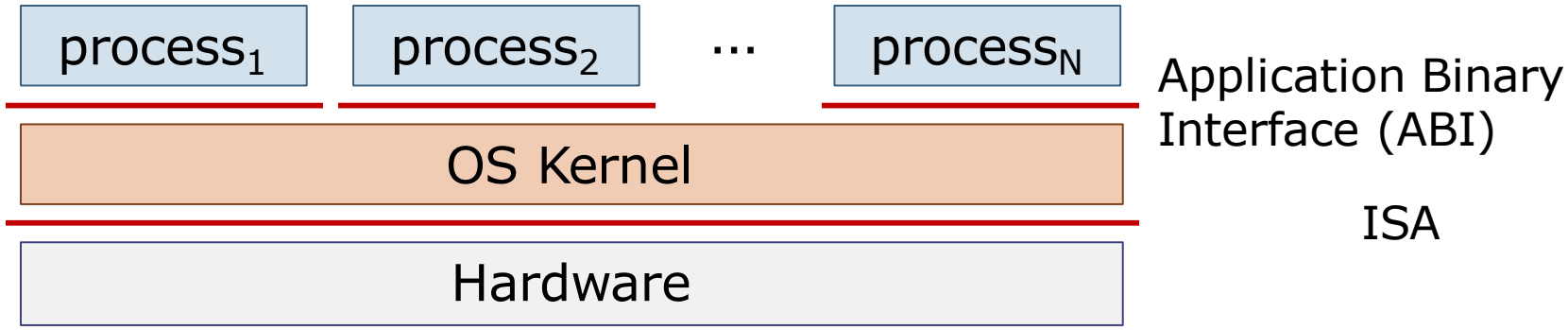
- Multiple executing programs share the machine
- Each executing program does not have direct access to hardware resources
- Instead, an operating system (OS) controls these programs and how they share hardware resources
  - Only the OS has unrestricted access to hardware

Operating Systems and Interfaces



- Instead, an operating system (OS) controls these programs and how they share hardware resources
  - Only the OS has unrestricted access to hardware
- The application binary interface (ABI) is the interface between programs and the OS

Goals of Operating Systems

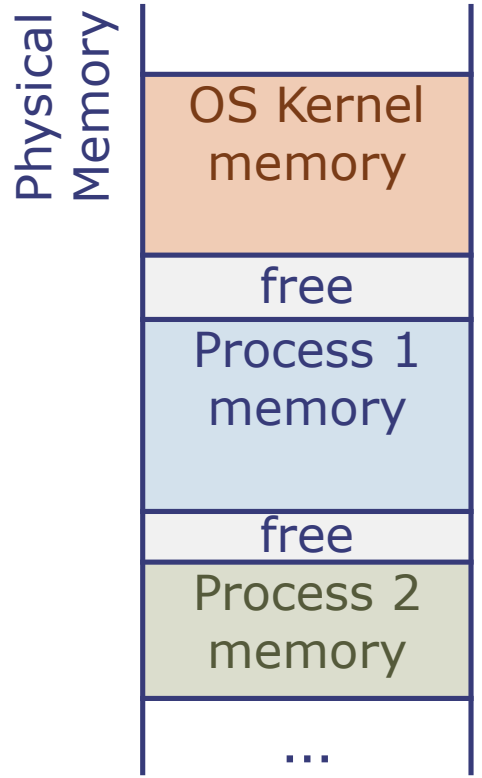


- **Protection and privacy:** Processes cannot access each other's data
- **Abstraction:** OS hides details of underlying hardware
- **Resource management:** OS controls how processes share hardware (CPU, memory, disk, etc.)

