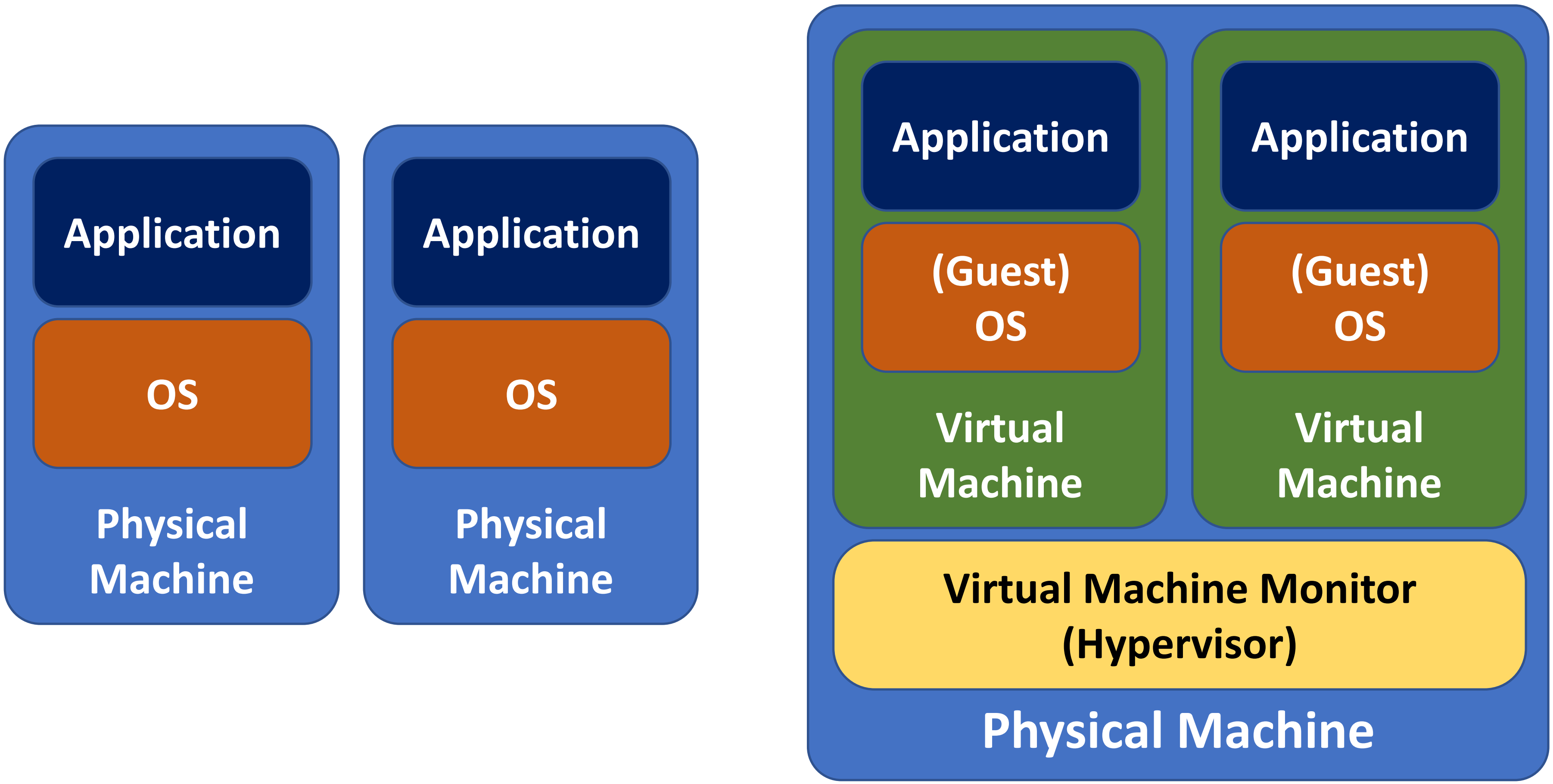
Virtualization

ENGR 689 (Sprint)

TEXAS A&M UNIVERSITY

What’s Virtualization?

Physical vs Virtual Machines



Without virtualization **With virtualization**

Why Virtualization in Cloud?

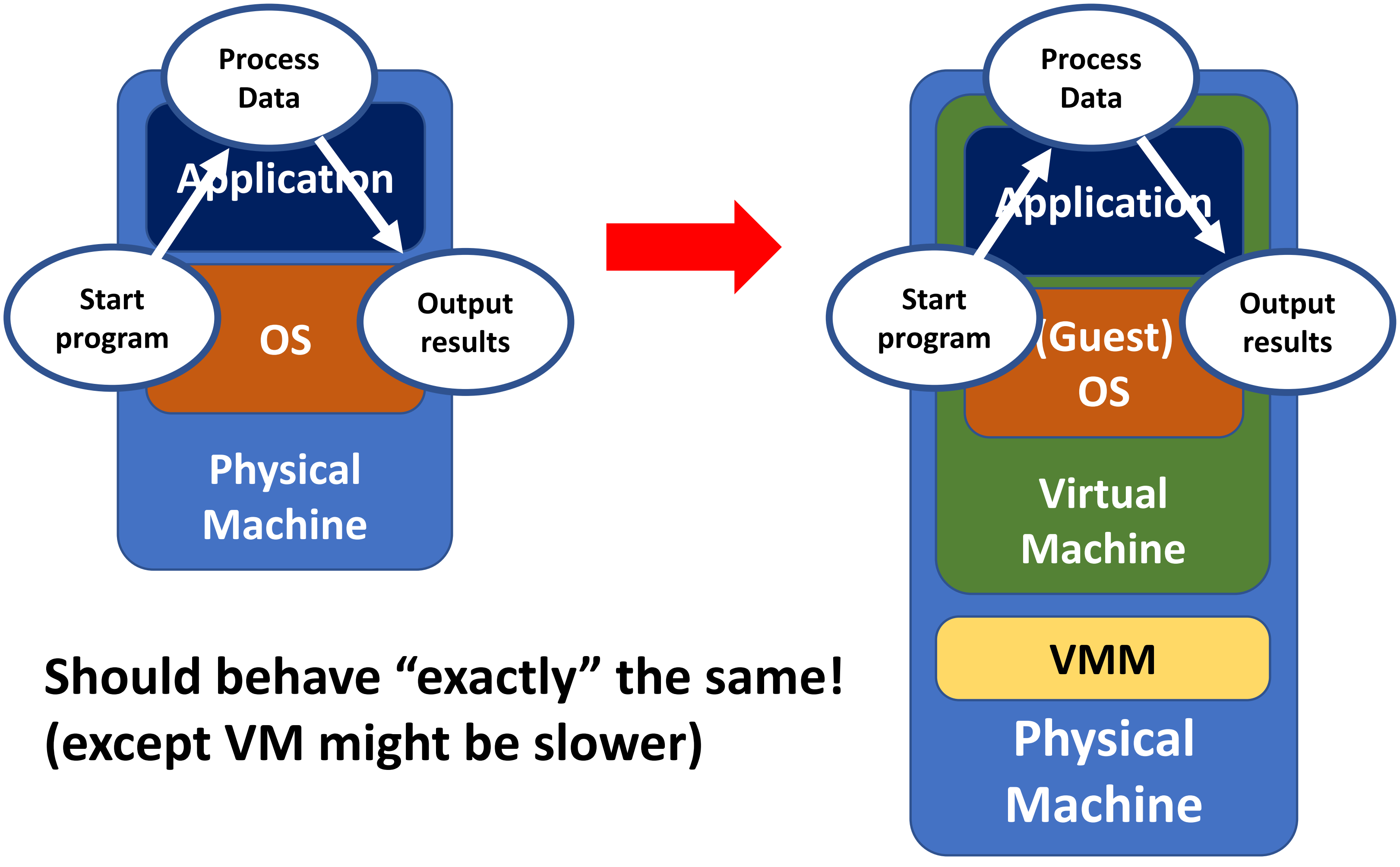
- **Ease of resource sharing:**
  splitting hardware resources for multiple tenants
- **Security Isolation:**
  tenants are isolated from each other
- **Legacy software:**
  tenants run their OSes and applications without modification
- **Flexibility of deployment:**
  tenants can be easily deployed, migrated, or removed from a physical host

Classic Virtualization

Popek & Goldberg (1974):

- **Identical environment:**
  Programs in VMs should have identical effects as running on real machines
- **Efficiency:** should not add too much overheads
- **Resource control:**
  Virtual machine monitor (VMM) has complete control of hardware resources

Identicality



Should behave "exactly" the same!
(except VM might be slower)

How to Achieve Virtualization?

- Software emulation
- Hardware-based virtualization
- Paravirtualization (not classic virtualization)
  - Discuss in next lecture

Software Emulation

Guest OS Binary

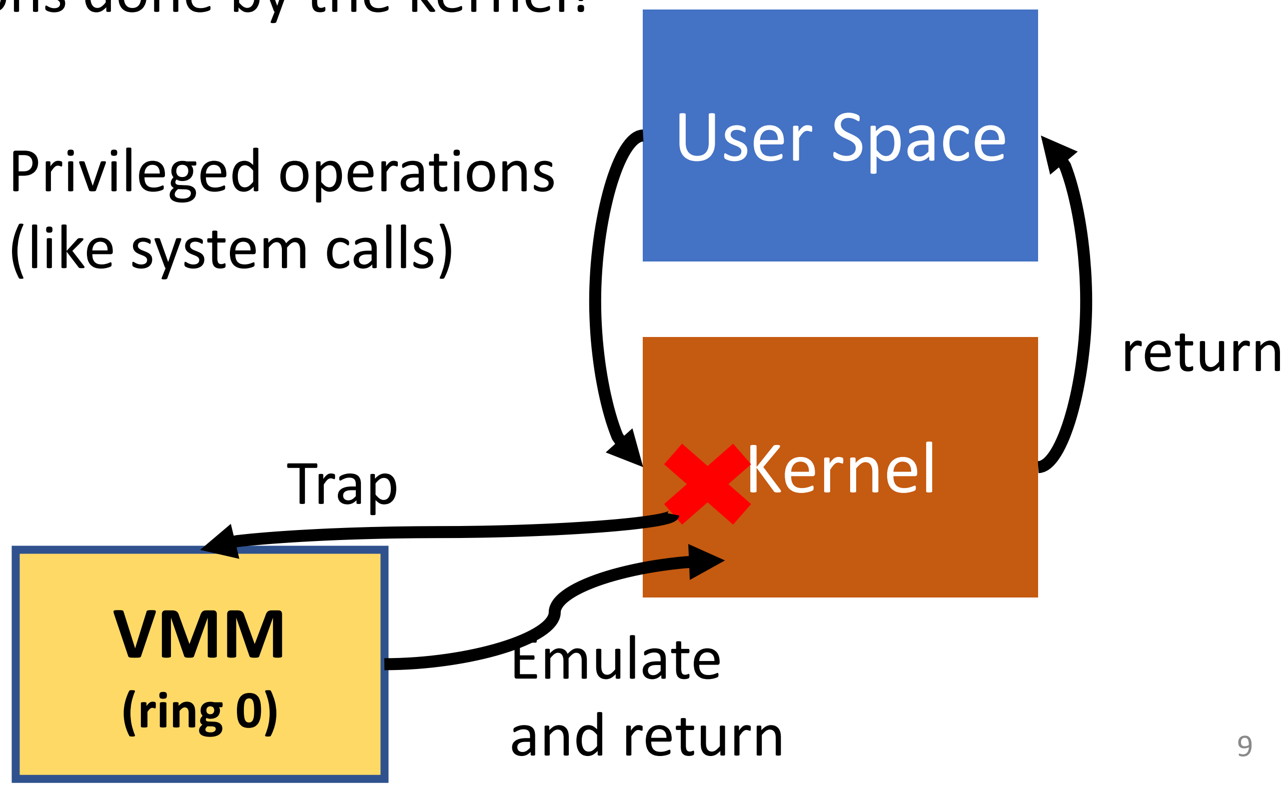
```
00000000    push    ebp
00000001    mov     ebp, esp
00000003    movzx   ecx, [ebp+arg_0]
00000007    pop     ebp
00000008    movzx   dx, cl
0000000C    lea     eax, [edx+edx]
0000000F    add     eax, edx
00000011    shl     eax, 2
00000014    add     eax, edx
00000016    shr     eax, 8
00000019    sub     cl, al
0000001B    shr     cl, 1
0000001D    add     al, cl
0000001F    shr     al, 5
00000022    movzx   eax, al
00000025    retn
```

