Virtualization and Xen

Dazhao Cheng

Outline

- Motivation and introduction
- Example: Xen
  - techniques
  - Evaluation

What is virtualization

- Partitioning one physical server to multiple virtual servers
  - Virtual machines are isolated
  - One VM failure will not affect the others
- Hypervisor software is the key
  - Or called virtualization manager
  - The layer between the hardware/OS and virtual machines
  - Manages the partitioning and isolation of system resources

Broader concept of virtualization

- Combine multiple physical resources into one single virtual resource
  - Storage virtualization
- Application virtualization: JVM, .Net
- Network virtualization
- Desktop virtualization

Benefits

- **Save money.**
  - Many companies require one app on one machine for reliability
- **Save energy**
  - Less physical servers, less energy consumption
- **Save time**
  - Deploy, setup, startup quickly
- **Agile development**
  - Developer can use multiple virtual OSes to simulate and test cross-platform software

History

- Introduced by IBM in the 1960s
  - To boost utilization of large, expensive mainframe systems
- Gave away to C/S in 80s and 90s
- Become hot again
  - Servers are cheap and powerful
  - Become the key component of cloud computing

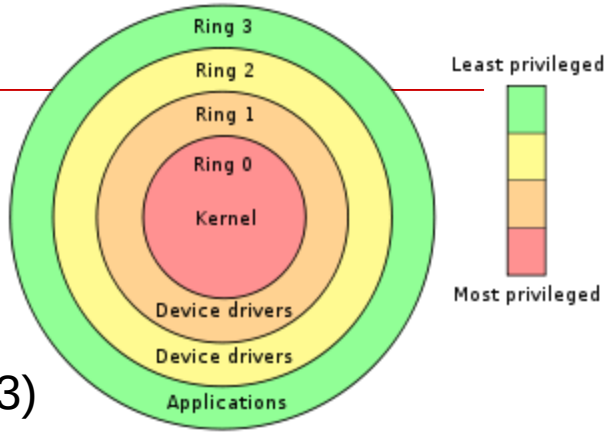
Basic ideas

- Virtualize resources
  - CPU
  - Memory
  - Network
  - Disk

- Key: the layer between hardware and guest OSs – hypervisor software
  - Partitioning, isolating, and scheduling resources between guest Oss

Preliminary (normal OS)

Protection rings



APPS

User space (lower privilege: ring 3)

System call/ trap

Kernel space (high privilege: ring 0)

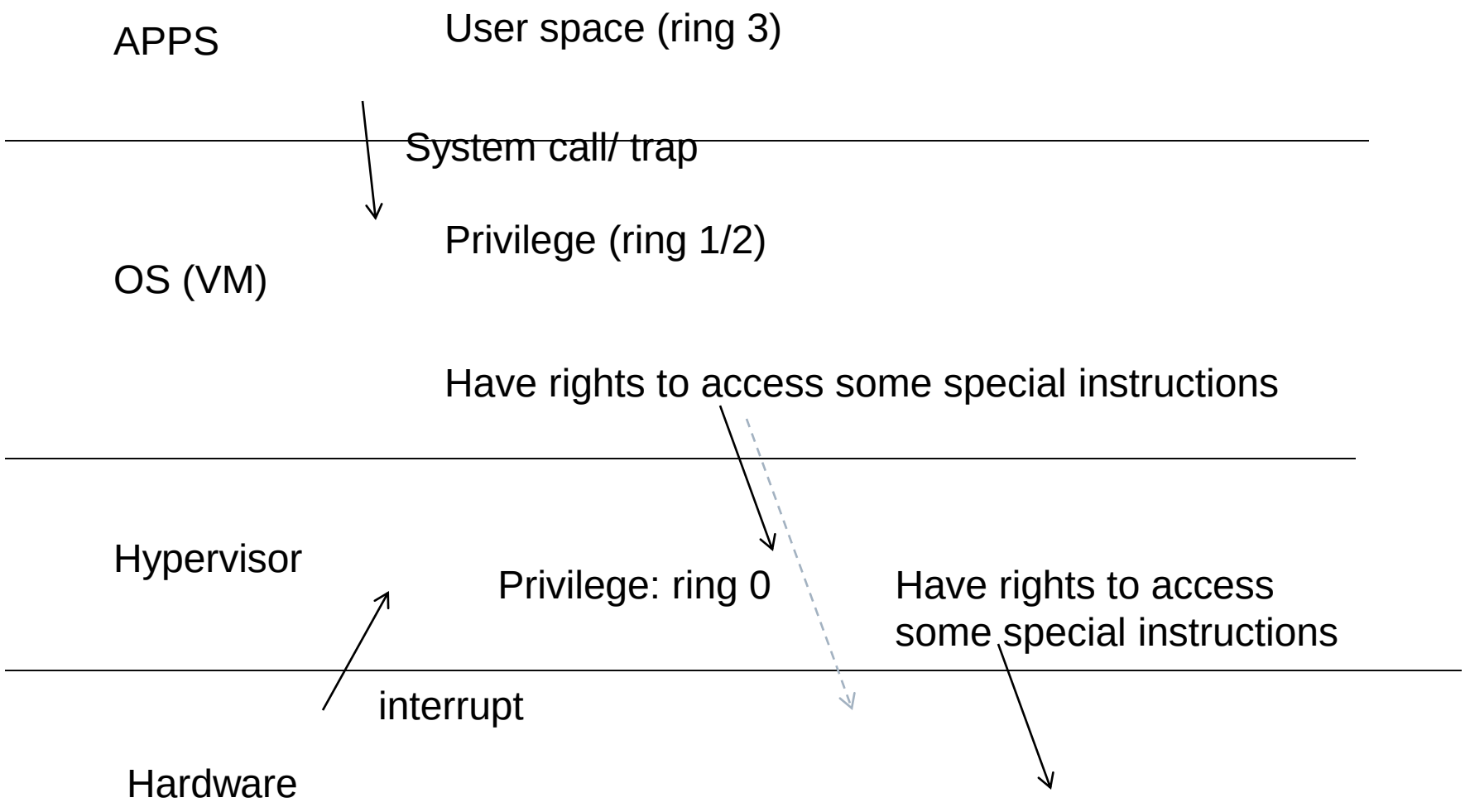
OS
(supervisor mode)

