Distributed Systems
CS 15-440

Lecture 25, April 30, 2013

Gregory Kesden

Borrowed from our good friends in Doha:

Majd F. Sakr, Mohammad Hammoud andVinay Kolar

Objectives

Discussion on Virtualization

Why virtualization, and virtualization properties

Virtualization, para-virtualization, virtual machines and hypervisors

Virtual machine types

Partitioning and Multiprocessor virtualization

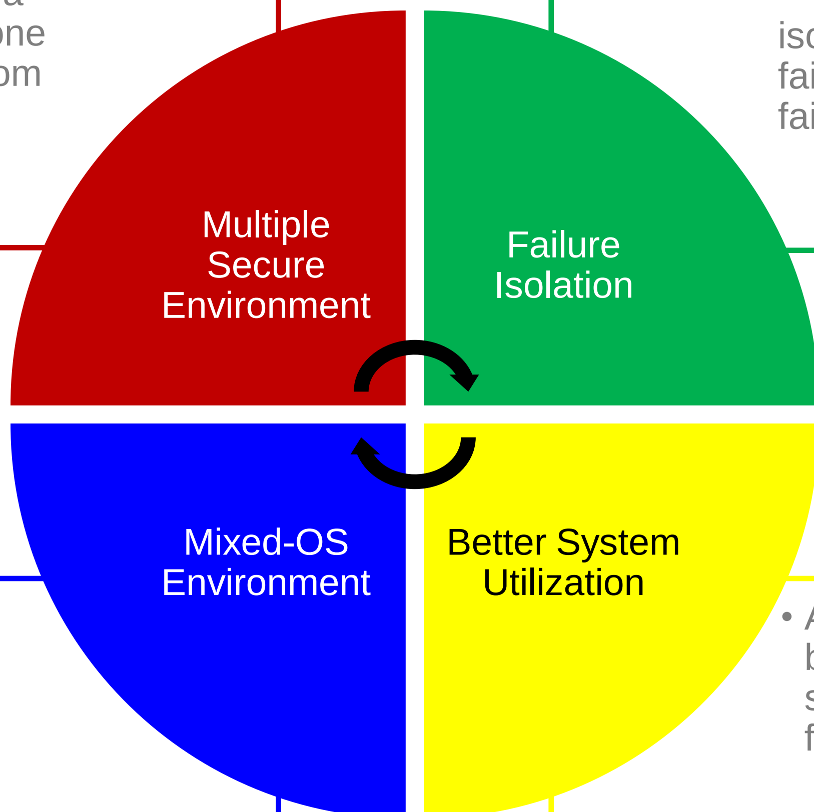
Resource virtualization

Benefits of Virtualization

- Here are *some* of the benefits that are typically provided by a virtualized system

- A system VM provides a sandbox that isolates one system environment from other environments

- Virtualization helps isolate the effects of a failure to the VM where the failure occurred



- A single hardware platform can support multiple operating systems concurrently

- A virtualized system can be (dynamically or statically) re-configured for changing needs

Operating Systems Limtations

- OSs provide a way of virtualizing hardware resources among *processes*

- This may help isolate *processes* from one another

- However, this does not provide a *virtual machine* to a user who may wish to run a different OS

- Having hardware resources managed by a single OS limits the flexibility of the system in terms of available software, security, and failure isolation

- Virtualization typically provides a way of relaxing constraints and increasing flexibility

Virtualization Properties

- Fault Isolation
- Software Isolation
- Performance Isolation (accomplished through scheduling and resource allocation)

Isolation

1

- All VM state can be captured into a file (i.e., you can operate on VM by operating on file– cp, rm)
- Complexity is proportional to virtual HW model and independent of guest software configuration

Encapsulation

2

- All guest actions go through the virtualizing software which can inspect, modify, and deny operations

Interposition

3

What is Virtualization?

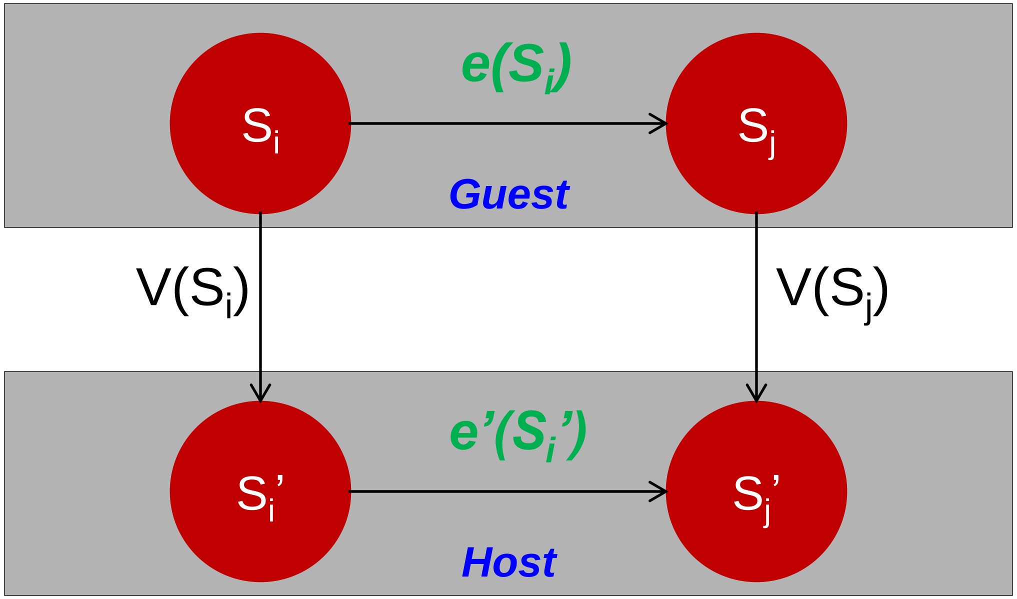
- Informally, a virtualized system (or subsystem) is a *mapping* of its interface, and all resources visible through that interface, to the interface and resources of a real system

