CS 5264/4224; ECE 5414/4414
(Advanced) Linux Kernel Programming
Lecture 22

Virtualization

May 6, 2025

Huaicheng Li

https://people.cs.vt.edu/huaicheng/lkp-sp25/

Acknowledgement: Credits to Dr. Changwoo Min for the original LKP lecture slides.

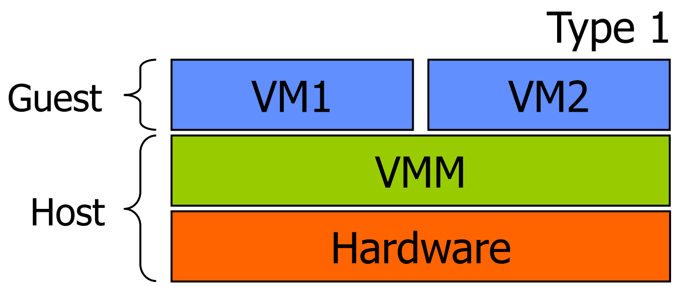
Outline

- Virtualization Overview
- Memory Management
- I/O Virtualization
- Wrap-up

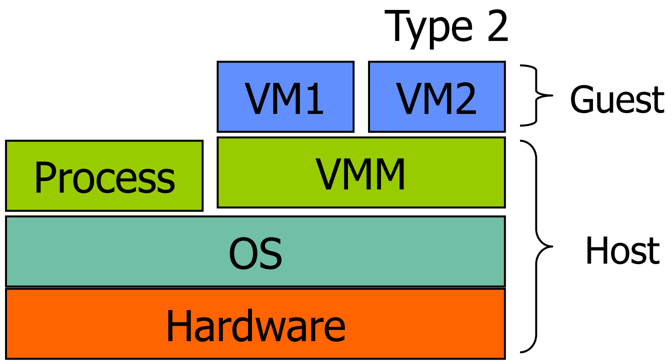
Introduction

- What is virtualization?
  - Virtualization is a way to run multiple operating systems and user applications on the same hardware
  - Virtual Machine Monitor (Hypervisor) is a software layer that allows several virtual machines to run on a physical machine

- Types of VMMs
  - Type-1: hypervisor runs directly on hardware
  - Type-2: hypervisor runs on a host OS



Xen, VMware ESX, Microsoft Hyper-V

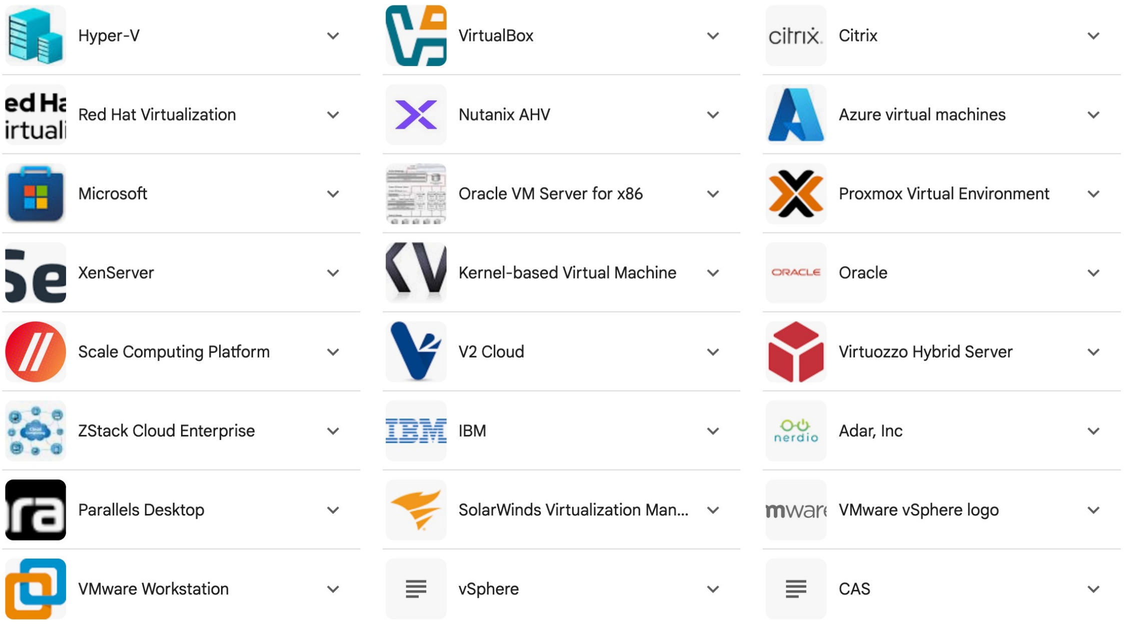


KVM, VMware Workstation, Sun VirtualBox, QEMU

- *Virtual Machine Monitor (VMM) = Hypervisor = Host OS*
- *Virtual Machine (VM) = Guest OS*

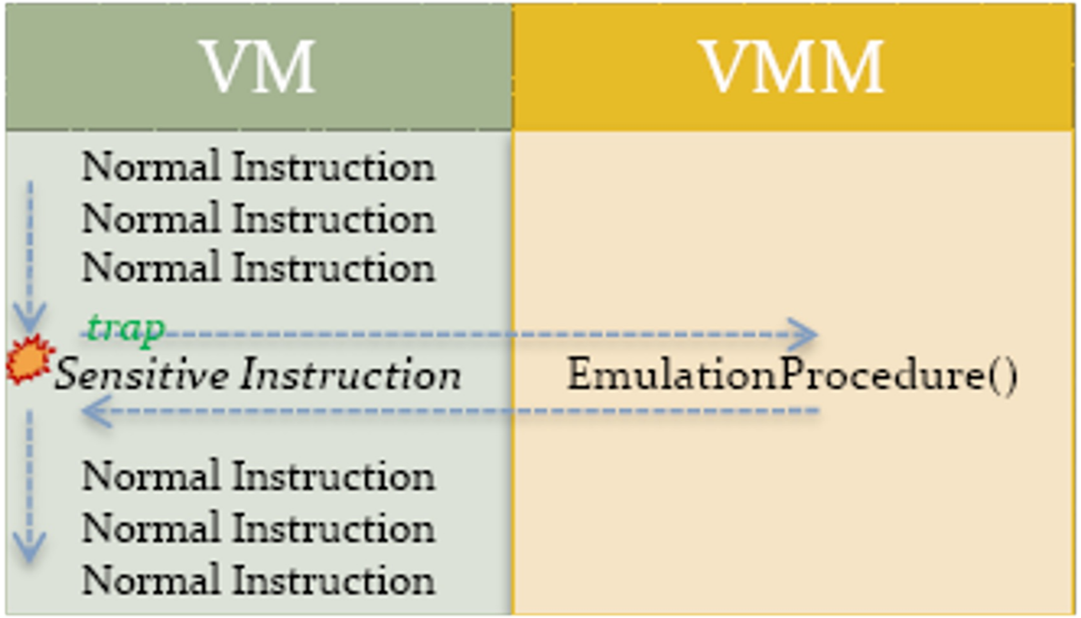
- Advantages
  - Isolation
    - Limits security exposures
    - Reduces spread of risks
  - Roll-Back
    - Quickly recovers from security breaches
  - Abstraction
    - Limits direct access to hardware
  - Portability
    - Disaster recovery
    - Switches to “standby” VMs
  - Deployment
    - Distributes workloads
    - Customizes guest OS security settings

