Virtualization

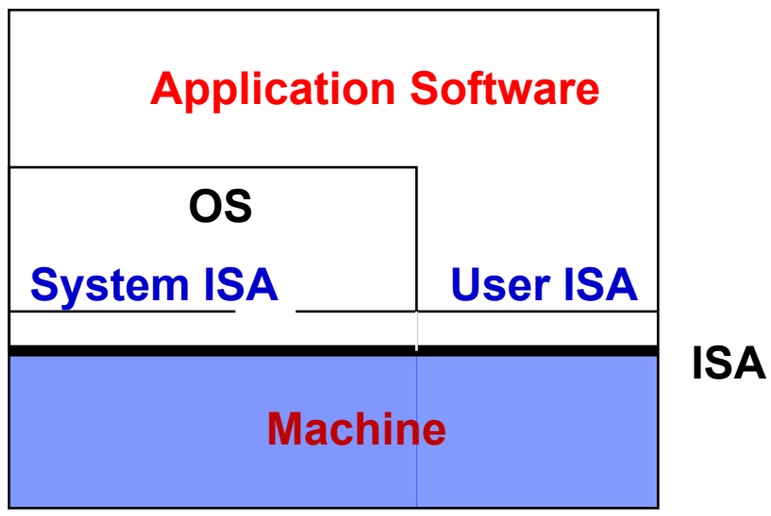
What is Virtualization?

"Virtualization is the simulation of the software and/or hardware upon which other software runs. This simulated environment is called a virtual machine"

--Wikipedia

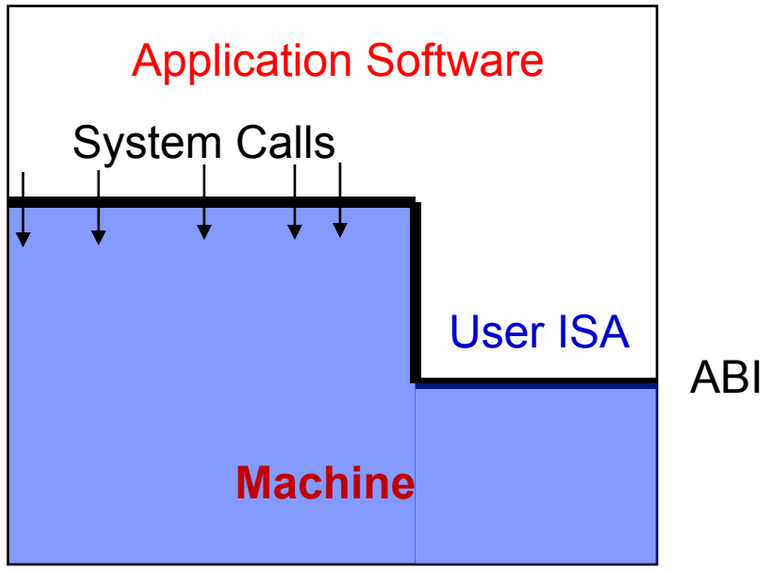
Computer Systems Arch.

- Instruction set arch. (ISA), introduced in IBM 360 series in early 60's, provides an interface between HW and SW, so that HW could be implemented in various ways
- OS provides a first layer of abstraction, that hides specifics of the HW from programs.
  - data types, instructions, registers
  - addressing mode, mem hierarchy
  - interrupt, I/O handling



Application Binary Interface

- From the perspective of a user process, the machine is a combination of the OS and the underlying user-level HW, defined by the ABI interface



Application
Binary Interface

Virtual Machine

- Mapping of virtual resources or state (e.g. registers, memory, files, etc) to real resources
- User of real machine instructions and/or system calls to carry out the actions specified by VM instructions and/or system calls (e.g. emulation of the VM ABI or ISA)
- Two types of VM
  - Process VM from the perspective of user process
  - System VM from the perspective of OS

Process Virtual Machine

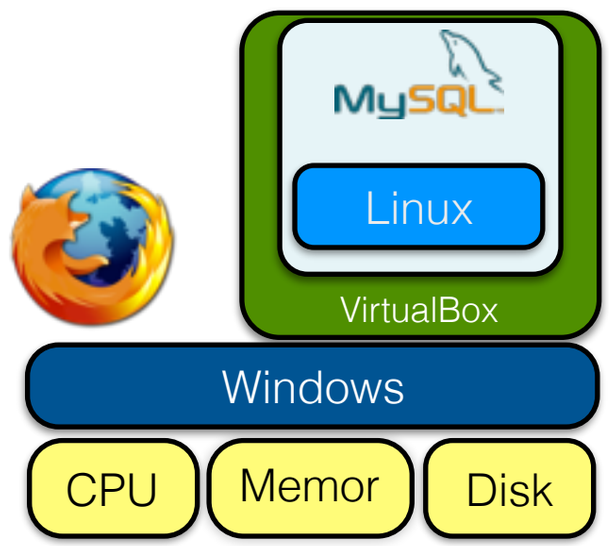
- Process-level (application) VMs provide user apps with a virtual ABI environment
- Types of process-level VMs
  - Multiprogramming
  - Emulators and Dynamic Binary Translators
  - Same-ISA Binary Optimizers
  - High-Level Language Virtual Machines (Platform Independence)
    - JVM