- Formally, virtualization involves the construction of an isomorphism that *maps* a virtual *guest* system to a real *host* system (Popek and Goldberg 1974)

✓ Function V maps the guest state to the host state

✓ For a sequence of operations, **e**, that modifies a guest state, there is a corresponding **e'** in the host that performs an equivalent modification

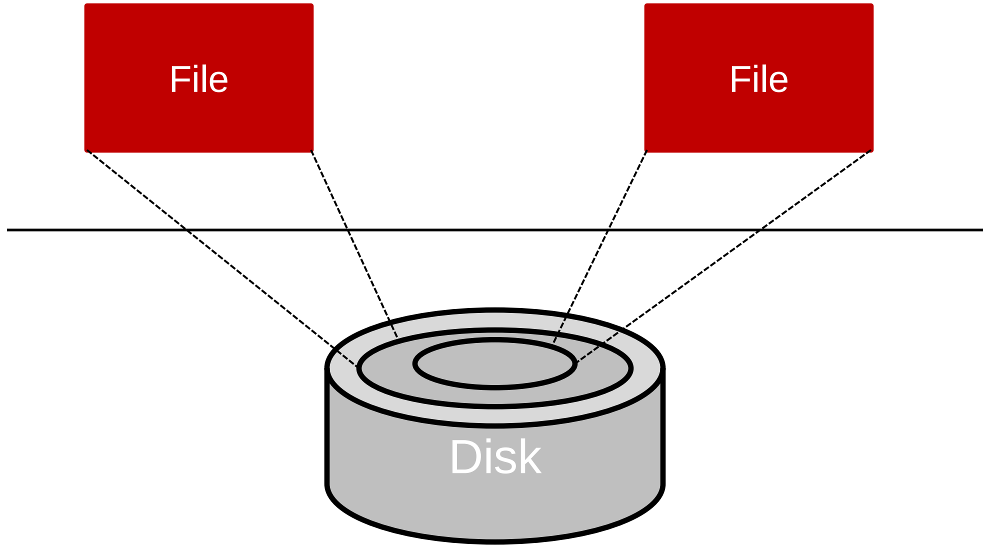
✓ How can this be managed?



Abstraction

- The key to managing complexity in computer systems is their division into *levels of abstraction* separated by *well-defined interfaces*

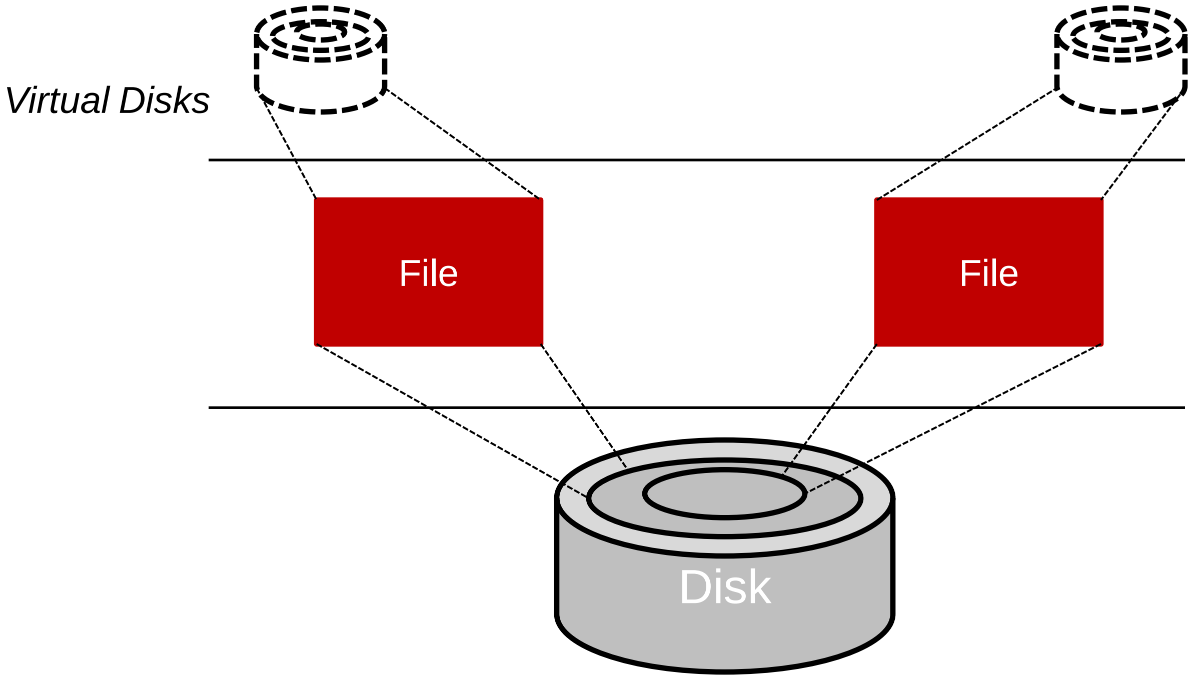
- Levels of abstraction allow implementation details at lower levels of a design to be ignored or simplified