- Applications
  - Server virtualization
    - Green IT
    - Xen, VMware ESX Server
  - Desktop virtualization
    - VMware, VirtualBox, Citrix's Xen HDX
  - Mobile virtualization
    - Secure execution
    - Xen on ARM
  - Cloud computing
    - Storage/platform cloud services
    - Amazon EC2, MS Azure, Google AppEngine
    - Serverless
  - Emulation
    - iPhone/Android emulator
    - Qemu, Bochs

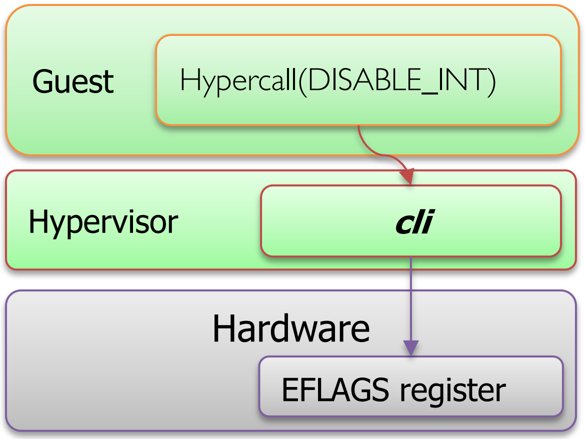


Processor Virtualization

- Classic virtualization
  - Trap and emulate: For an architecture to be virtualizable, all sensitive instructions must be handled by VMM
  - **Sensitive instructions** include
    - Instruction that changes processor mode
    - Instruction that accesses hardware directly
    - Instruction whose behavior is different in user/kernel mode



- Para-virtualization
  - Requires modifications to the guest OS
    - Guest is aware that it is running on a VM
  - Example
    - Instead of doing “cli” (turn off interrupts), guest OS should do hypercall(DISABLE_INT)
  - Pros
    - Near-native performance
    - No hardware support required
  - Cons
    - Requires specifically modified guest
  - Solutions : Xen



- Full-virtualization (**Emulation**)
  - Process of implementing the interface and functionality of one system on a different system
  - Do whatever the CPU does, but in software
    - CPU emulation
      - Fetches and decodes the next instruction
      - Executes using the emulated registers and memory

Pros

- No hardware support required
- Simple

Cons

- Very slow

- Solutions : Bochs

```
addl %ebx, %eax
```

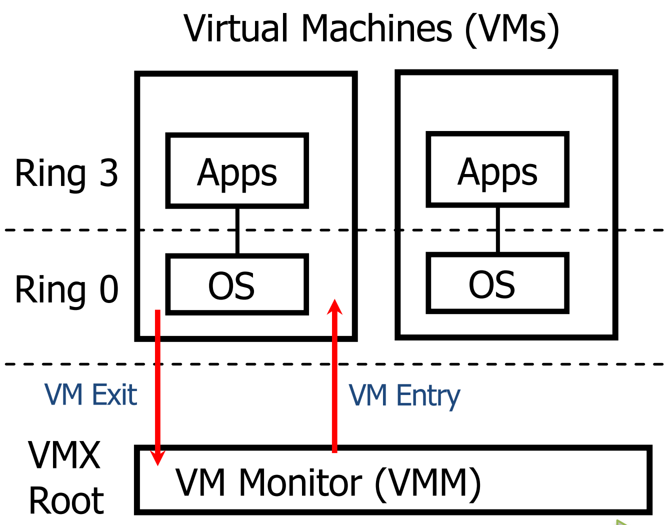
```
unsigned long regs[8];
regs[EAX] +=
regs[EBX];
```

Processor Virtualization – Full-Virtualization

- Binary translation
  - Translates code block to safe code block (like JIT) directly
  - Dynamically translates privileged instructions to normal instructions which can be executed in user mode
  - Pros
    - » No hardware support required
    - » Fast
  - Cons
    - » Hard to implement
    - » VMM needs x86-to-x86 binary compiler
  - Solutions: VMware, QEMU

JIT: Just in time compilation

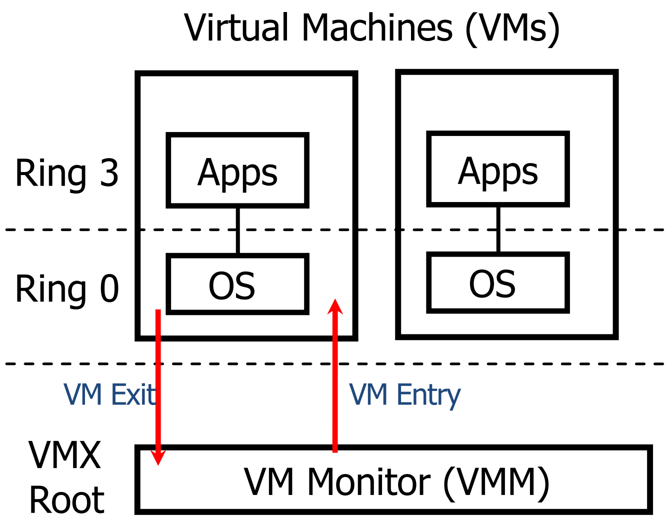
- Full-virtualization
  - Hardware-assisted virtualization
    - Runs the VM directly on the CPU
      - No emulation
    - Integrates new execution mode into the CPU by extending the instruction set and control structure
    - Pros
      - Fast
    - Cons
      - Need hardware support
        - » AMD SVM
        - » Intel VT
    - Solutions : KVM, Xen