Operating Systems: The Big Picture

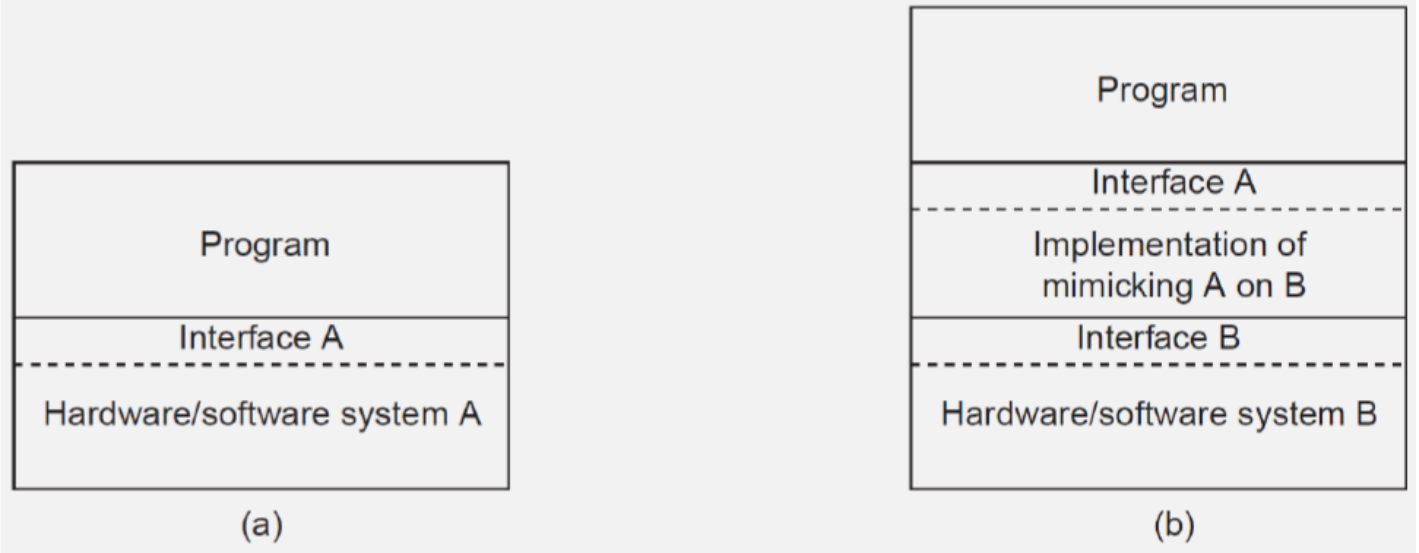
- The OS kernel provides a private address space to each process
  - Each process is allocated space in physical memory via OS
  - A process is not allowed to access the memory of other processes
- The OS kernel schedules processes into the CPU
  - Each process is given a fraction of CPU time
  - A process cannot use more CPU time than allowed

  - The OS kernel lets processes invoke system services (e.g., access files or network sockets) via system calls



Virtualization

- Virtualization: extend or replace an existing interface to mimic the behavior of another system.
  - Introduced in 1970s: run legacy software on newer mainframe hardware
- Handle platform diversity by running apps in VMs
  - Portability and flexibility

