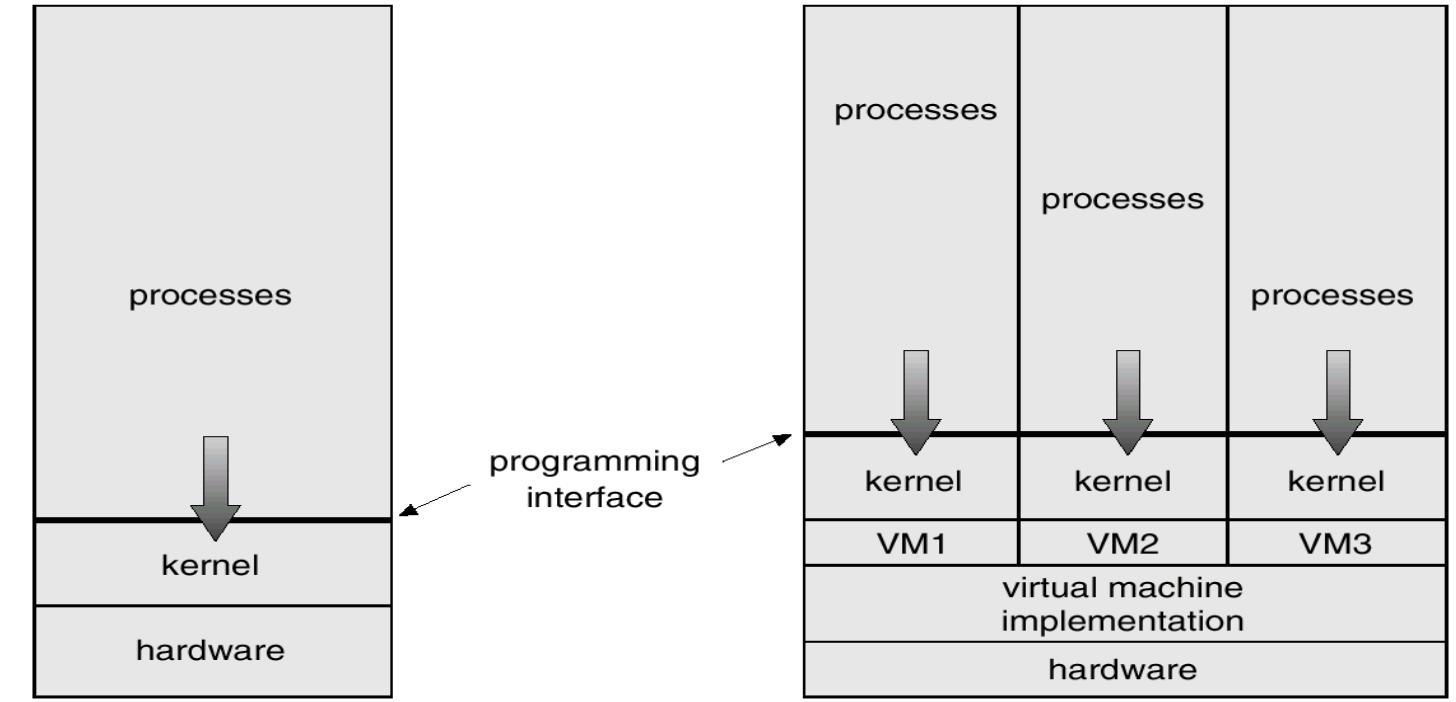
Virtual Machines

Virtual Machines

- *Virtual machine* (VM)
  - Provides *identical or similar* interface as underlying hardware in software
- OS
  - Creates the illusion of multiple processors
  - Each OS with own processor and (virtual) memory
- Run multiple VMs on one physical host
  - CPU scheduling -> users have their own processor
  - Spooling and a file system -> virtual input and output