AMD SVM : Secure Virtual Machine
Intel VT : Virtualization Technology

Intel VMX

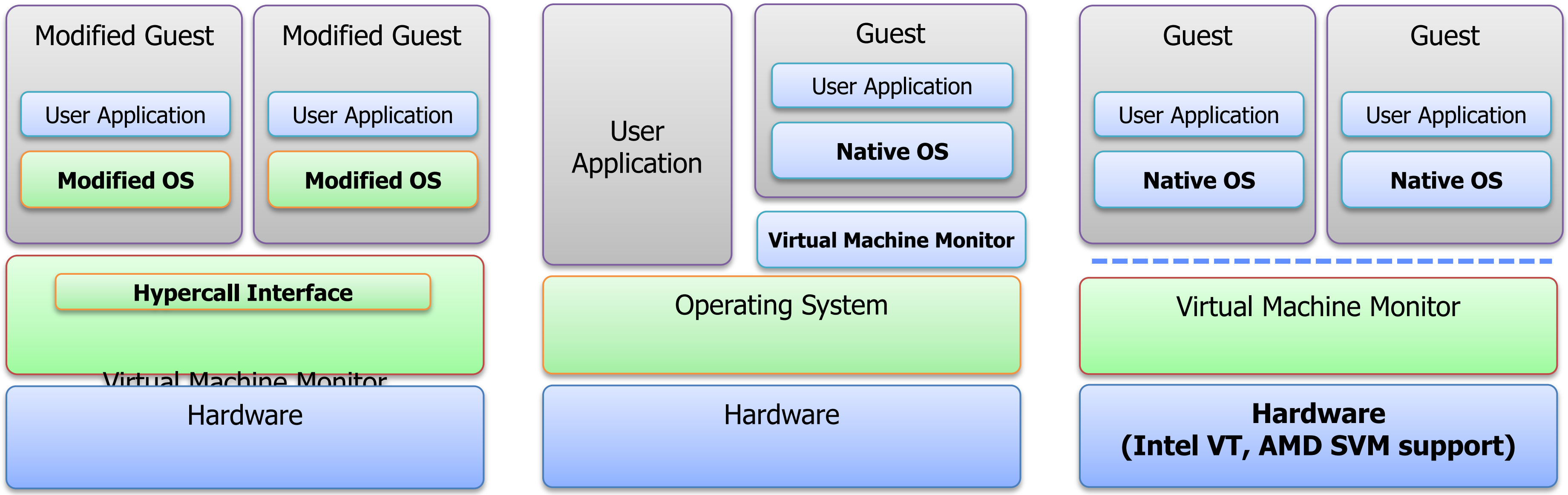
- VMX (Virtual Machine Extension) supports virtualization of processor hardware.
- Two new VT-x operating modes
  - Less-privileged mode (VMX non-root) for guest OSes
  - More-privileged mode (VMX root) for VMM
- Two new transitions
  - VM entry to non-root operation
  - VM exit to root operation



- Execution controls determine when exits occur
  - Access to privilege state, occurrence of exceptions, etc.
  - Flexibility provided to minimize unwanted exits
- VM Control Structure (VMCS) controls VMX operation
  - Also holds guest and host state

Processor Virtualization

- Comparison



Para-virtualization

Full-virtualization
(Emulation, Binary Translation)

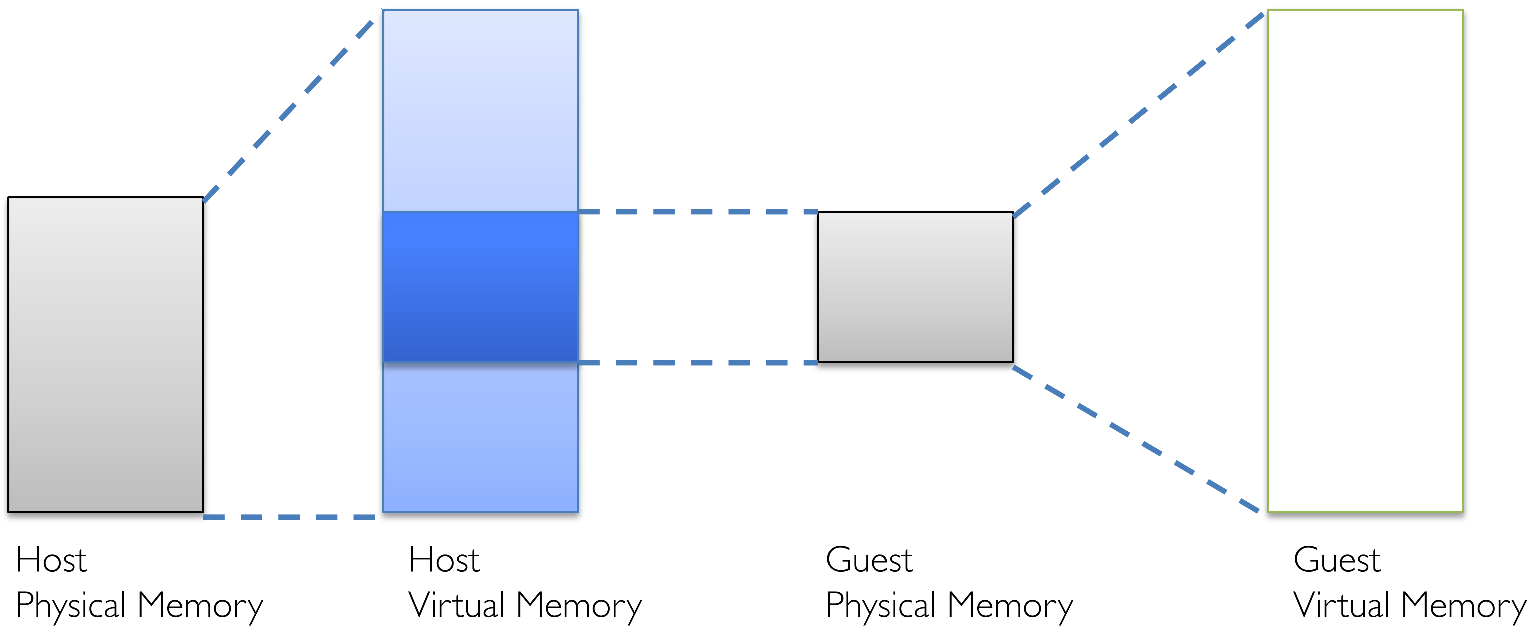
Full-virtualization
(Hardware-assisted VT)

Processor Virtualization (cont'd)