✓ **Files are an abstraction of a Disk**
✓ **A level of abstraction provides a simplified interface to underlying resources**

Virtualization and Abstraction

- Virtualization uses abstraction but is different in that it doesn't necessarily hide details; the level of detail in a virtual system is often the same as that in the underlying real system



✓ *Virtualization provides a different interface and/or resources at the same level of abstraction*

Objectives

Discussion on Virtualization

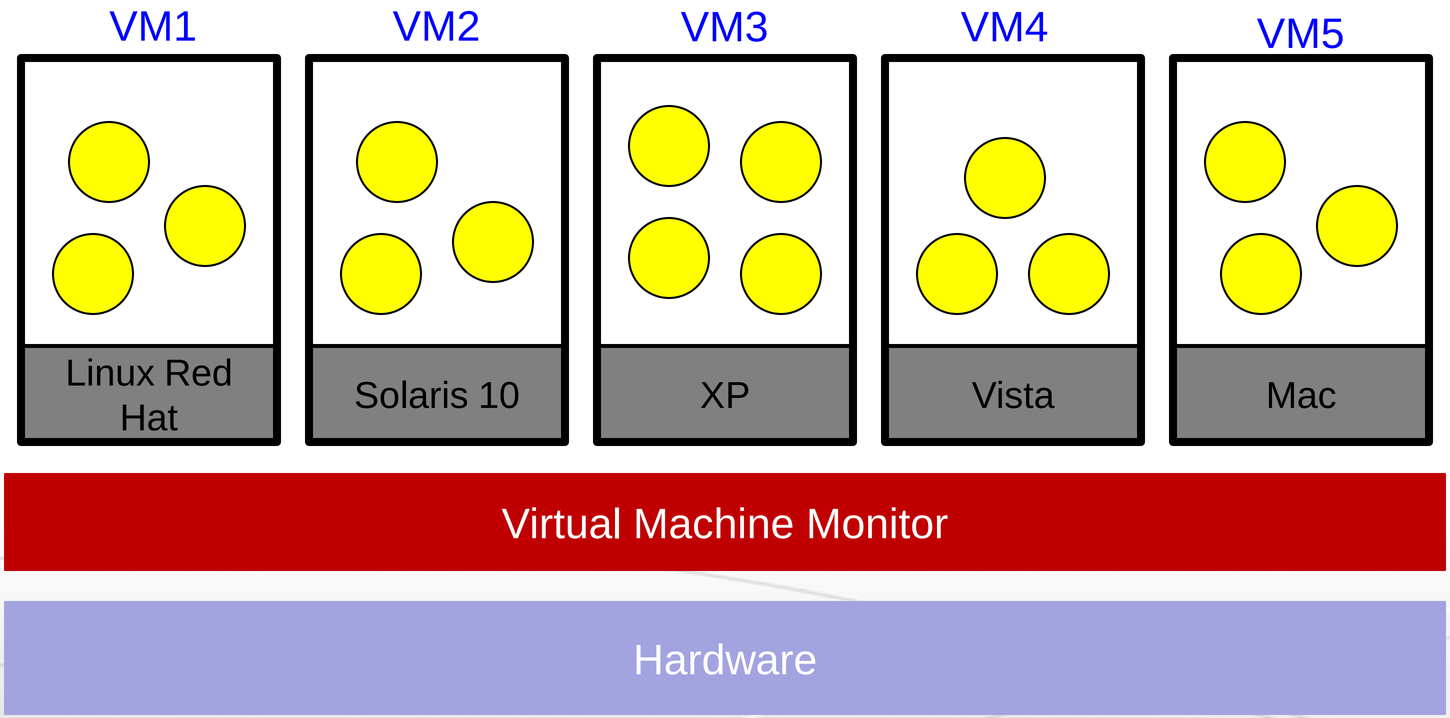
- Why virtualization, and virtualization properties
- Virtualization, para-virtualization, virtual machines and hypervisors
- Virtual machine types
- Partitioning and Multiprocessor virtualization
- Resource virtualization

Virtual Machines and Hypervisors

- The concept of virtualization can be applied not only to subsystems such as disks, but to an entire machine denoted as a virtual machine (VM)
- A VM is implemented by adding a *layer of software* to a real machine so as to support the desired VM's architecture
- This layer of software is often referred to as virtual machine monitor (VMM)
- Early VMMs are implemented in firmware
- Today, VMMs are often implemented as a co-designed firmware-software layer, referred to as the hypervisor

A Mixed OS Environment

- Multiple VMs can be implemented on a single hardware platform to provide individuals or user groups with their own OS environments



Full Virtualization

- Traditional VMMs provide full-virtualization:
  - The functionally provided is identical to the underlying physical hardware
  - The functionality is exposed to the VMs
  - They allow unmodified guest OSs to execute on the VMs
    - This might result in some performance degradation
  - E.g., *VMWare* provides full virtualization

Para-Virtualization

- Other types of VMMs provide para-virtualization:
  - They provide a virtual hardware abstraction that is *similar, but not identical* to the real hardware
  - They modify the guest OS to cooperate with the VMM
  - They result in lower overhead leading to better performance
  - E.g., *Xen* provides both para-virtualization as well as full-virtualization

Virtualization and Emulation

- VMs can employ *emulation techniques* to support cross-platform software compatibility
- Compatibility can be provided either at the system level (e.g., to run a Windows OS on Macintosh) or at the program or process level (e.g., to run Excel on a Sun Solaris/SPARC platform)
- Emulation is the process of implementing the interface and functionality of one system on a system having a different interface and functionality
- It can be argued that virtualization itself is simply a form of emulation