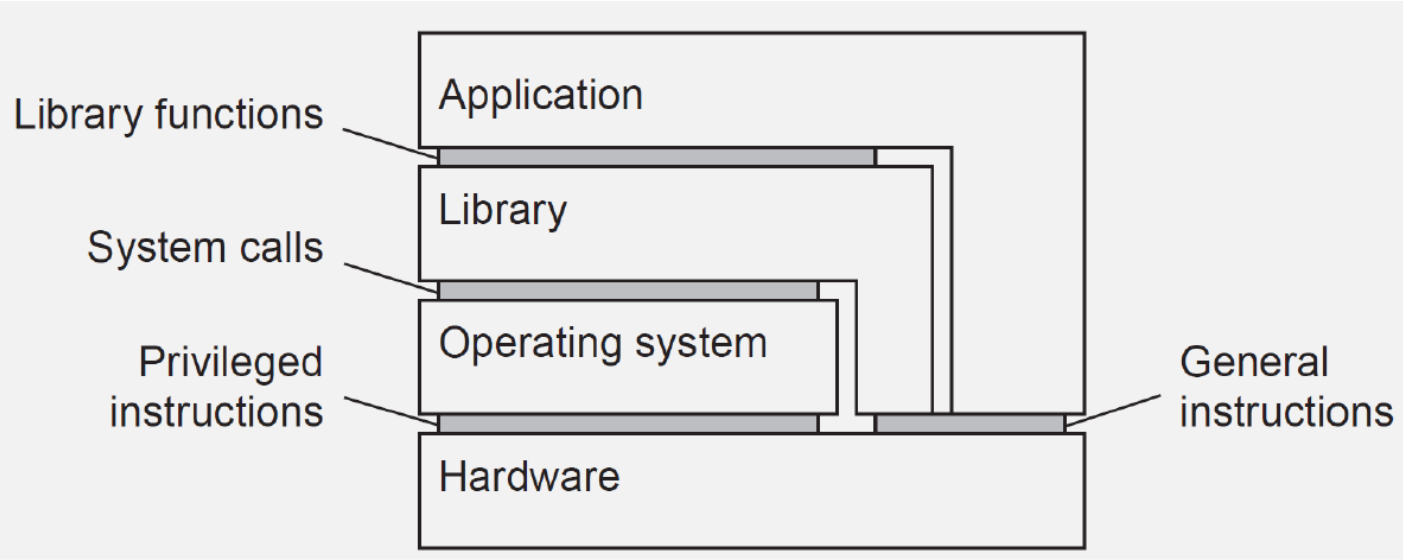


Machine Reference Model

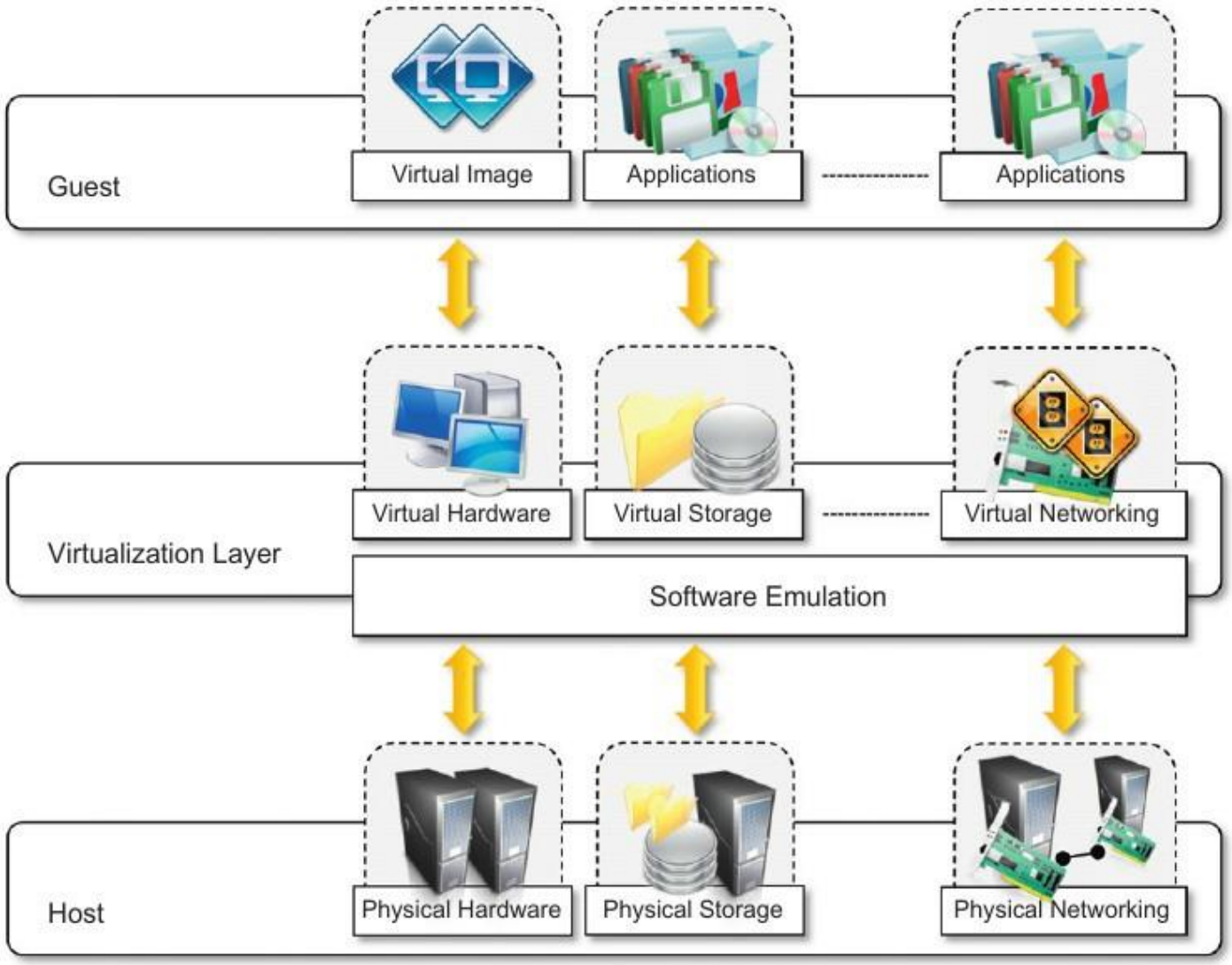
- It defines the interfaces between the levels of abstractions, which hide implementation details.
- Virtualization techniques replace one of the layers and intercept the calls that are directed towards it.
- Machine is defined by an interface and the interfaces can be virtualized