System Models



Non-virtual Machine

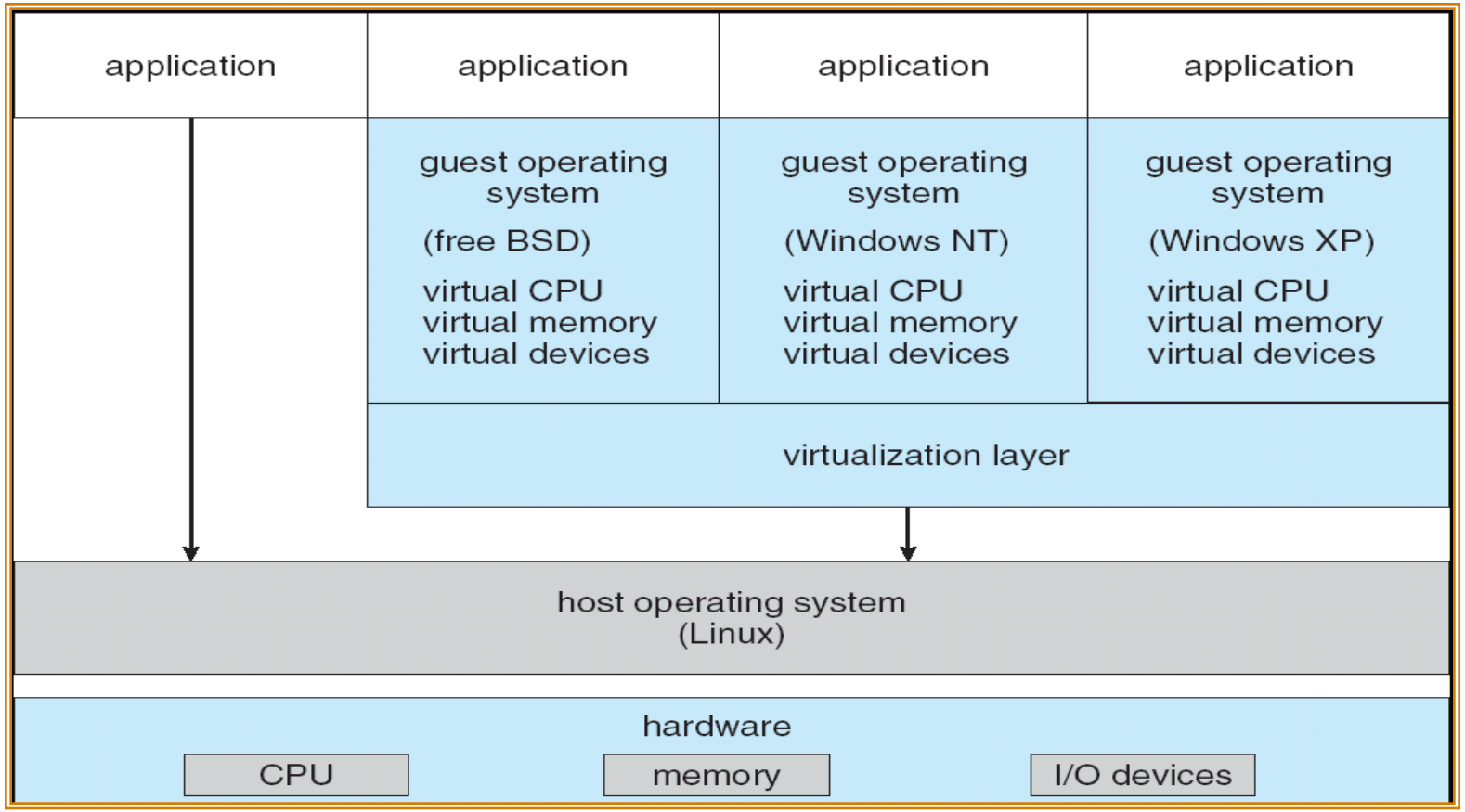
Virtual Machine

Advantages and disadvantage of VMs

- Isolation
  - Good: VMs don't adversely affect each other
  - Bad: no sharing of resources between VMs (cross "machine" boundary)
- Research & development
  - Good: Easy to develop & test kernel
- Implementation
  - Bad: hard
- Performance
  - Bad: some cost (overhead)