- Comparison

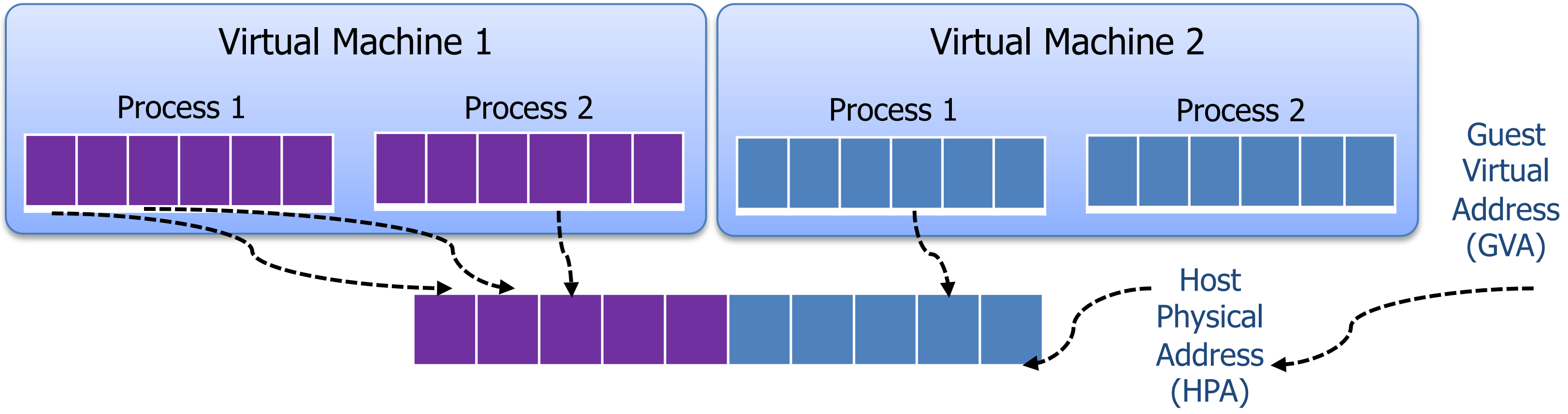
	Para-virtualization	Full-virtualization (Emulation)	Full-virtualization (Binary translation)	Full-virtualization (Hardware-assisted VT)
Speed	Very Fast (Almost Native)	Very Slow	Fast	Fast
Guest Kernel Modification	Yes	No	No	No
Support Other Arch	No	Yes	No	No
Solutions	Xen, VMWare ESX	Bochs	VMWare, QEMU	KVM, Xen
Purposes	Server virtualization	Emulator	Desktop virtualization	Desktop virtualization

Virtual Machine Memory Map



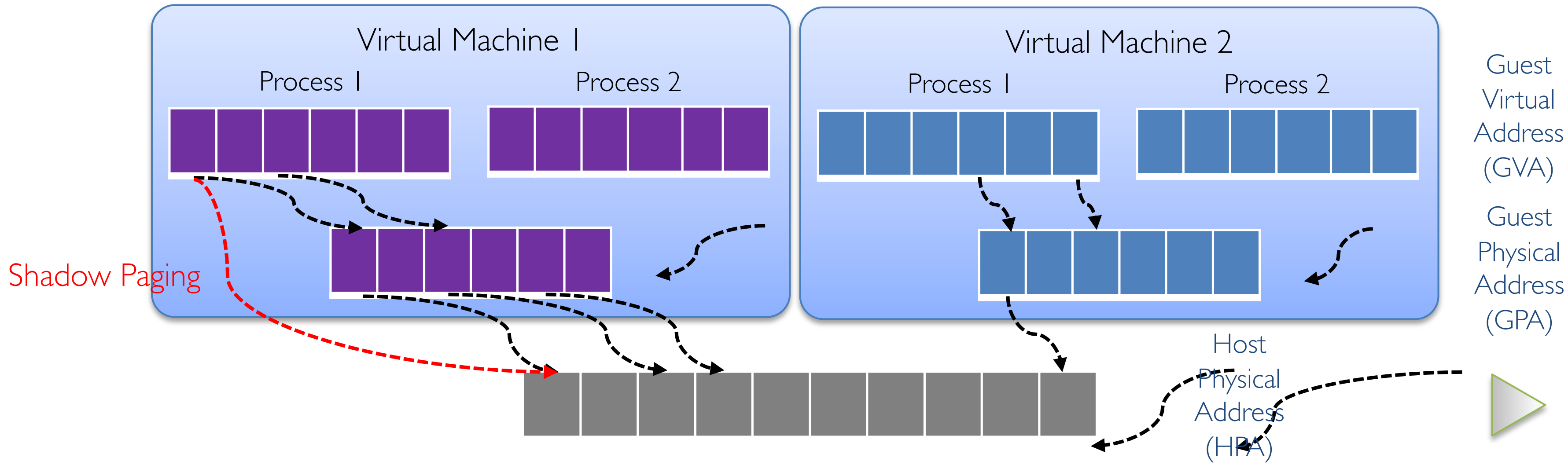
Memory Virtualization

- Direct Paging
  - Guest operating system directly maintains a mapping of Guest Virtual Address to Host Physical Address (GVA → HPA).
  - When a logical address is access, the hardware walks these page tables to determine the corresponding physical address.
  - Dedicated physical memory region is allocated at the initialization of guest OS.
  - Pros
    - Simple to implement
    - High performance (no virtualization overhead)
  - Cons
    - Need to modify guest kernel (not applicable to closed-source OS)
    - Inflexible memory management