System Virtual Machine

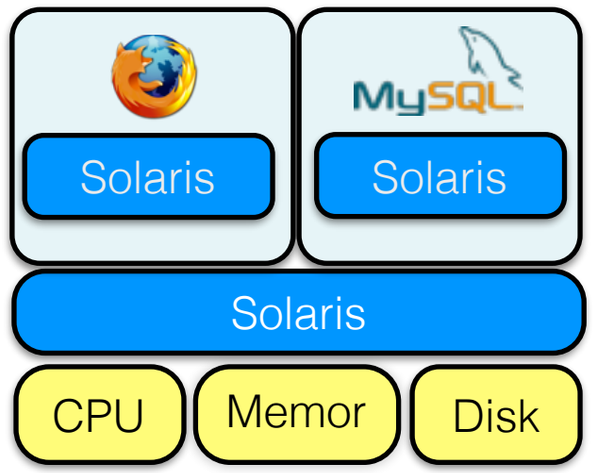
- provides a complete system platform which supports the execution of a complete operating system (OS)
  - supports multiple user processes
  - provides them with access to I/O devices
  - supports GUI if on the desktop

Types of System VM

- Hosted virtualization
  - simulates a OS in a process
  - VirtualBox, VMware player



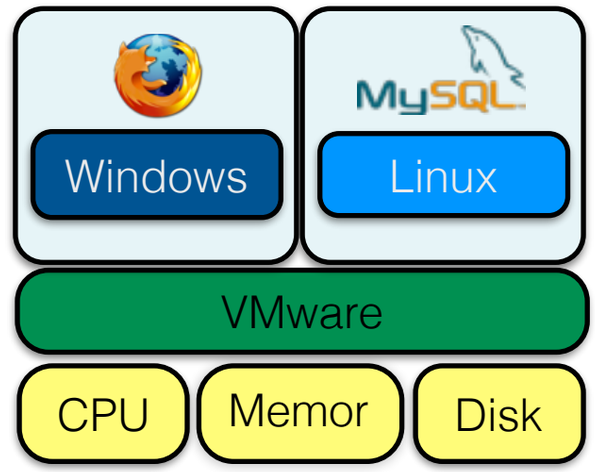
- OS-level virtualization
  - divides host OS into partitions
  - guest OS is the same as the host OS
  - Solaris containers, OpenVZ, Linux Vserver

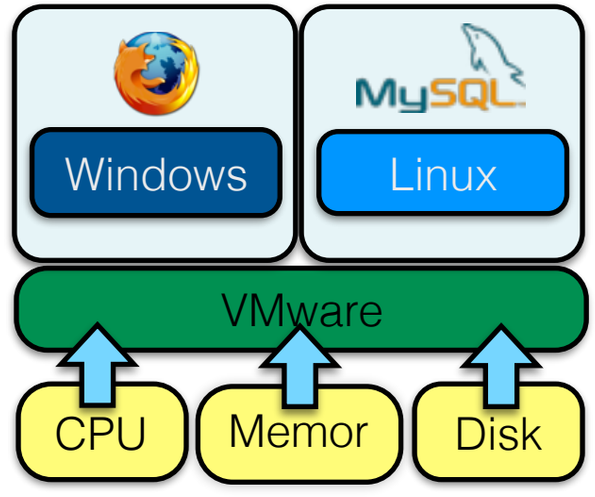


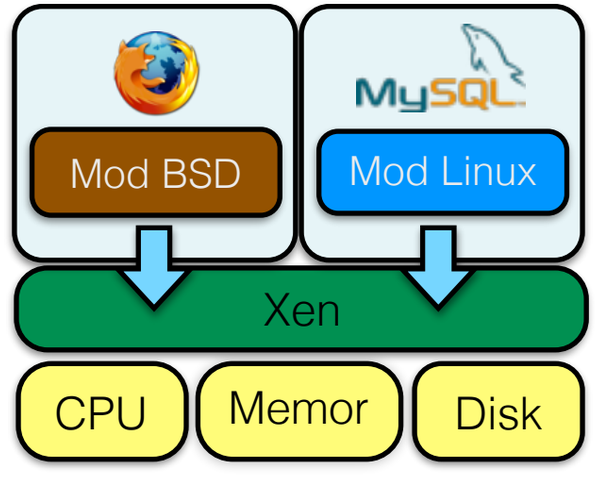
VM

Types of Virtualization (cont')

- Hardware (platform) virtualization
  - Full virtualization
    - unmodified OS runs in emulated hardware
    - IBM VM series, Parallel
  - Hardware-assisted virtualization (HV)
    - HW provides architectural support hosting VMs
  - Para-virtualization (PV)
    - modified OS runs in VM
    - Xen, VMware ESXi
  - PVHVM