Have rights to access some special CPU instructions

interrupt

Hardware

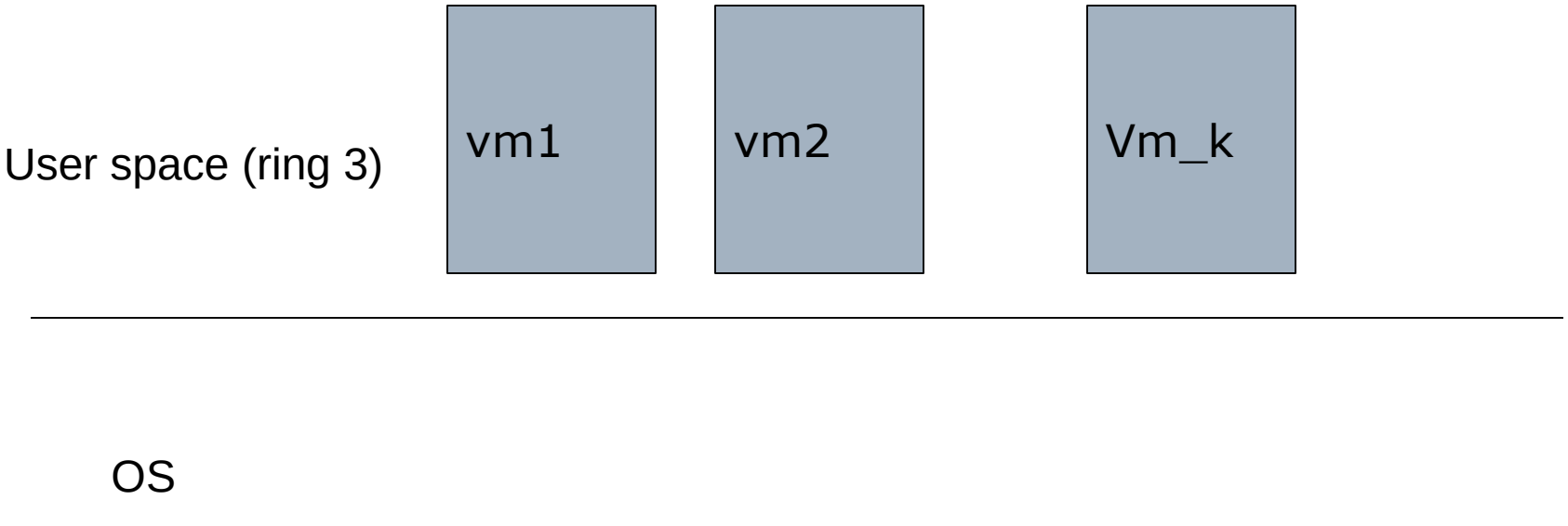
x86 virtualization



Types of virtualization

- Container virtualization
- Full virtualization
- Para-virtualization

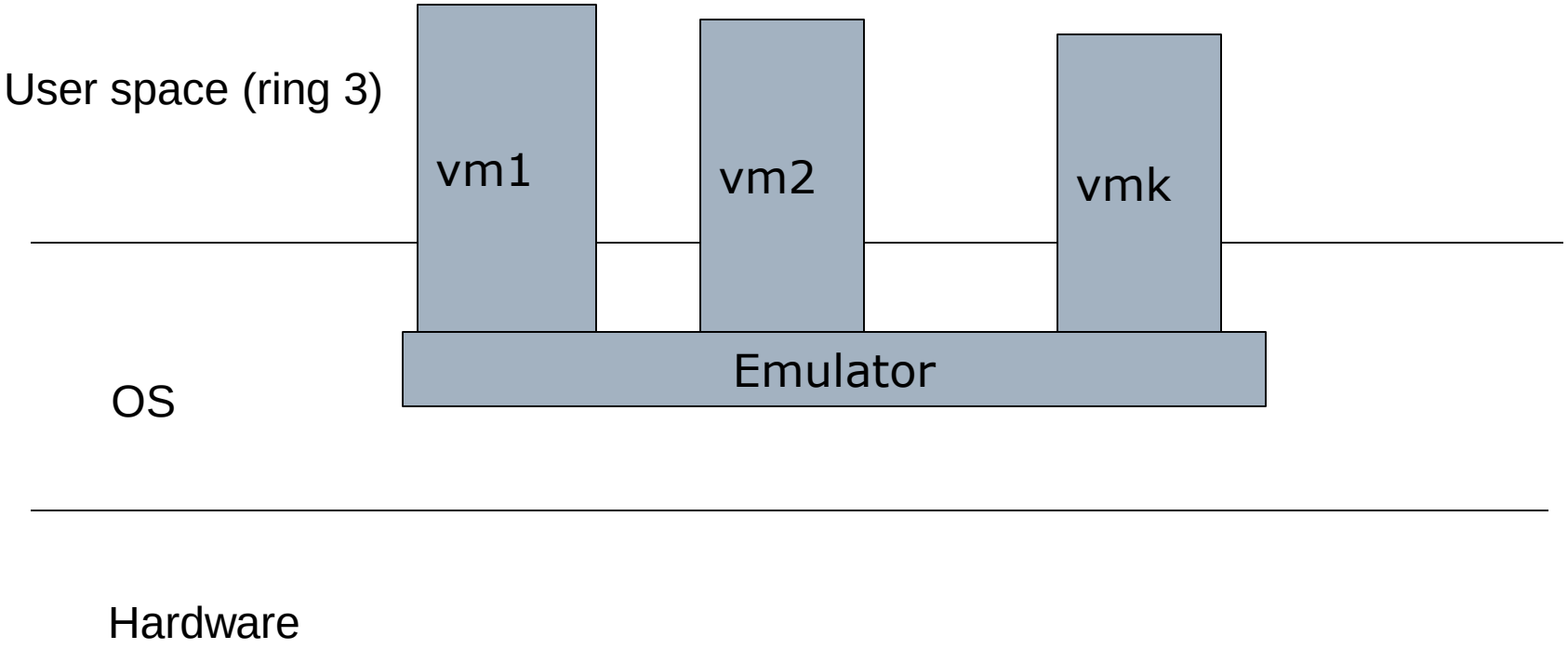
Container virtualization



Container virtualization

- User-space virtual machines
- All guests share the same filesystem tree.
- Same kernel on all virtual machines
- Unprivileged VMs can’t mount drives or change network settings
- Provide extra-level of security
- Native Speeds, no emulation overhead
- OpenVZ, Virtuozzo, Solaris Containers, FreeBSD Jails, Linux-Vserver