VMware Architecture

- Full Virtualization



KVM

- Kernel based virtual machine
  - Full virtualization solution for Linux on x86
  - Requires Intel VT or AMD-V hardware support
  - Consists of a loadable kernel module, kvm.ko

Full Virtualization

- Pros
  - Easy portability
  - Good isolation
- Cons
  - High cost
  - Implementation is complex

Xen and the Art of Virtualization

Barham et al. SOSP 2003

Design Principles

- Support unmodified application binary interface
- Support full multi-task Operating Systems
- Para-virtualization
  - Good tradeoff between performance and isolation
    - For example, virtualizing page tables can result in many expensive traps
  - Complete virtualization of guest OS risks correctness and performance
    - For example, the VM need to know real time (and not just virtual time) to handle things like time out

Xen

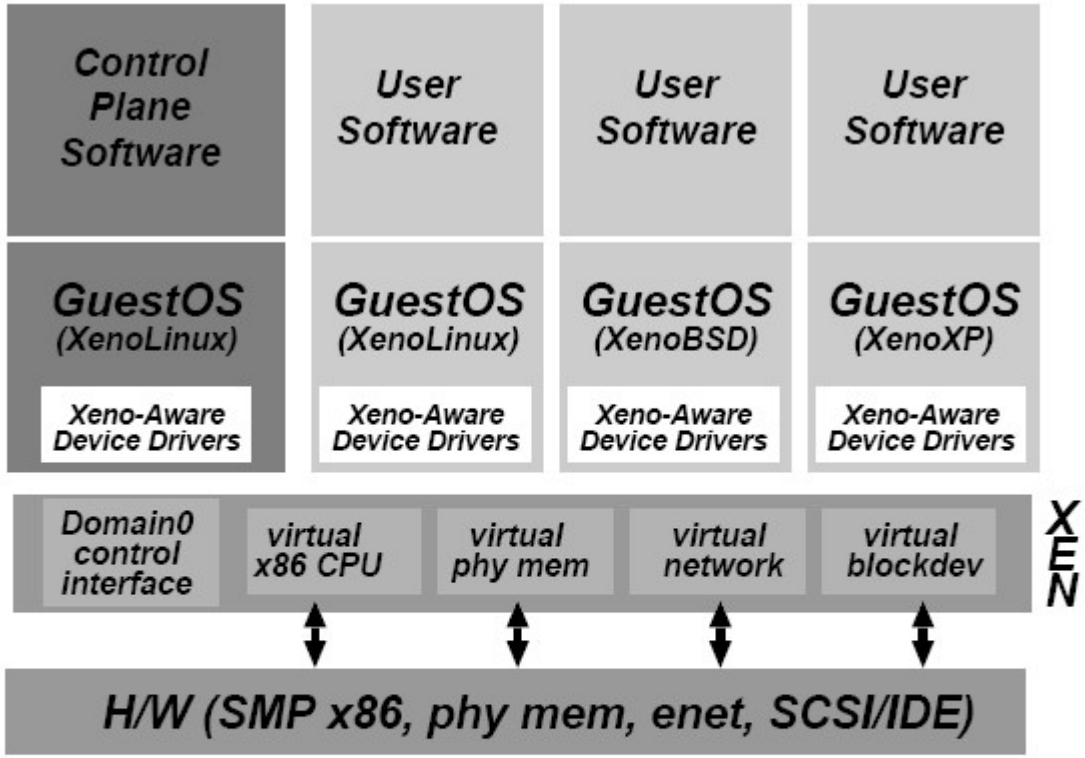
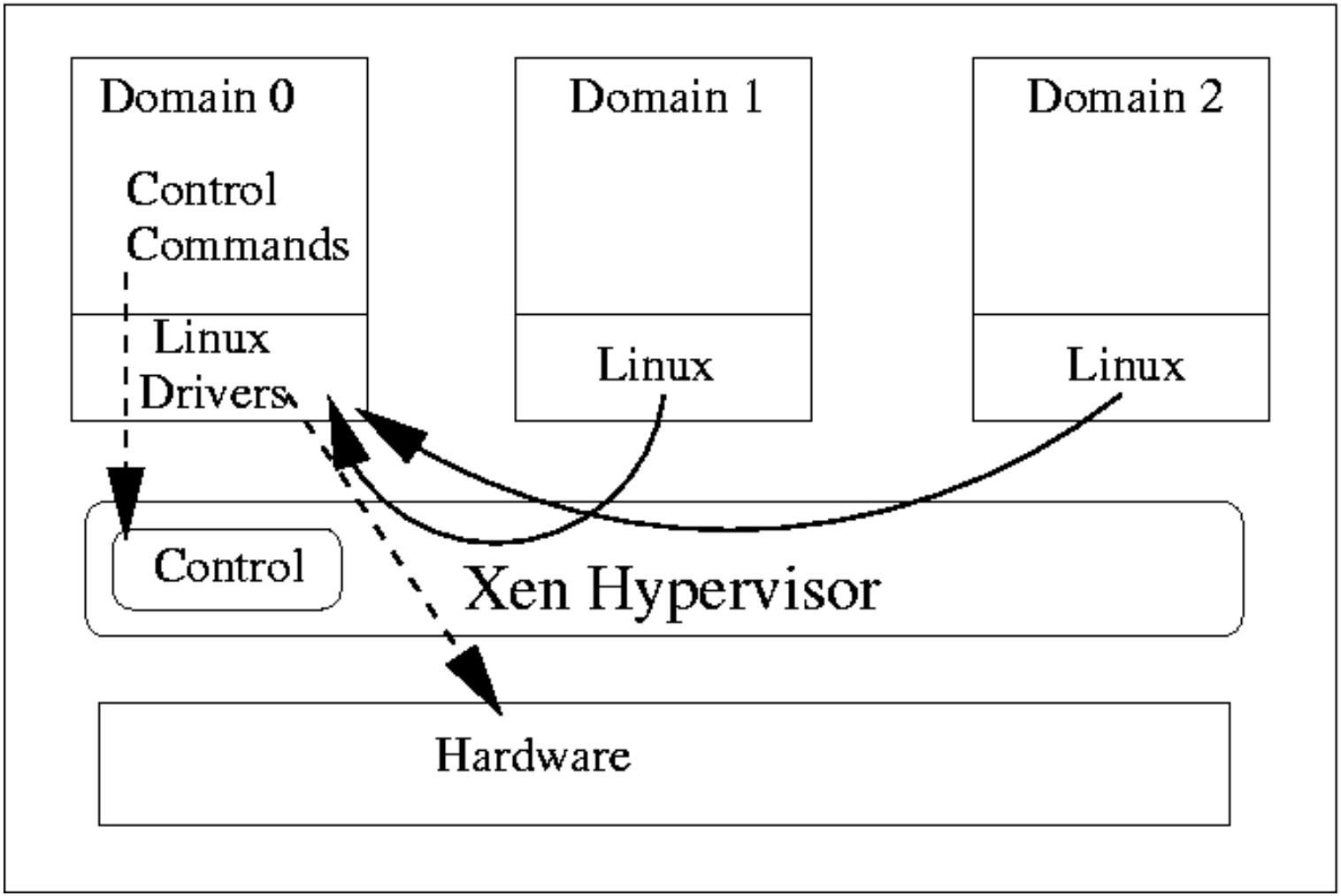