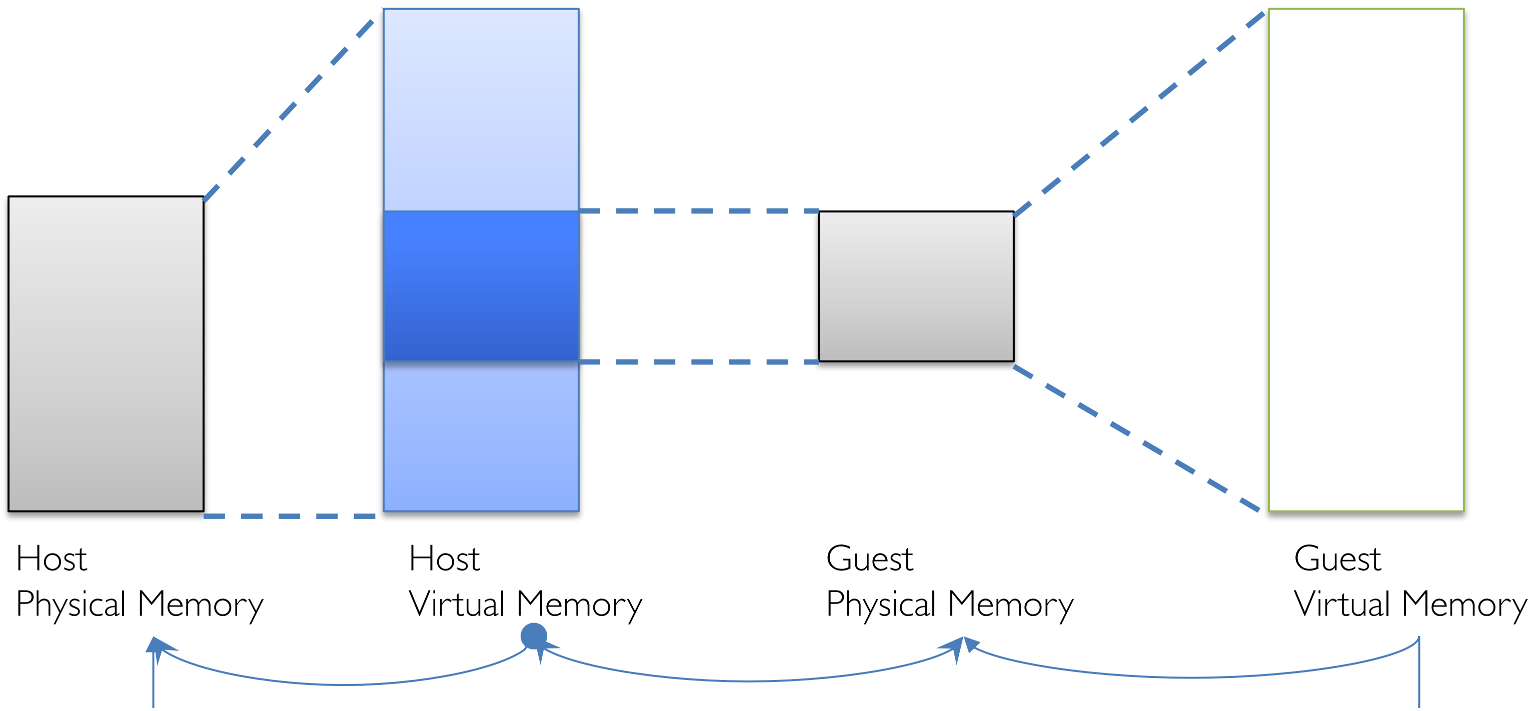


Memory Virtualization (cont'd)

- Shadow Paging
  - VMM maintains GVA→GPA mappings in its internal data structures and stores GVA→HPA mappings in shadow page tables that are exposed to the hardware.
  - The VMM keeps these shadow page tables synchronized to the guest page tables.
  - This synchronization introduces virtualization overhead when the guest updates its page tables.
  - Pros
    - Support unmodified guest OS
  - Cons
    - Hard to implement and maintain
    - Large virtualization overhead



Shadow Paging



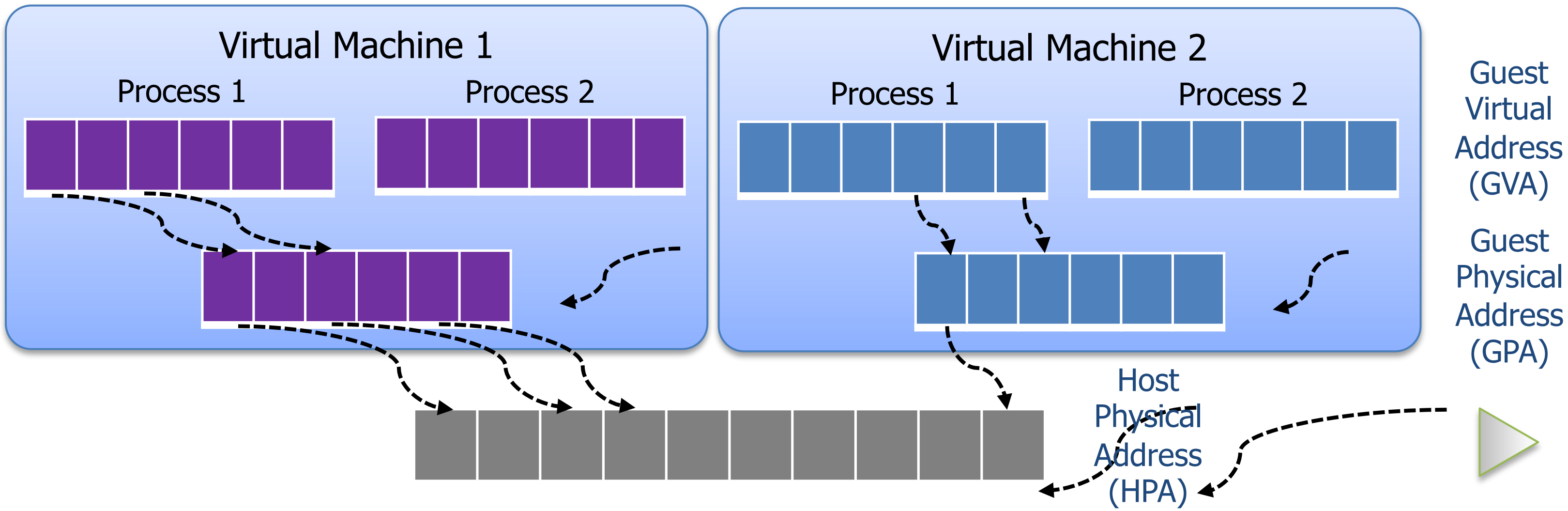
- MMU with host page table created by host kernel
- One-to-one mapping

- Guest page table
- It represents some 'logical' address in guest environment. It cannot be used for MMU, because guest physical address is another virtual address.

- MMU with shadow page table created by hypervisor

Memory Virtualization (cont'd)

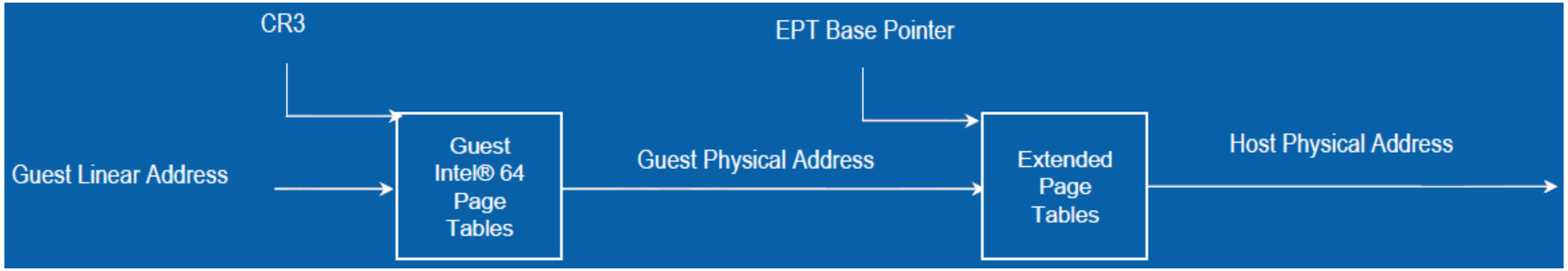
- Nested Paging
  - Guest operating system continues to maintain GVA-->GPA mappings in the guest page tables.
  - But the VMM maintains GPA->HPA mappings in an additional level of page tables, called nested page tables.
  - Both the guest page tables and the nested page tables are exposed to the hardware.
  - When a logical address is accessed, the hardware walks the guest page tables as in the case of native execution, but for every GPA accessed during the guest page table walk, the hardware also walks the nested page tables to determine the corresponding HPA.
  - Pros
    - Simple to implement
    - Support unmodified guest OS
  - Cons
    - H/W supports is needed.
    - Larger TLB footprint