Objectives

Discussion on Virtualization

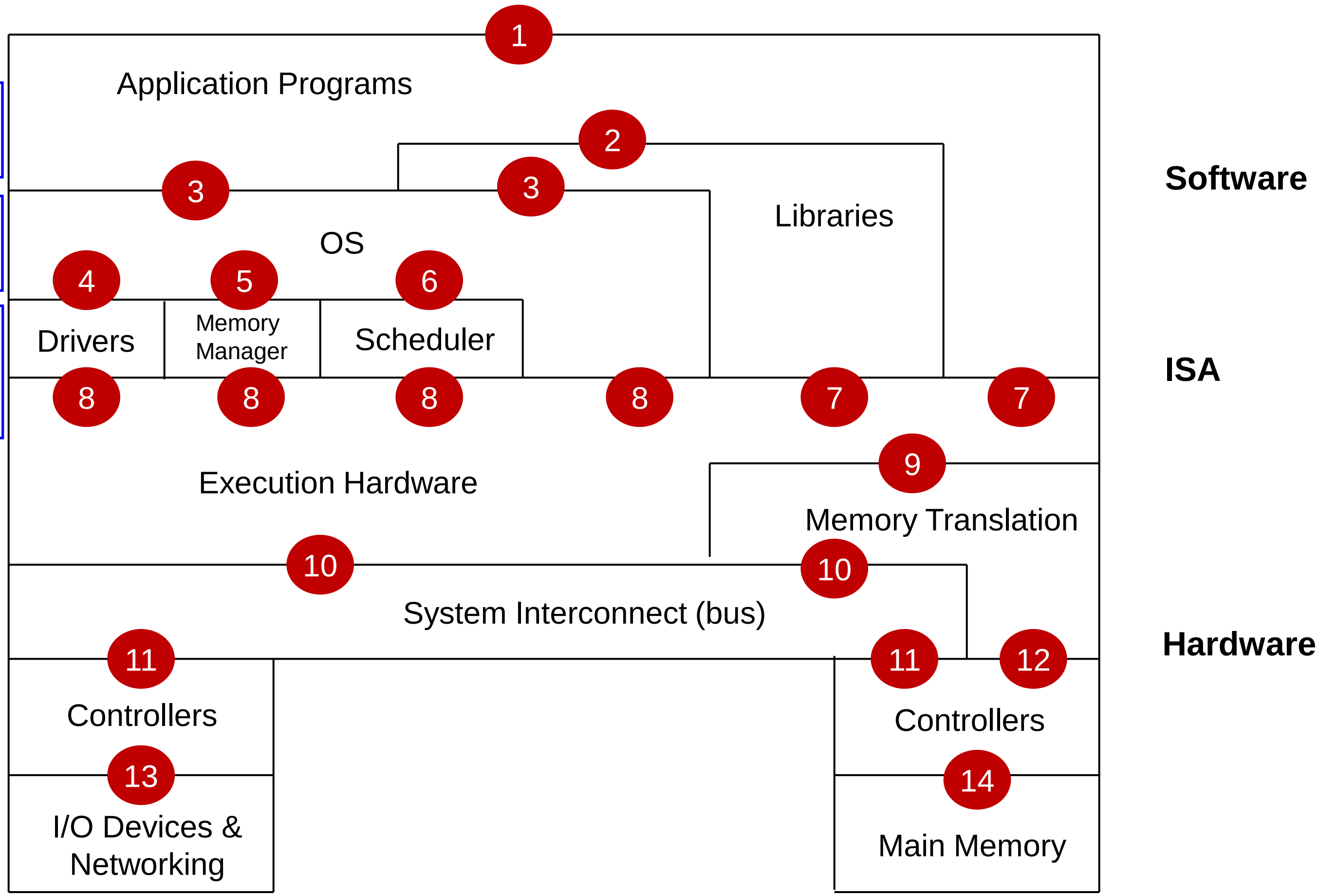
- Why virtualization, and virtualization properties
- Virtualization, para-virtualization, virtual machines and hypervisors
- Virtual machine types
- Partitioning and Multiprocessor virtualization
- Resource virtualization

Background: Computer System Architectures

Instruction Set Architecture (ISA): 7 & 8

Application Binary Interface (ABI): 3 & 7

Application Programming Interface (API): 2 & 7



Types of Virtual Machines

- As there is a process perspective and a system perspective of machines, there are also process-level and system-level VMs

- Virtual machines can be of two types:

