Virtualization

Dilma da Silva
dilmasilva@us.ibm.com
Advanced Operating Systems Department
IBM TJ Watson Research Center

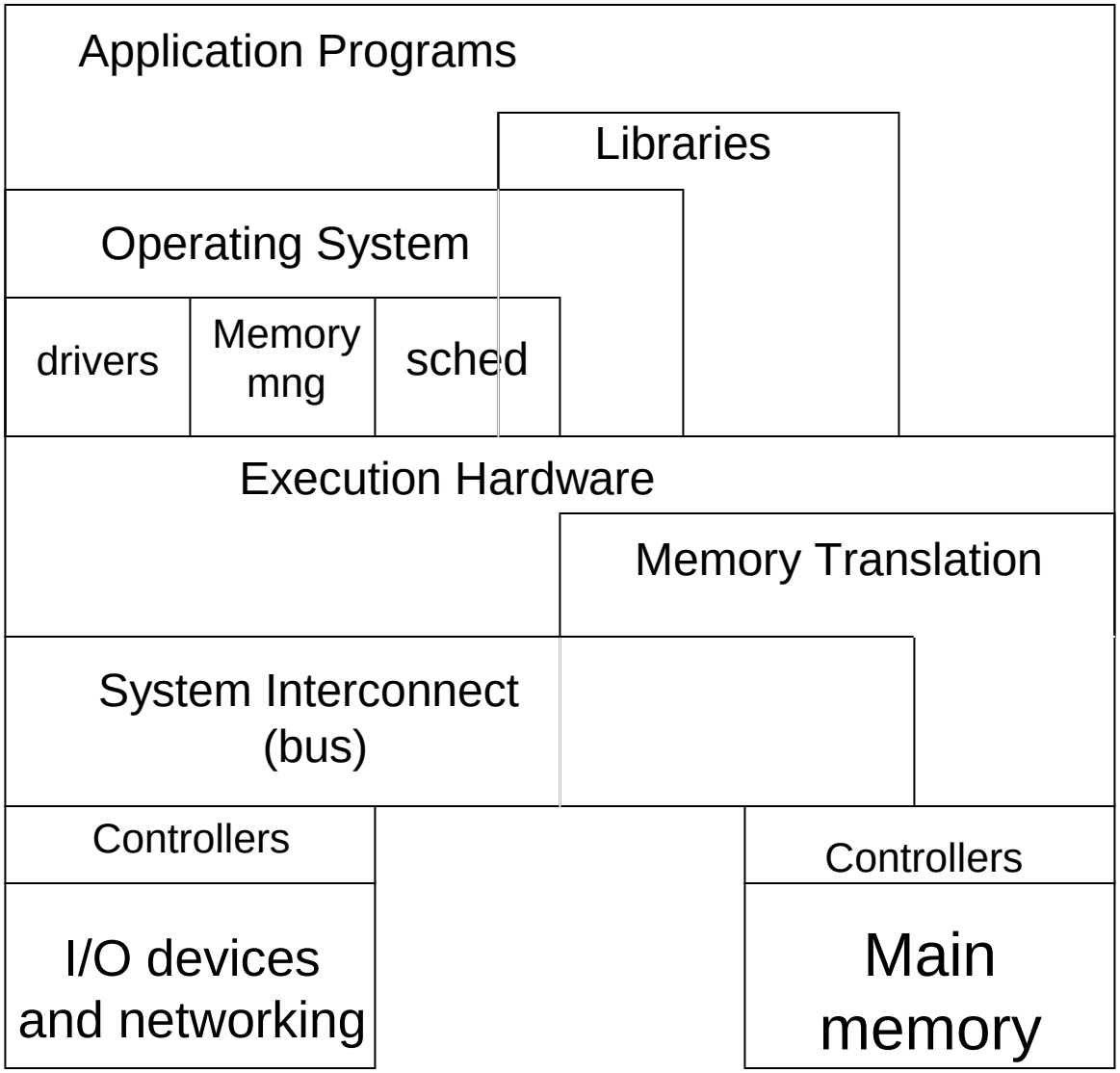
Outline

- Virtualization Basics
- Case Studies
  - VMware
  - Xen
- Current landscape
  - Impact of KVM, Veridian
- New usages for virtualization
  - Virtual appliances
  - Utility computing
  - Multicore architectures
  - Specialized execution environment

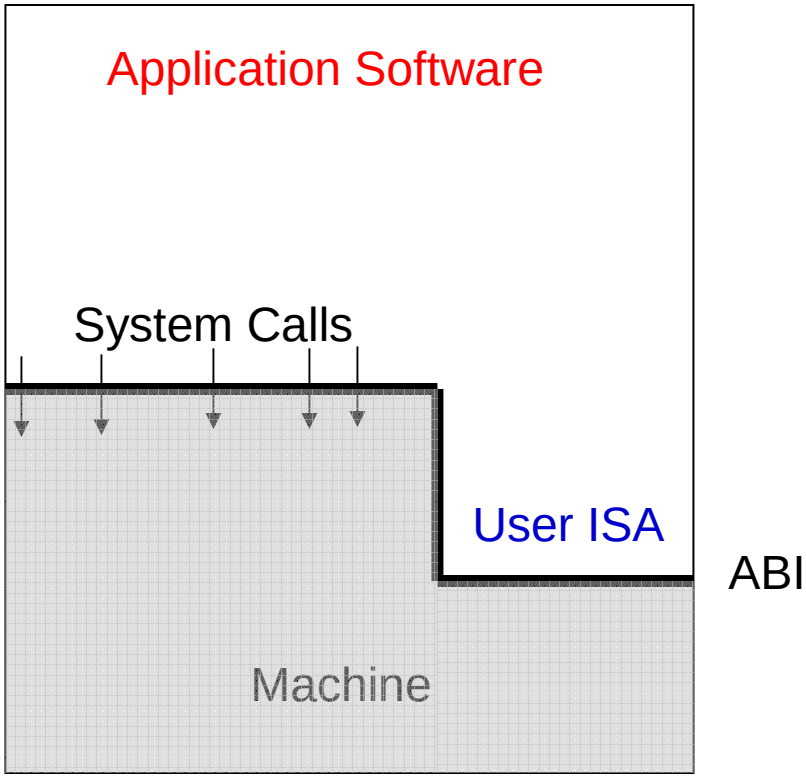
Recap: the role of Operating Systems

- Processes
- Multitasking
- System API
- Privileged mode
- I/O services
- Complaints ?
  - QoS
  - Reliability
  - Security
  - Evolution

Recap: Computer Architecture

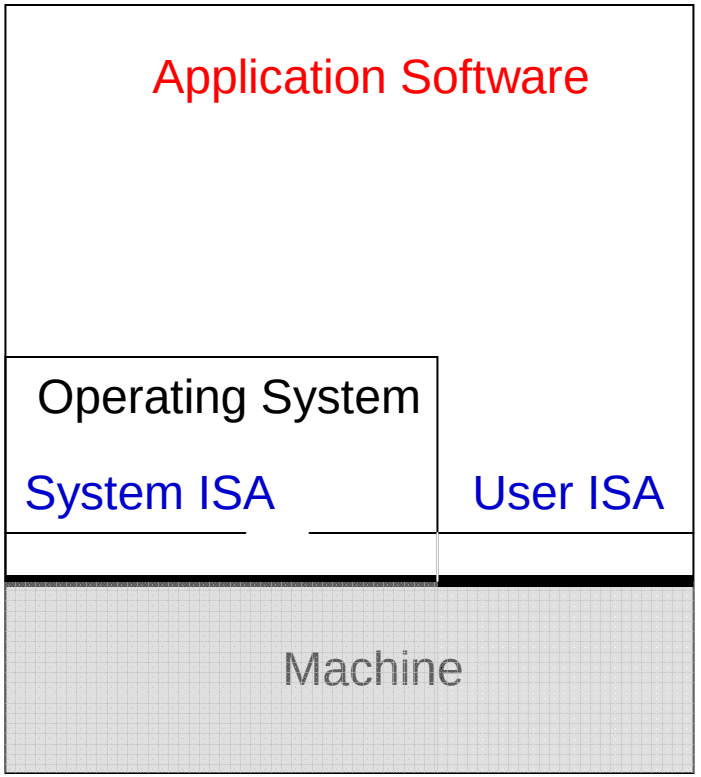


Machine Interfaces



ABI

ABI



ISA

ISA

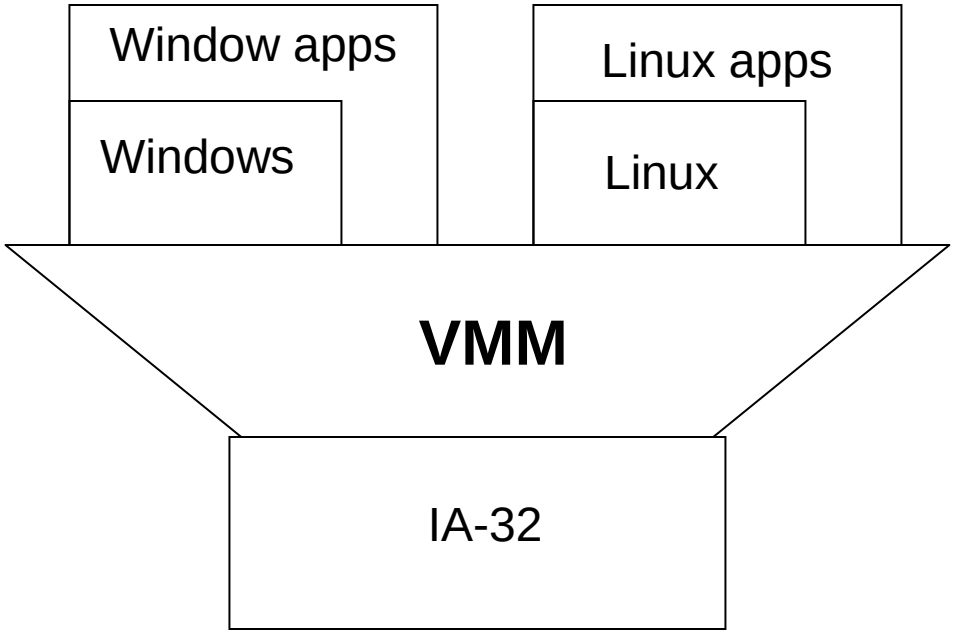
Process Virtual Machines

Process-level VMs provide user apps with a virtual ABI environment

- Multiprogramming
- Emulators and Dynamic Binary Translators
- Same-ISA Binary Optimizers
- High-Level Language Virtual Machines (Platform Independence)

System Virtual Machines

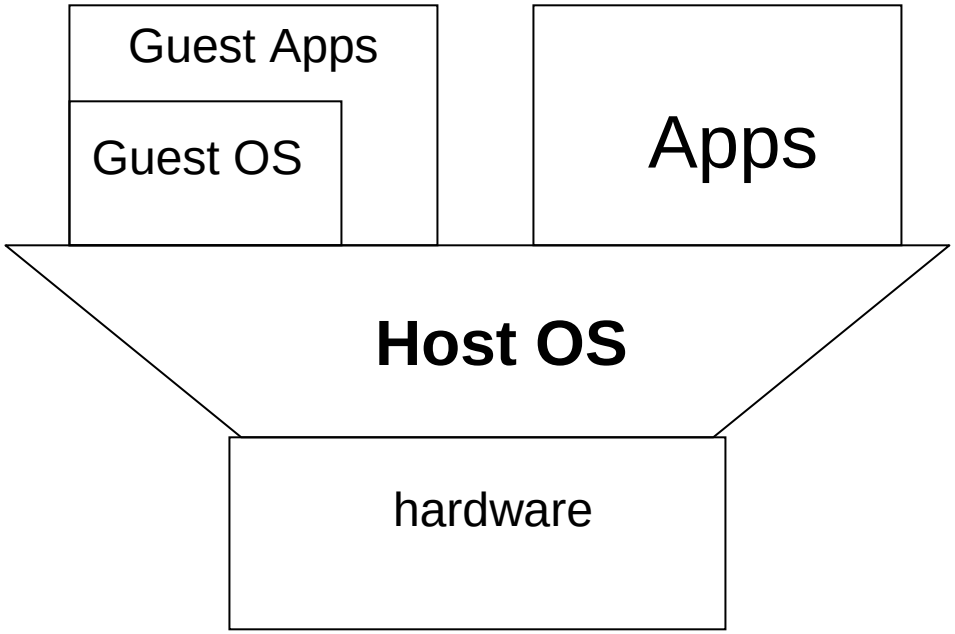
Provide a complete system environment in which many processes, possibly belongingt o multiple users, can coexist.