Types of Interfaces

- Different types of interfaces
  - Assembly instructions (Hardware virtualization)
  - System calls (OS-level virtualization)
  - APIs (Application-level virtualization)
- Depending on what is replaced /mimicked, we obtain different forms of virtualization



Virtualization Reference Model



Types of Virtualization

- Emulation
  - VM emulates/simulates complete hardware (CPU, disk, NIC)
  - Unmodified guest OS for a different PC can be run

- Full/native Virtualization
  - VM simulates “enough” hardware to allow an unmodified guest OS to be run in isolation
    - Same hardware CPU
  - IBM VM family, VMWare Workstation, Parallels, VirtualBox

Types of Virtualization

- Para-virtualization
  - VM does not simulate hardware
  - Use special API that a modified guest OS must use
- OS-level virtualization
  - OS allows multiple secure virtual servers to be run
  - Guest OS is the same as the host OS, but appears isolated
- Application-level virtualization
  - Application is giving its own copy of components that are not shared

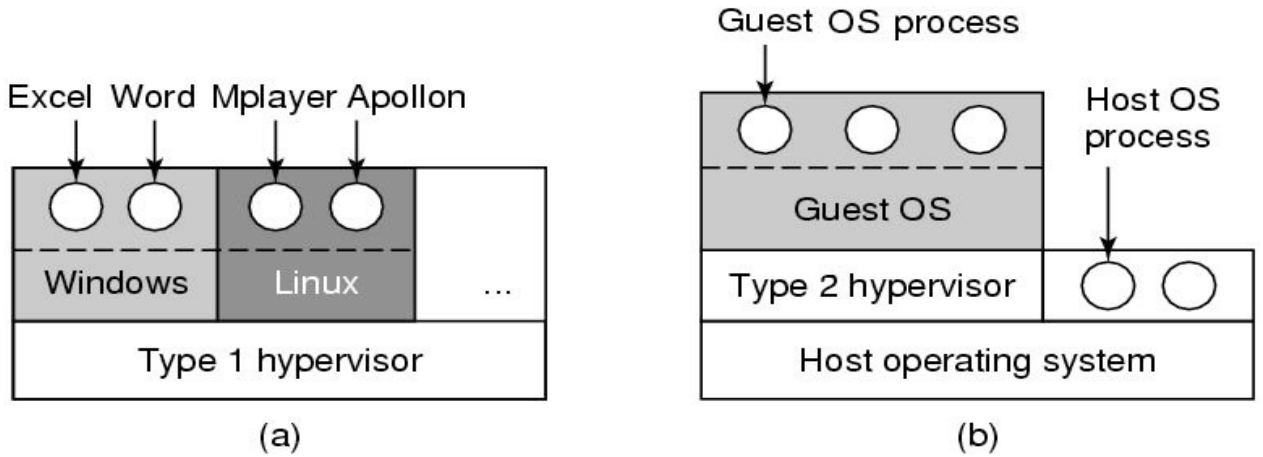
How Virtualization works?