Emulate the execution
of instructions one by one.

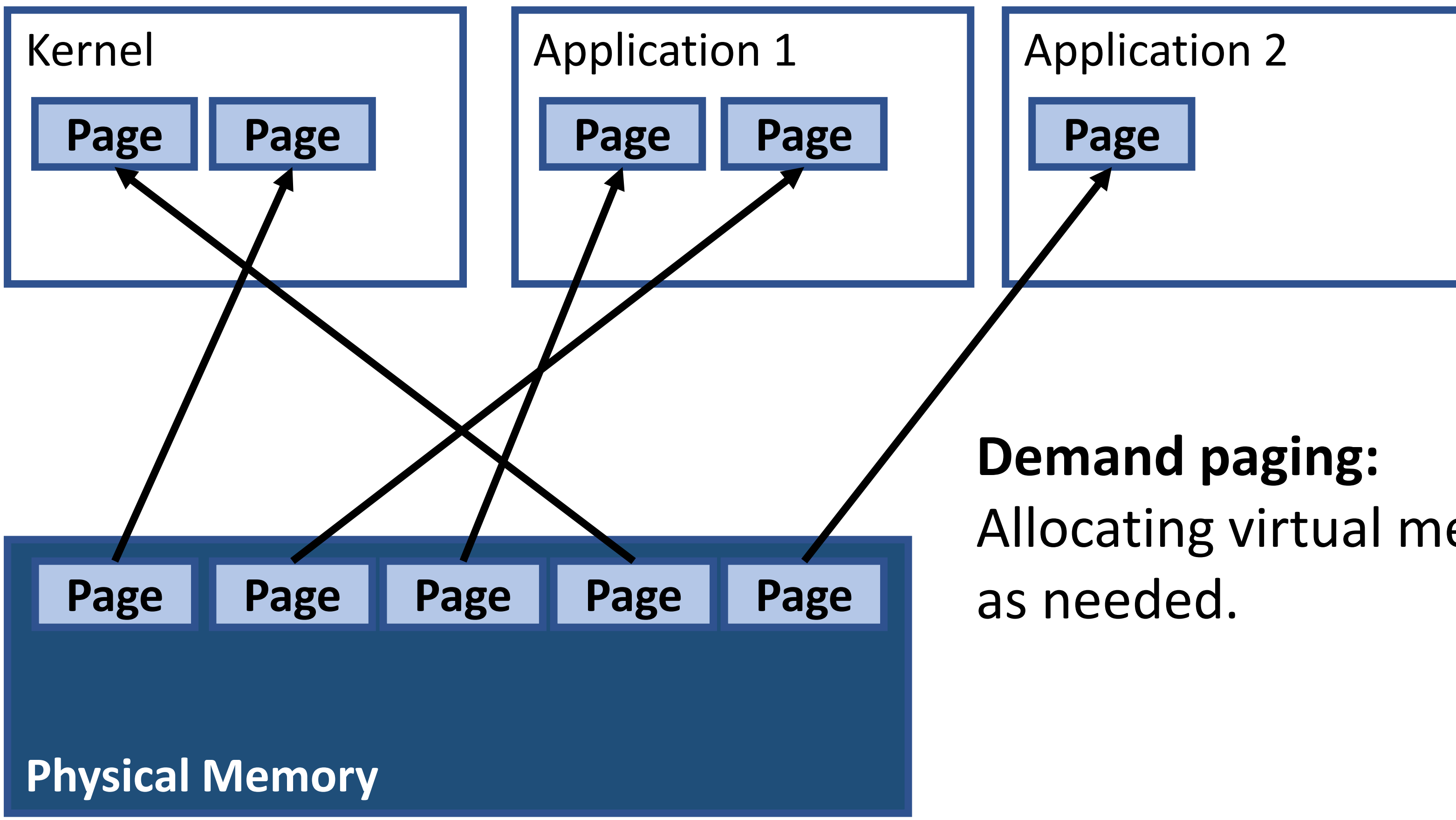
Emulator
(e.g., Qemu)

Trap & Emulate

- For efficiency, only “privileged operations” need to be emulated
  - Operations done by the kernel!