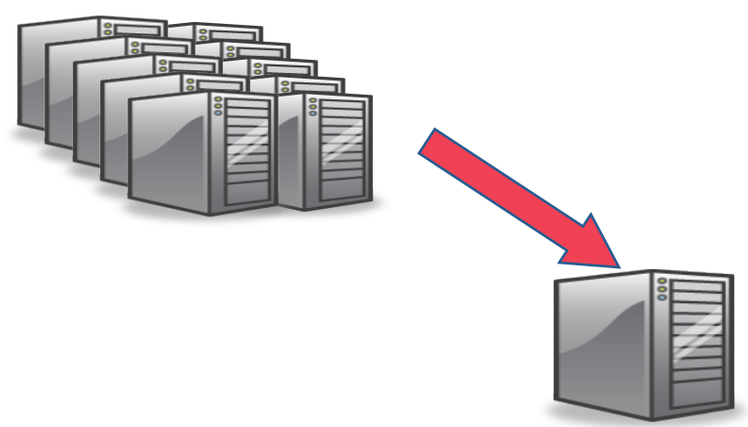






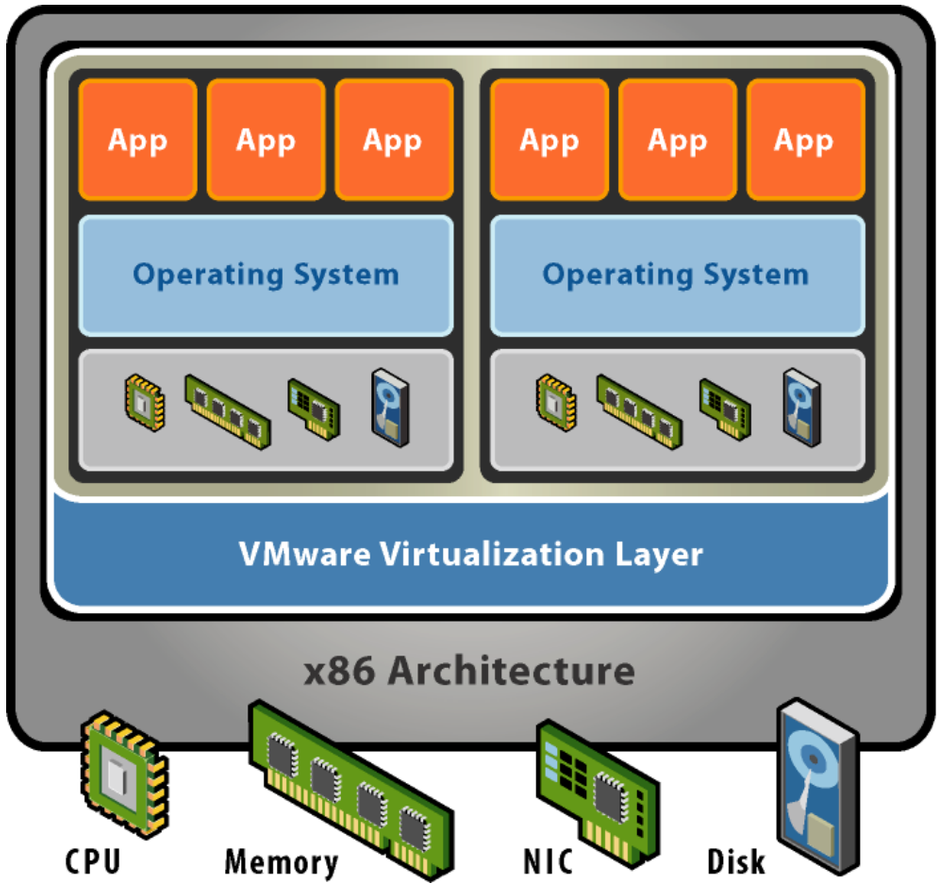
System VM: Why?

- Reduce total cost of ownership (TCO)
  - Increased systems utilization (current servers have less than 10% average utilization, less than 50% peak utilization)
  - Reduce hardware (25% of the TCO)
  - Space, electricity, cooling (50% of the operating cost of a data center)