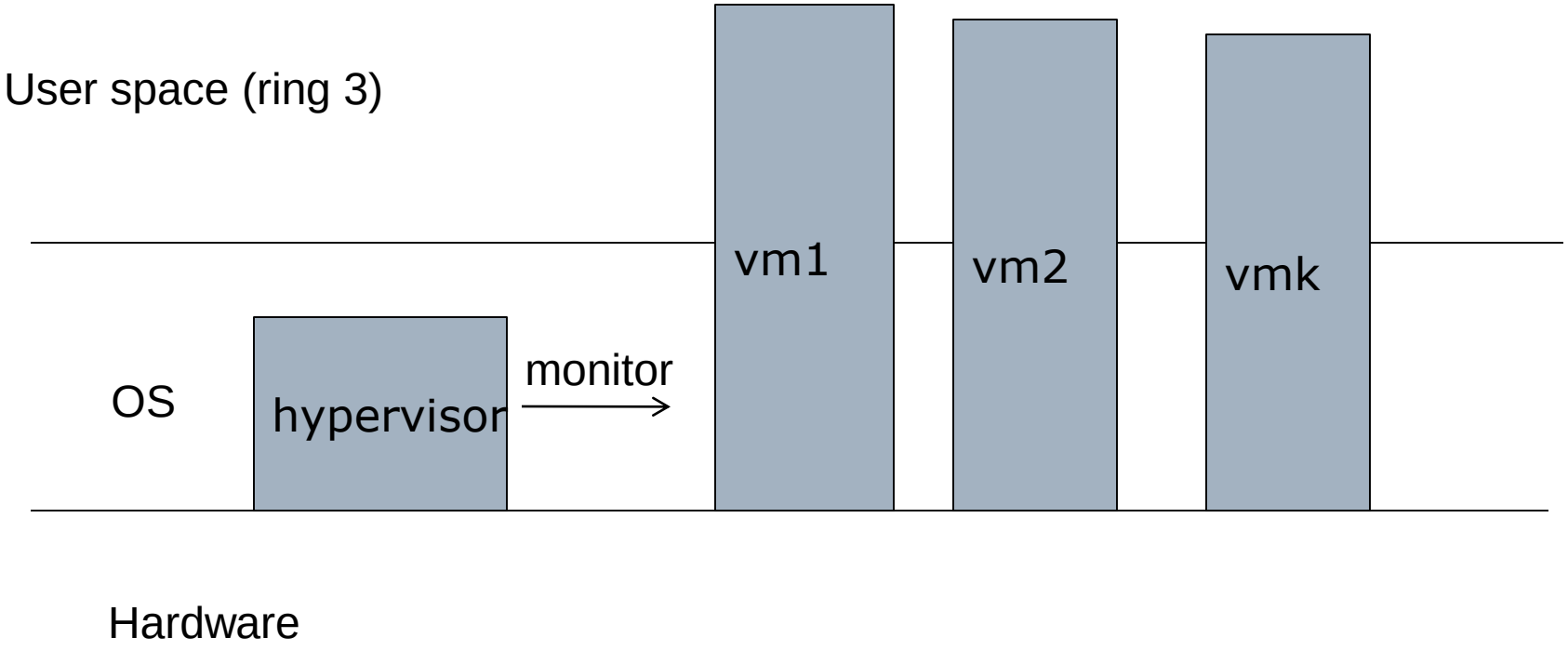
Full virtualization



Full virtualization

- Runs **unmodified** guests
- Simulates bios, communicates with VMs through ACPI emulation, BIOS emulation, sometimes custom drivers
  - Guests cannot access hardware
- Generally worst performance, but often acceptable
- VMWare, Xen HVM, KVM, Microsoft VM, Parallels, virtualbox