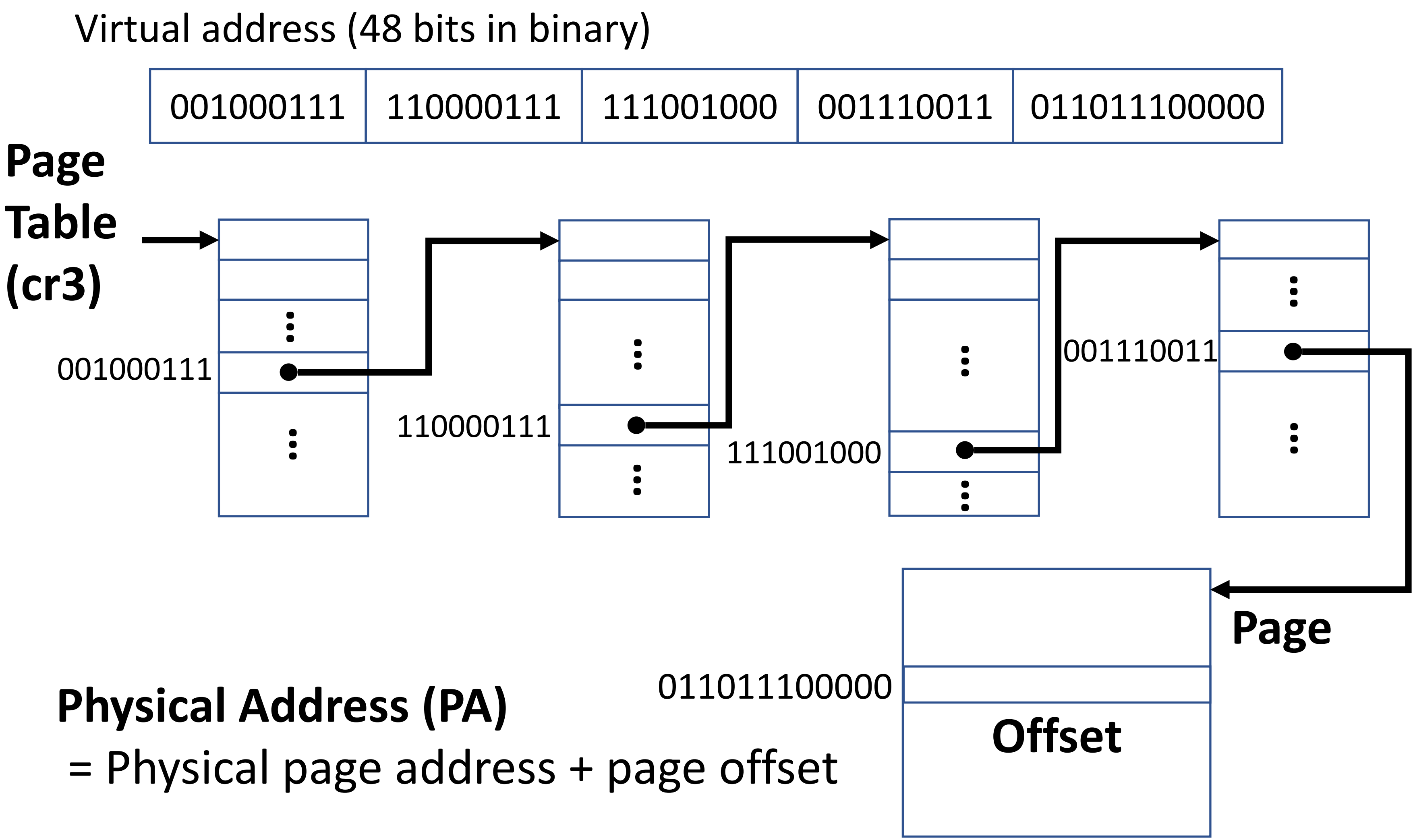


Example: Memory Management



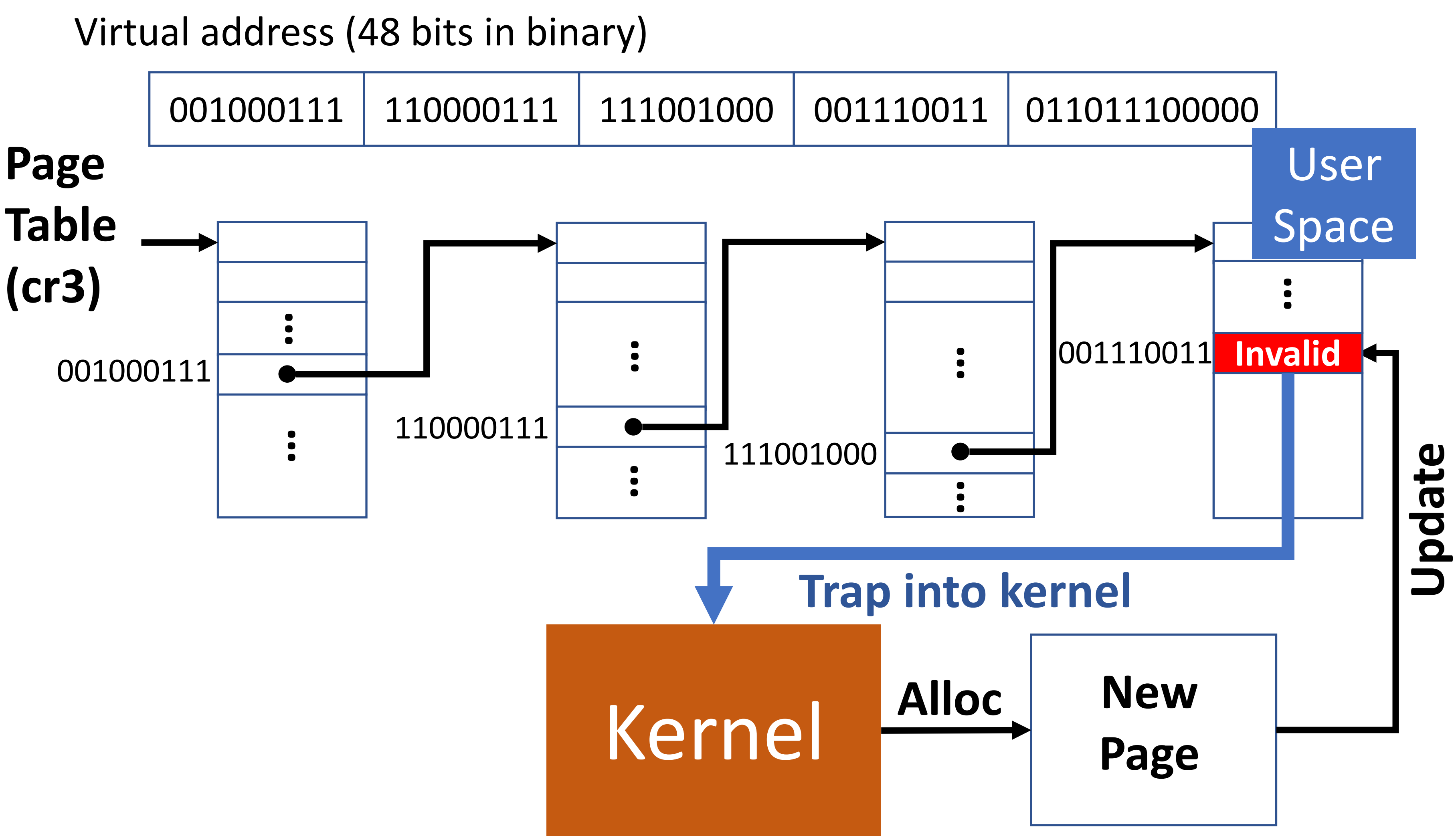
Demand paging:
Allocating virtual memory as needed.

Virtual Address → Physical Address

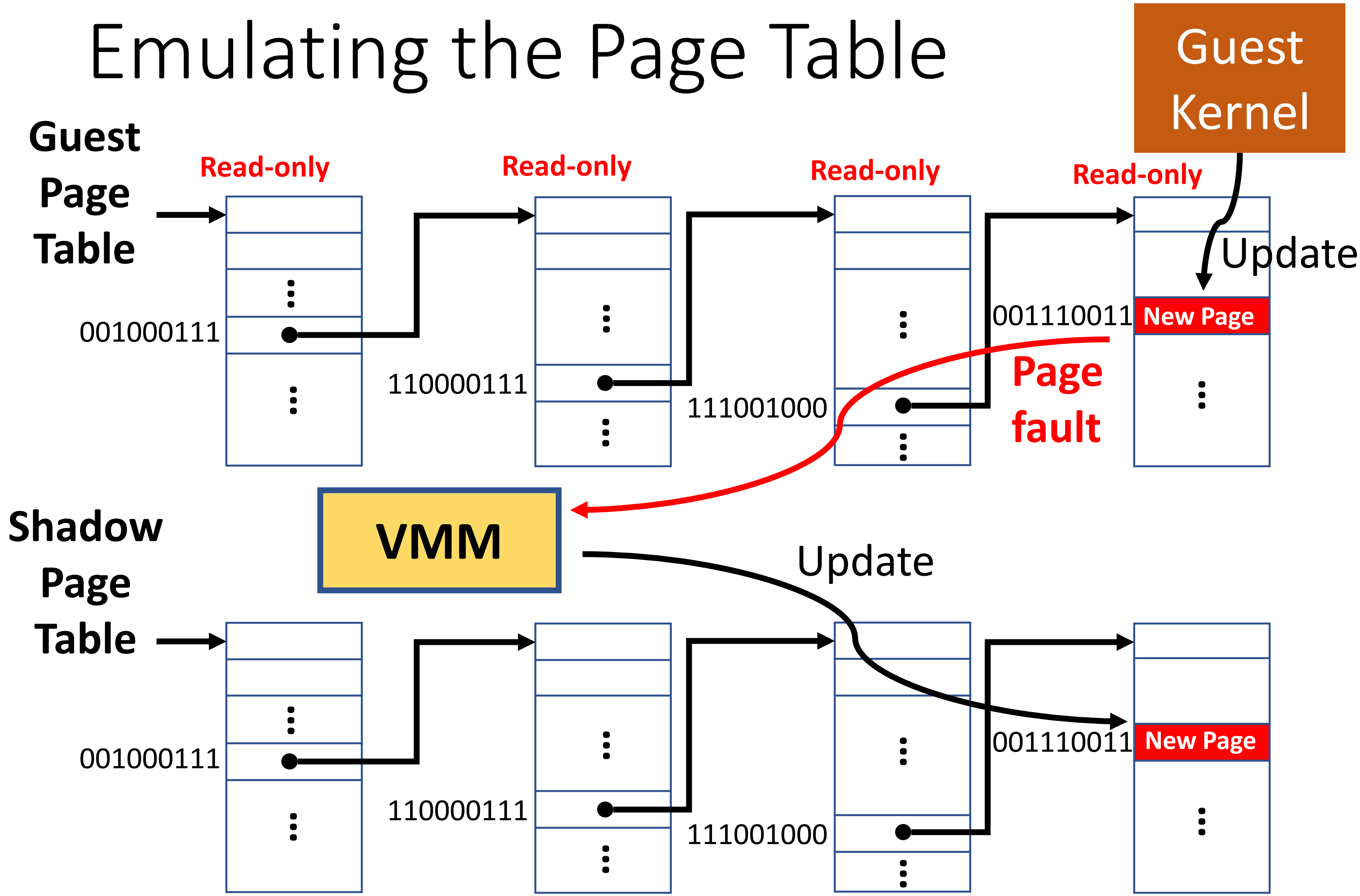


Physical Address (PA)
= Physical page address + page offset

Page Fault Handling



Emulating the Page Table



Kernel vs VMM

- An OS relies on hardware protections to ensure all resources are controlled by kernel
- VMM has the exact same requirement

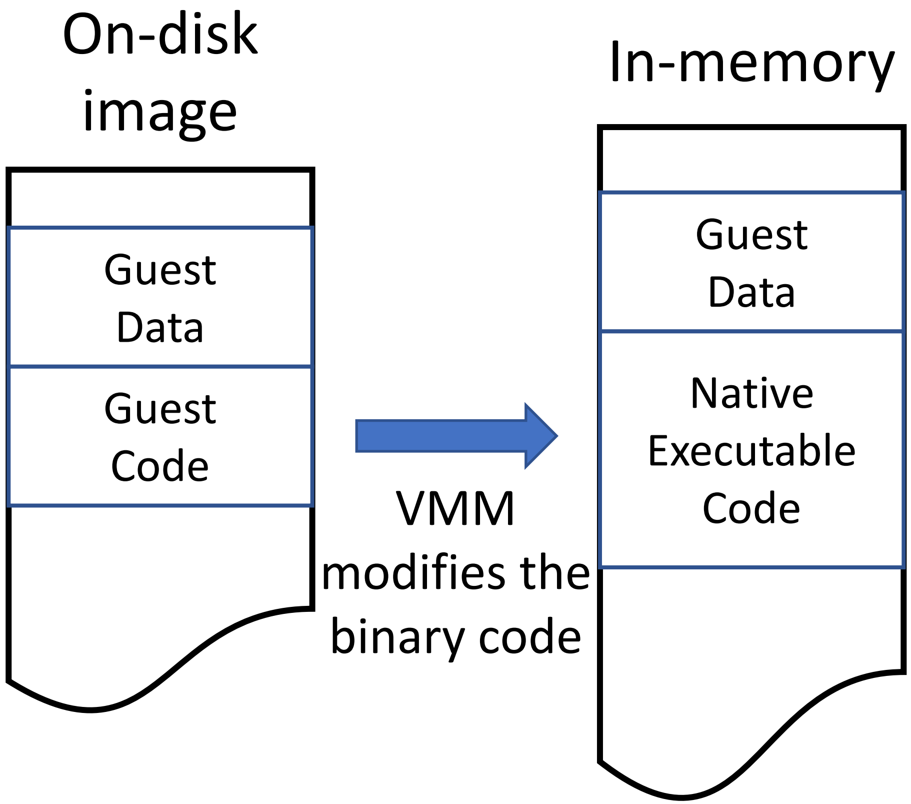
➔ The main challenge: How to subvert the control of OS kernel when the VMM is in charge?

Virtualization: The VMWare Approach

x86 Was Not Virtualizable

- Popek & Goldberg: all privileged instructions need to be trapped in non-kernel mode (ring > 0)
- Many x86 instructions are not trappable
  - Example 1: PUSH %cs pushes current protection level on the stack, so guest kernel can see ring != 0
  - Example 2: POPF can enable/disable interrupt in ring 0, but silently ignored in ring > 0
  - **VMM never gets the chance to emulate!**