Classic

Approach

Alternative System VMM implementation



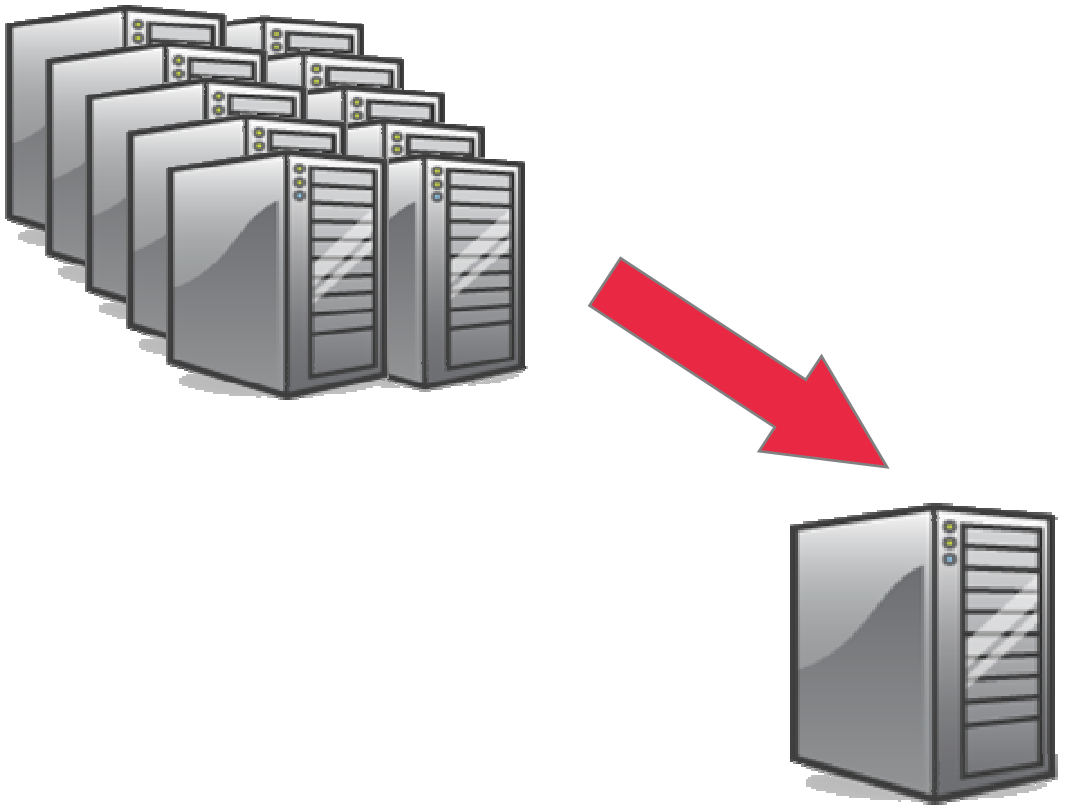
Virtualization

- Multiple consumers share a resource while maintaining the illusion that each consumer owns the full resource
  - Memory, processor(s), storage, peripherals, entire machines
- Virtual Machine Monitor (VMM) or hypervisor is the software layer that provides one or more Virtual Machine (VM) abstractions

System Virtual Machines: 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)

Data Center Consolidation



System Virtual Machines Applications

- Implementing Multiprogramming
- Multiple single-application virtual machines
- Multiple secure environments
- Managed application environments
- Mixed-OS environments
- Legacy applications
- Multiplatform application development
- New system transition

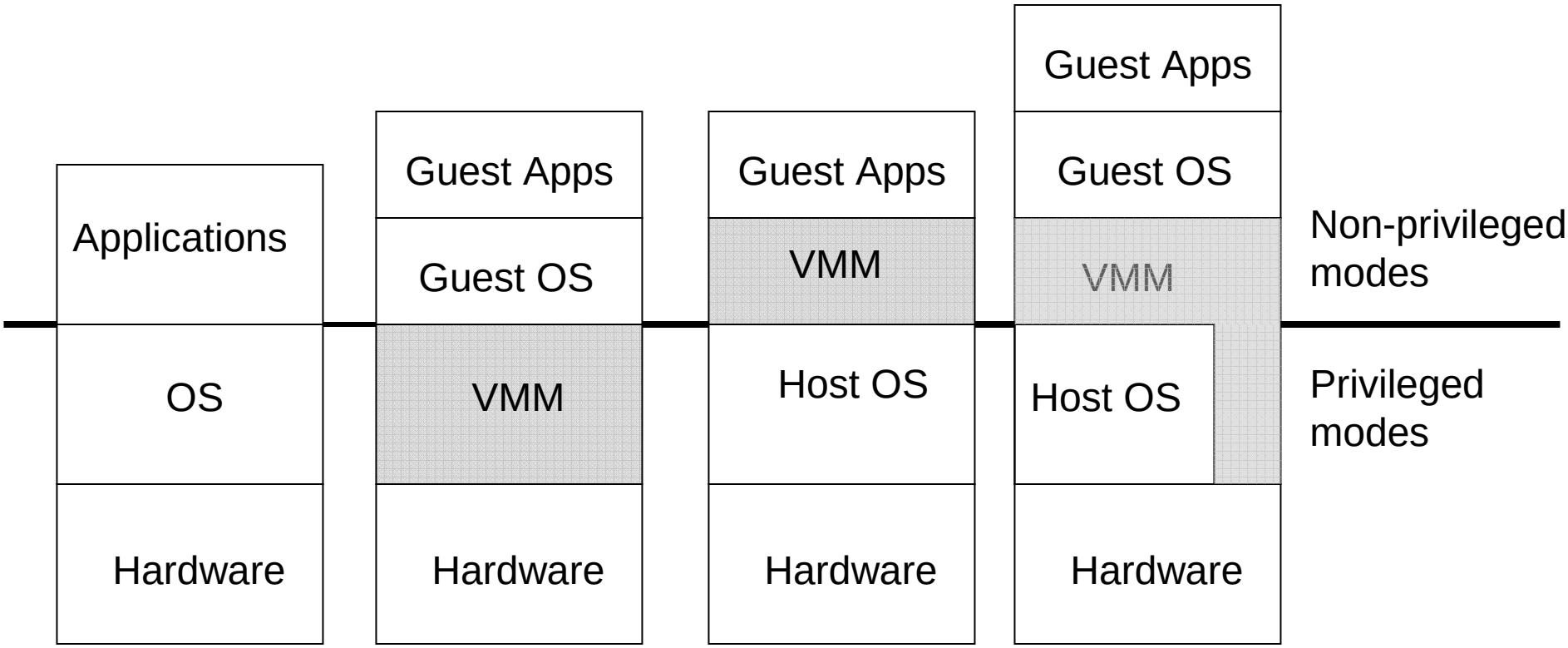
System Virtual Machines Applications (cont)

- System Software Development
- Operating system training
- Help desk support
- Operating system instrumentation
  - IBM Keefe (68), UMLinux (2003)
- Event monitoring
  - Replay
- System encapsulation

System Virtual Machines Applications (cont)

- Management simplification
  - Dynamic provisioning
  - Workload management/isolation
  - Virtual machine migration
  - Reconfiguration
- Virtualization protects IT investment
- Virtualization is a true scalable multi-core work load

Native and Hosted VM Systems



Resource Virtualization - Processors

- Execution of the guest instructions (both system and user level)
  - Emulation
    - Performance is an issue
  - Direct native execution
    - Not always possible

Privileged and Sensitive 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

Privileged and Sensitive Instructions (cont)

- IA-32 POPF instruction: pops the flag registers from a stack held in memory.
- On of the register is the interrupt-enable flag, which can be modified only in privileged mode. In user mode, this instruction overwrites all flags except the interrupt-enable flag

- POPF is sensitive but not privileged!

Sufficient conditions for ISA Virtualizability (1974)

- Assumptions:
  1. Hardware consists of a processor and a uniformly addressable memory
  2. Processor can operate in one of two modes: system mode or user mode
  3. Some subset of the instruction set is available only on system mode
  4. Memory addressing is done relative to the contents of a relocation register
- (I/O was not considered)

Sufficient conditions for ISA Virtualizability (cont)

A VMM may be constructed if the set of sensitive instructions is a subset of the privileged instructions

- POPF is sensitive but not privileged (critical), so we can't virtualize IA 32 ?????
- VMM could intercept POPF (and other critical instructions) and deal with them …
  - performance issue
- ... Or Intel/AMD can fix architecture
  - legacy issue

Patching critical instructions:

- basic block scan with instruction replaced with trap to VMM
- Caching emulation code

Resource Virtualization: Memory

- Native platform (without VMM) :
  - Operating systems keep maps from virtual address space to real memory which is physical memory
- Virtualized platform (with VMM):
  - Guest's real memory must undergo further mapping to determine address in physical memory of host hardware
- Combined total size of real memory of all guests can be bigger than available physical memory ➔ VMM maintains its own swap space

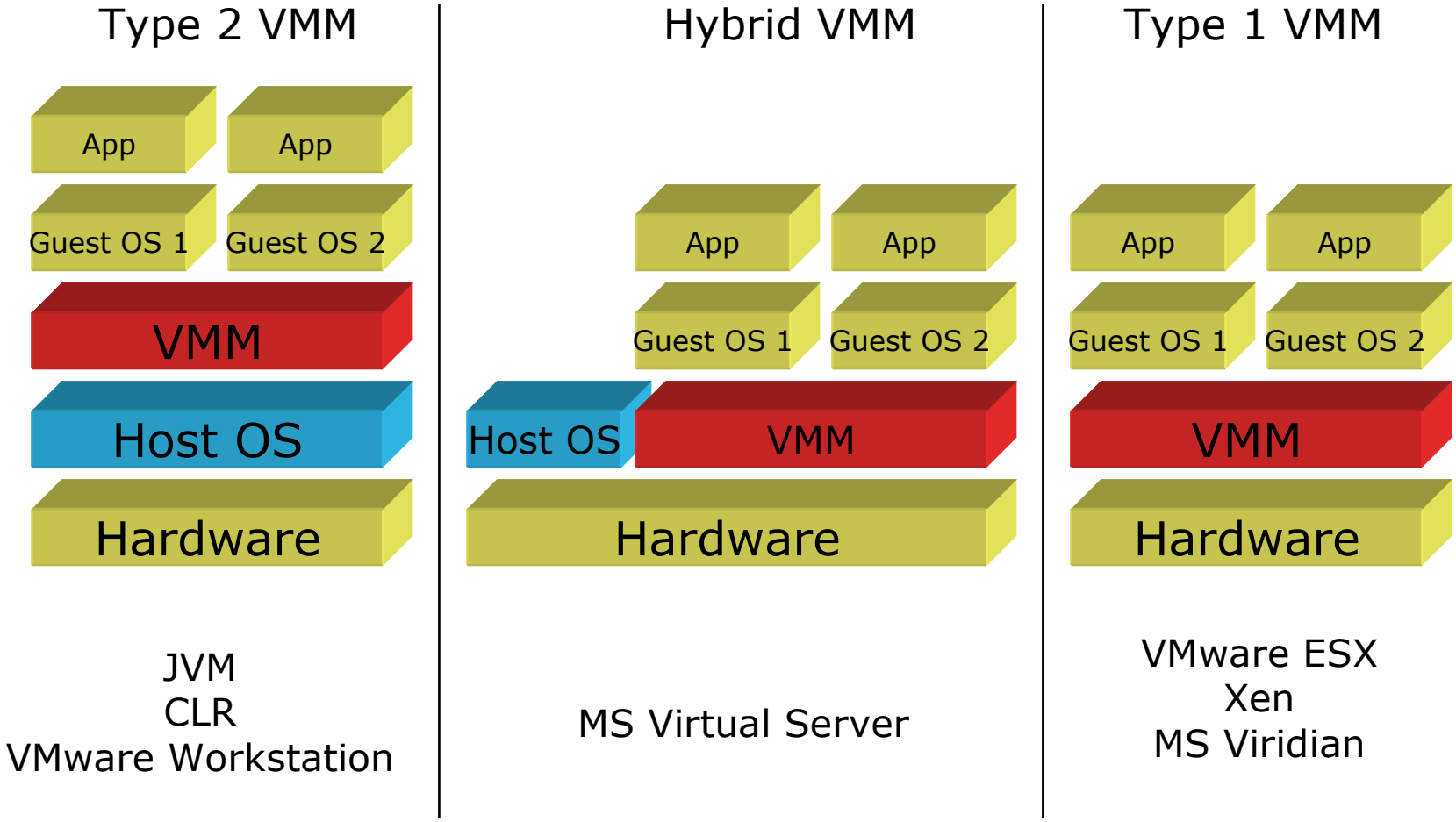
Resource Virtualization: Memory (cont)

- Architected page tables
  - Virtual-to-physical mapping kept by the VMM in shadow page tables used by hardware to translate virtual addresses and to keep TLB up-to-date
- Page table register is virtualized. VMM updates it when it activates a guest VM
- When a guest tries to access the PTP, either to read it or write it, the read or write instruction traps (either automatically or through patched code)
- Architected, software-managed TLBs
  - If tags available, flushes minimized

Resource Virtualization: I/O

- Difficult!
- For a given I/O device type, construct a virtual version of the device and then virtualize I/O activity directed at the device
- When guest VM makes request to use virtual device, request is intercepted and converted to the equivalent on the physical device
- Dedicated devices: mouse, console, keyboard…
- Partitioned devices: disk
- Shared devices: network adapter