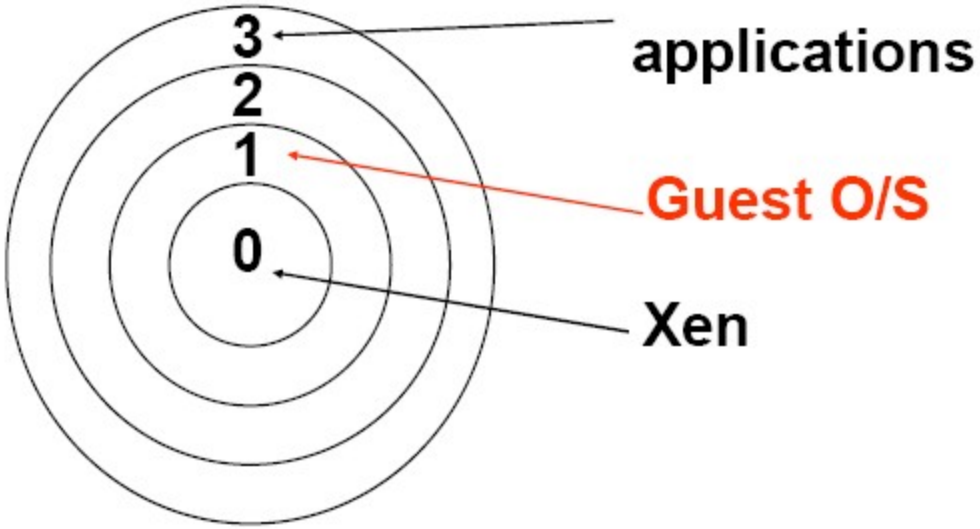
Figure 1: The structure of a machine running the Xen hypervisor, hosting a number of different guest operating systems, including *Domain0* running control software in a XenoLinux environment.

Xen Architecture



Privilege Rings

- Protection: leverages availability of multiple “rings”
  - Intermediate rings have not been used in practice since OS/2; x86-specific
  - An O/S written to only use rings 0 and 3 can be ported; needs to modify kernel to run in ring 1



Xen VM interface: Memory

- Memory management
  - Guest OS cannot install highest privilege level segment descriptors
  - Top end of linear address space is reserved for Xen
  - Guest OS has direct (not trapped) read access to hardware page tables; writes are trapped and handled by the Xen hypervisor
  - Physical memory presented to guest is not necessarily contiguous

Details: Memory

- TLB: challenging
  - Software TLB can be virtualized without flushing TLB entries between VM switches
  - Hardware TLBs tagged with address space identifiers can also be leveraged to avoid flushing TLB between switches
  - x86 is hardware-managed and has no tags…
- Decisions:
  - Guest O/Ss allocate and manage their own hardware page tables with minimal involvement of Xen for better safety and isolation
  - Xen VMM exists in a 64MB section at the top of a VM's address space that is not accessible from the guest and avoid TLB flush during trap from guest to hypervisor

Details: Memory

- Guest O/S has direct read access to hardware page tables, but updates are validated by the VMM
  - Through “hypercalls” into Xen
  - Also for segment descriptor tables
  - VMM must ensure access to the Xen 64MB section is not allowed
  - Guest O/S may “batch” update requests to amortize cost of entering hypervisor

Xen VM interface: CPU

- CPU
  - Guest runs at lower privilege than VMM
  - Exception handlers must be registered with VMM
  - Fast system call handler can be serviced without trapping to VMM
  - Hardware interrupts replaced by lightweight event notification system
  - Timer interface: both real and virtual time

Details: CPU

- Frequent exceptions:
  - Software interrupts for system calls
  - Page faults
- Allow “guest” to register a ‘fast’ exception handler for system calls that can be accessed directly by CPU in ring 1, without switching to ring-0/Xen
  - Handler is validated before installing in hardware exception table: To make sure nothing executed in Ring 0 privilege.
  - Doesn’t work for Page Fault