Dynamic Binary Translation



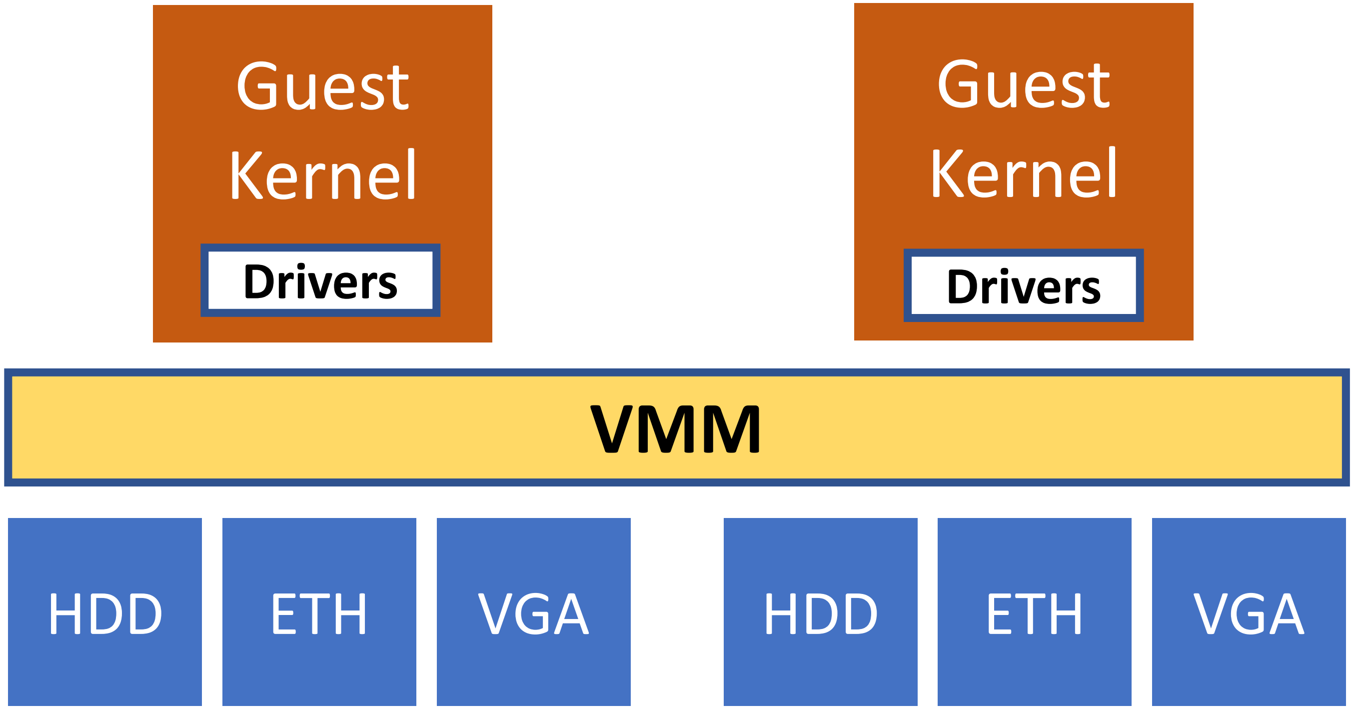
Untrapped privileged instructions can be either:

(1) Replaced with emulation code which originally runs in VMM

(2) Injected with other trappable instructions (e.g., syscall)

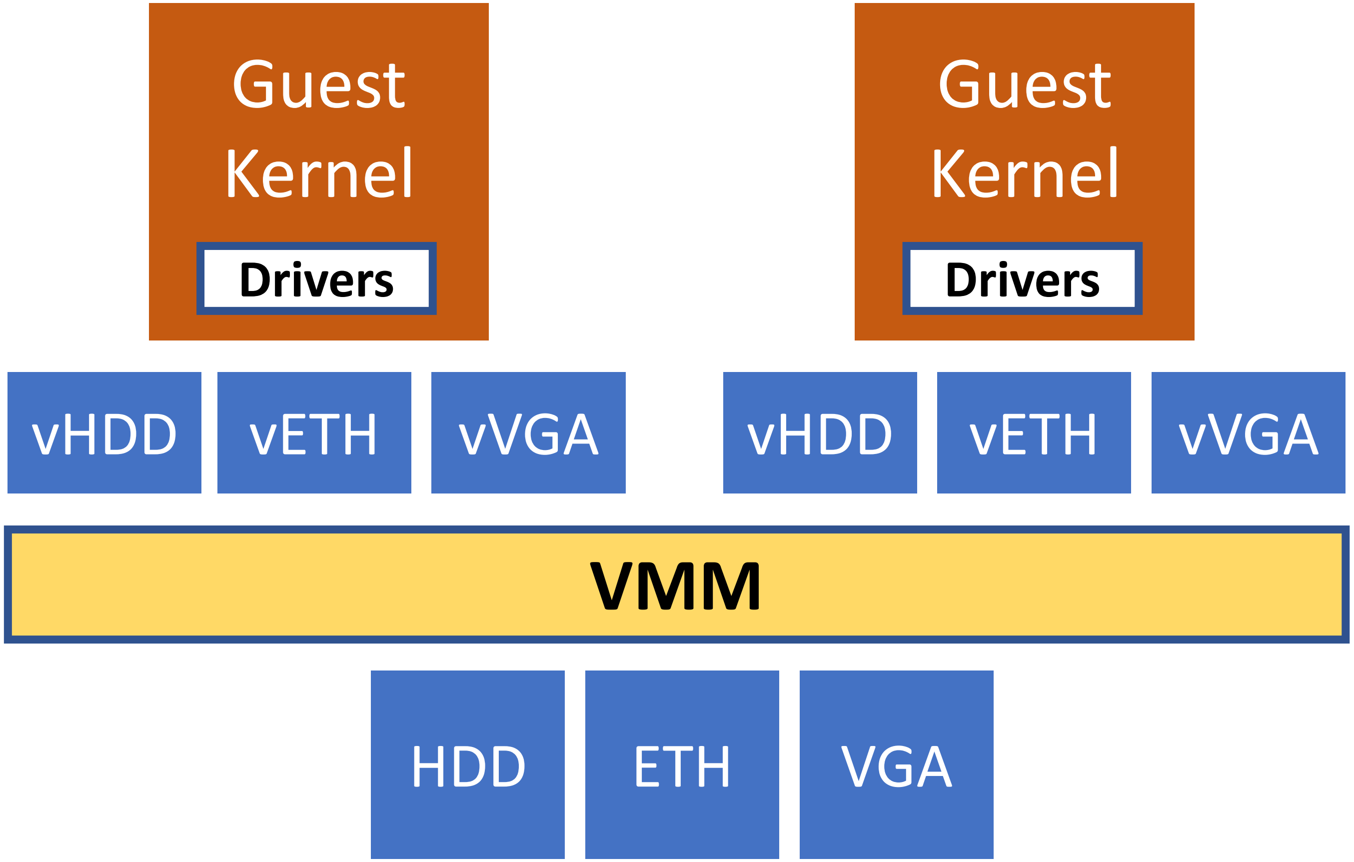
Virtualizing I/O Devices (1/2)

- Assigning physical devices to guest kernels pose engineering and security challenges because guest drivers can't access devices directly

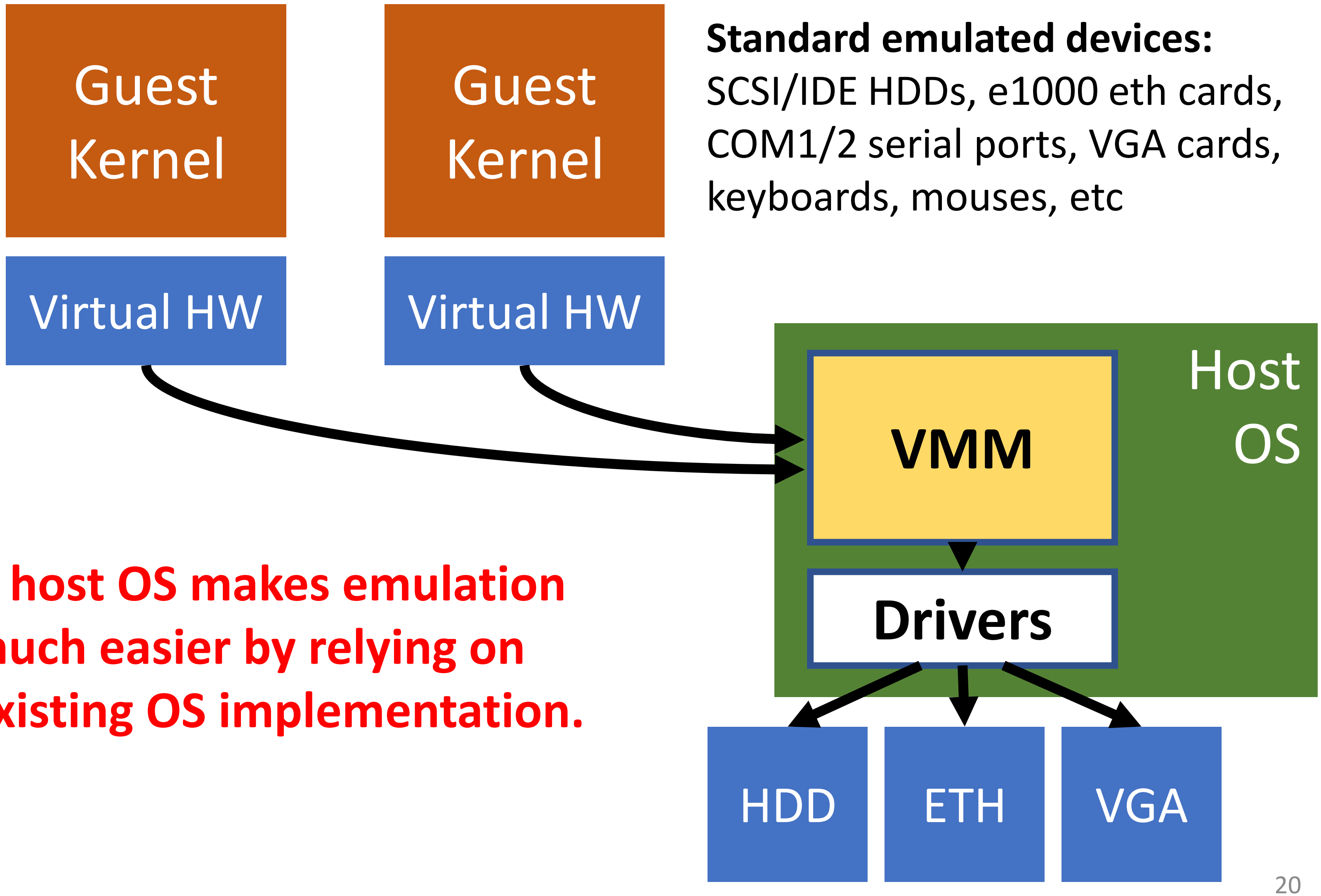


Virtualizing I/O Devices (1/2)

- VMM creates virtual devices to multiplex access to I/O resources



Use of A Host Operating System



Standard emulated devices:
SCSI/IDE HDDs, e1000 eth cards, COM1/2 serial ports, VGA cards, keyboards, mouses, etc

A host OS makes emulation much easier by relying on existing OS implementation.