- CPU supports kernel and user mode (ring0, ring3)
  - Set of instructions that can only be executed in kernel mode
    - I/O, change MMU settings, etc -- *sensitive instructions*
  - Privileged instructions: cause a trap when executed in user mode
- Result: type 1 virtualization feasible if sensitive instruction subset of privileged instructions
- Intel x86: ignores sensitive instructions in user mode
  - Can not support type 1 virtualization
- Recent Intel/AMD CPUs have hardware support
  - Intel VT, AMD SVM
    - Create containers where a VM and guest can run

Hypervisor

- The hypervisor runs in the supervisor mode.
- It recreates a h/w environment.
- It is a piece of s/w that enables running one or more VMs on a physical server(host).
- Two major types of hypervisor
  - ***Type -I***
  - ***Type-II***

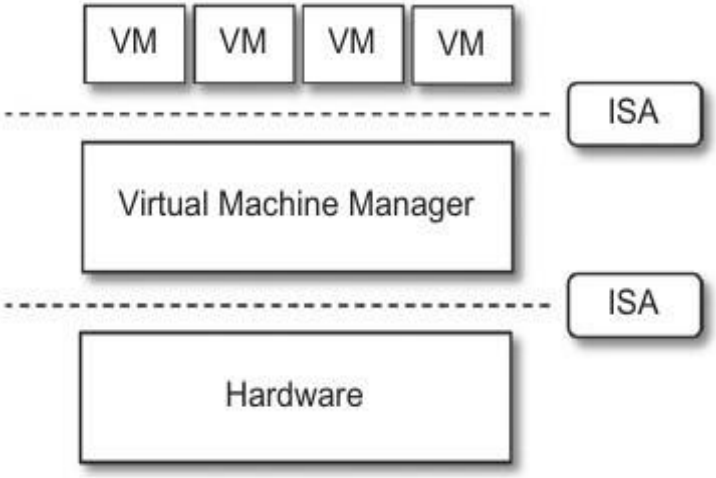
Types of Hypervisors



- Hypervisor/VMM: virtualization layer
  - resource management, isolation, scheduling, …
- Type 1: hypervisor runs on “bare metal”
- Type 2: hypervisor runs on a host OS
  - Guest OS runs inside the hypervisor
- Both VM types act like real hardware