Virtual Machine Monitor Approaches



Performance of Virtualization

- Reasons for performance degradation
  - Setup
  - Emulation
  - Interrupt handling
  - State saving
  - Bookkeeping
  - Time elongation
- Systems such as System/370 introduced instructions to reduce overhead
- Guest OSes can also work on different mode (e.g. real-mode only) to alleviate extra costs

Outline

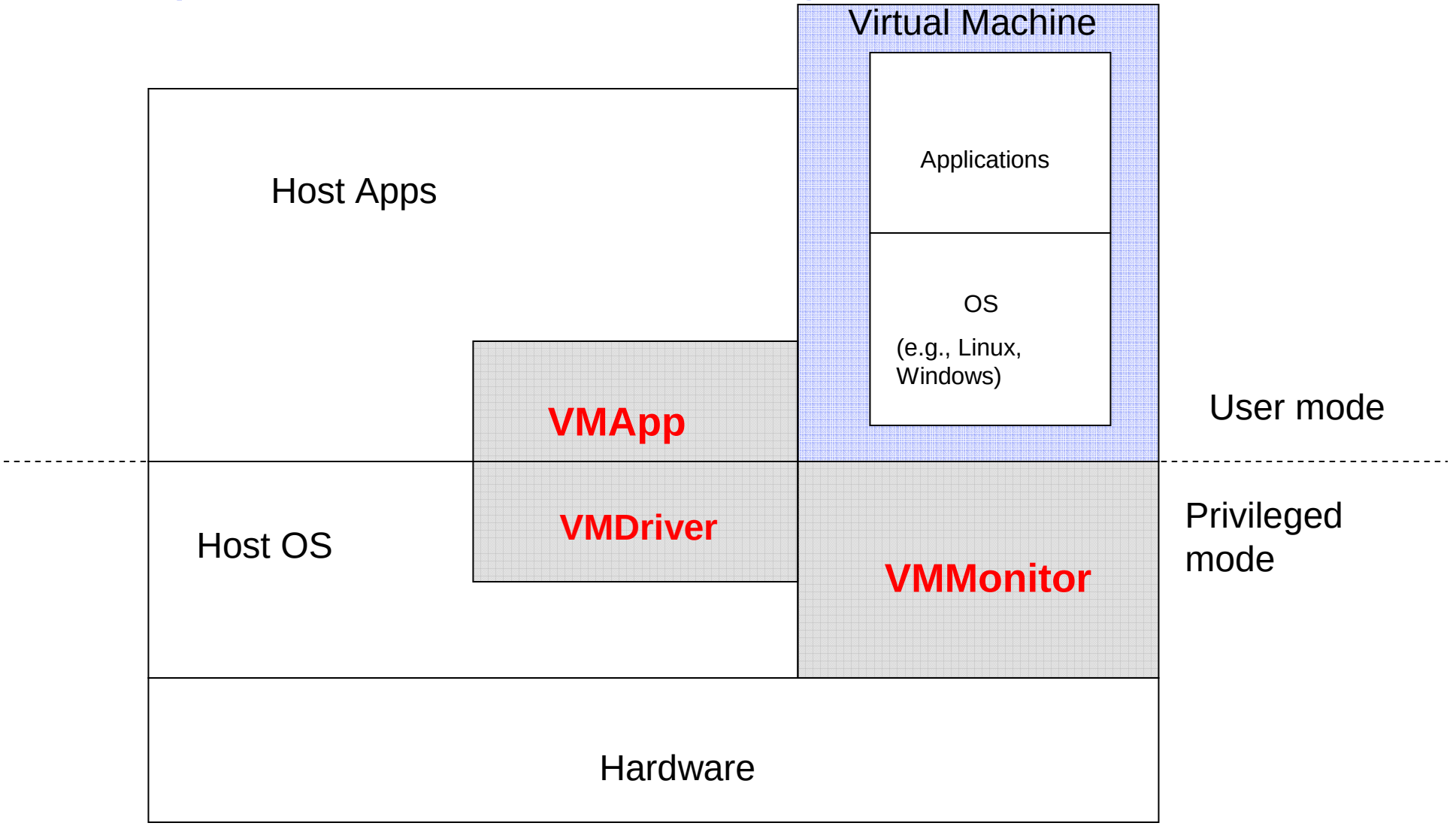
- Virtualization Basics
- Case Studies
  - VMware
  - Xen
- Current landscape
  - Impact of KVM, Veridian
- New usages for virtualization
  - Virtual appliances, utility computing
  - Multicore architectures
  - Specialized execution environment

VMware virtual platform

- VMware is an EMC company going IPO soon
- Free: VMware Server, VMware player, (try)
- VMware Infrastructure 3: VMware ESX Server, VMware Virtual Center, Consolidated Backup
- VMware Server is a hosted virtual machine system
- VMware ESX Server has included native virtualization architecture

- ia-32 has not been designed for large systems supporting multiple users

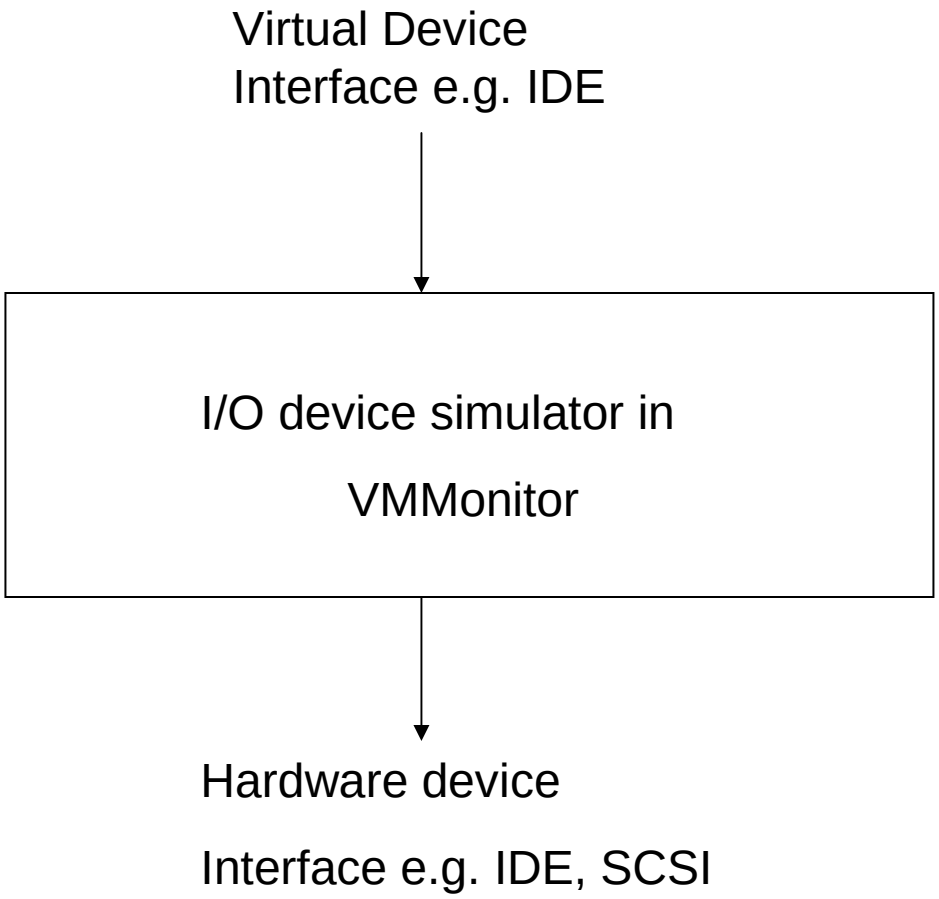
Components of the VMware System

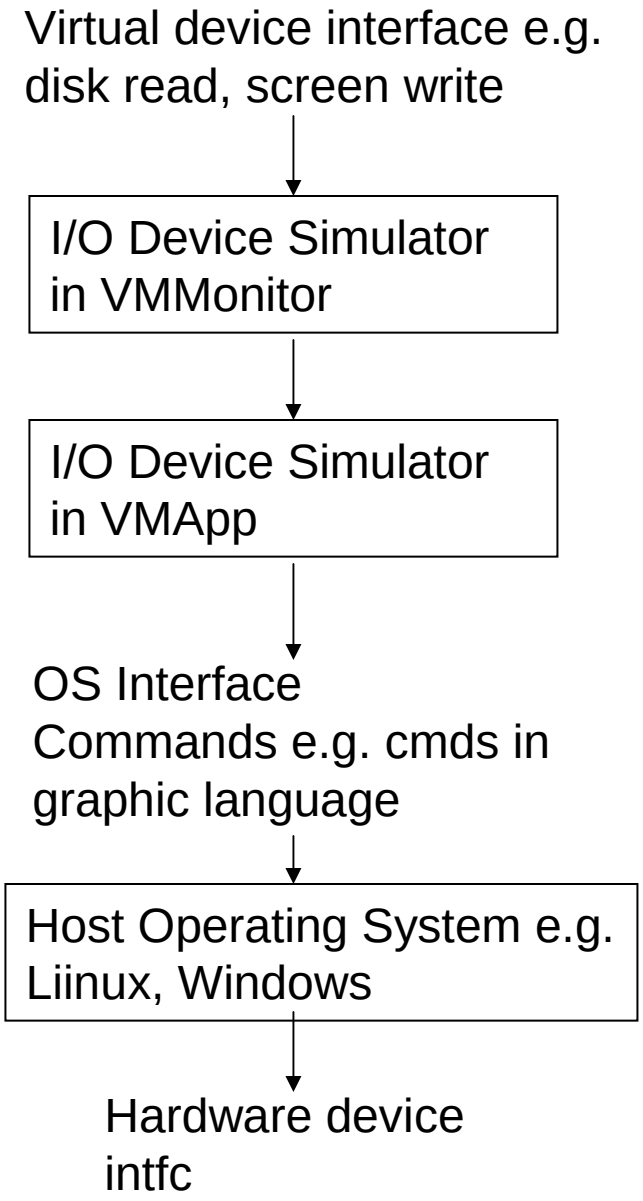


VMware's processor virtualization for IA-32

- IA-32 has 17 instructions that are critical
- VMMonitor scans instruction stream and detects the presence of instructions such as popfd
- The instruction is replaced with code that takes the processor into privileged state and emulates the action of orignal code

I/O virtualization




Virtual device interface e.g. disk read, screen write
I/O Device Simulator in VMMonitor
I/O Device Simulator in VMApp
OS Interface Commands e.g. cmds in graphic language
Host Operating System e.g. Liinux, Windows
Hardware device intfc

VMware's memory virtualization

- VMMonitor virtualizes physical memory of a virtual machine by using the host operating system to allocate or release the real machine's physical memory
- A ballooning technique reclaims the pages considered least valuable by the operating system running in a virtual machine
- An idle memory tax achieves efficient memory utilization while maintaining performance isolation guarantees
- Content-based page sharing and hot I/O page remapping exploit transparent page remapping to eliminate redundancy and reduce copying overheads.

How to use it ?

- Download free version for your host OS
- Create a virtual machine
  - Be prepared to have an image to install
- Run your image
- Notice things changed in your host OS

Revisiting what we learned so far …

x86 Virtualization Approaches

- Full virtualization
  - Binary rewriting
    - Inspect each basic block, rewrite privileged instructions
    - VMware, Virtual PC, qemu
  - Hardware assist (AMD SVM, Intel VT-x)
    - Conceptually, introduce a new CPU mode
    - Xen, KVM, MS Viridian, (VMware)
- Paravirtualization
  - Modify guest OS to cooperate with the VMM
  - Xen, L4, Denali
- Hybrid combinations
  - MS Viridian's enlightements
  - Vmware's Virtual Machine Interface (VMI)

CPU Virtualization Techniques Comparison

	Performance	Legacy guest support	VMM complexity
Binary rewriting	**medium**	**yes**	**high**
paravirtualization	**high**	**no**	**medium**
Hardware assist (current gen)	**low**	**yes**	**medium-low**
Hardware assist (next gen)	**medium**	**yes**	**medium-low**
Future hardware assist	**high**	**yes**	**low**

low medium high

Xen (let's look at motivations again!)

Motivations:
- server consolidadtion
- co-located hosting facilities
- distributed web services
- secure computing platforms
- application mobility

Challenges:
- isolation (including performance isolation)
- heterogeneity of guest OSes
- small performance overhead

- Target was running 100 guests

Why not simply run multiples apps on a hardware?
- get performance isolation (hard to get when resources are oversubscribed or users are uncooperative); OSes tried this with recourse containers,
- Linux/RK, Qlinux, SILK … But it's hard to account for resource usage:
  - charge the right app ... given how e.g. buffer caches and page caches work
- sysadm costs of dealing with requirements from configuration interactions
- certain apps require specific OSes/libraries

Full- versus Para-virtualization

- Xen developers advocate that there are situations in which full virtualization is not desirable
  - OS may want to see physical time (not only virtual) and real machine addresses
- Xen does paravirtualization:
  - presents a VM abstraction similar but not identical to hardware
  - it requires modifications to the guest OS
    - but apps do not change ... well, glibc for x86 does change

Xen virtualization of I/O

- Xen offers a set of clean device abstractions
- I/O data is transferred to/from domUs through Xen (using shmem async buffer-descriptor rings)
- Xen supports a lightweight event delivery mechanisms to let the OSes know that there are notifications ... OS can hold off on the callbacks as long as it wants ...