Xen VM interface: I/O

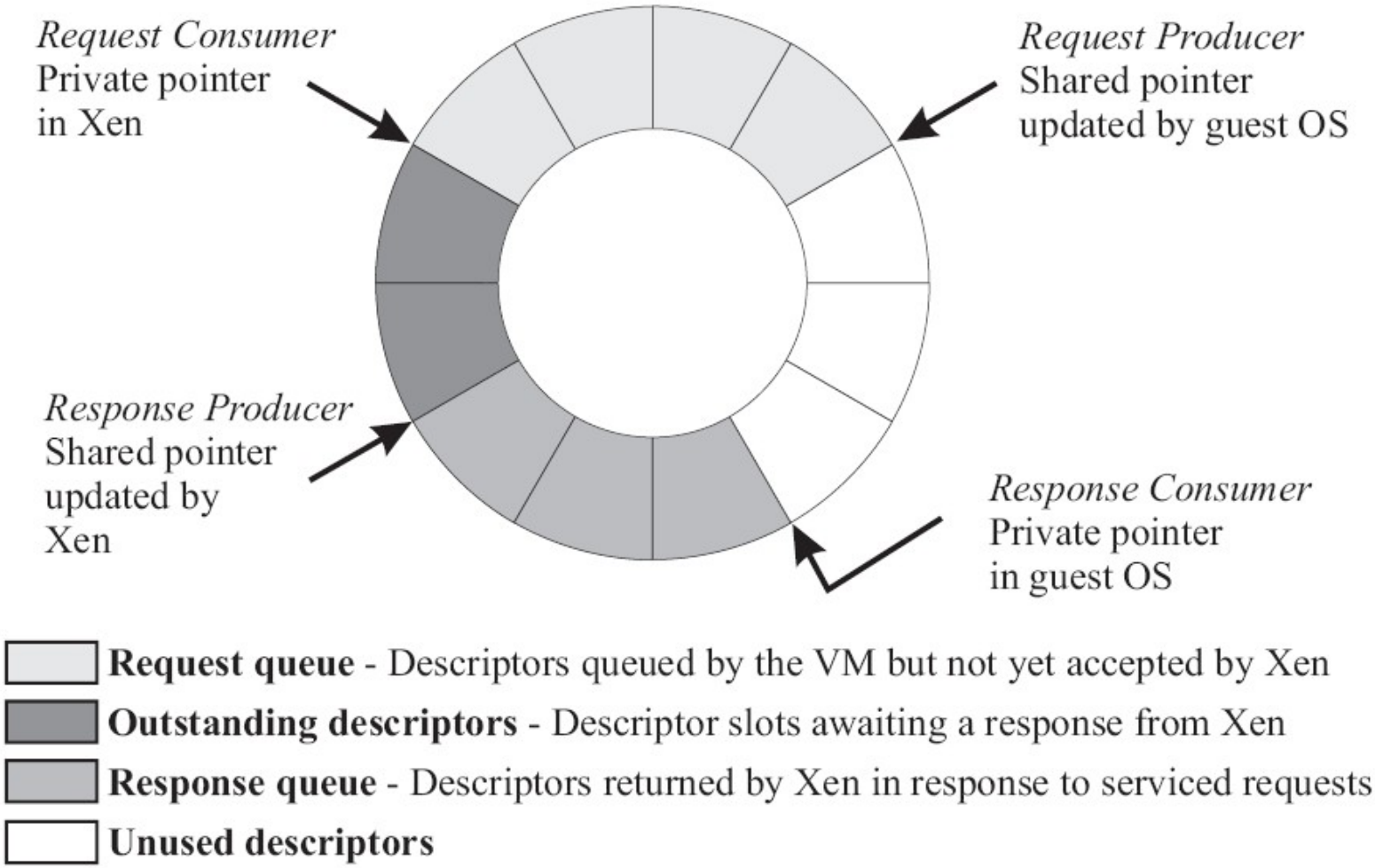
- I/O
  - Virtual devices exposed as asynchronous I/O rings to guests
  - Event notification replaces interrupts