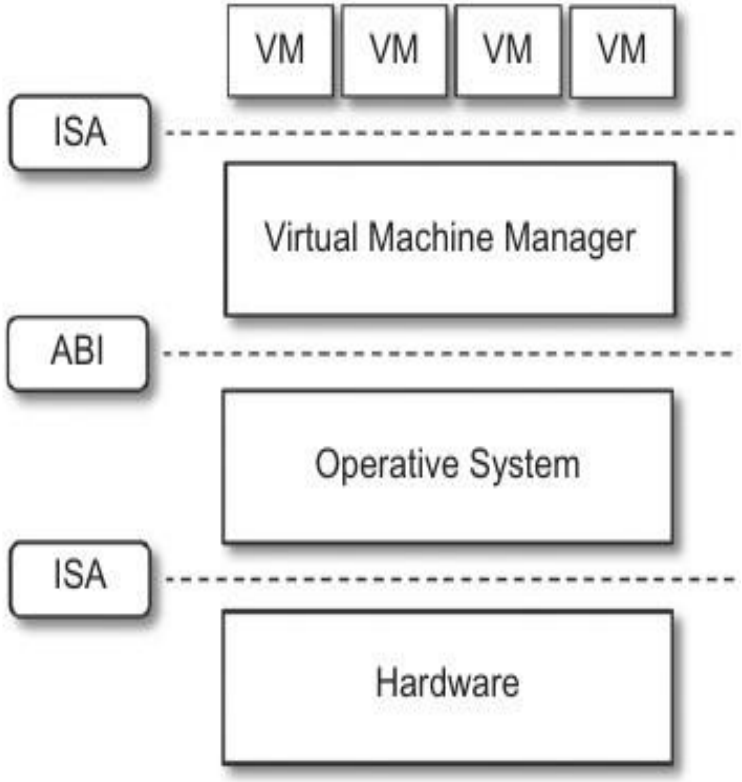
Type-I Hypervisor

- It runs directly on top of the hardware.
- Takes place of the OS.
- Directly interact with the ISA exposed by the underlying hardware.
- Also known as *native virtual machine*.



Type-II Hypervisor

- It require the support of an operating system to provide virtualization services.
- Programs managed by the OS.
- Emulate the ISA of virtual h/w.
- Also called hosted virtual machine.
- Hypervisor performs binary translation on the fly.



Paravirtualization

- Both type 1 and 2 hypervisors work on unmodified OS
- It was developed as a workaround for Type 1 hypervisor that is needed to run on old hardware which does not cause traps on sensitive instructions.
- Paravirtualization: modify OS kernel to replace all sensitive instructions with hyper calls
  - OS behaves like a user program making system calls
- Hypervisor executes the privileged operation invoked by hypercall.

Memory virtualization

- OS manages page tables
  - Create a new page table that is sensitive -> traps to the hypervisor
- hypervisor manages multiple OS
  - Need a second shadow page table
  - OS: VM virtual pages to VM's physical pages
  - Hypervisor maps to the actual page in a shadow page table
  - Two-level mapping
  - Need to catch changes to page table (not privileged)
    - Change PT to read-only

I/O Virtualization

- Each guest OS thinks it “owns” the disk
- Hypervisor creates “virtual disks”
  - Large empty files on the physical disk that appear as “disks” to the guest OS
    - Hypervisor converts block # to file offset for I/O
- NIC Virtualization