- dom0: responsible for hosting app-level mng software
- control itfc can create/destroy domains, specify scheduling parameters,
- physical mem alloc, access to physical disks and net devices (creation of
- virtual itfcs and virtual block dev)

- hypercalls: synch calls from domain to Xen
- notifications from Xen to domains through async events (e.g. delivery of
- net pack, completion of virtual disk request)

Xen CPU virtualization

- hypervisor is most privileged piece of code
- if only two privilege levels exist, OS had to share level of privilege with apps
- The OS calls the hypervisor to pass control to apps
- In x86 there are 4 levels (but on x86_64 there are only two)
  - In x86 only ring 0 can run privileged instructions. Apps run on ring 3 and nothing really runs on rings 1 and 2, so we can have the OS running on level 2
- Xen validates and executes the privileged instructions:
  - installing a new page table
  - yielding the processor when idle

Xen CPU virtualization (cont)

- exceptions (including memory faults and software traps)
  - a table describing the handler for each type is registered with Xen for validation
    - not much change in the handlers ... only the page fault one because it used to run the fault address from a privileged register
- When an exception occurs outside of ring 0, Xen will be invoked and it will create a exception stack frame and pass it to the OS (to the registered handler)
- Frequent exceptions are page fault and system calls.
  - To make syscalls fast, the OS can register the handler (validated by Xen) and then the handler will be invoked without crossing to ring 0

- validation of handlers only necessary if they specify execution on ring 0
- If the OS registers a routine that is not paged in memory, then Xen will take a fault on "iret" instruction that would go to the hander
  - Xen detects this double faults and terminates the offending OS

Xen memory virtualization

- guest OSes are responsible allocating and managing hardware page tables
  - hypervisor has to do something to ensure safety and isolation
- Xen lives on the top of every address space, so getting in and out of hypervisor doesn't require TLB flush
  - this is not used by any commom x86 ABI, so this doesn't break anything
- when a guest OS needs a new page table (e.g. process creation), it allocates and initializes a page from its own memory and registers it with Xen.
- Guest OS can read paging maps from page table directly, but updates of mappings may be validated from Xen
  - updates are batched
- No shadow pages
- segmentation is virtualized in a similar way

Xen virtualization of I/O

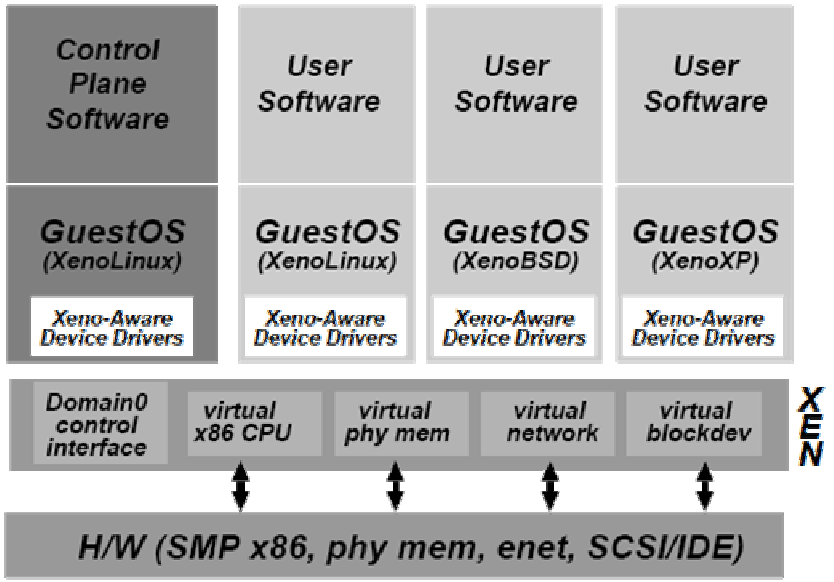
- Xen offers a set of clean device abstractions
- I/O data is transferred to/from domUs through Xen (using shmem async buffer-descriptor rings).

The Cost of Porting an OS to Xen

- Privileged instructions
- Page table access
- Network driver
- Block device driver
- <2% of code-base

Control Management

- Separation of policy and mechanism
- Domain0 hosts the application-level management software
  - Creation and deletion of virtual network interfaces and block devices

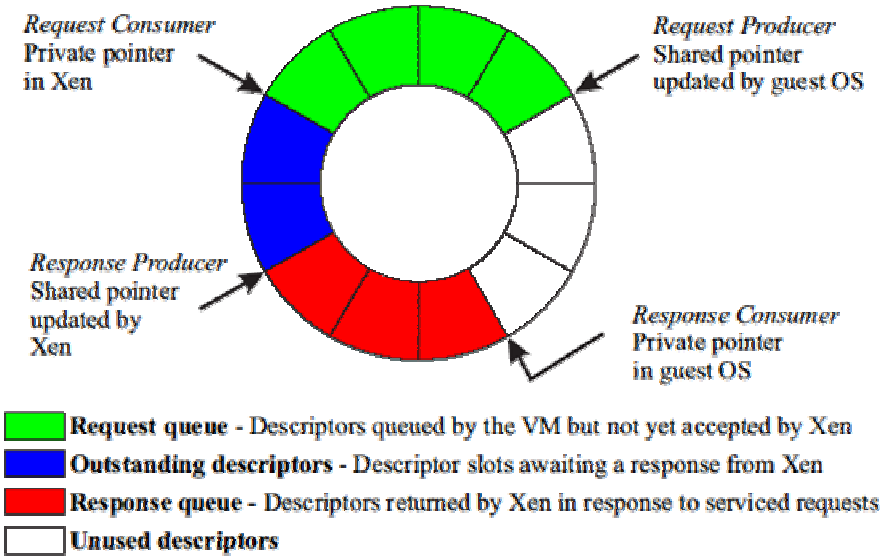


Control Transfer: Hypercalls and Events

- Hypercall: synchronous calls from a domain to Xen
  - Analogous to system calls
- Events: asynchronous notifications from Xen to domains
  - Replace device interrupts

Data Transfer: I/O Rings

- Zero-copy semantics



CPU Scheduling

- Borrowed virtual time scheduling
  - Allows temporary violations of fair sharing to favor recently-woken domains
  - Goal: reduce wake-up latency

Time and Timers

- Xen provides each guest OS with
  - Real time (since machine boot)
  - Virtual time (time spent for execution)
  - Wall-clock time
- Each guest OS can program a pair of alarm timers
  - Real time
  - Virtual time

Physical Memory

- Reserved at domain creation times
- Memory statically partitioned among domains

Network

- Virtual firewall-router attached to all domains
- Round-robin packet scheduler
- To send a packet, enqueue a buffer descriptor into the transmit rang
- Use scatter-gather DMA (no packet copying)
  - A domain needs to exchange page frame to avoid copying
  - Page-aligned buffering

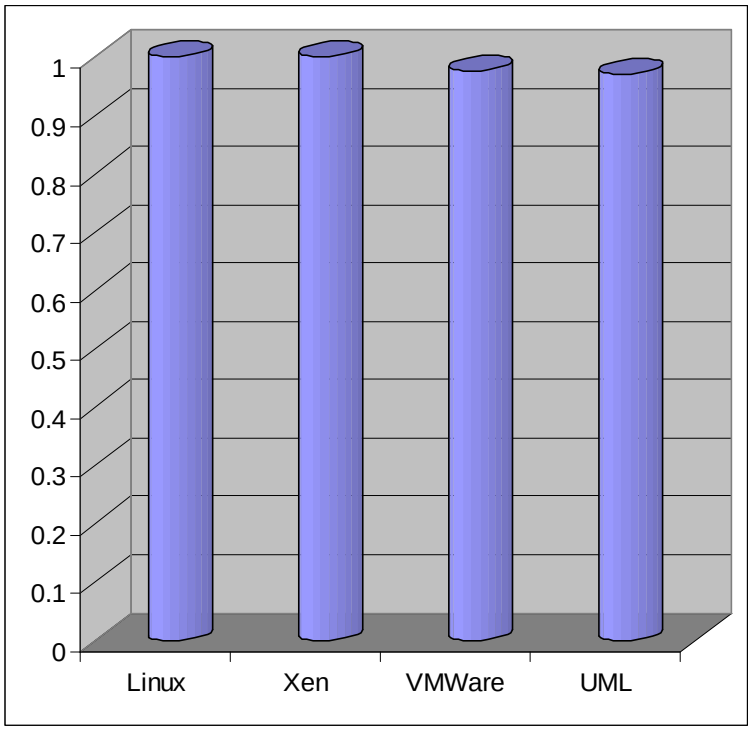
Disk

- 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
  - If permitted, Xen will also reorder requests to improve performance
- Use DMA (zero copy)

Evaluation

- Dell 2650 dual processor
- 2.4 GHz Xeon server
- 2GB RAM
- 3 Gb Ethernet NIC
- 1 Hitachi DK32eJ 146 GB 10k RPM SCSI disk
- Linux 2.4.21 (native)

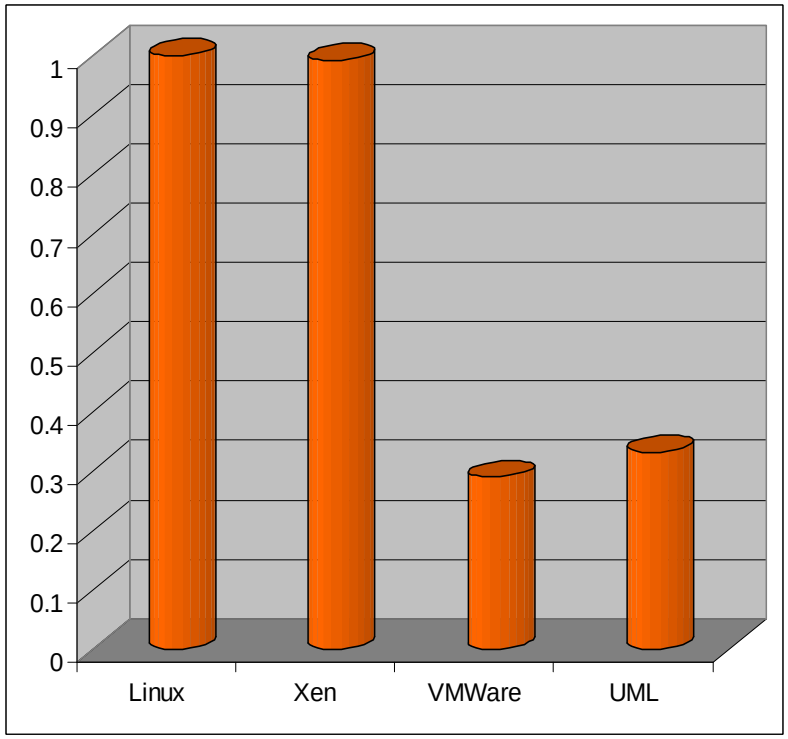
Relative Performance



SPEC INT2000 score

CPU Intensive

Little I/O and OS interaction



SPEC WEB99

180Mb/s TCP traffic

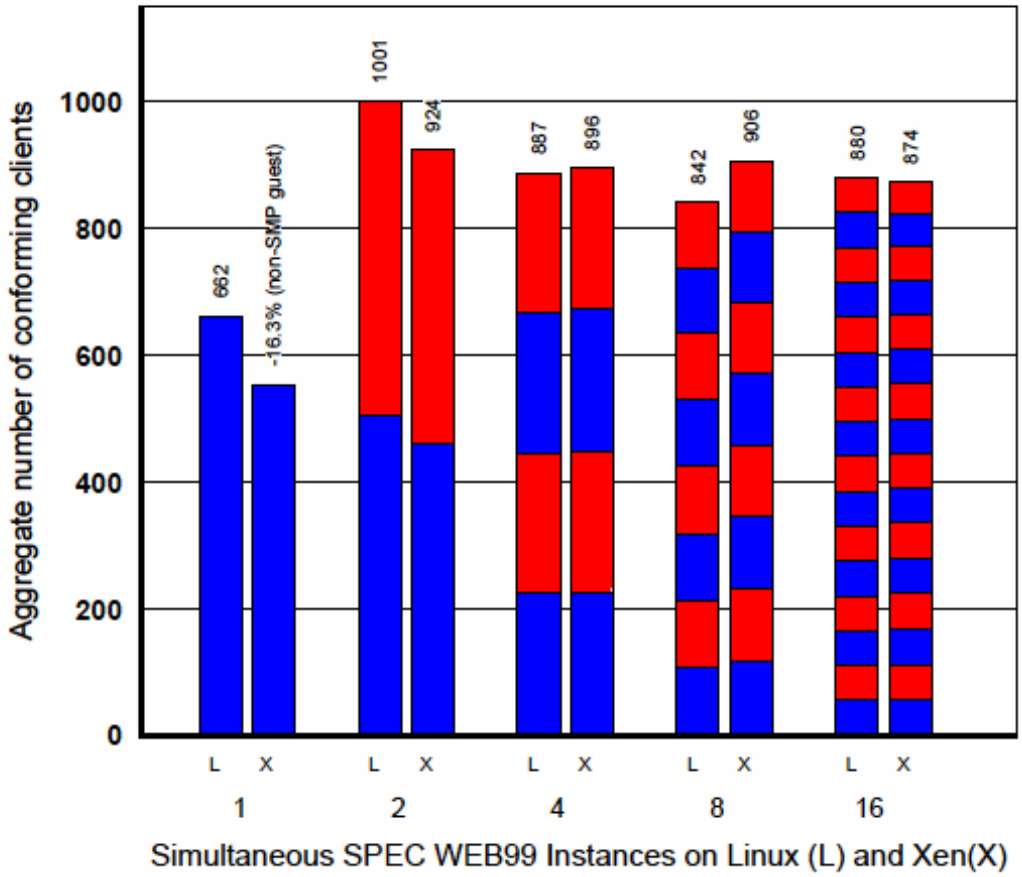
Disk read-write on 2GB dataset

Concurrent Virtual Machines

Multiple Apache processes in Linux

vs.