Paravirtualization



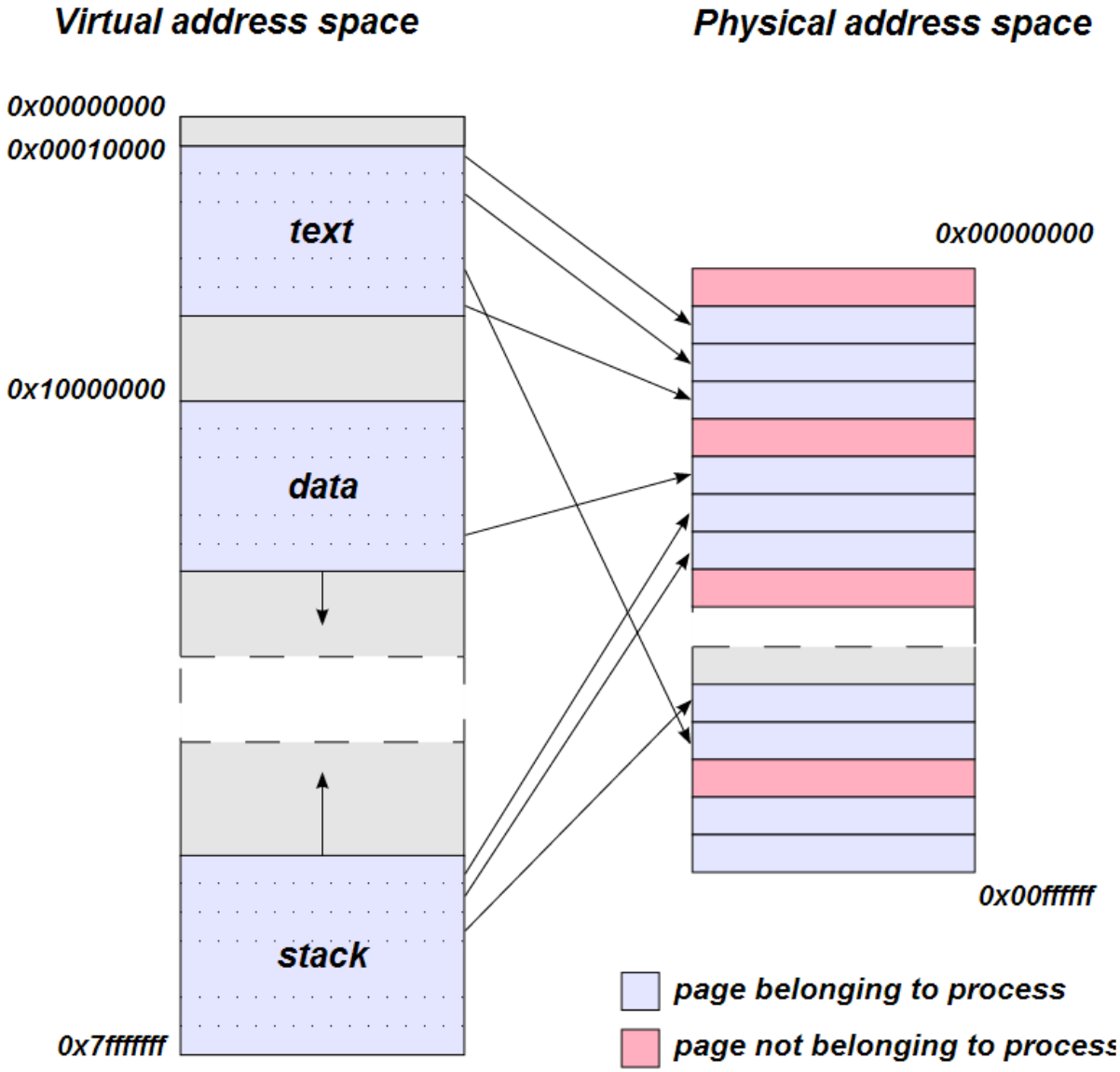
Paravirtualization

- Do not try to emulate everything
  - Work as a guard
  - Pass safe instructions directly to CPU and device
  - Guests have some exposure to the hardware
- Better performance
- Need to slightly modify guest OS, but no need to modify applications
- Xen, Sun Logical Domains