Resource Virtualization

- Processor
- Memory
- Device and I/O

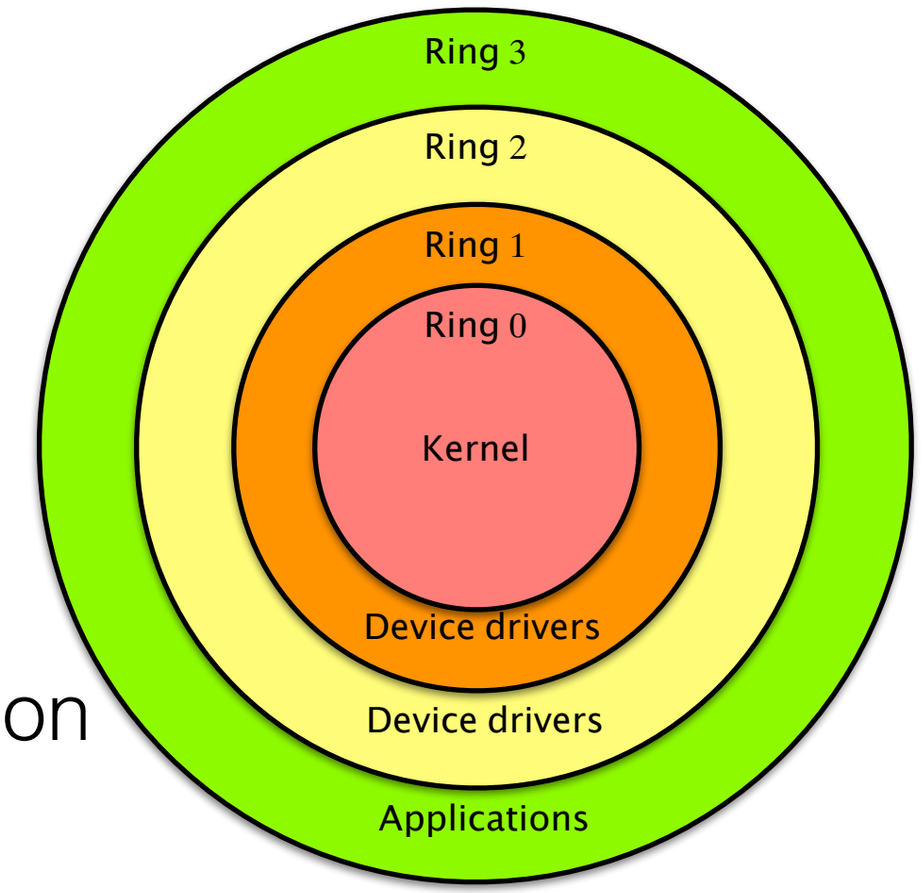


Popek and Goldberg Virtualization Requirements (1974)

- Fidelity
  - A program running under the VMM should exhibit a behavior essentially identical to that demonstrated when running on an equivalent machine directly
- Safety
  - The VMM must be in complete control of the virtualized resources
- Performance
  - A statistically dominant fraction of machine instructions must be executed without VMM intervention

CPU Rings

- User and kernel mode are controlled by CPU
- Multiple CPU protection rings
  - traditional OS runs in ring 0
  - OS in VM runs in ring 1-3
  - must handle ring 3 to ring 0 transition



Sufficient Conditions for Virtualization

- Classification of Instructions:
  - Privileged instruction traps if the machine is in user mode and does not trap if in system mode
  - Control-sensitive instructions attempt to change the configuration of resources in the system
  - Behavior-sensitive instructions: results produced depend on the configuration of resources
- A VMM may be constructed if the set of sensitive instructions is a subset of the privileged instructions
  - Intuitively, it is sufficient that all instructions that could affect the correct functioning of the VMM (sensitive instructions) always trap and pass control to the VMM.

Challenges for X86 Virtualization

- IA-32 contains 16 sensitive, but non-privileged instructions

  - Sensitive register instructions: read or change sensitive registers and/or memory locations such as a clock register or interrupt registers:

    - SGDT, SIDT, SLDT, SMSW, PUSHF, POPF

  - Protection system instructions: reference the storage protection system, memory or address relocation system:

    - LAR, LSL, VERR, VERW, POP, PUSH, CALL, JMP, INT n, RET, STR, MOV

Binary Translation

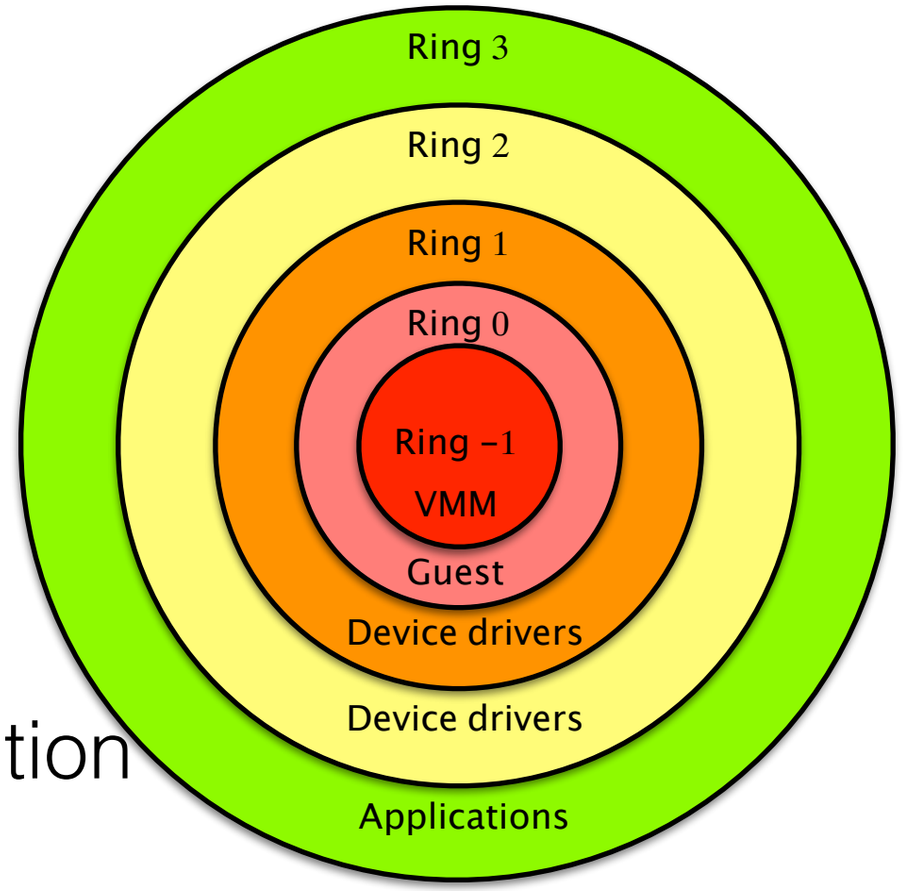
- dynamic translate native binary code into host instructions
  - preprocess OS binary running in VM
  - detect sensitive instructions
  - call out to the VMM