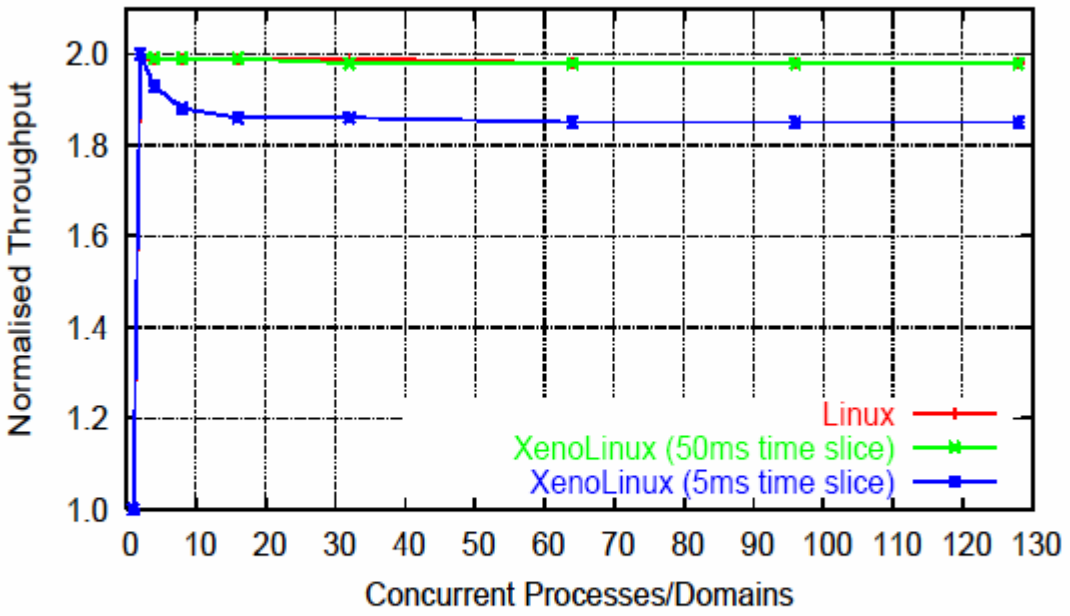
One Apache process in each guest OS



Performance Isolation

- 4 Domains
- 2 running benchmarks
- 1 running dd
- 1 running a fork bomb in the background
- 2 antisocial domains contributed only 4% performance degradation

Scalability



Normalized aggregate performance of a subset of SPEC CINT2000 running concurrently on 1-128 domains

How to experiment with OS

- Download and build it
  - Update packages, grub menu
- Boot with “XenoLinux” as your dom0

How to experiment with OS (cont): create your image

- dd if=/dev/zero of=/virtual/images/vm_base.imb bs=1024k count=xxx
- dd if=/dev/zero of=/virtual/images/vm_base-swap.img bs=…
- mkfs.ext3 /virtual/images/vm_base.img
- mkswap /virtula/imgaes/vm_base-swap.img
- mount –o loop /virtual/images.bm_base.img /virtual/vm_base
- debootstrap –arch i386 sarge /virtual/vm_base http://ftp2.de.debian.org/debian
- chroot /virtual/vm_base
- apt-setup; apt-get update; apt-get installl localeconf
- configure with base-config
- rm –f /etc/hostname
- Edit /etc/network/interfaces:

```
auto lo
Iface io inet loopback
  address 127.0.0.1
  netmask 255.0.0.0
```

- Edit /etc./fstab and /etc/hosts
- Copy kernel moldues to our virutal images
  - cp –dpR /lib/modules/2.6.12.6-xenU /virtual/vm_base/lib/modules
  - mv /virtual/vm_base/lib/tls /virtual/vm_base/tls.disabled
- umount /virtual/bm_base

How to experiment with OS (cont): create your image

- Create virtual domains: create a configuration file for your domU image using provided examples
  - name=…
  - kernel=…
  - root=/dev/hda1
  - memory=64
  - disk=['file:/virtual/images/vm01.img,hda1,w',"file:/virtual/images/vm01-swap.img.hda2,w']
  - vif=['']
  - dhcp='off'
  - ip=…
  - netmask=…, getaway, hostname …
  - Extr="3"
- Use xen tools
  - xm create –c myfirstdomain.cfg

Xen positioning in the virtualization landscape

- Many industry partners; backed by main distributions
- Derived from Linux 2.4 kernel base
- Good performance by para-virtualizing guest OS
- Optimized around hardware sweet-spot of 2003
- patches didn't make into Linux
- tools …
- Performs poorly for full virtualization without modified device drivers due to dependence on QEMU
- XenSource commercial offering includes para-virt drivers for Windows

Other x86 Players

- Paravirt, KVM
  - Generic para-virtualization interfaced released in mainline Linux kernel 2.6.20
  - KVM: Qumranet provided kernel extension for native VM support
    - Enables access to Intel's VT and AMD's SVM
    - User-level VMM: a regular Linux process
    - Loadable kernel module
    - Very new
    - Does not support advanced features such as migration
    - QEMU's devices models

Linux is perceived as stable, high-performance, scalable, and improving

Xen vs KVM: Xen

The Good

- Virtual Machine abstraction allows for easy CPU and memory hot-plugging to be supported by Guest OS
- Theoretically easier to support HW hot-plugging than in Linux (though this work does not exist yet)
- Efficient memory use to increase server consolidation scenarios
- Mature management model
- Mature VIO capabilities
- Distros have picked up and support the Linux changes
- Full virtualization comes from improvement in QEMU emulator

The Bad

- Is only as stable as the Linux that runs in Dom0
- Xen is based on old Linux 2.6.9 code that has known scalability issues, although that code is being improved with original code.
- Admitted scalability issues, especially with CMP systems on the horizon
- Efficient memory use conflicts with large/super pages and therefore performance
- Continues to grow in size and complexity and becoming yet another kernel
- Smaller, less nimble community
- XenSource contributors changes are rarely peer-reviewed

Xen vs KVM: KVM

The Good

- Capitalizes on existing Linux kernel services that are always peer-reviewed and improving
- Larger reviewing community than Xen
- Loadable module so at any time the Linux you are running can become a Hypervisor
- All Drivers and VIO are in the "Hypervisor"
- Full virtualization comes from improvement in QEMU emulator and is the same that Xen uses.
- Simpler management model, and existing non-Xen tools should port quite easy

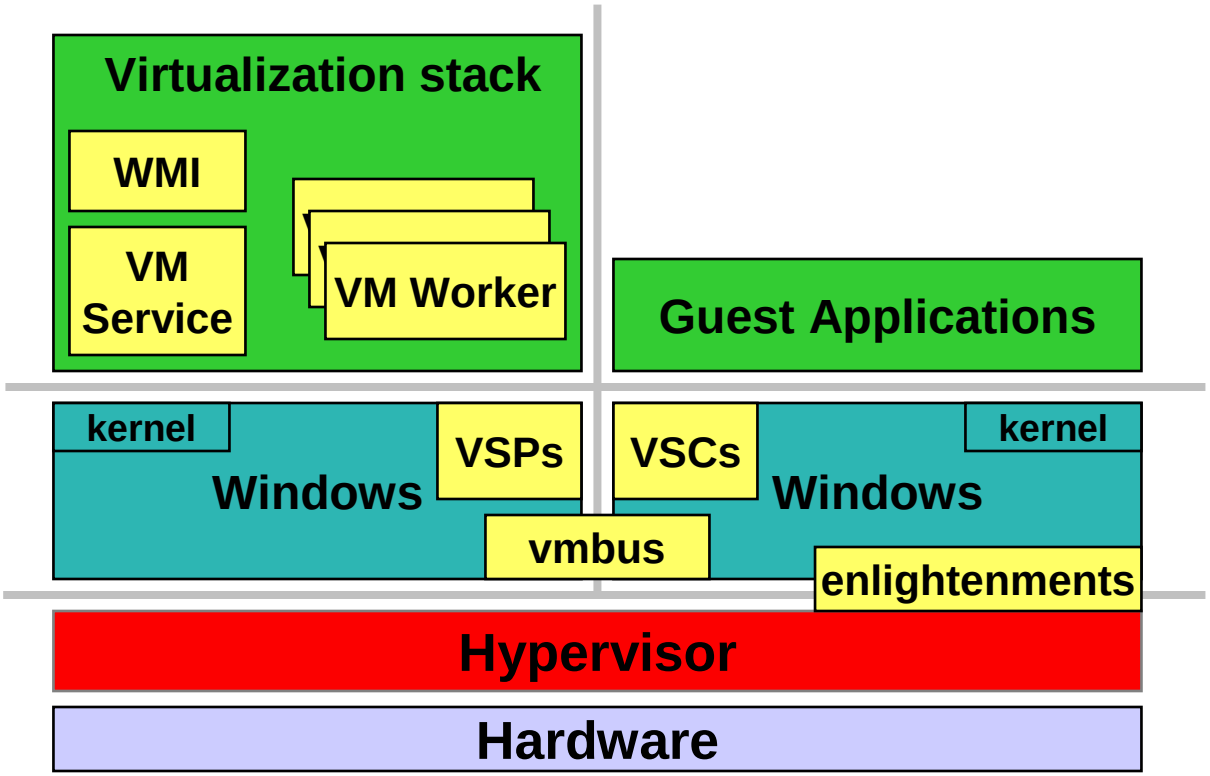
The Bad

- Currently immature, but already has a larger "free" community then Xen
- No VIO but the patch is coming tomorrow
- Could take a year to catch up with Xen-3.0.4 in terms of all functionality.