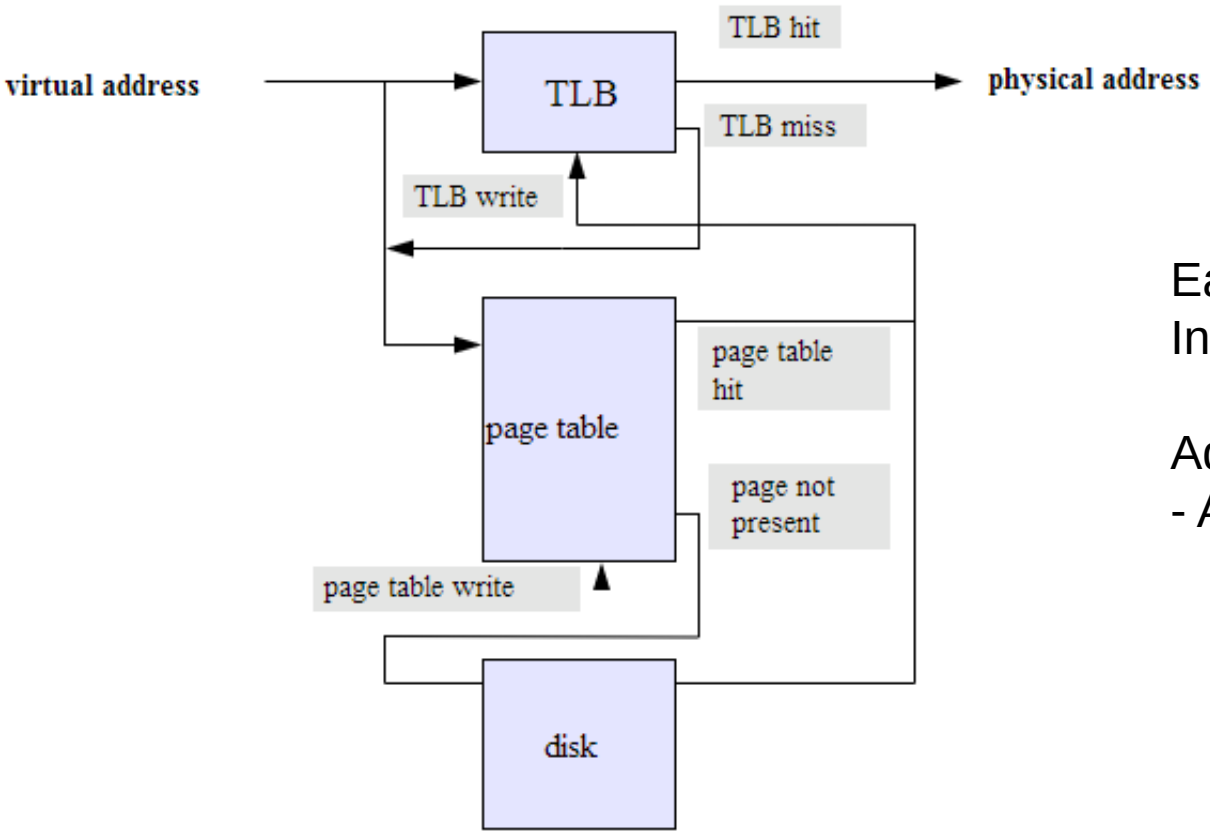
Xen: introduction

- Paravirtualization
- Faster than full virtualization
- Need to slightly change some guest OS
- Domain (1-) : guest OS

virtual memory management



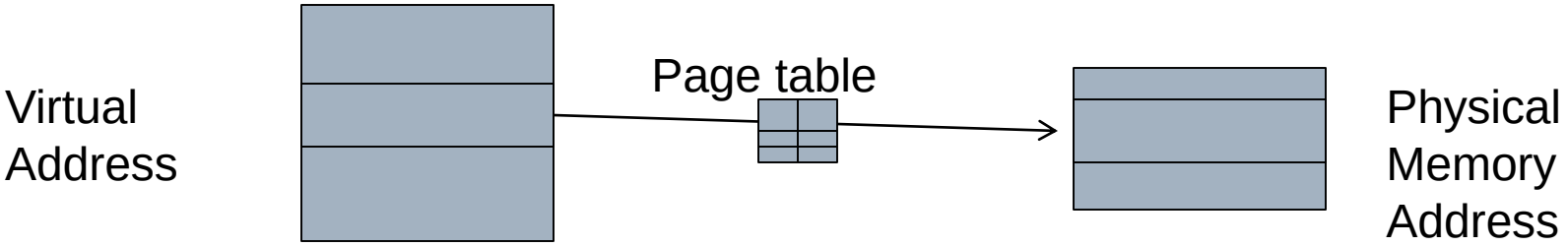
Translation



Each context switch needs to Invalidate TLB – TLB flushing

Add a tag to TLB. No need to flush
- Address Space ID (8bits)

Xen: virtual memory management



- TLB(translation lookaside buffer) flushing
  - CPU cache of page table entrieS
  - X86 needs TLB flushing for context switching
- To avoid TLB flushing
  - Updates are batched and validated by the hypervisor
  - Xen exists in a 64MB session at the top of every address space