Details: I/O 1

- Xen does not emulate hardware devices
  - Exposes device abstractions for simplicity and performance
  - I/O data transferred to/from guest VM via Xen using shared-memory buffers
  - Virtualized interrupts: light-weight event delivery mechanism from Xen to guest
    - Update a bitmap in shared memory
    - Optional call-back handlers registered by guest OS

Details: I/O 2

- I/O Descriptor Ring:



Control Transfer

- Guest synchronously call into VMM
  - Explicit control transfer from guest O/S to VMM
  - “hypercalls”
- VMM delivers notifications to guest O/S
  - E.g. data from an I/O device ready
  - Asynchronous event mechanism; guest O/S does not see hardware interrupts, only Xen notifications

Event notification

- Pending events stored in per-domain bitmask
  - E.g. incoming network packet received
  - Updated by Xen before invoking guest O/S handler
  - Xen-readable flag may be set by a domain
    - To defer handling, based on time or number of pending requests
    - Analogous to interrupt disabling

Data Transfer: Descriptor Ring

- Descriptors are allocated by a domain (guest) and accessible from Xen
- Descriptors do not contain I/O data; instead, point to data buffers also allocated by domain (guest)
  - Facilitate zero-copy transfers of I/O data into a domain

Network Virtualization

- Each domain has 1+ virtual network interfaces (VIFs)
  - Each VIF has 2 I/O rings (send, receive)
  - Each direction also has rules of the form (<pattern>,<action>) that are inserted by domain 0 (management)
- Xen models a virtual firewall+router (VFR) to which all domain VIFs connect

Network Virtualization

- Packet transmission:
  - Guest adds request to I/O ring
  - Xen copies packet header, applies matching filter rules
    - e.g. change header IP source address for NAT
    - No change to payload; pages with payload must be pinned to physical memory until DMA to physical NIC for transmission is complete
  - Round-robin packet scheduler

Network Virtualization

- Packet reception:
  - Xen applies pattern-matching rules to determine destination VIF
  - Guest O/S required to exchange unused page frame for each packet received
    - Xen exchanges packet buffer for page frame in VIF's receive ring
    - If no receive frame is available, the packet is dropped
    - Avoids Xen-guest copies; requires page aligned receive buffers to be queued at VIF's receive ring

Disk virtualization

- Domain0 has access to physical disks
  - Currently: SCSI and IDE
- All other domains: virtual block device (VBD)
  - Created & configured by management software at domain0
  - Accessed via I/O ring mechanism
  - Possible reordering by Xen based on knowledge about disk layout

Disk virtualization

- Xen maintains translation tables for each VBD
  - Used to map requests for VBD (ID,offset) to corresponding physical device and sector address
  - Zero-copy data transfers take place using DMA between memory pages pinned by requesting domain
- Scheduling: batches of requests in round-robin fashion across domains

OS Porting Cost

- Number of lines of code modified or added compared with original x86 code base (excluding device drivers)
  - Linux: 2995 (1.36%)
  - Windows XP: 4620 (0.04%)
- Re-writing of privileged routines;
- Removing low-level system initialization code