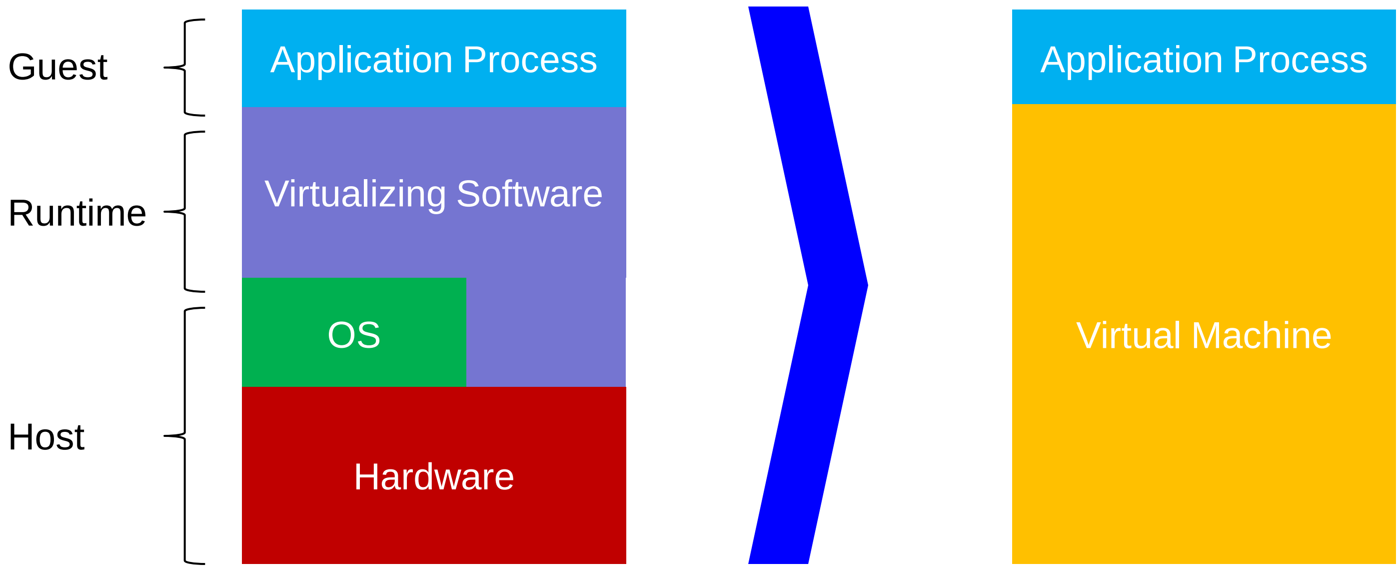
1. Process VM

- Capable of supporting an individual process

2. System VM

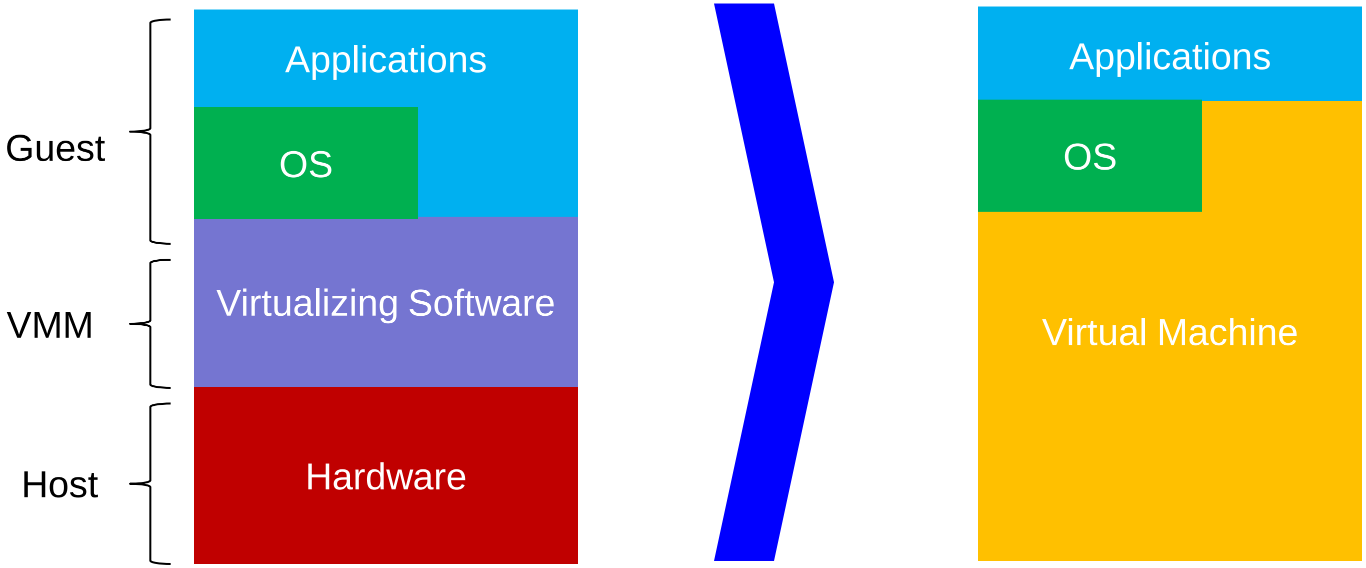
- Provides a complete system environment
- Supports an OS with potentially many types of processes