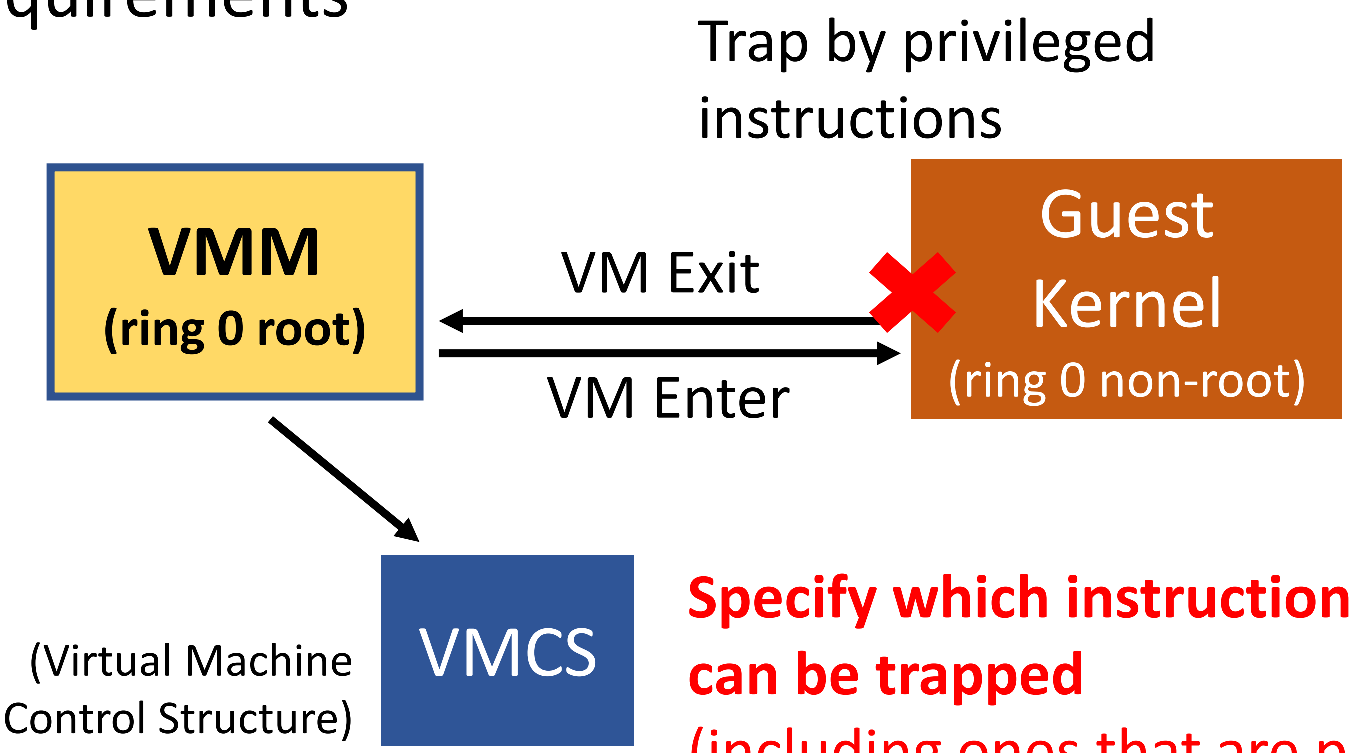
Virtualization:
The Hardware Approach

Modern Virtualization Solutions

- Virtualization nowadays are assisted by hardware
  - x86 is now virtualizable
  - Hardware-assisted paging replaced shadow paging
  - Virtualization-friendly I/O devices

Fixing x86 Virtualization

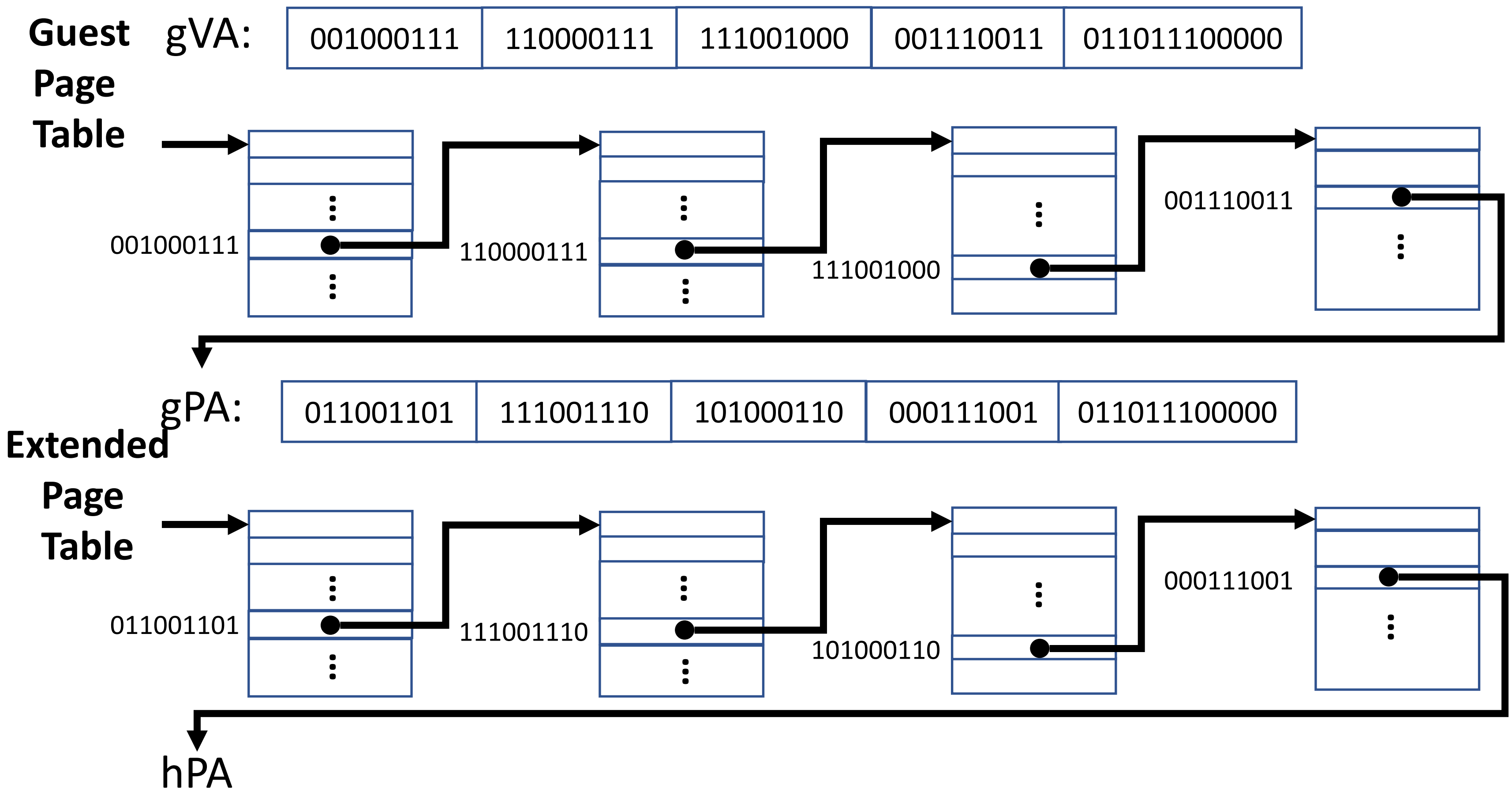
- Intel VT or AMD-V fulfilled the Popek & Goldberg requirements



Specify which instructions can be trapped
(including ones that are previously untrappable, e.g., POPF)

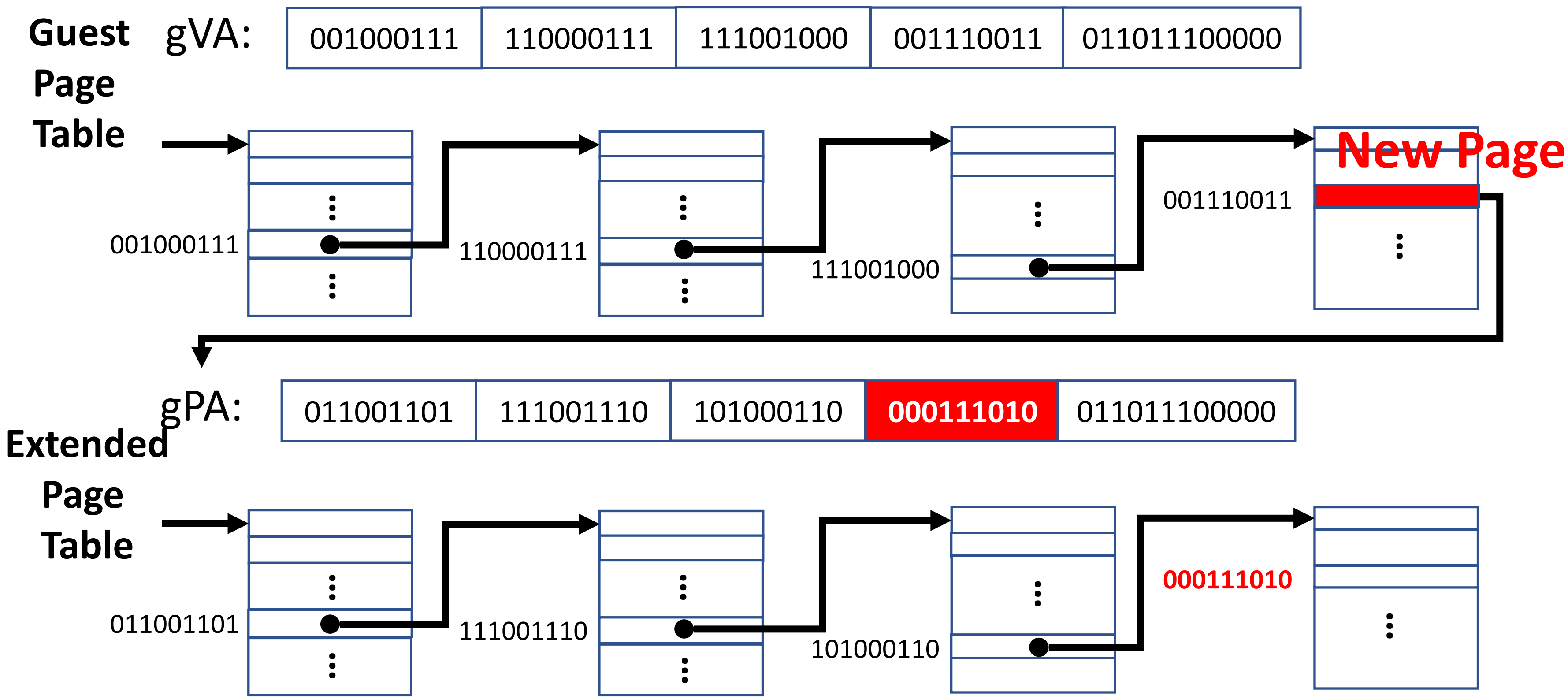
Memory Virtualization (1/3)

- Intel VT-x (extended page table) & AMD SVM (nested page table)