Para-virtualize Privileged Instructions

- Execution of privileged instructions requires validation in the VMM
  - modify OS to exit into VMM for validation and execution
  - Hypercalls in Xen
  - Optimizations
    - batching
    - validation at initialization

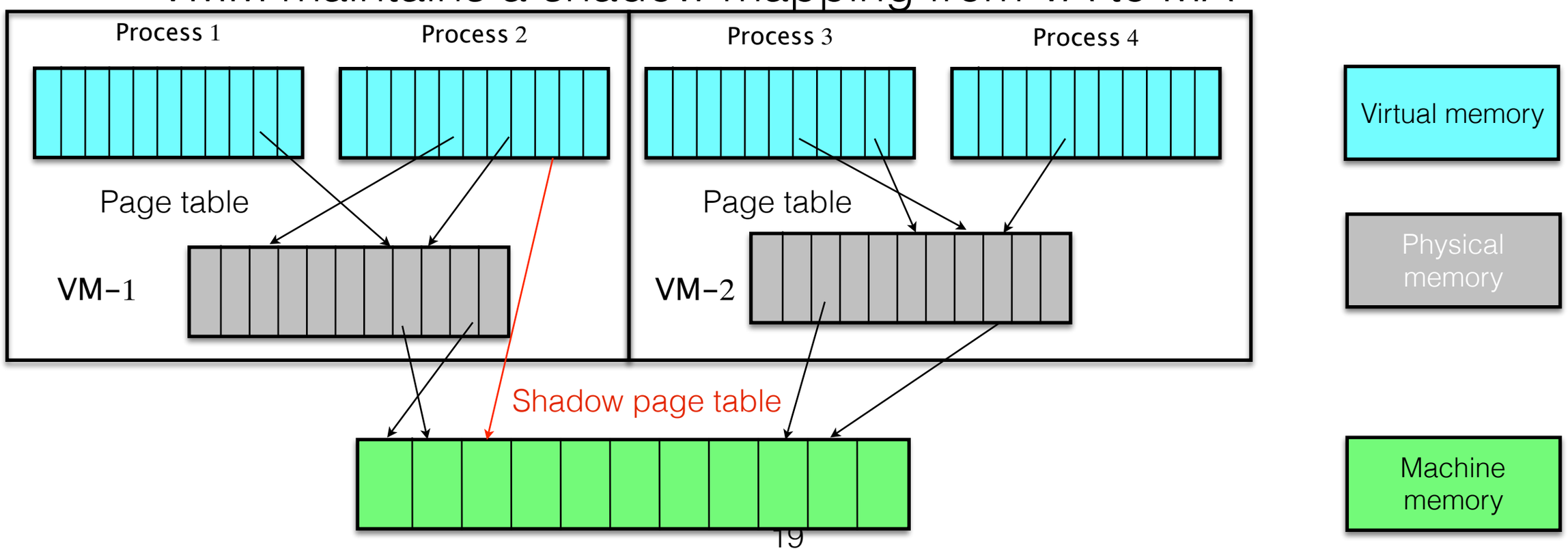
Hardware-Assisted CPU Virtualization

- CPU hardware support for virtualization
  - Intel VT and AMD-V
  - Hypervisor runs in ring -1 (root)
  - Guest OS runs in ring 0 (non-root)
  - New instructions for VM/VMM transition
    - VM exit and VM entry



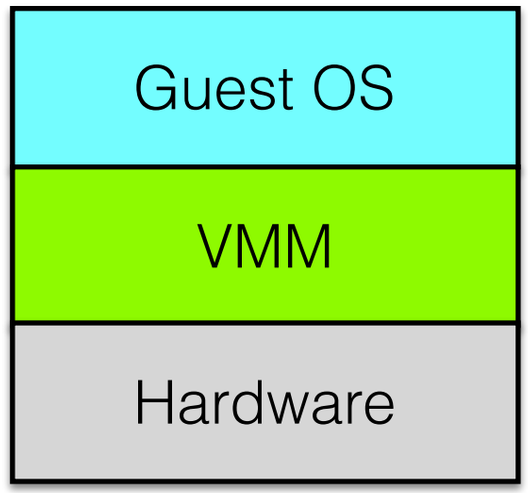
Virtualizing Memory

- Three memory addresses
  - virtual memory (process), physical memory (OS), machine memory (VMM)
  - VMM maintains a shadow mapping from VA to MA