- Minimize complexity
  - Let guest OSes allocate and manage the hardware page tables
  - Minimal involvement to ensure safety and isolation

Xen: memory allocation

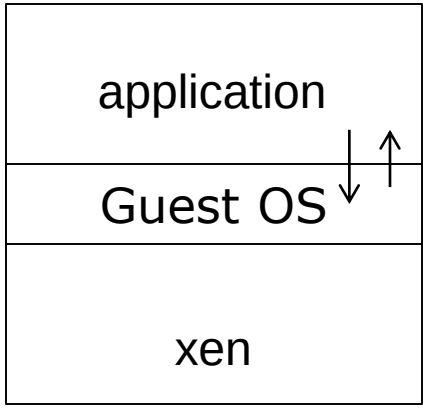
- At the beginning of creating guest OS
  - A fixed amount of physical memory is allocated (*reservation*)
  - Claim additional memory from Xen, when needed; release memory to Xen after finish
- Allocated memory are not contiguous
  - "Physical memory" a virtual view of contiguous memory by guest OS
  - "hardware memory": real physical memory
  - Guest OS builds a map between physical memory and hardware memory

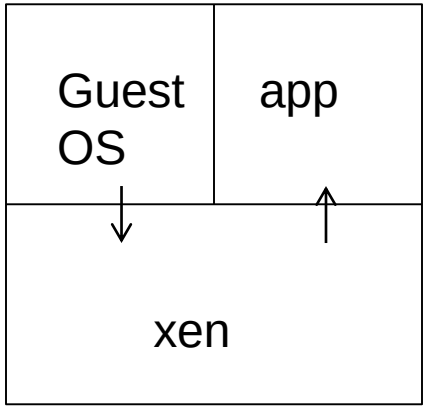
When start a new process

- Guest OS requires a new page table
- Allocates and initializes a page from its own memory reservation and register it with Xen
- Relinquish write privileges to the page-table memory – all updates must be validated by Xen

Xen: CPU scheduling

- Guest OS runs at a lower privilege level than Xen
- Guest OS must register exception (trap) handlers with Xen
  - Xen will check the handler
  - Page fault is handled differently
- System calls : no Xen intervention
- Use a lightweight event system to handle hardware interrupts





More than two privilege levels

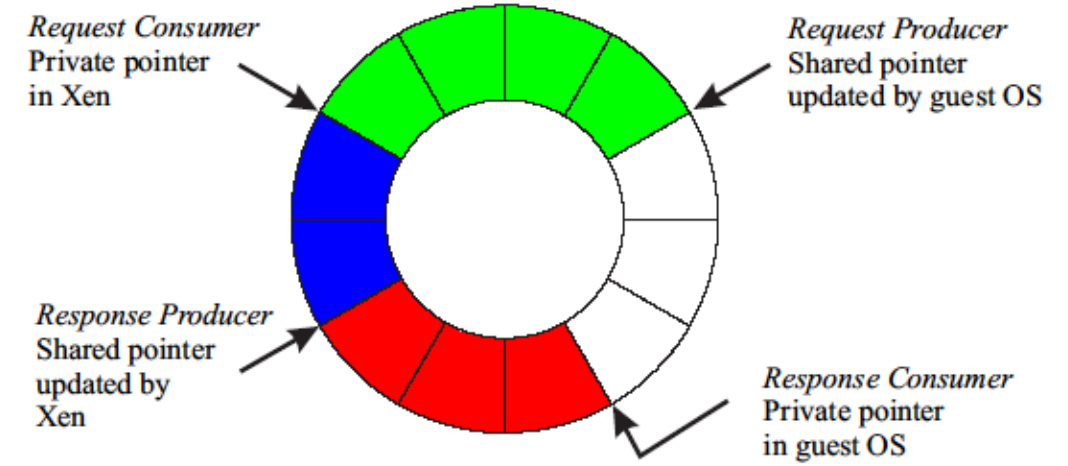
only two privilege levels for some processors

X86 provides 4 levels of privilege – rings
Xen at ring 0, guest OS at ring 1, apps at ring 3

- Two types of frequent exception
  - System calls
  - Page faults
- Improve performance of system calls
  - A fast exception handler accessed directly by the processor without via ring 0; validated before installing it in the hardware exception table
  - Validation: check the handler's code segment – no execution in ring 0

Xen: device I/O

- Events: asynchronous notifications from Xen to domains
  - Allocated by the domain; replace device interrupts
  - Guest OS manages data buffers



Xen: device I/O

- Only Domain0 has direct access to disks
- Other domains need to use virtual block devices
  - Use the I/O ring
  - Reorder requests prior to enqueuing them on the ring
  - use DMA (zero copy)

Xen: network

- Virtual firewall-router attached to all domains
- To send a packet, enqueue a buffer descriptor into the I/O ring
- Use DMA (no packet copying)