Memory Virtualization (cont'd)

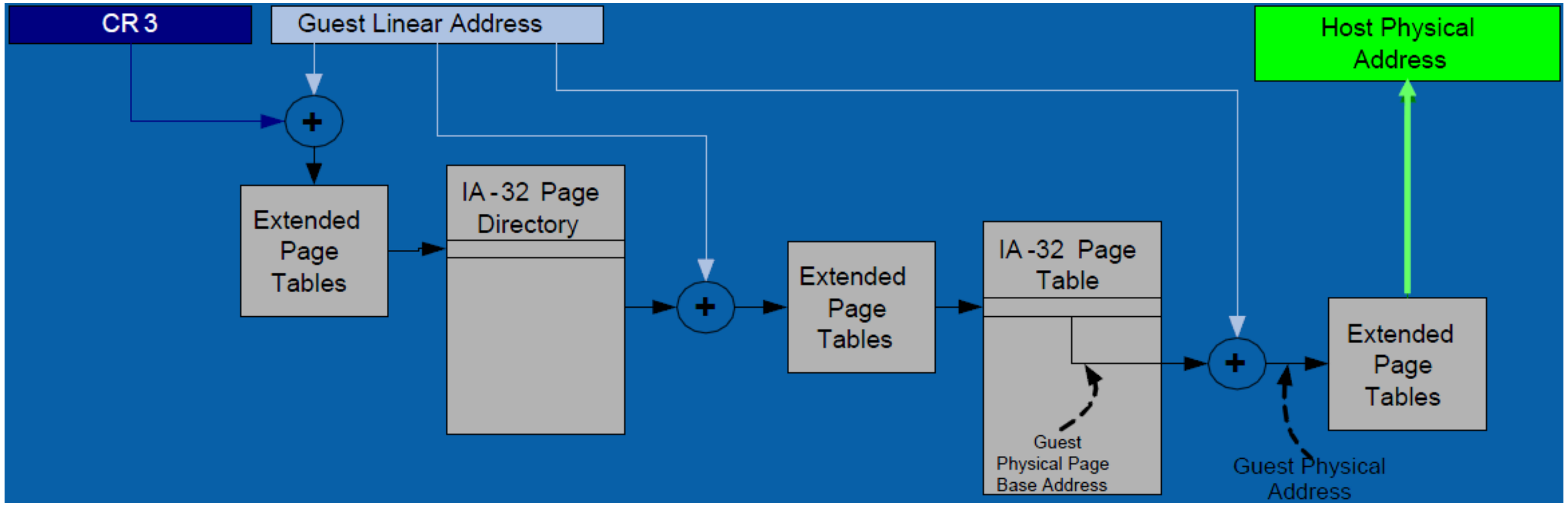
	Direct Paging	Shadow Paging	Nested Paging
Speed	Very Fast (Almost Native)	Very Slow	Fast
Guest Kernel Modification	Yes	No	No
Need H/W Support	No	No	Yes
Complexity	Simple	Complex	Very Simple

EPT



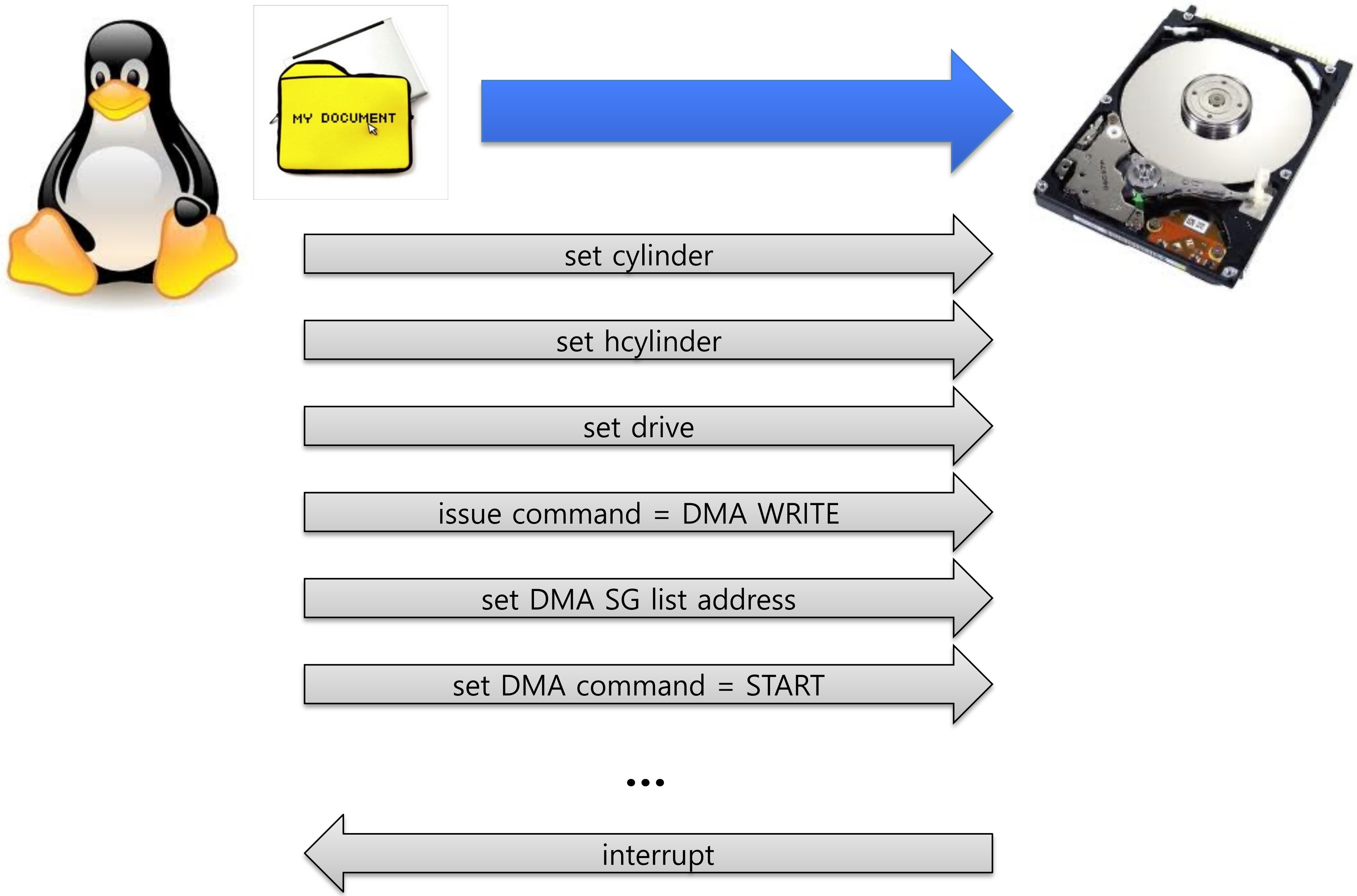
- The extended page-table mechanism (EPT) is a feature that can be used to support the virtualization of physical memory.
- Guest-physical addresses are translated by traversing a set of EPT paging structures to produce physical addresses that are used to access memory.
- Guest can have full control over page tables/events
  - CR3, CR0, CR4 paging bits, INVLPG, page fault
- VMM controls Extended Page Tables
- CPU uses both tables, guest paging structure and EPT paging structure
- EPT activated on VM entry
  - When EPT active, EPT base pointer (loaded on VM entry from VMCS) points to extended page tables
- EPT deactivated on VM exit