Process Virtual Machine



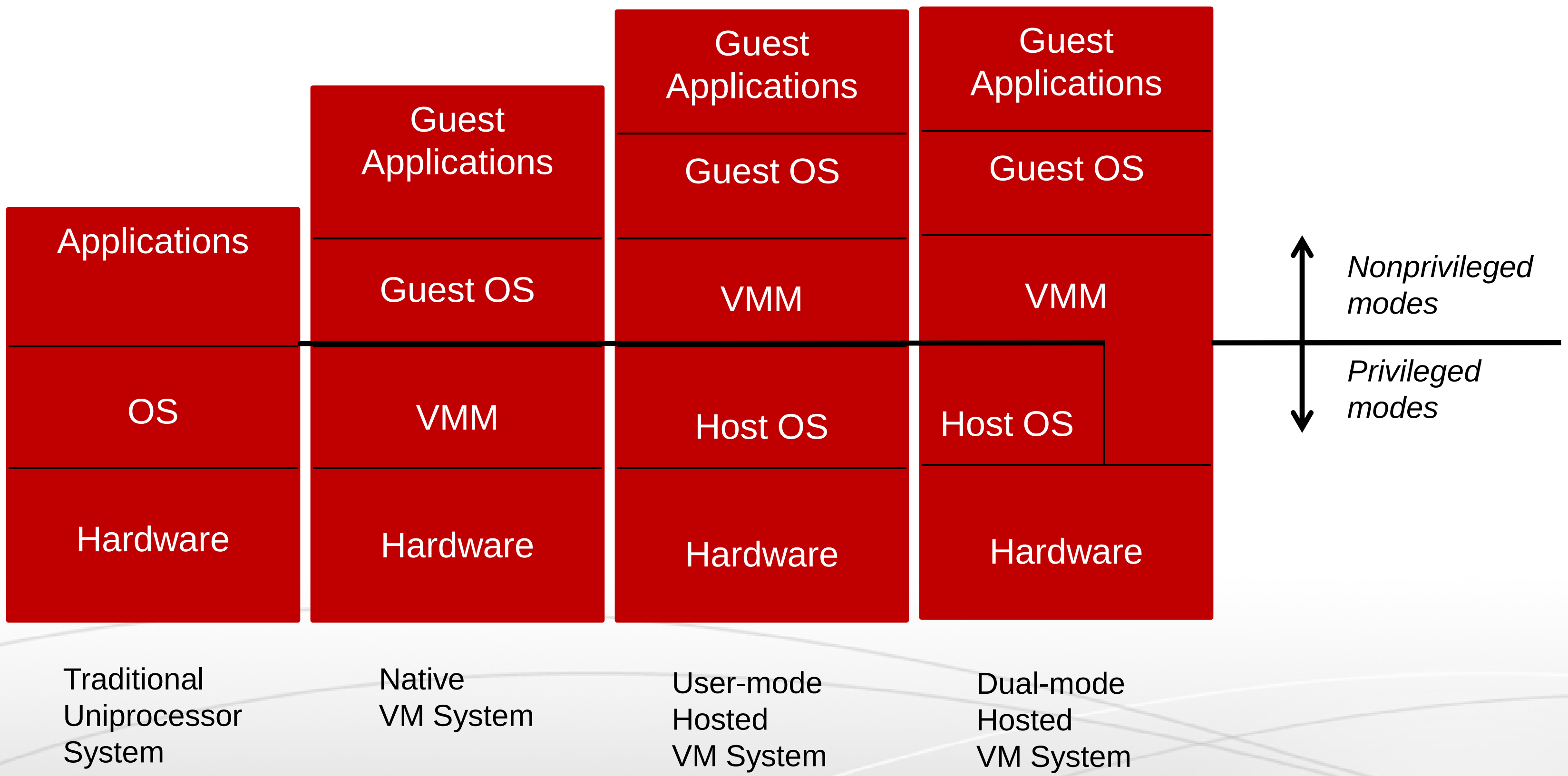
- ✓ Runtime is placed at the ABI interface
- ✓ Runtime emulates both user-level instructions and OS system calls