Virtualizing Memory (cont')

- High virtualization overhead with shadow page table
  - frequent guest OS to VMM transition and TLB flush
  - Xen's optimization
    - directly register guest PG to MMU
    - Read to PG bypass VMM
    - VMM traps updates to VMM
    - Batch updates
    - Reserve top 64MB for VMM to avoid TLB flush due to guest/VMM switch



0	123
2	239
5	100

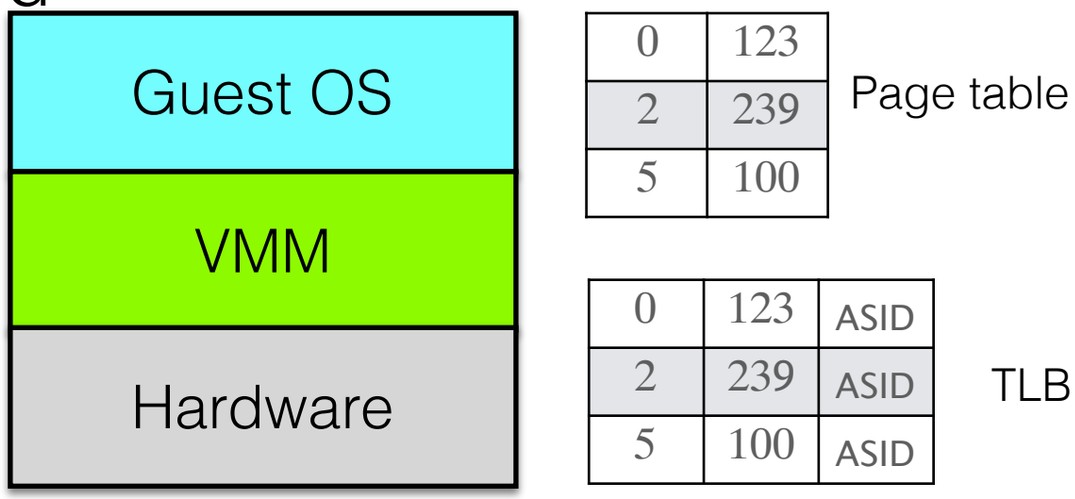
Page table

0	250
2	453
5	23

Shadow page table

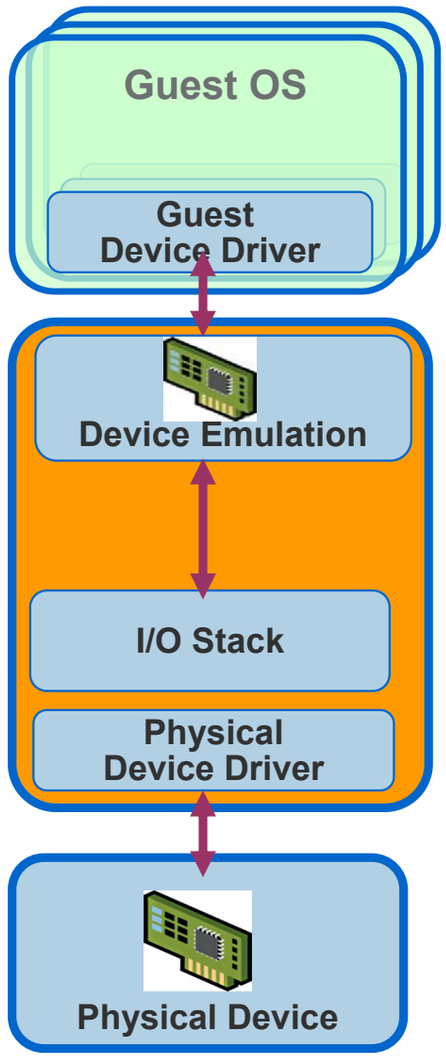
Hardware Support

- Extended/Nested page tables
  - Intel VT-x and AMD-V
  - no shadow page table is needed
  - Two hardware PGs
    - VA->PA and PA->MA
  - Tagged TLB entry
  - costly page walk



Virtualizing I/O

- I/O virtualization architecture
  - guest driver
  - generic virtual device, e.g., Intel e1000
  - virtualization I/O stack
  - real device driver
  - hardware device