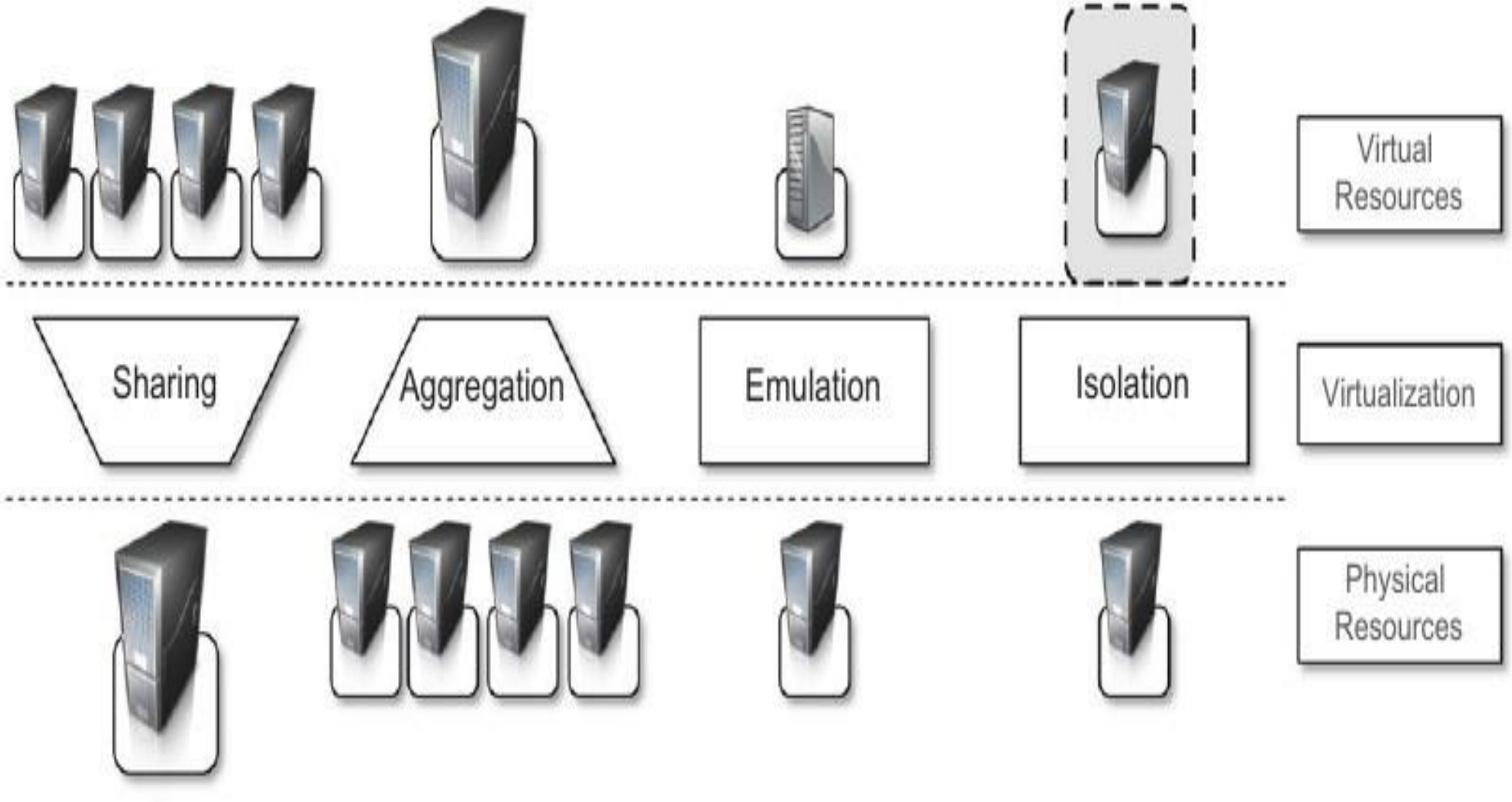
Advantages of Virtualization

- **Increased Security**
  - Ability to control the execution of a guest
  - Guest is executed in emulated environment.
  - Virtual Machine Manager control and filter the activity of the guest.
  - Hidding of resources.
  - Having no effect on other users/guest environment.

Advantages of Virtualization

- **Managed Execution types**
  - **Sharing**
    - Creating separate computing environment within the same host.
    - Underline host is fully utilized.
  - **Aggregation**
    - A group of separate hosts can be tied together and represented as single virtual host.
  - **Emulation**
    - Controlling & Tuning the environment exposed to guest.
  - **Isolation**
    - Complete separate environment for guests.