Other x86 players

- Microsoft
  - Current: Virtual PC and Virtual Server
  - In development: Veridian
  - Device para-virtualization to speed up device access

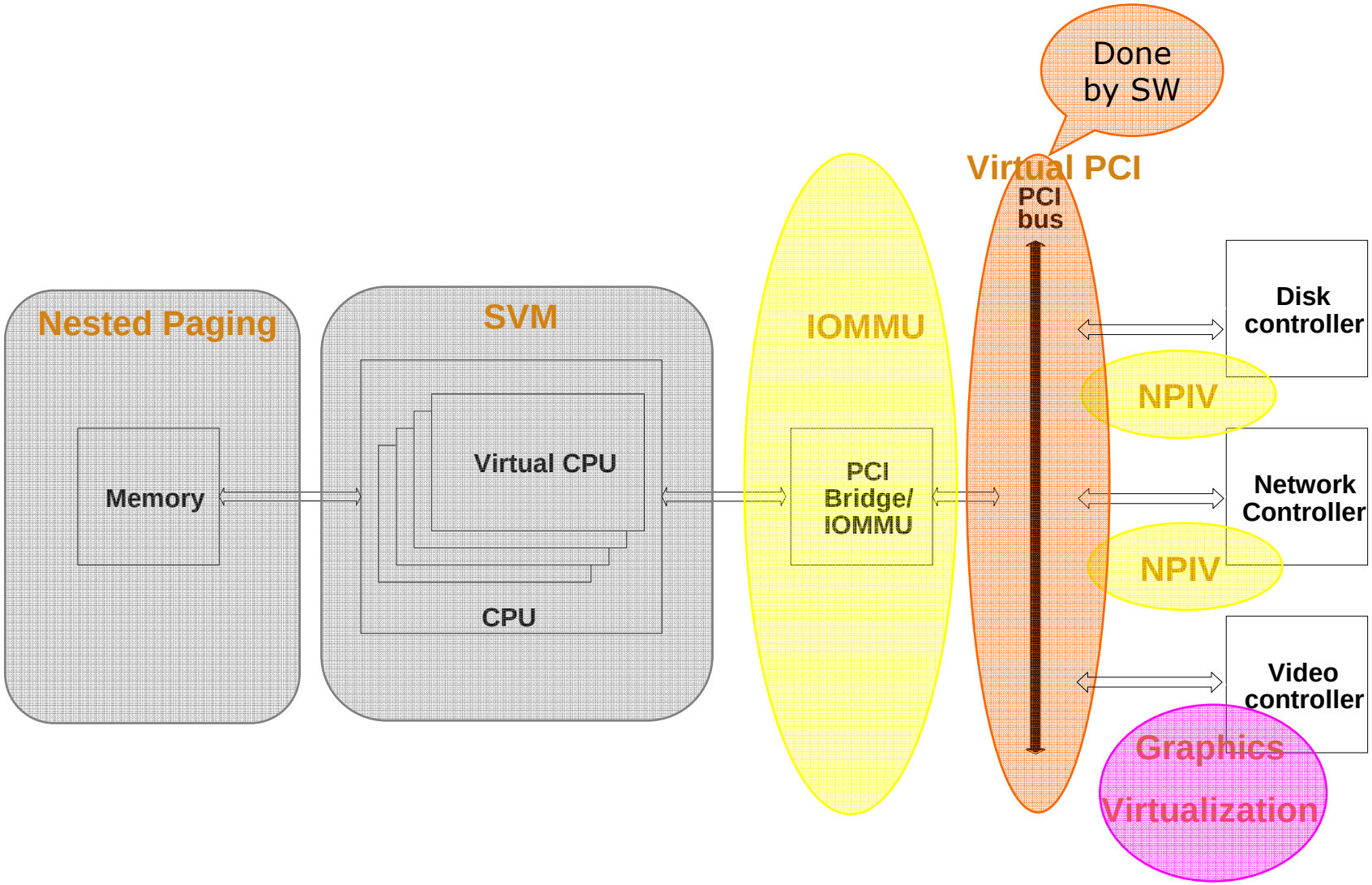
Virtualization Software Stack Microsoft Viridian



Viridian runs Windows and Linux guests
Uses AMD SVM, Intel VT-x and paravirtualization (enlightenments)

Hardware Virtualization Trends

Virtualizing The x86 Platform

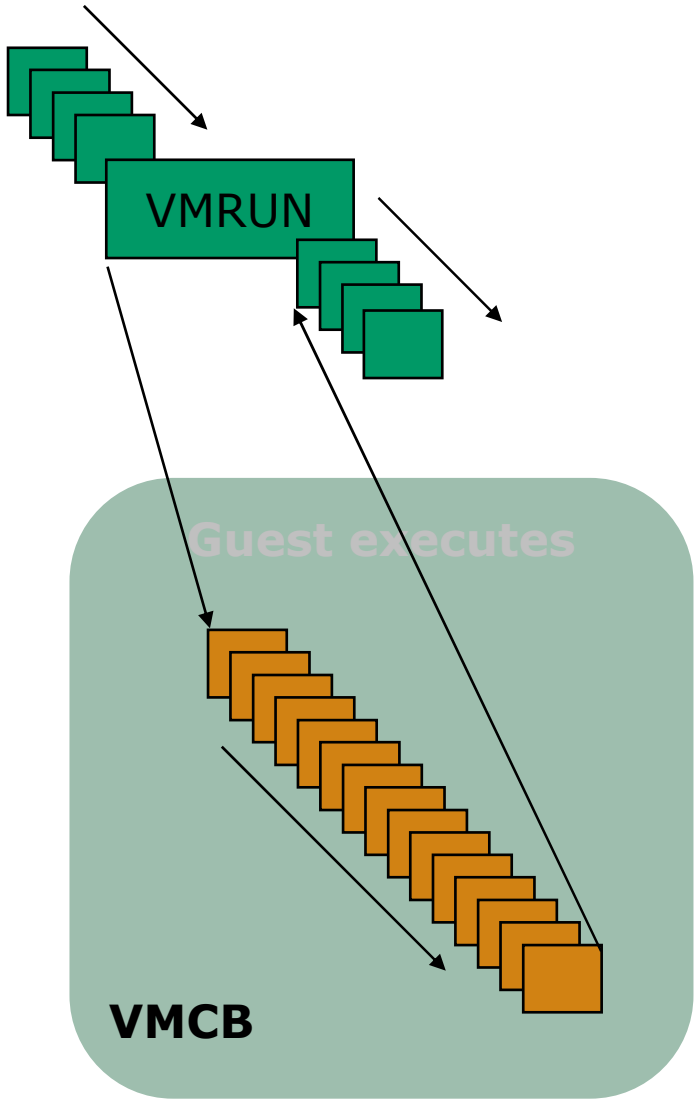


Processor Virtualization Features

- Both AMD and Intel defined processor extensions for their CPU architectures
- AMD: Secure Virtual Machine (Pacifica, SVM, AMD-V), Rev F, Rev G, Barcelona, …
- Intel: Vanderpool Technology (VT-x, VT-x2)
- From 10,000 ft. both look very similar
  - Container model (similar to mainframe SIE, start interpretive execution)

SVM In A Nutshell

- Virtualization based on VMRUN instruction (similar to SIE)
- VMRUN executed by host causes the guest to run
- Guest runs until it exits back to the host
- Host resumes at the instruction following VMRUN
- World-switch: host → guest → host
- World switches are not cheap

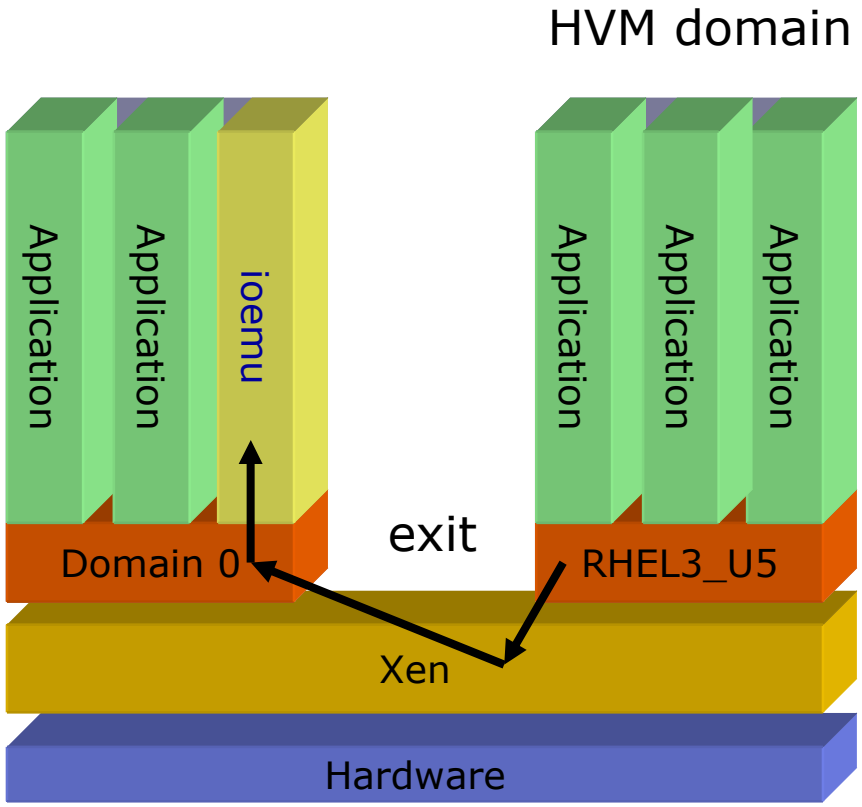


Intercepts and Exits

- A guest runs until
  - it performs an action that causes an exit
  - it executes a VMCALL/VMMCALL
- Exit conditions are specified per guest
  - Exceptions (e.g., page faults) and interrupts
  - Instruction intercepts (CLTS, HLT, IN, OUT, INVLPG, MONITOR, MOV CR/DR, MWAIT, PAUSE, RDTSC …)
- AMD-V has paged real-mode support
- Intel VT-x has shadow registers

Example: Full Virtualization Support for Xen

- Most device emulation is implemented in ioemu (PCI, VGA, IDE, NE2100, …)
- High performance drivers, such as ioapic, lapic, vpit are implemented in Xen
- Developed by Intel, AMD and IBM



Sample #VMEXIT Distribution

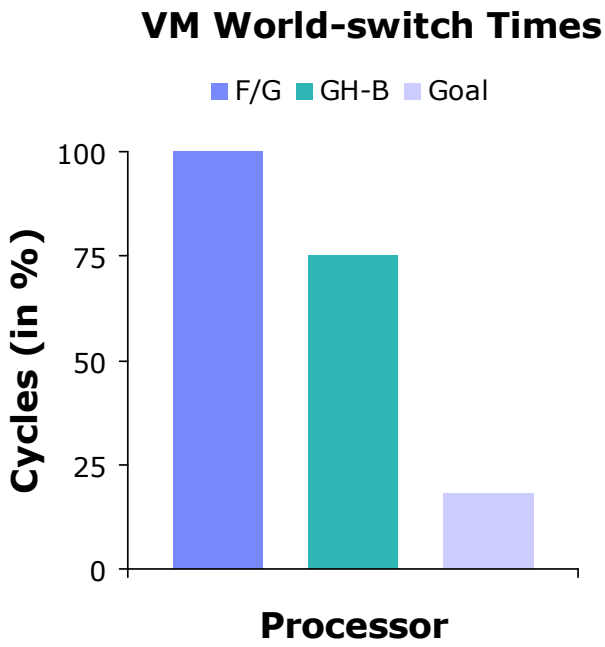
READ_CR0	634749	0%
READ_CR3	1935734	0%
READ_CR4	75	0%
WRITE_CR0	958506	0%
WRITE_CR3	3255402	0%
WRITE_CR4	146	0%
WRITE_DR0	1	0%
WRITE_DR1	1	0%
WRITE_DR2	1	0%
WRITE_DR3	1	0%
WRITE_DR7	1	0%
EXCEPTION_PF	1201225361	90%
INTR	2151104	0%
NMI	7105	0%
CPUID	48111299	3%
HLT	9370980	0%
IOIO	61350890	4%
MSR	24	0%

Performance benchmark

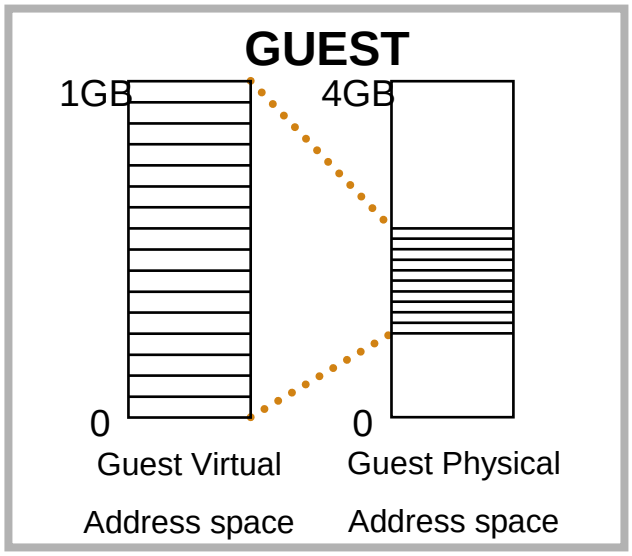
- kernbench -M
- Host: linux-2.6.20.2 + kvm-16, x86_64
- Guest: FC6, x86_64, 1.5GB
- Guest is not paging

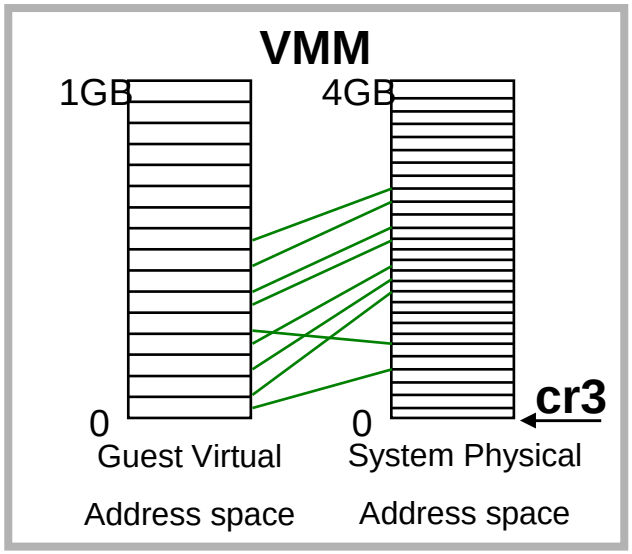
Virtualization Challenge

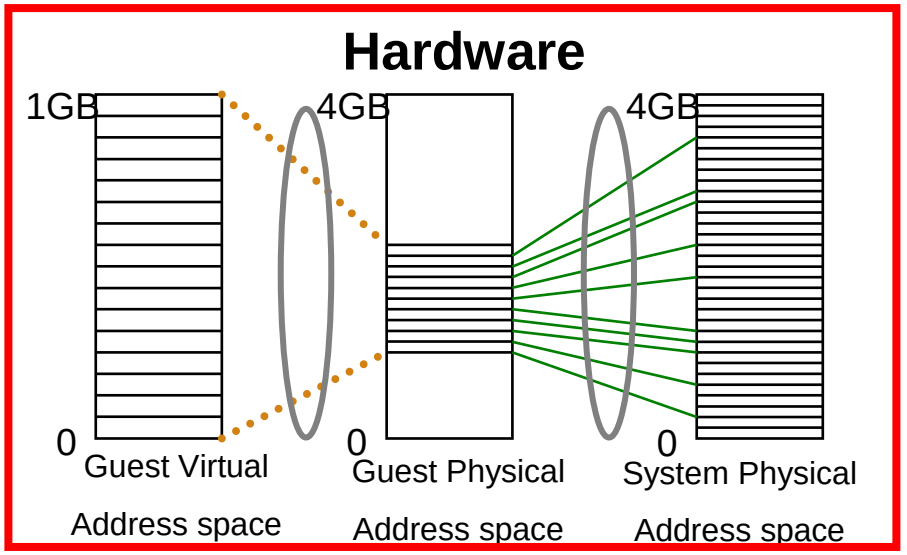
- The key problem is how to scale the number or VMs?
  - Reduce overall world-switch times
  - Eliminate world switches
  - Over commit (memory) resources
- Reduce world-switch times
  - Better caching of VMCB state
  - Selective reload of VMCB state
  - Tag TLB by ASID
- Eliminate world switches
  - Nested paging (Barcelona)
  - Direct device assignment (IOMMU)
- Additional features
  - APIC, clock, exit delays, precise exits, performance counters, etc.