Evaluation

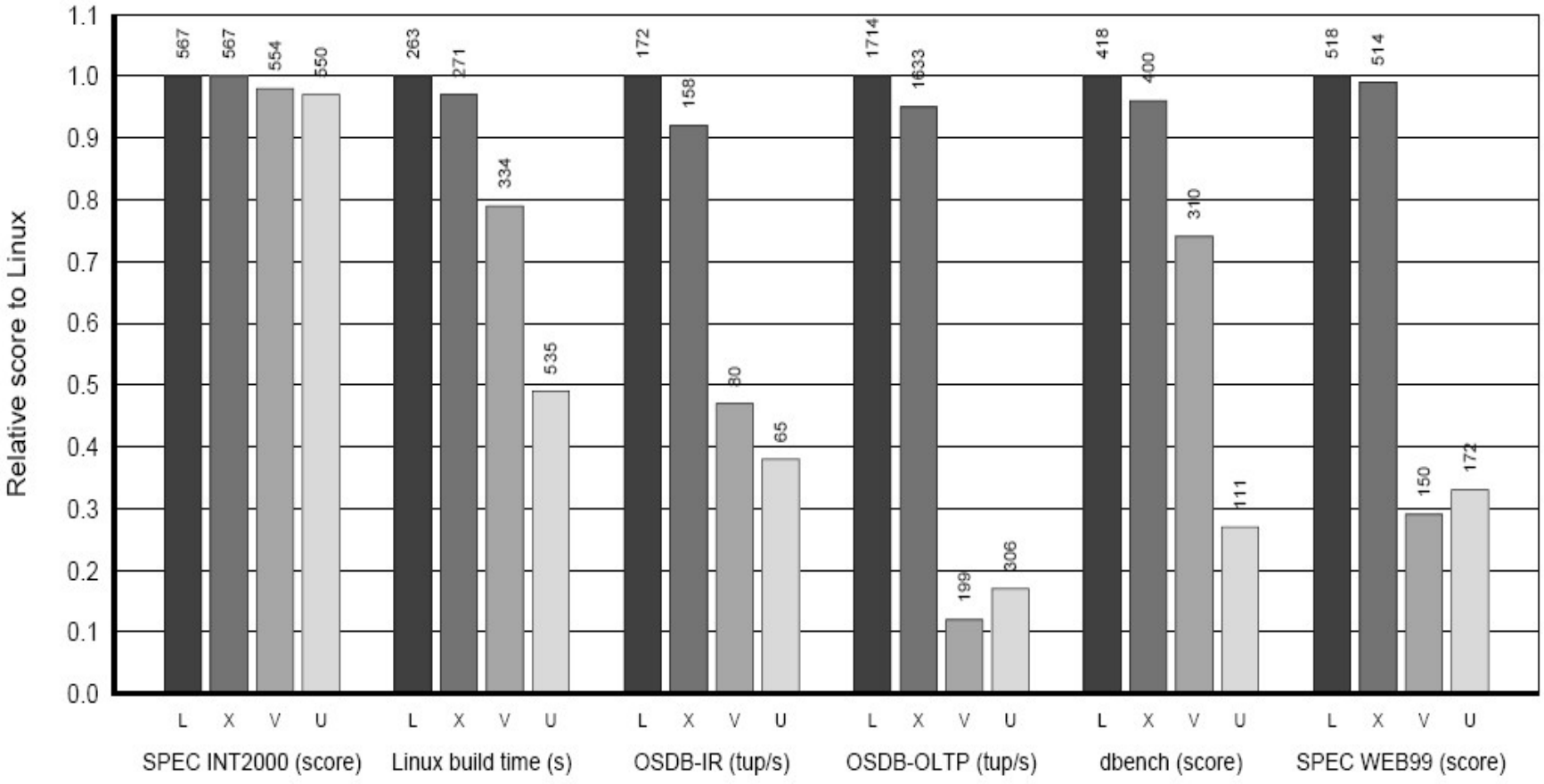


Figure 3: Relative performance of native Linux (L), XenoLinux (X), VMware workstation 3.2 (V) and User-Mode Linux (U).

Microbenchmarks

- Stat, open, close, fork, exec, etc
- Xen shows overheads of up to 2x with respect to native Linux
  - (context switch across 16 processes; mmap latency)
- VMware shows up to 20x overheads
  - (context switch; mmap latencies)
- UML shows up to 200x overheads
  - Fork, exec, mmap; better than VMware in context switches

Summary

- Paravirtualization benefits
  - Better performance
  - Allow VMs to be isolated
  - Support up to 100 OS instances
- Paravirtualization disadvantages
  - OS must be aware of virtualization
- Contorl and management of Xen itself is done from a special guest OS, i.e., domain 0