Managed Execution



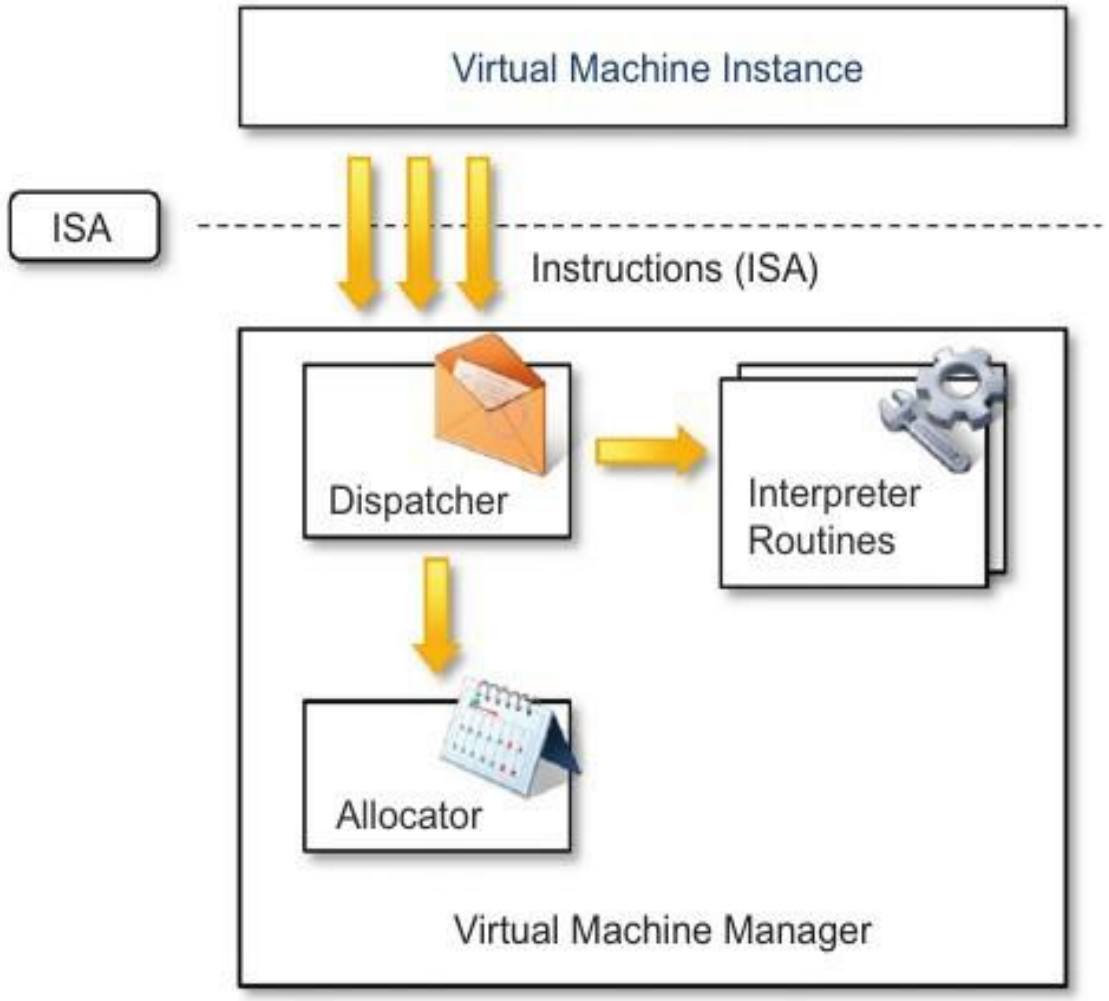
Advantages of Virtualization

- **PerformanceTuning**
  - control the performance of guest.
- **Virtual Machine Migration**
  - move virtual image into another machine.
- **Portability**
  - safely moved and executed on top of different virtual machine.

Virtual Machine Monitor/Manager (VMM)

- Main Modules :-
  - **Dispatcher**
    - Entry Point of VMM
    - Reroutes the instructions issued by VM instance.
  - **Allocator**
    - Deciding the system resources to be provided to the VM.
    - Invoked by dispatcher
  - **Interpreter**
    - Consists of interpreter routines
    - Executed whenever a VM executes a privileged instruction.
    - Trap is triggered and the corresponding routine is executed.

Virtual Machine Monitor/Manager (VMM)



Use of Virtualization Today

- Data centers:
  - server consolidation: pack multiple virtual servers onto a smaller number of physical server
    - saves hardware costs, power and cooling costs
- Cloud computing: rent virtual servers
  - cloud provider controls physical machines and mapping of virtual servers to physical hosts
  - User gets root access on virtual server
- Desktop computing:
  - Multi-platform software development
  - Testing machines
  - Run apps from another platform

References

- Clark et. al. Live migration of virtual machines NSDI 2005
- Post-copy migration Hines and Gopalan. Post-Copy Based Live Virtual Machine Migration Using Adaptive Pre-Paging and Dynamic Self-Ballooning. VEE 2009.
- Distributed Systems: Principles and Paradigms by Tanenbaum and van Steen, chapter 3.2