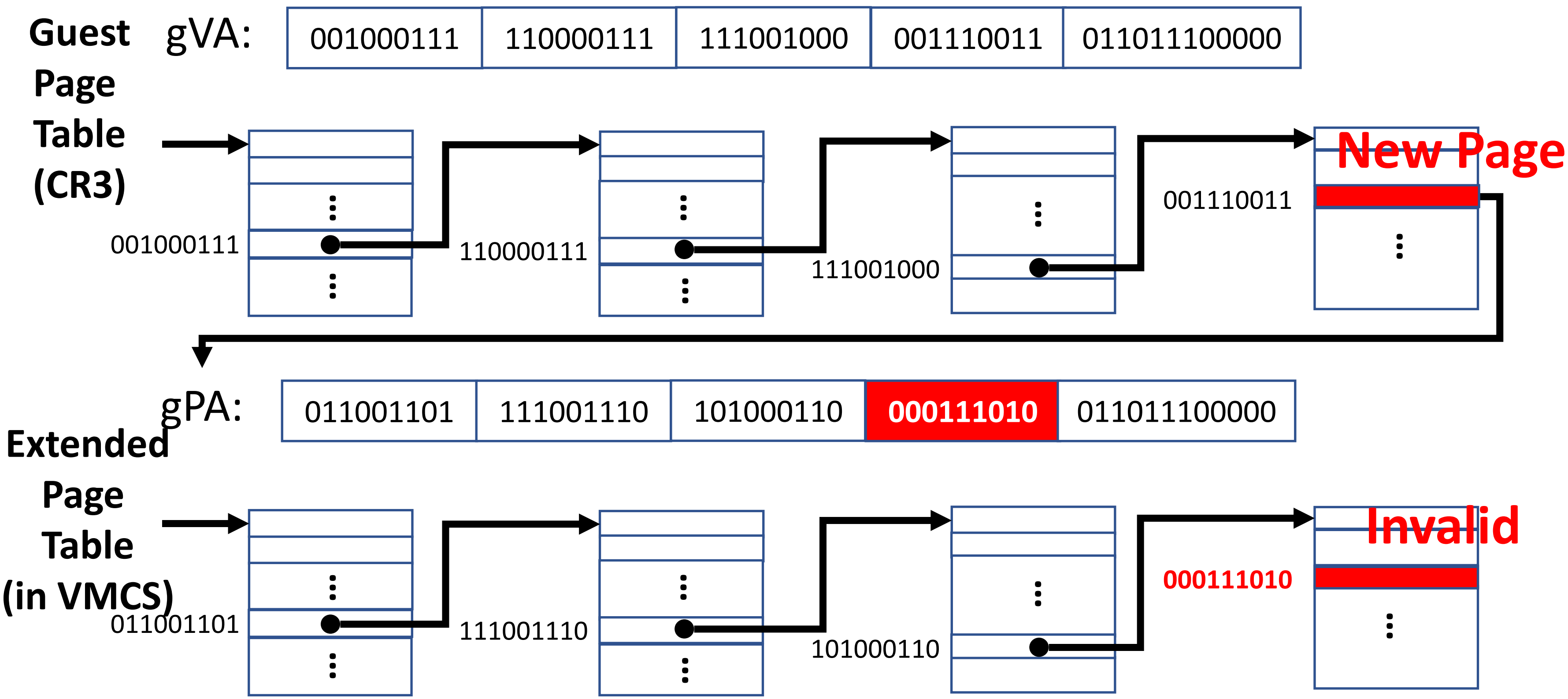
Memory Virtualization (2/3)

- Guest can update CR3 or page tables w/o trapping into VMM



Memory Virtualization (3/3)

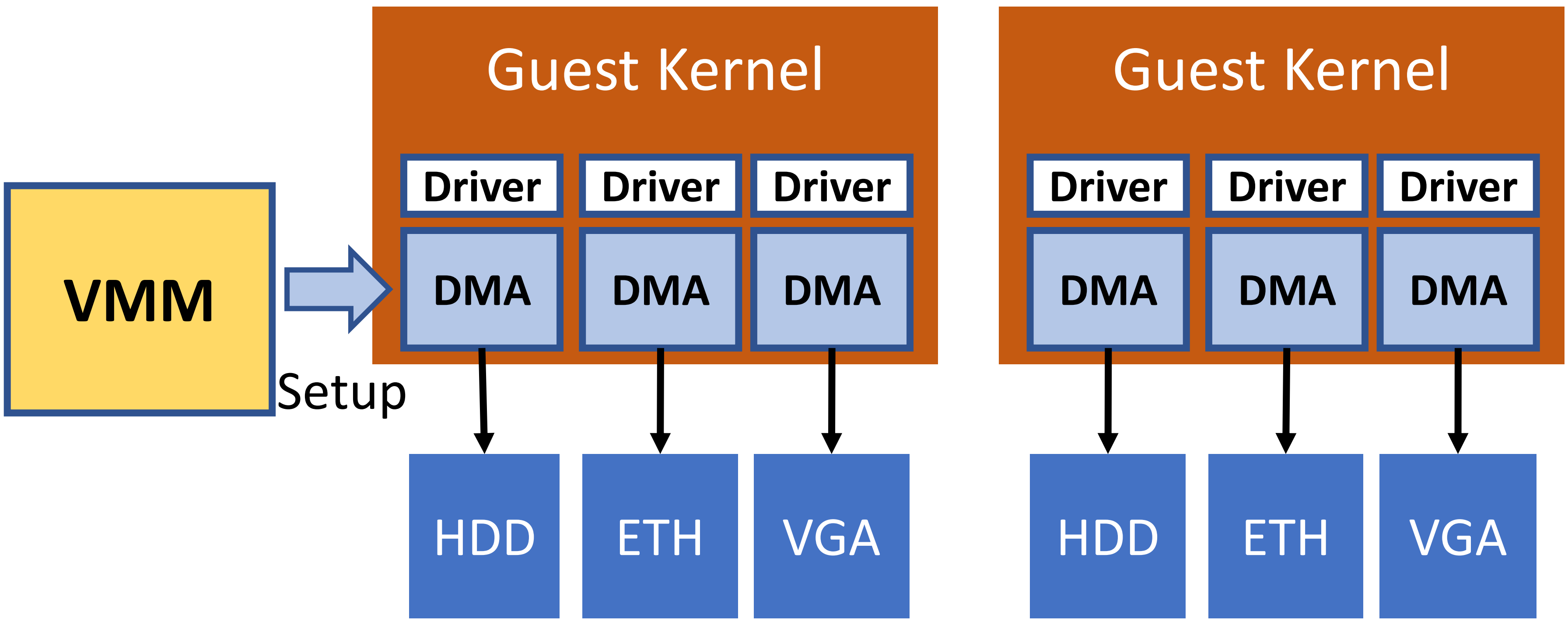
- VMM only manages the guest physical pages



Not your physical page!!!

I/O Virtualization (1/2)

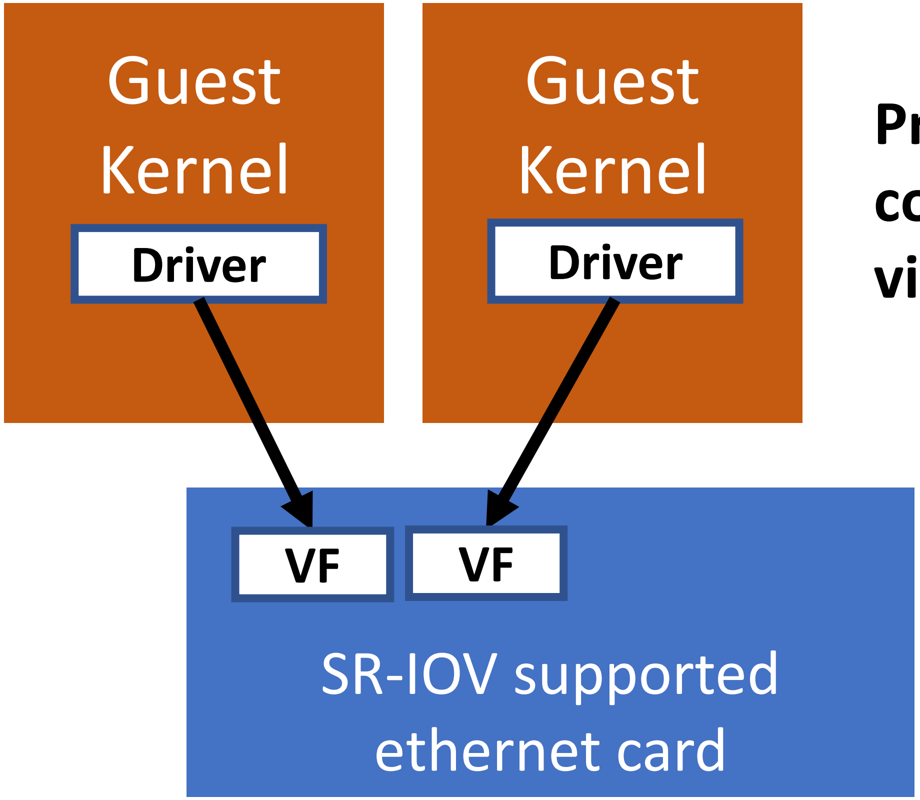
- Intel VT-d and AMD-Vi allow direct I/O to assigned devices
  ➔ Virtual IOMMU (vIOMMU)



Still can’t share devices!

I/O Virtualization (2/2)

- **Single-Root I/O Virtualization (SR-IOV):**
  A specification for sharing PCI-e devices with multiple guests