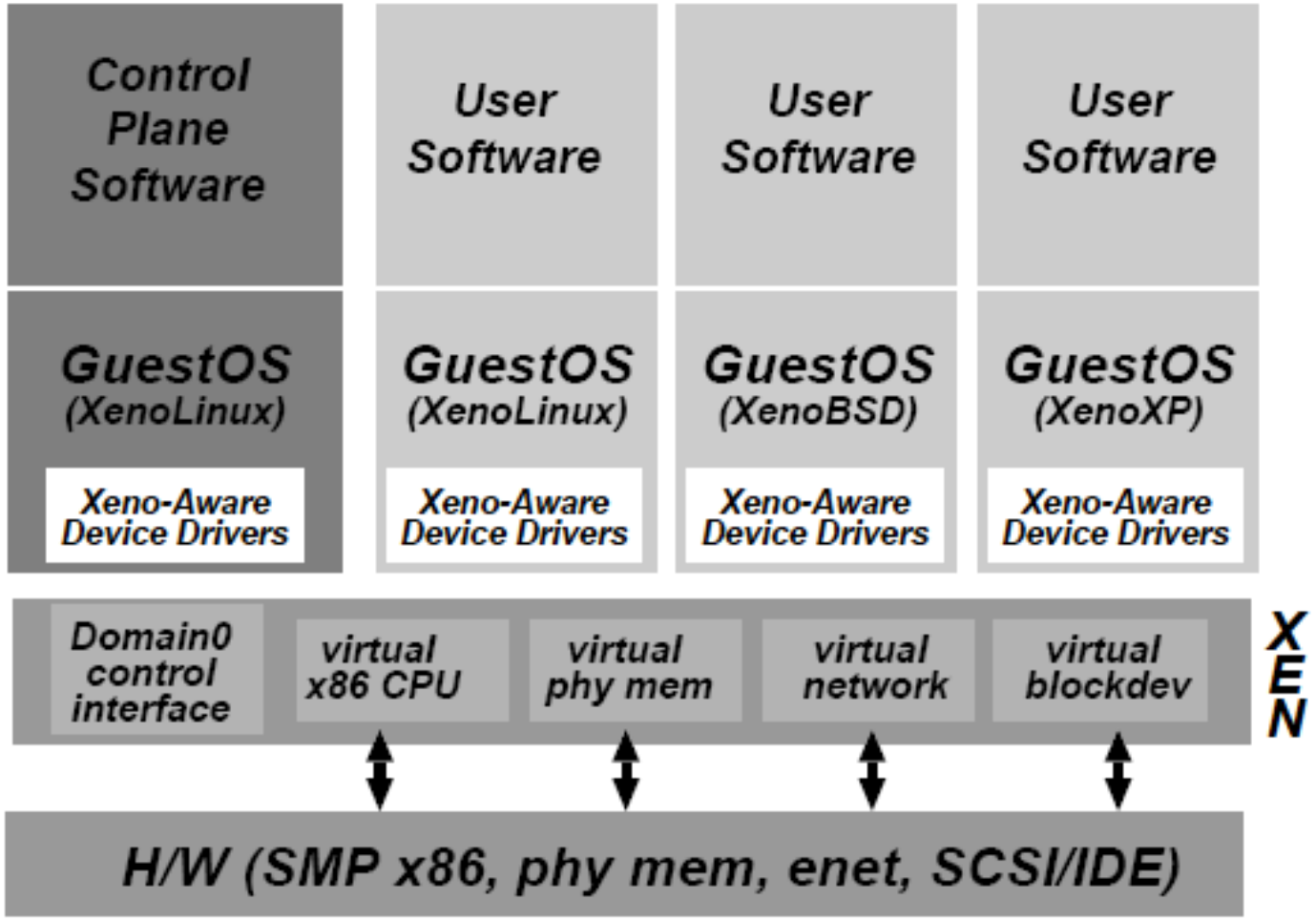
Partitioning resources between guest OSes

- Memory- preallocated physical memory
- Disk – quota
- CPU and network
  - Involves more complicated procedures

Domain 0

- The representative to the Xen hypervisor
- Provide bootstrap code for different types of VMs
- Creating/deleting virtual network interfaces and virtual block devices for other domains

System looks like



Cost of porting a guest OS to Xen

OS subsection	# lines	
	Linux	XP
Architecture-independent	78	1299
Virtual network driver	484	–
Virtual block-device driver	1070	–
Xen-specific (non-driver)	1363	3321
Total	**2995**	**4620**
(Portion of total x86 code base	**1.36%**	**0.04%)**

Table 2: The simplicity of porting commodity OSes to Xen. The cost metric is the number of lines of reasonably commented and formatted code which are modified or added compared with the original x86 code base (excluding device drivers).