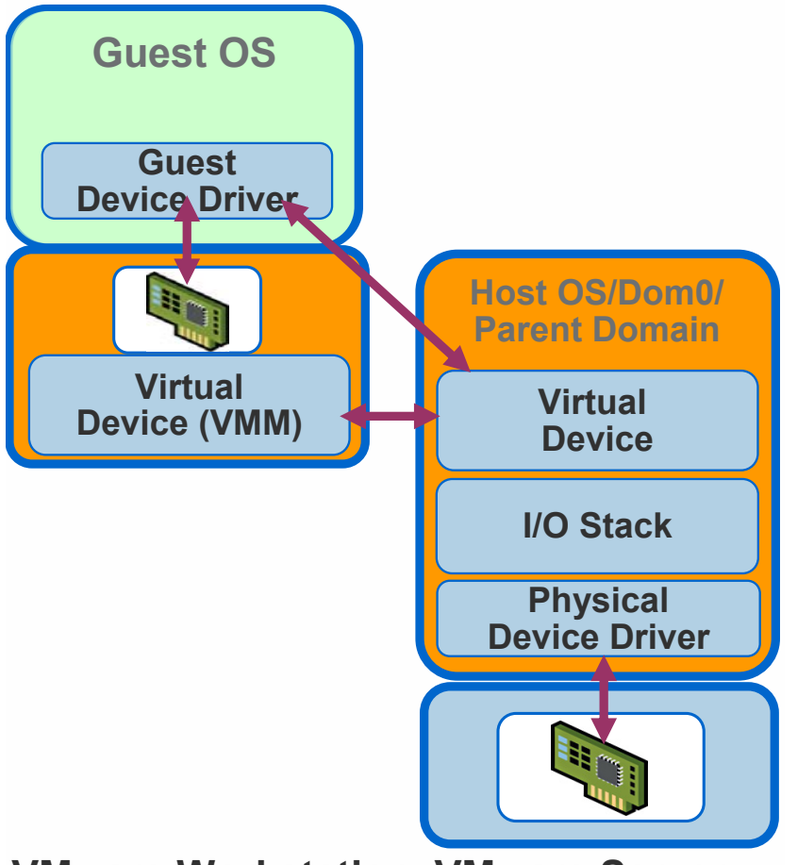


*Adapted from Mallik's presentation at VMworld 2006

I/O Virtualization Implementations

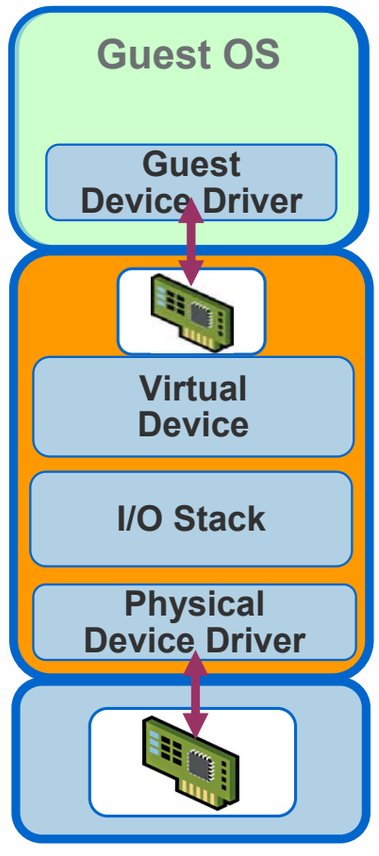
Virtualized I/O

Hosted or Split



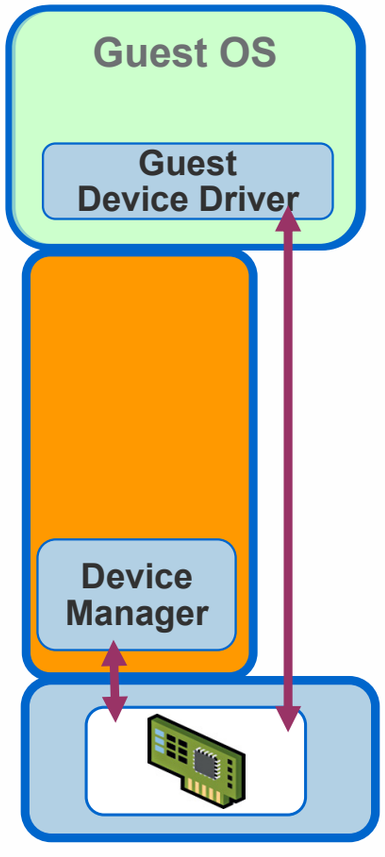
VMware Workstation, VMware Server, VMware ESX Server
Microsoft Viridian & Virtual Server, Xen

Hypervisor Direct



VMware ESX Server (storage and network)