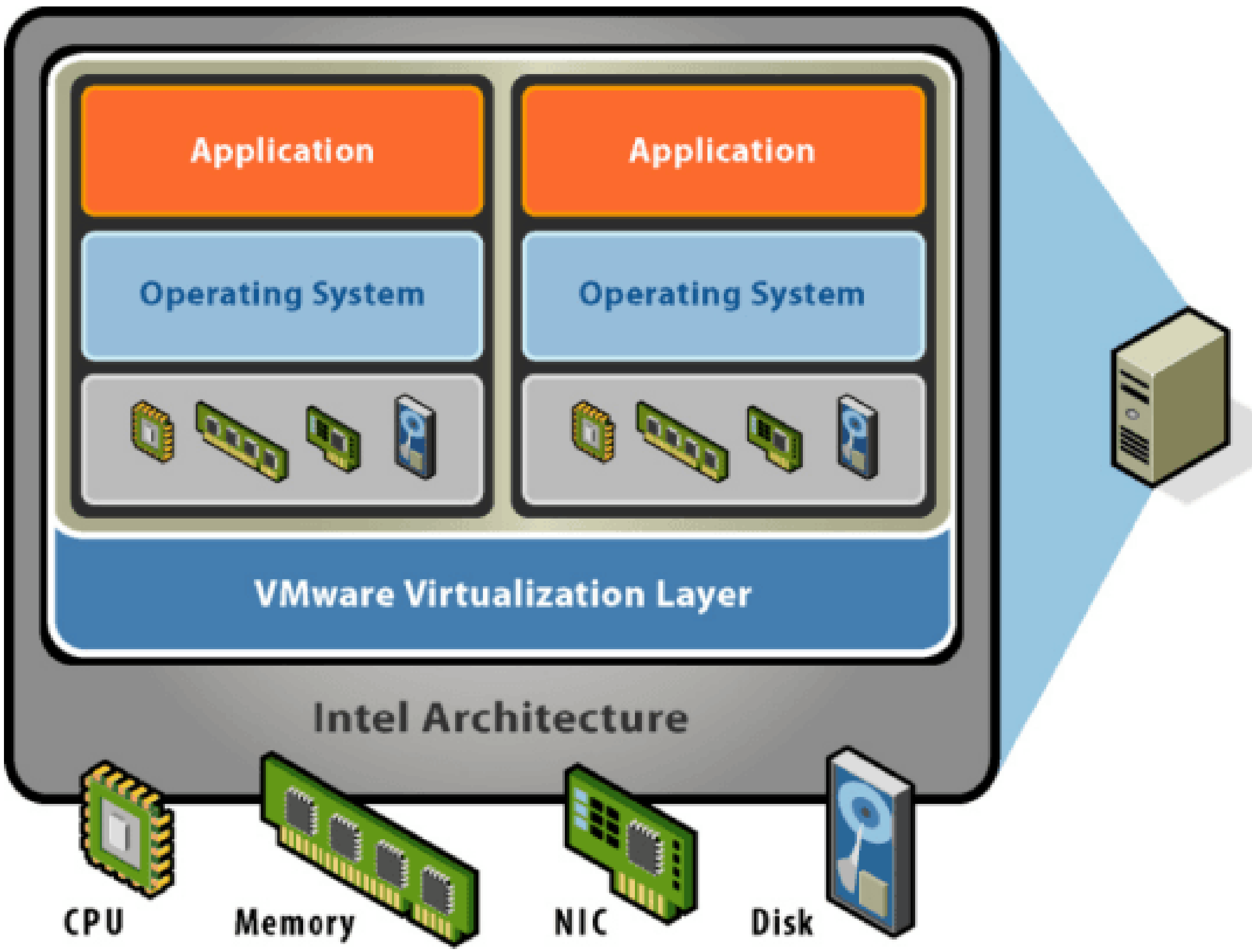


Presented as multiple devices controlled by PCI-e virtual functions (VFs)

Modern Hypervisors

VMWare ESX/ESXi

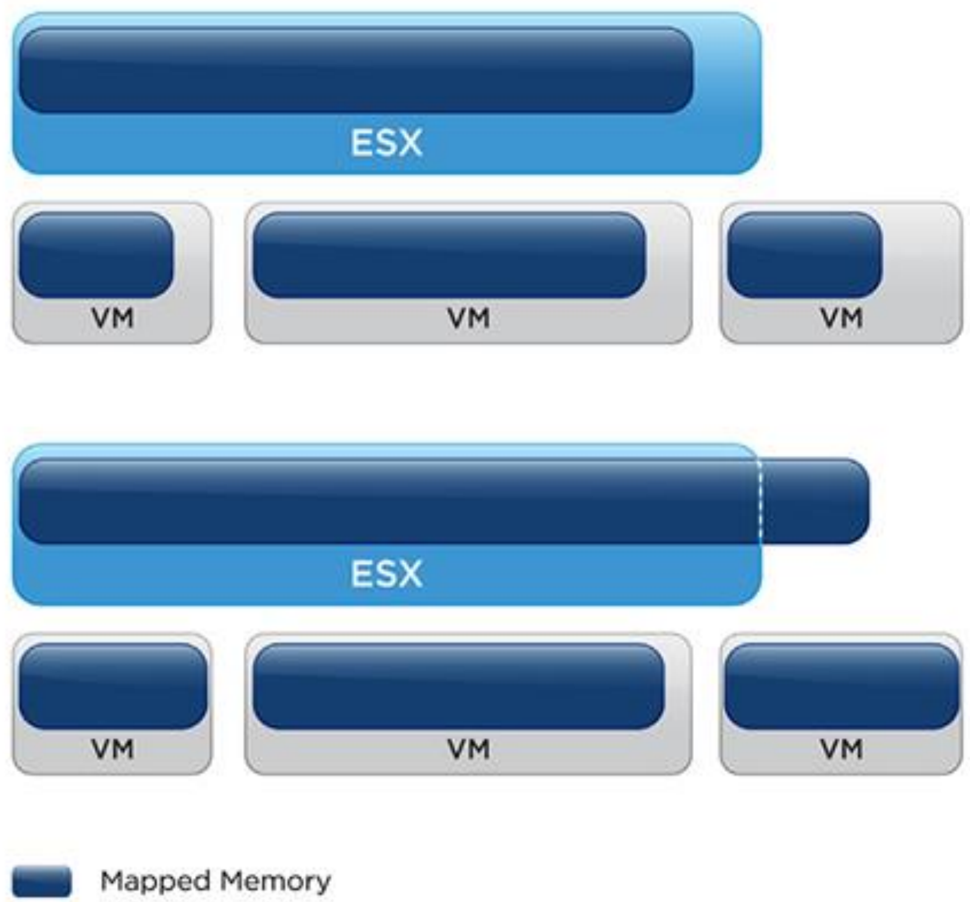
"Bare-metal Hypervisor"



Memory Overcommitment

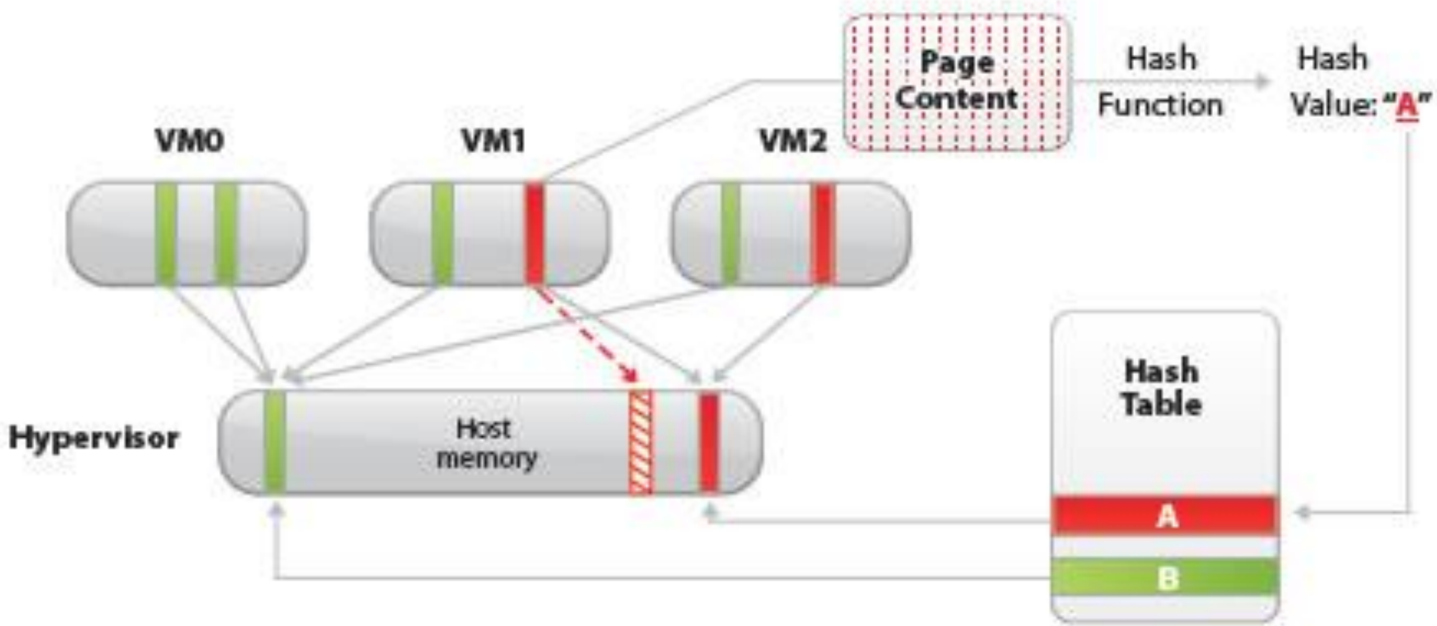
Not overcommitted:

Overcommitted:



Memory Sharing

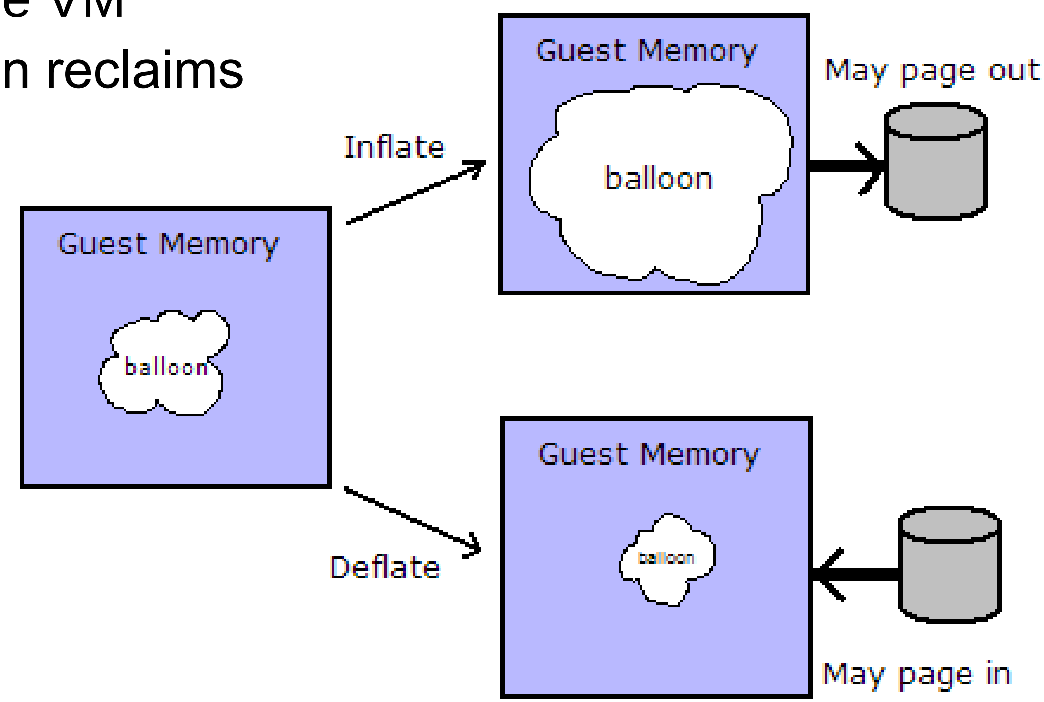
Same paging merging:
If VM1 and VM2 contain pages with exactly same contents, merge them into one physical page.