System Virtual Machine

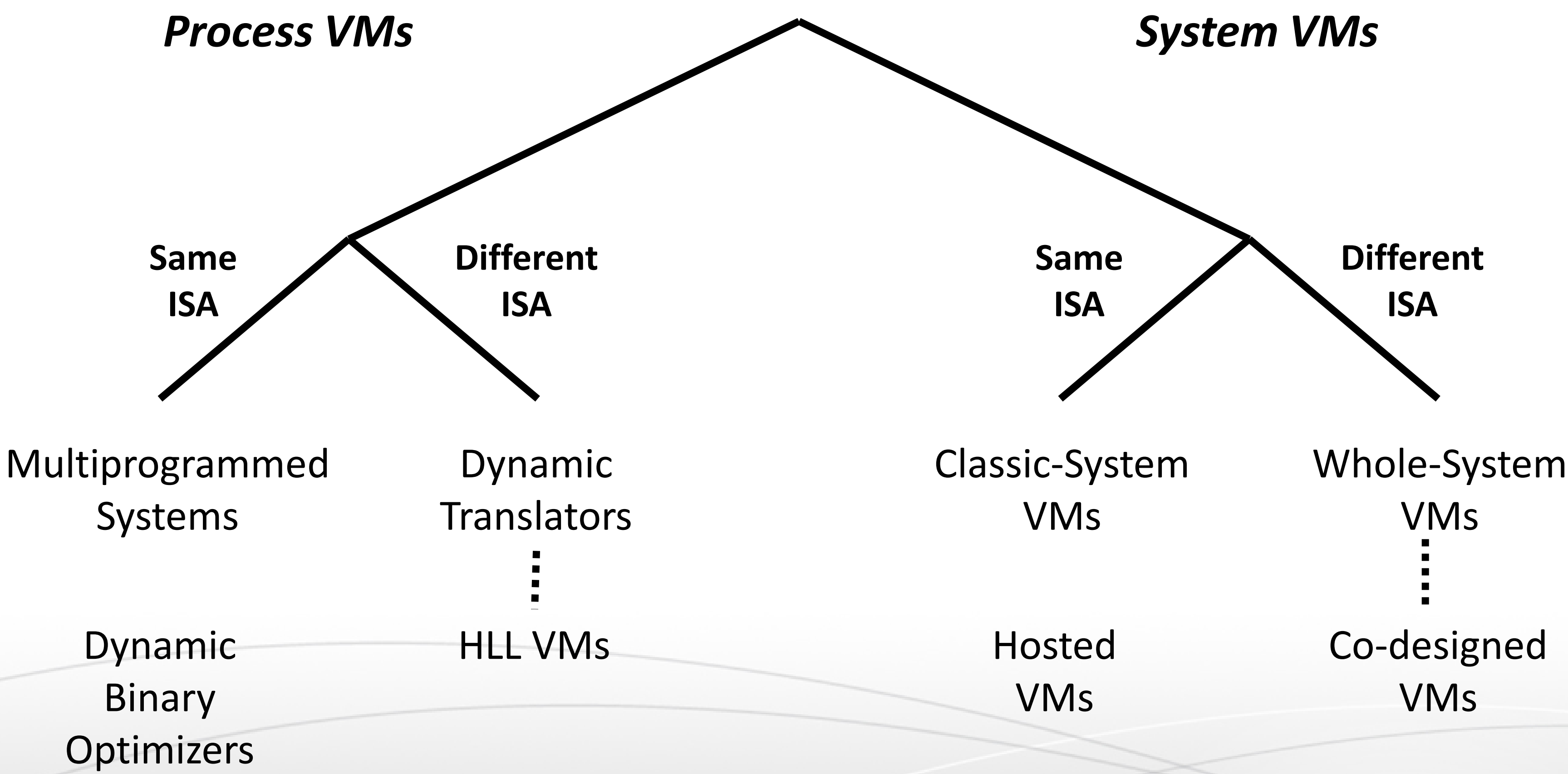


- ✓ VMM emulates the ISA used by one hardware platform to another, forming a system VM
- ✓ A system VM is capable of executing a system software environment developed for a different set of hardware

Native and Hosted VM Systems



A Taxonomy


Process VMs
System VMs
Same ISA
Different ISA
Same ISA
Different ISA
Multiprogrammed Systems
Dynamic Translators
Classic-System VMs
Whole-System VMs
Dynamic Binary Optimizers
HLL VMs
Hosted VMs
Co-designed VMs

The Versatility of VMs

Java Application

↓ *JVM*

Linux IA-32

↓ *VMWare*

Windows IA-32

↓ *Code Morphing*

Crusoe VLIW

Objectives

Discussion on Virtualization

- Why virtualization, and virtualization properties
- Virtualization, para-virtualization, virtual machines and hypervisors
- Virtual machine types
- Partitioning and Multiprocessor virtualization
- Resource virtualization

Multiprocessor Systems

- Multiprocessor systems might have 1000s of processors connected to TBs of memory and PBs of disk capacity

- Often there is a mismatch between the ideal number of processors an application needs and the actual number of physical processors available

- It is more often the case that applications cannot exploit more than a fraction of the processors available. The is mainly because of:

  - Limitations in the parallelism available in the programs

  - Limitations in the scalability of applications due to the overhead of communication between processors

Partitioning

- The increasing availability of multiprocessor systems has led to the examination of techniques that can help *utilize* them more effectively

- Techniques have been developed in which the multiprocessor system can be partitioned into multiple partitions

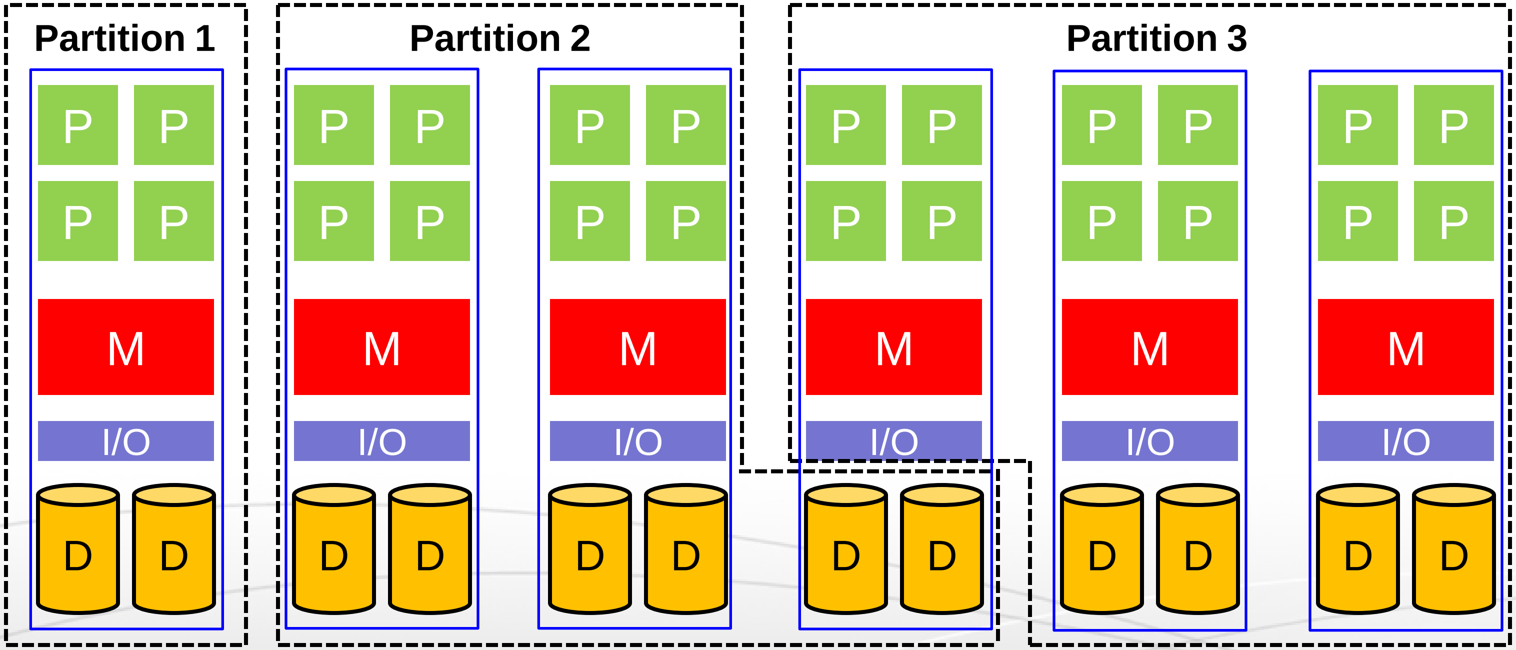
  - A partition is given a subset of the resources available on the system