Nested Page Tables

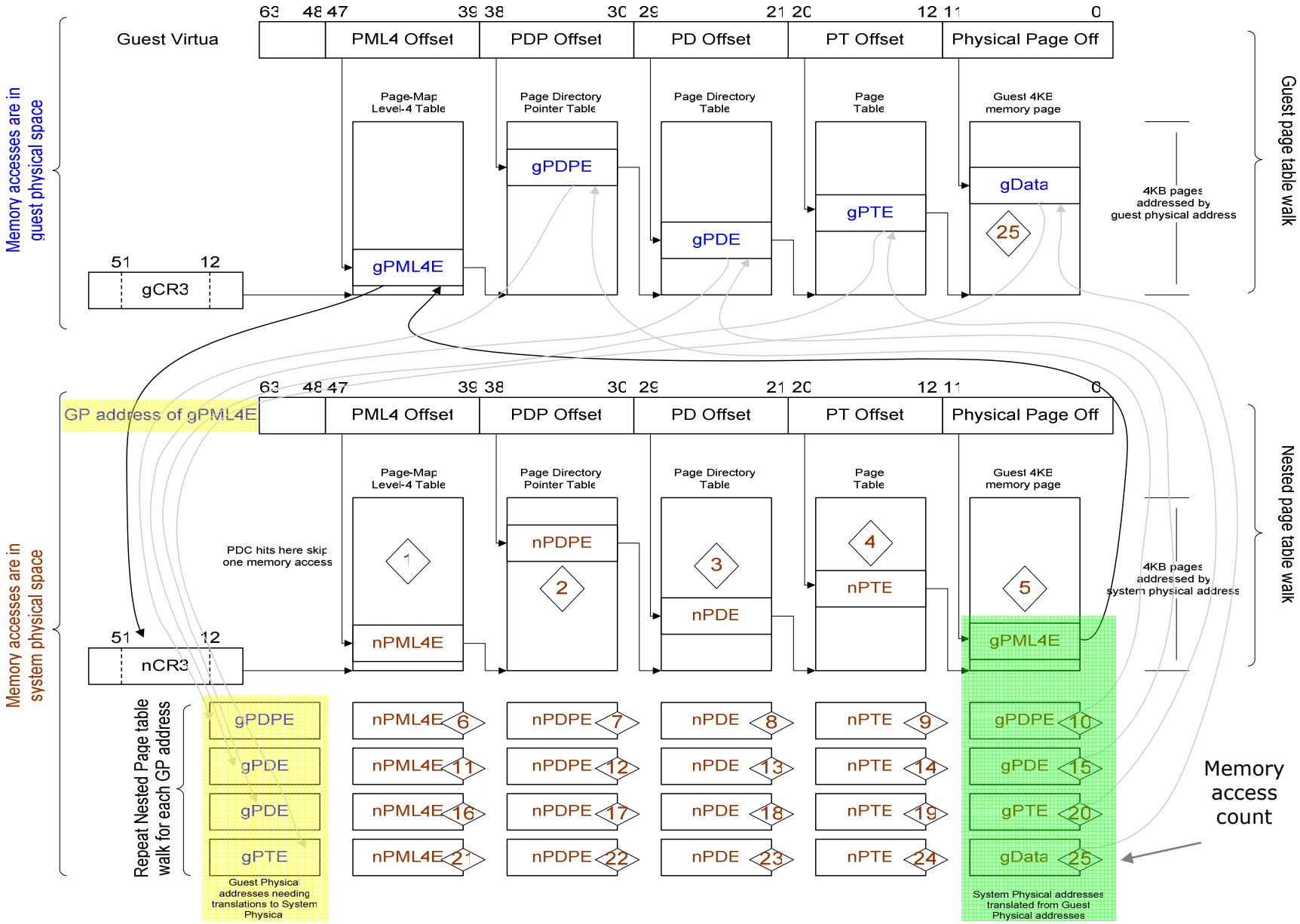




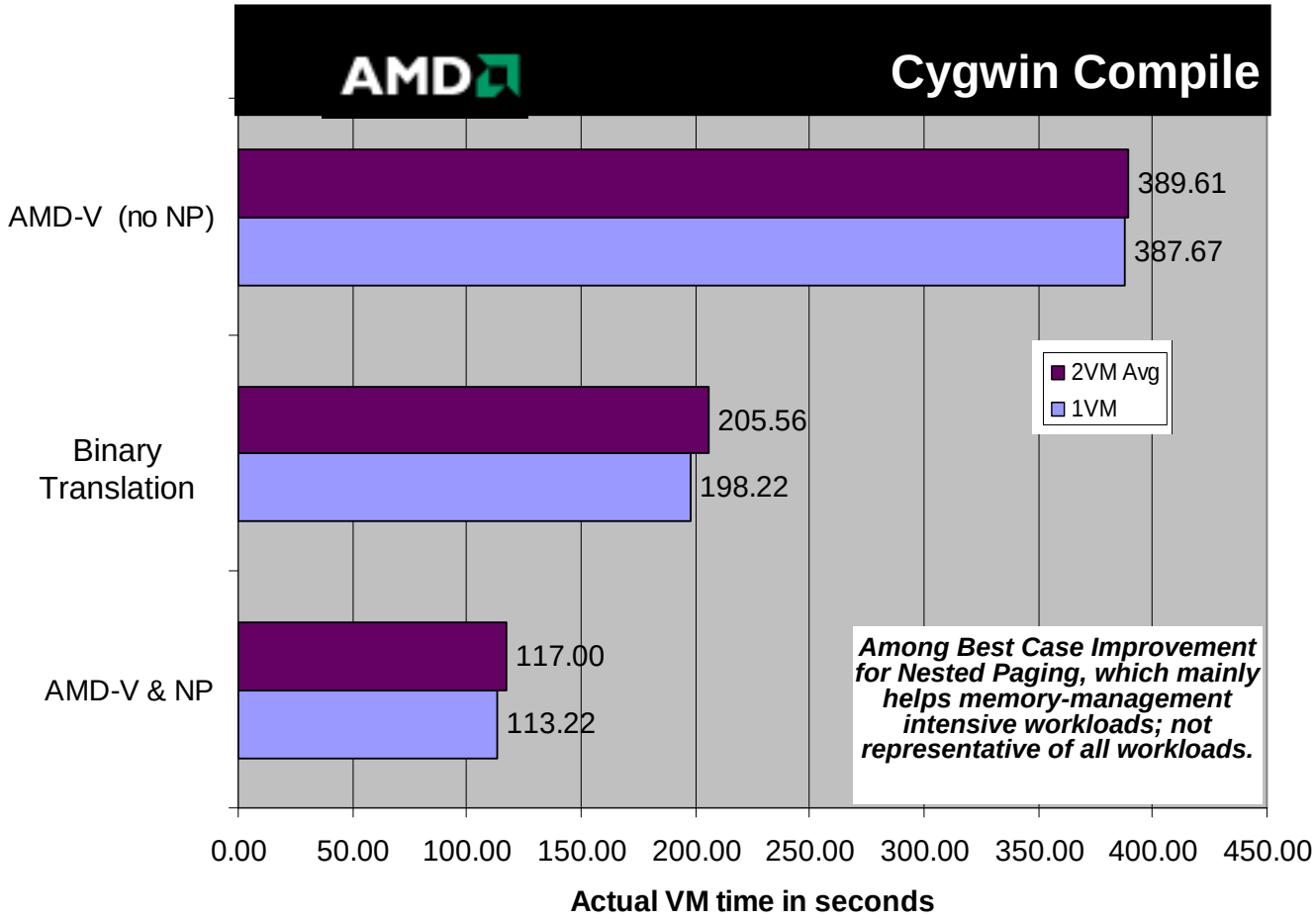


- Traditionally the hypervisor maintains shadow page tables:
  - Expensive to emulate correct behavior (accessed/modified bits)
- Nested paging eliminates this by performing a recursive walk
  - Available in Barcelona
  - Reduces number of #VMEXITs by 40-70%