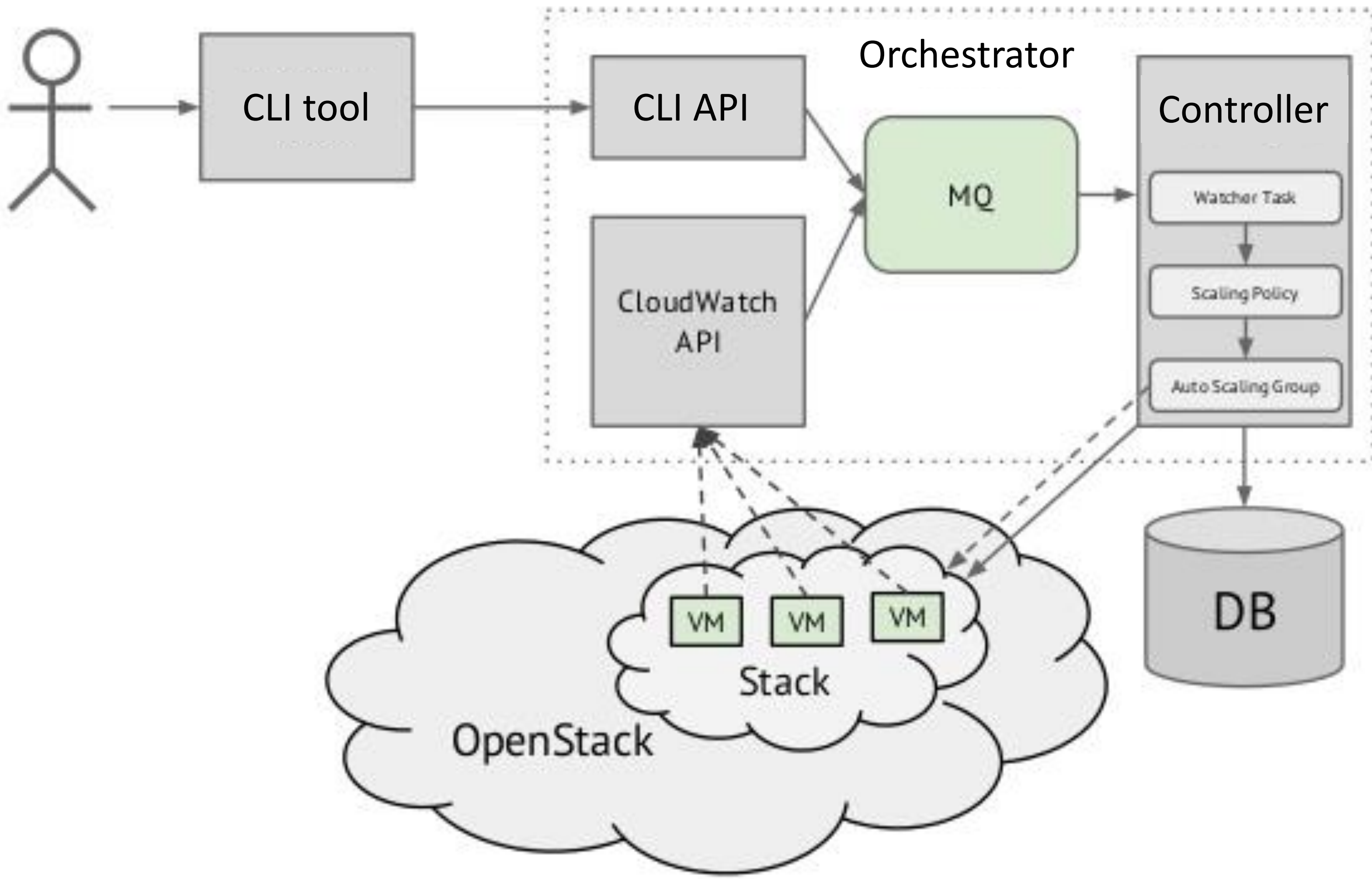
Ballooning

<u>Reclaim memory from VMs</u>

- A balloon module is loaded into the guests
- The balloon works on pinned physical pages in the VM
- "Inflating" the balloon reclaims memory



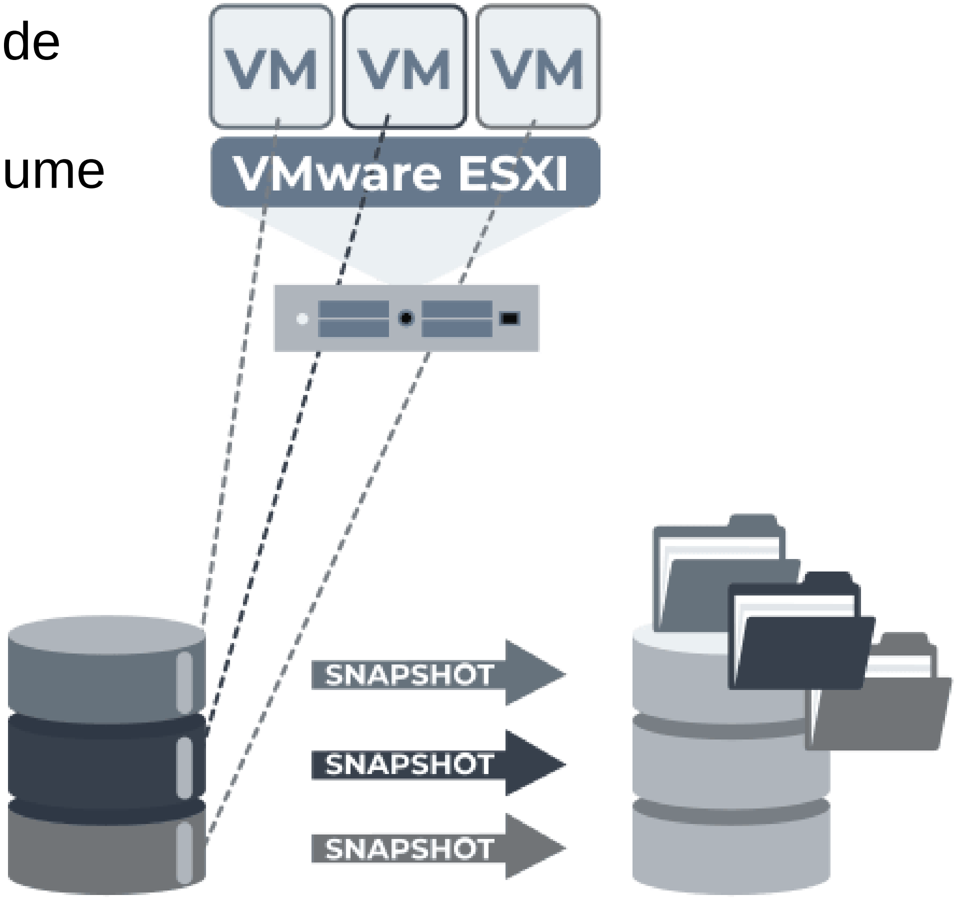
VM Orchestration



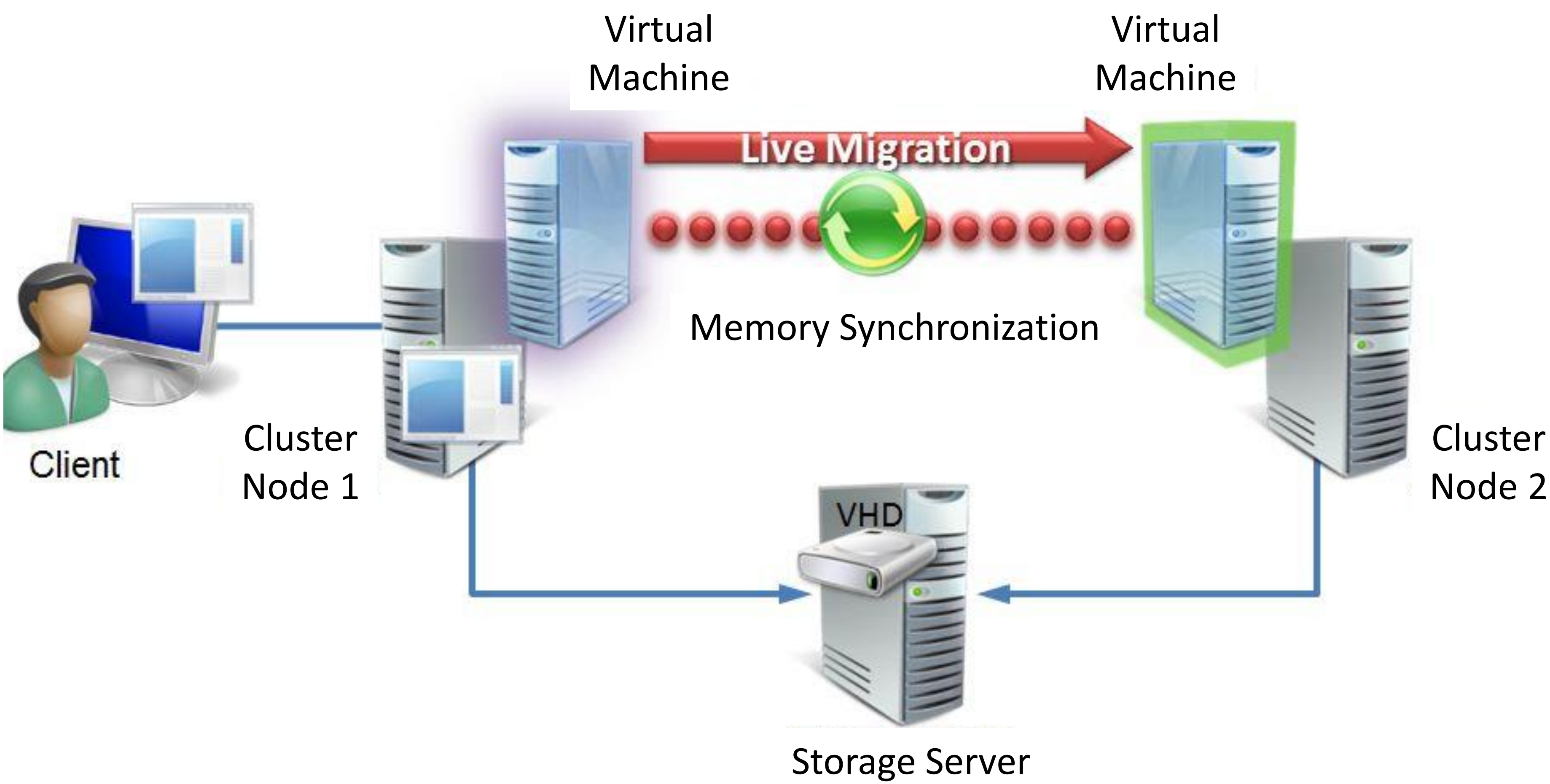
Snapshot/Checkpoint/Migration

Snapshot

- Save the state of VM inside the disk
- Easily restore later to resume the prior execution



Live Migration



VM Introspection