- Hence, using partitioning, multiple applications can simultaneously exploit the available resources of the system

- Partitioning can be achieved:
  - Either in-space (referred to as physical partitioning)
  - Or in-time (referred to as logical partitioning)

Physical Partitioning

- With physical partitioning, each partition is assigned resources that are physically distinct from the resources used by the other partitions



Physical Partitioning

- Physical partitioning allows a partition to *own* its resources physically

- It is not permissible for two partitions to share the resources of a single system board

- Partitions are configured by a *central control unit* that receives commands from the console of the system admin and provisions hardware resources accordingly

- The number of partitions that can be supported in physically partitioned systems is limited to the number of available physical processors

Physical Partitioning- Advantages

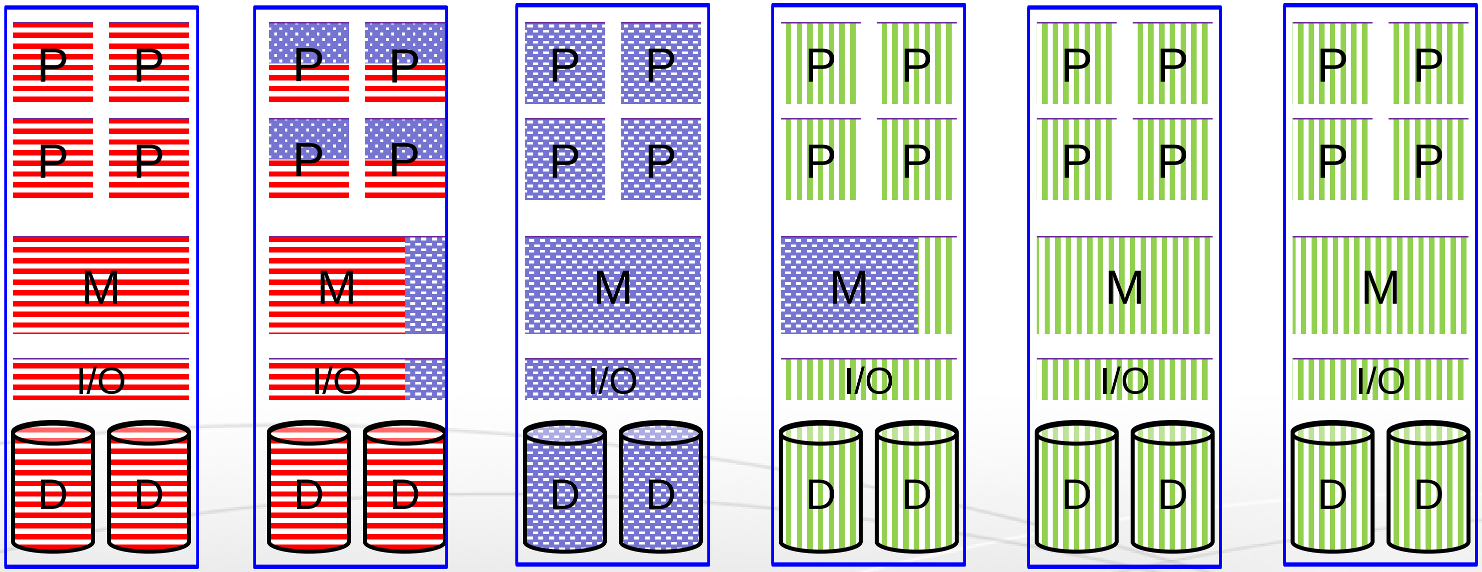
- Physical partitioning provides:
  - Failure Isolation: it ensures that in the event of a failure, only the part of the physical system that houses the failing partition will be affected
  - Better security isolation: Each partition is protected from the possibility of intentional or unintentional denial-of-service attacks by other partitions
  - Better ability to meet system-level objectives (these result from contracts between system owners and users of the system)
  - Easier management of resources: no need of sophisticated algorithms for scheduling and management of resources

Physical Partitioning- Disadvantages

- While physical partitioning has a number of attractive features, it has some major disadvantages:
  - System utilization: Physical partitioning is probably not the ideal solution if system utilization is to be optimized
    - It is often the case that each of the physical partitions is underutilized
  - Load balancing: with physical partitioning, dynamic workload balancing becomes difficult to implement

Logical Partitioning

- With logical partitioning, partitions share some of the physical resources, usually in a *time-multiplexed* manner



Logical Partitioning

- With logical partitioning it is permissible for two partitions to share the resources of a single system board

- Logical partitioning makes it possible to partition an ***n-way*** system into a system with more than ***n*** partitions, if so desired

- Logical partitioning is more flexible than physical partitioning but needs additional mechanisms to provide safe and efficient way of sharing resources

- Logical partitioning is usually done through a VMM or a hypervisor and provides what is referred to as *multiprocessor virtualization*

Multiprocessor Virtualization

- A virtualized multiprocessor gives the appearance of a system that may or may not reflect the exact configuration of the underlying physical system