EPT Translation



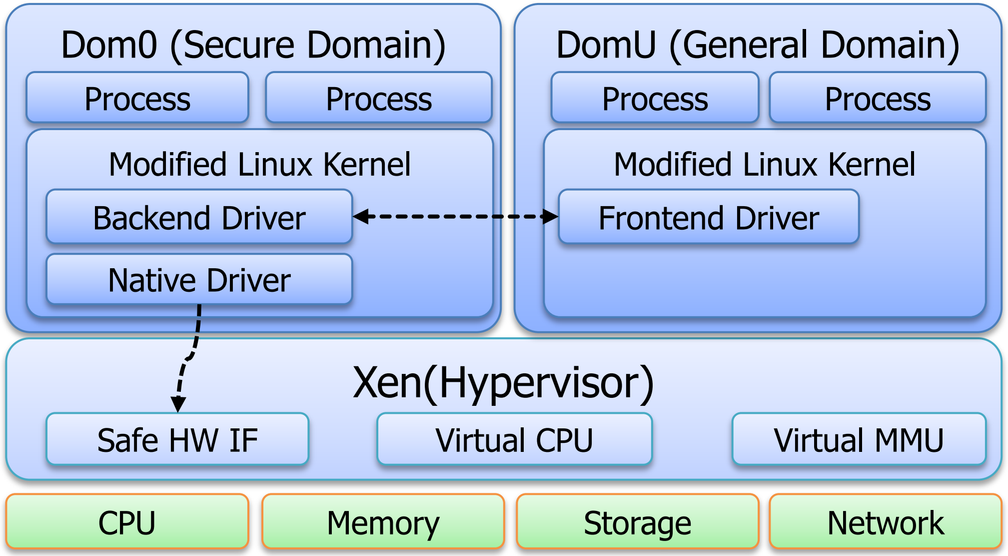
- All guest-physical addresses go through extended page tables
  – Includes address in CR3, address in PDE, address in PTE, etc.
- In addition to translating a guest-physical address to a physical address, EPT specifies the privileges that software is allowed when accessing the address. Attempts at disallowed accesses are called EPT violations and cause VM exits.

How OS and a device interacts



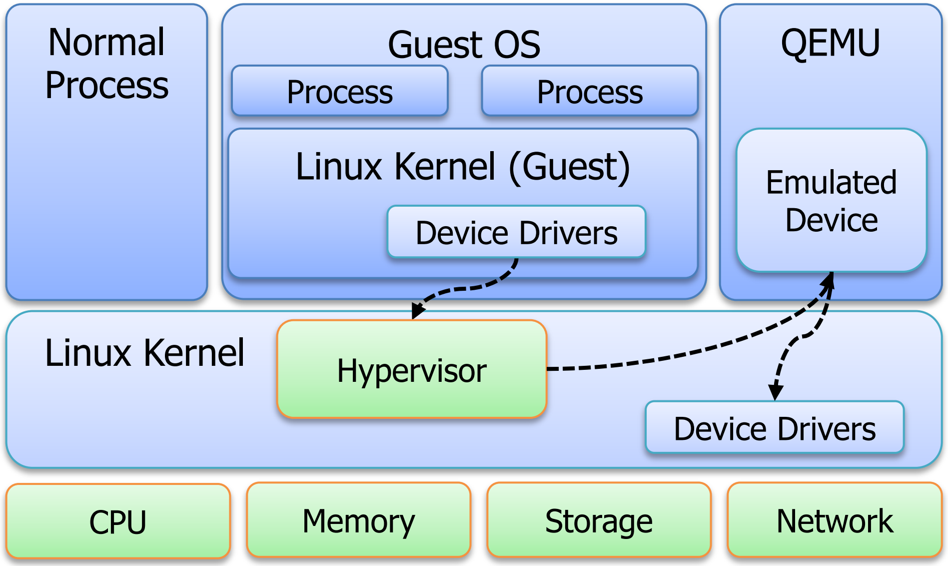
IO Virtualization

- Front-end/Back-end Driver Model
  - Guest OS uses para-virtualized front-end driver to send requests to backend driver.
  - Back-end driver on secure domain receives the requests, performs actual IO using the native driver.