Passthrough I/O

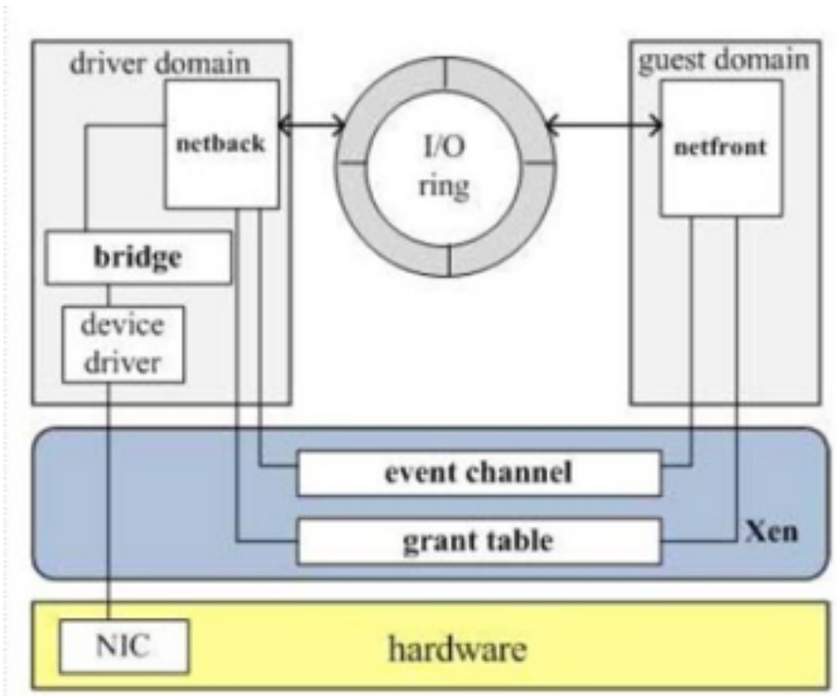


A Future Option

*Adapted from Mallik's presentation at VMworld 2006

Xen's I/O Structure (split)

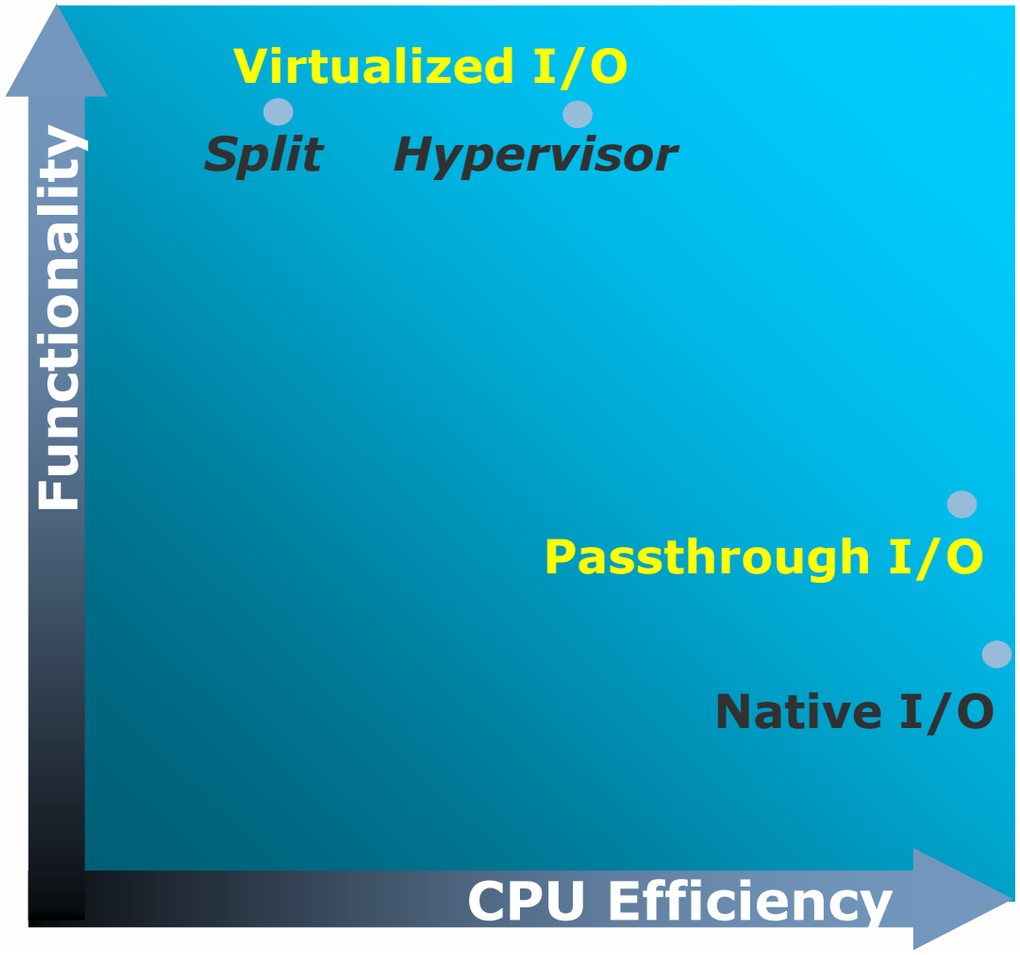
- Event-channel for inter-domain communication and interrupt handling
- I/O ring buffer for submitting request and retrieving responses
- Grant table for DMA access



I/O Channel
-I/O ring
-Event channel
-Grant table

Trade-offs

- Virtualized I/O provides rich functionality
- Passthrough I/O reduces CPU utilization and better performance



*Adapted from Mallik's presentation at VMworld 2006

Passthrough I/O

- Guest uses I/O device directly
  - suitable for I/O appliance and high performance VMs
  - requires hardware support
    - IO MMU for DMA address translation and protection (Intel VT-d)
    - Partitionable I/O devices (PCI-SG IOV SR/MR)
      - physical functions (PF) and virtual functions (VF)

*Adapted from Mallik's presentation at VMworld 2006