Linux kernel 2.4

Issues

- Performance isolation vs. maximizing overall system utilization
  - Easy to partition memory and disk
  - Not easy to partition CPU and network
    - Time issue

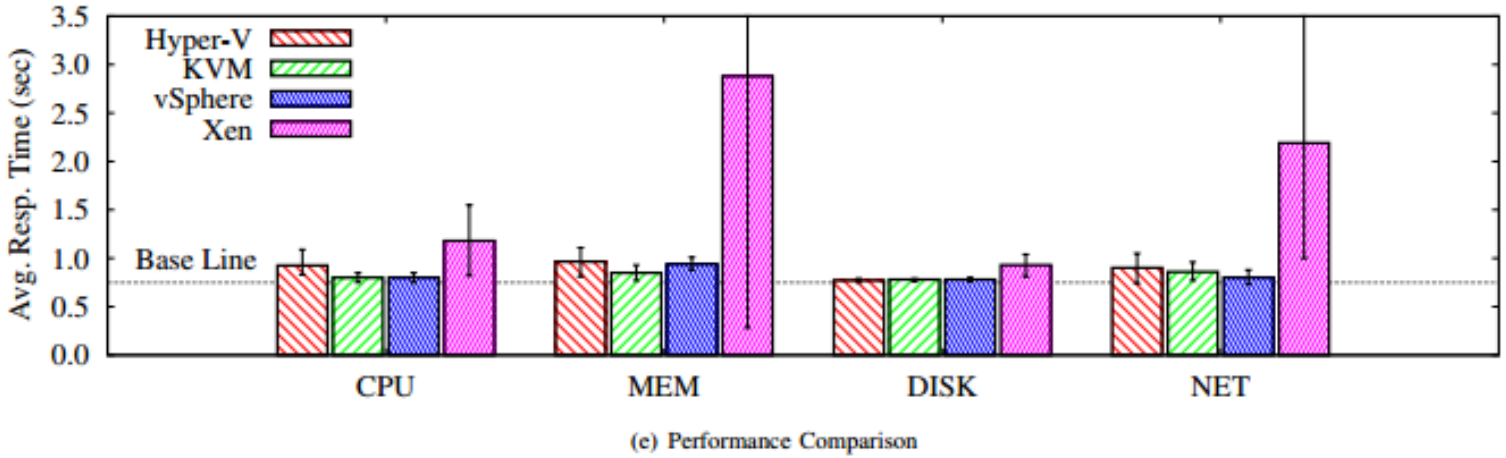
Recent development

- Kernel based virtual machine (KVM)
  - A part of the linux kernel (vs. Xen as a standalone hypervisor
  - 2008 result

Table 1. Overall performance of base Linux, Xen, and KVM

	Linux	Xen	KVM
CPU	1.000	0.999	0.993
Kernel Compile	1.000	0.487	0.384
IOzone Write	1.000	0.855	0.934
IOzone Read	1.000	0.852	0.994

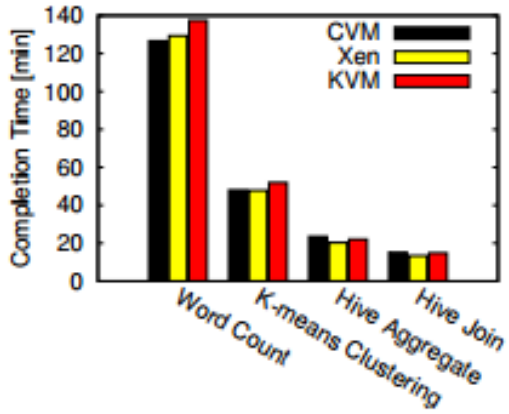
2013



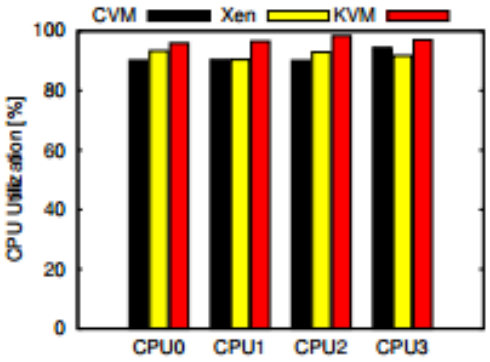
(e) Performance Comparison

Fig. 9. Interference Impact for Web Requests: 4 VMs (1 web server, 3 workload generators) are used. 3 VMs run the same workload at the same time. The workloads run in the sequence of CPU, memory, disk, and network workloads over time span. We can easily identify 4 interference sections from each graph.

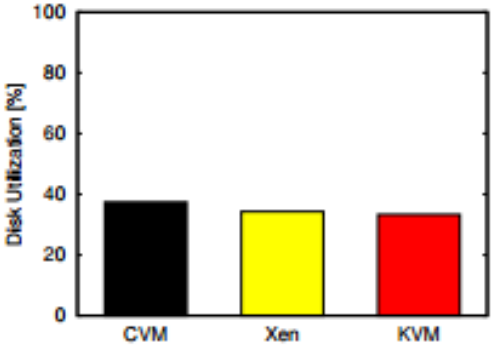
Hadoop workloads (2013)



(a) CPU-Bound Benchmark Completion Times. The performance difference is small.



(b) Word Count Average CPU at VM Level. The CPU for each VM is heavily saturated.



(c) Word Count Average Disk Utilization at VM Level. The disk utilization for the VM is low.

Fig. 1: CPU-Bound Benchmark Results and Word Count Statistics. The performance difference for these benchmarks as seen in Figure 1(a) is negligible between the different hypervisors. A representative benchmark, Wordcount, shows high CPU utilization and low disk utilization during the job as seen in Figure 1(b) and 1(c).

Conclusion

- Xen is a complete and robust GPL VMM
- Outstanding performance and scalability
- Excellent resource control and protection
- Linux 2.6 port required no modifications to core code*