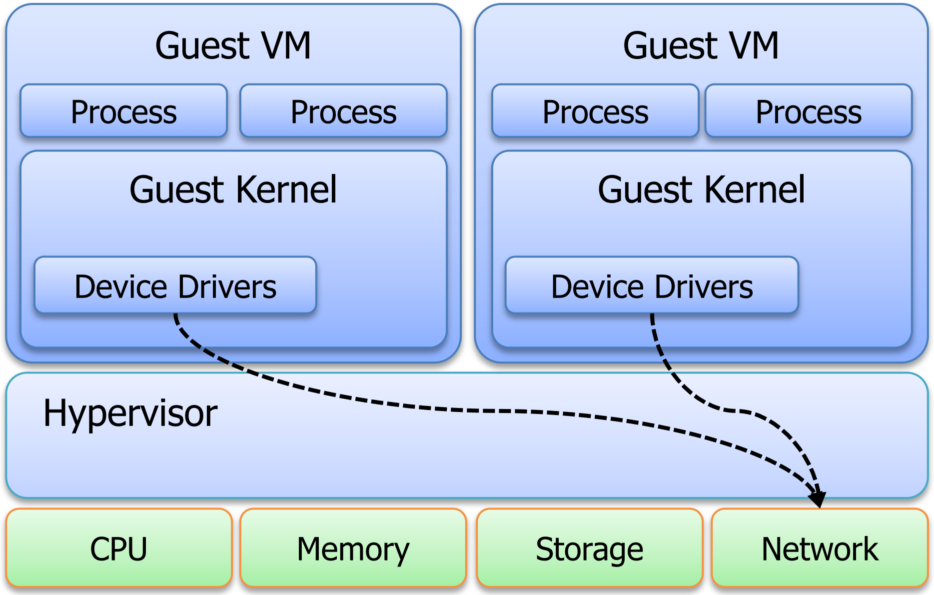
IO Virtualization (cont'd)

- Emulation
  - Behavior of a particular device is emulated as a software module.
  - Guest OS uses the native device driver for the particular device.
  - VMM intercepts all the access from guest OS to the device.
  - The intercepted accesses are sent to the emulated device.
  - The Emulated device do the actual IO operations.

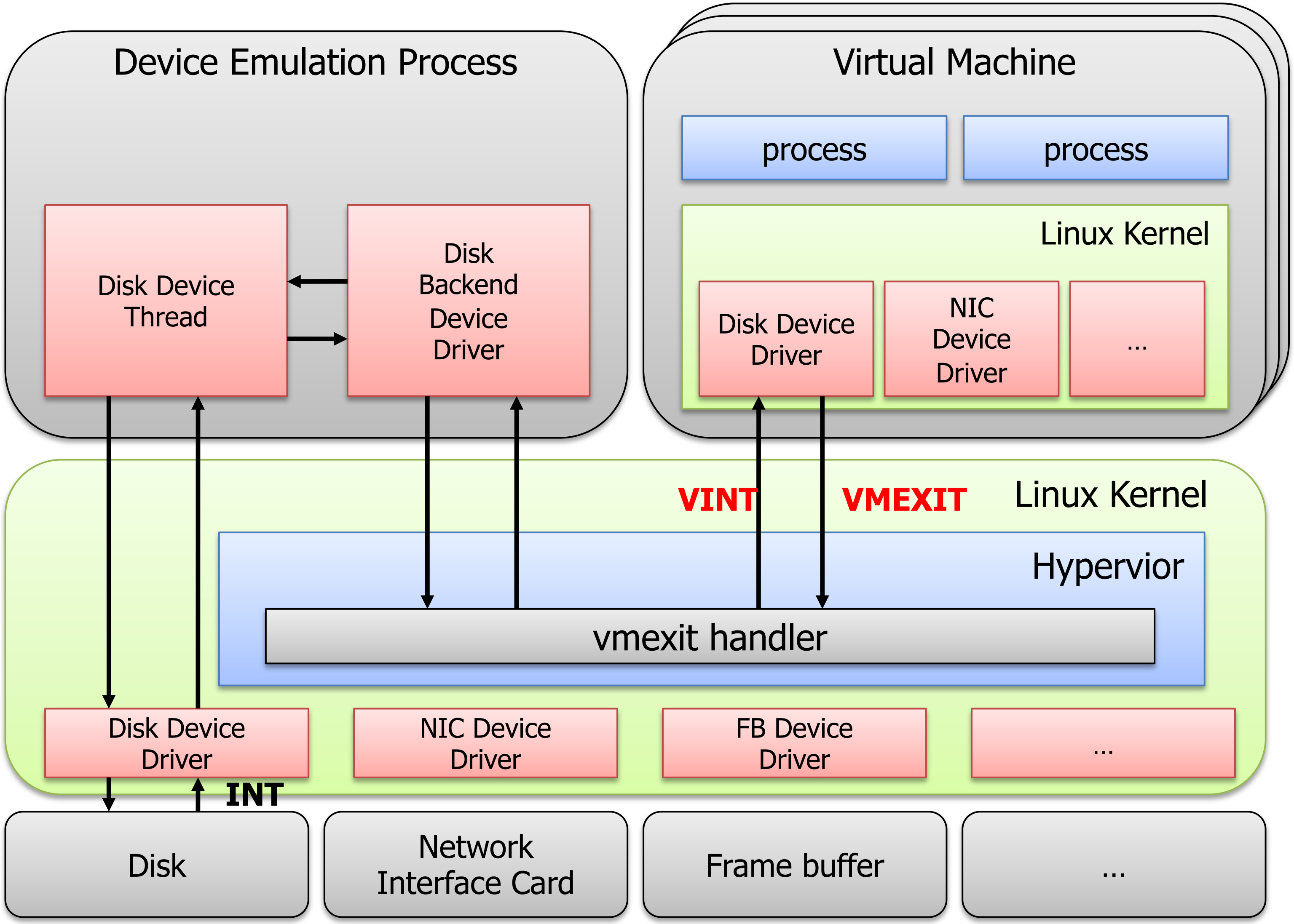


IO Virtualization (cont'd)

- H/W Assisted IO Virtualization
  - A specially designed H/W supports concurrent accesses from multiple guest OS.
  - Guest OS use the unmodified device driver.
  - Guest OS can access arbitrary host physical memory through DMA.
    - Intel VT-d controls the host physical memory access from the guest OS through DMA.



IO Virtualization Model Revisited
Front-end/Back-end Model



IO Virtualization (cont'd)

	Front-end/Back-end Driver Model	Emulation	H/W Assisted IO Virtualization
Speed	Very Slow	Very Slow	Fast
Device Driver Modification	Yes	No	No
Need H/W Support	No	No	Yes
Complexity	Simple	Complex	Simple

- Virtualization Overview
  - Introduction
  - Processor Virtualization
    - Para-virtualization, Emulation, Binary Translation, H/W assisted VT
  - Memory Virtualization
    - Direct Paging, Shadow Paging, Nested Paging
  - IO Virtualization
    - Front-end/Back-end Driver Model, Emulation, H/W assisted IO Virtualization

- Memory Management
  - Monitoring page access and swap device
  - VMM swapping without double paging
  - QoS aware memory allocation
  - Ballooning

- I/O Virtualization
  - Accelerating Virtual Machine Storage I/O for Multicore Systems
    - Virtualization overhead = direct cost + indirect cost + synchronous cost
  - "Towards Bare-metal Network Performance via Para-virtualized Socket Library and Exitless I/O"
    - Exitless IO
    - Paravirtualized Network Library