Nested Paging Page Entry Accesses



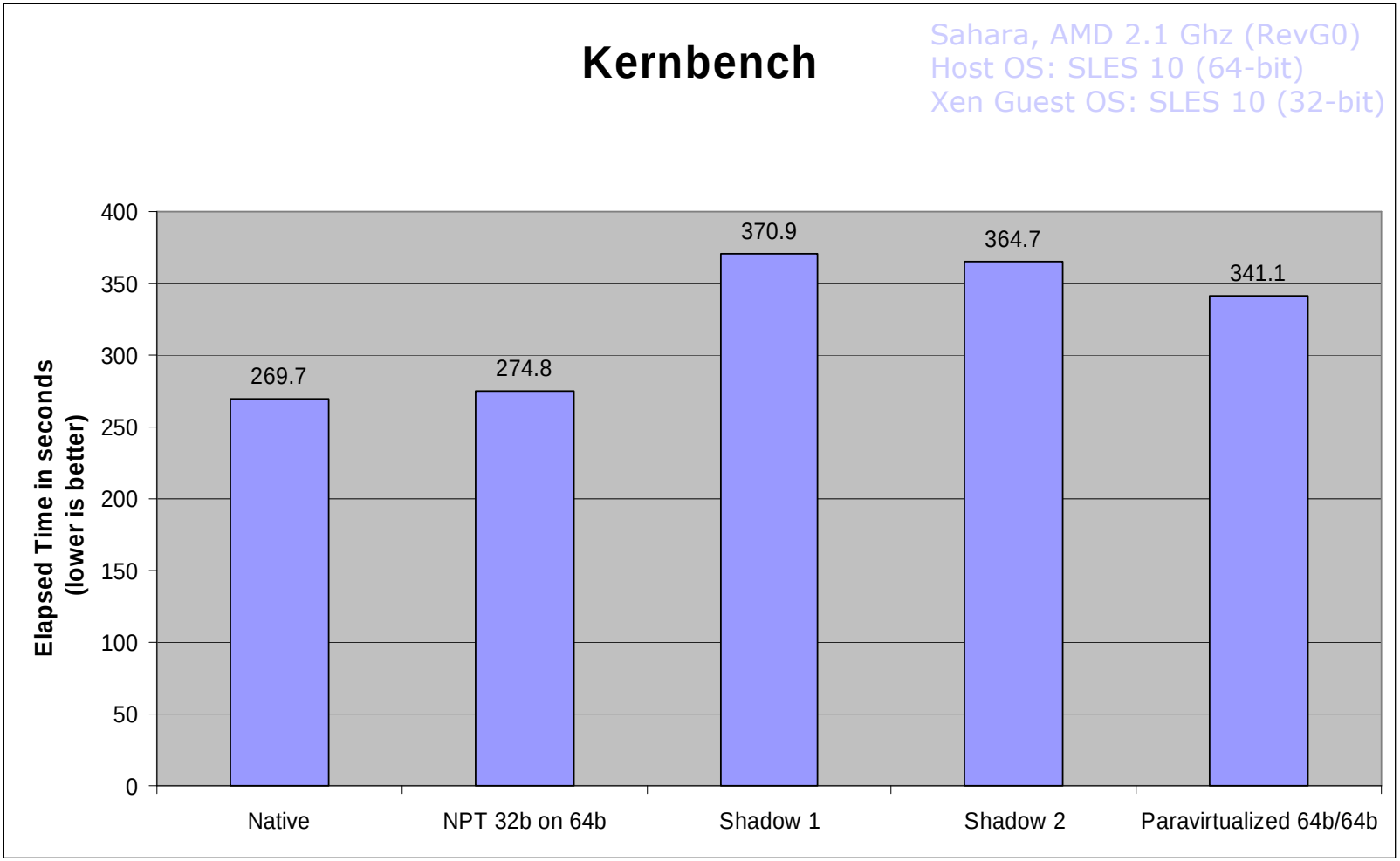
Cygwin compile with AMD Nested Paging



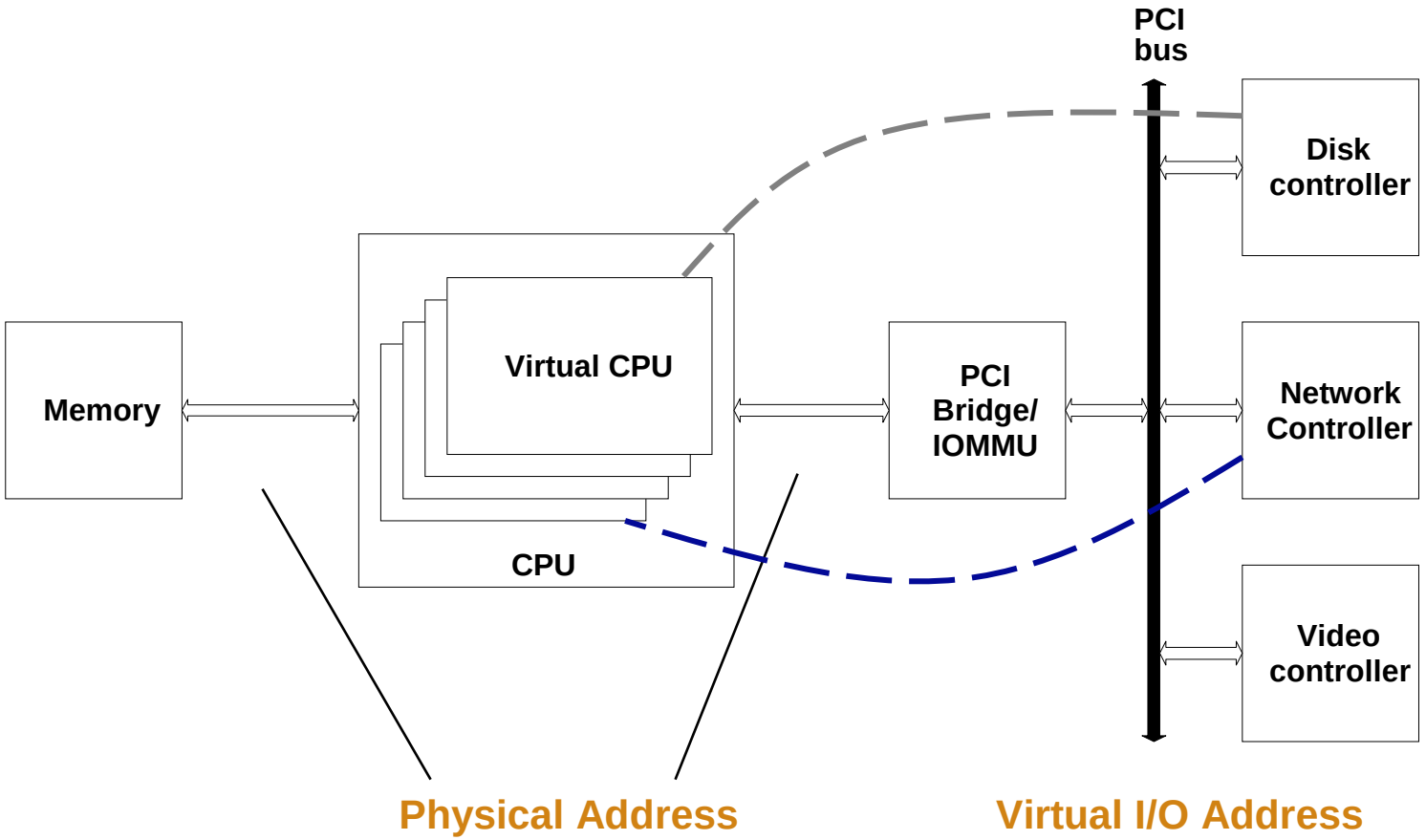
Nested Paging reduces compile time by 43%

Platform: Experimental AMD Processor with Nested Paging running experimental build of VMware Workstation.

Nested Page Table Performance



Direct Device Assignment



- Assign devices directly to a guest VM
- Eliminate IPCs to service OS
- IOMMU isolates busmaster capable devices

Over Committing Memory Resources

- Scaling the number of VMs per core requires memory over commitment
  - Per core: 32 VMs x 2G *versus* 32 VMs x 100 MB (working set)
  - Use paging or memory compaction
  - VMWare collapses memory pages with the same content into one and uses copy-on-write to disaggregate if necessary
  - Depending on workloads, this results in 7-33% memory compaction (*Memory Resource Management in VMware ESX Server*, OSDI'02)
- This does not work for the first generation IOMMU designs
  - You cannot restart PCI operations
  - Even if you make PCI restartable or pinning you still have to deal with devices that do not do end-to-end flow control signaling
  - How to deal with VM migration?
- Hardware support for memory compaction?

Virtual Machine Migration

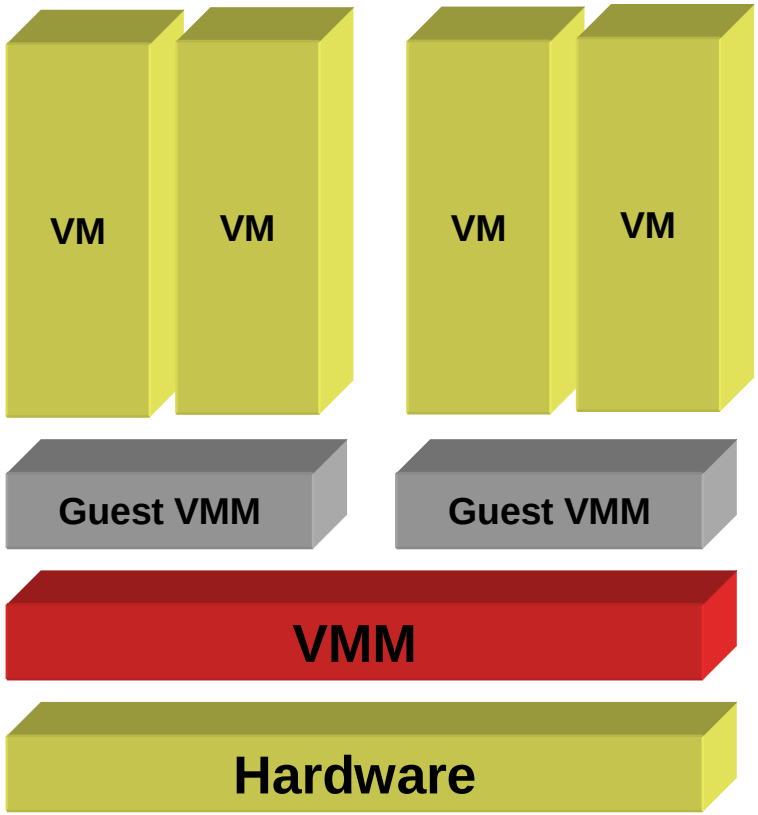
- Move a running VM to another machine
  - For example: Maintenance and load rebalancing
- *Easy* when moving between same CPU models
- Issues with migrating between different CPU models?
  - CPUID masquerading
  - New CPU opcodes means no longer cause #UD
  - Emulating new opcodes on old CPUs
  - Emulating old opcodes on new CPUs
  - Differences in FP significance
- Do you provide a bit vector to enable/disable features?
- Do you support *N* generations (Power6)?
- How much of a problem is this actually?
  - Software really should obey CPUID, but doesn't always
  - Vendors want 100% case coverage; is this really needed?
  - Opcode set enable is filled with problems

Improve Platform Reliability

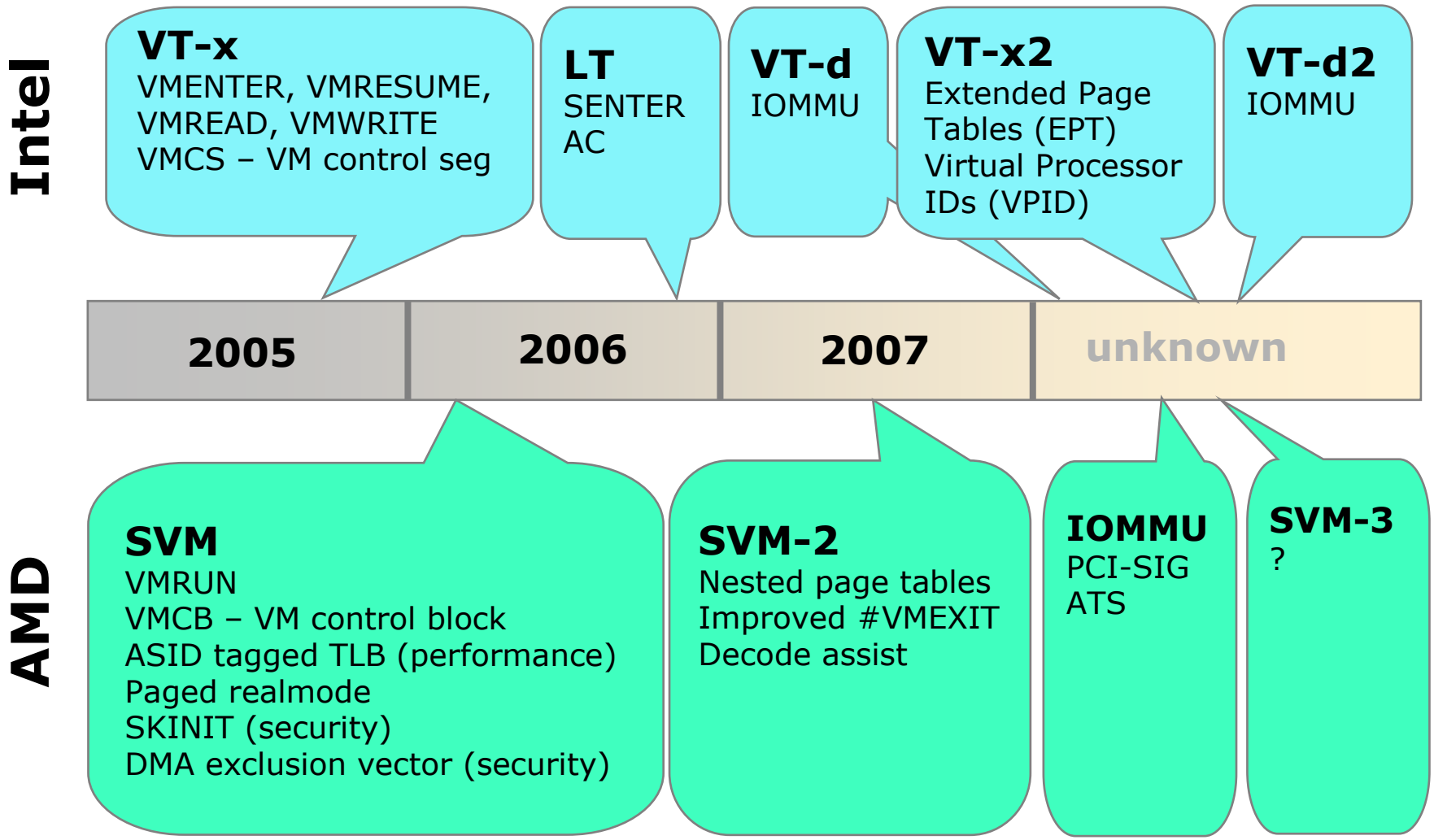
- What does it take for customers to virtualize their production environments or to enable utility computing?
  - Improved Reliability, Availability, and Serviceability (RAS)
- Not economic to have mainframe RAS capabilities in x86 commodity space
  - In most cases it is not necessary to give dual execution guarantees on all customer data
  - At reduced performance, you can implement active VM replication using a VMM
  - You need error detection and a certain level of repair (e.g. sparing, data poisoning)
  - And a notification mechanism so that management software can migrate VMs away from the faulting platform

Nested Virtualization

- Enable VMMs to run as guests
  - Akin to z/VM 2nd level guests
  - Allows different hypervisors to co-exist
  - Use binary translation for the 1st level guest?
  - Make VMM aware of nesting, 1..N-1 aware, N can be unaware
- Open issues
  - Is it transparent to the VMM?
  - Performance impact & complexity?
  - z/VM is mainly used by devtest
  - Could we partition cores instead?



Intel / AMD Comparison

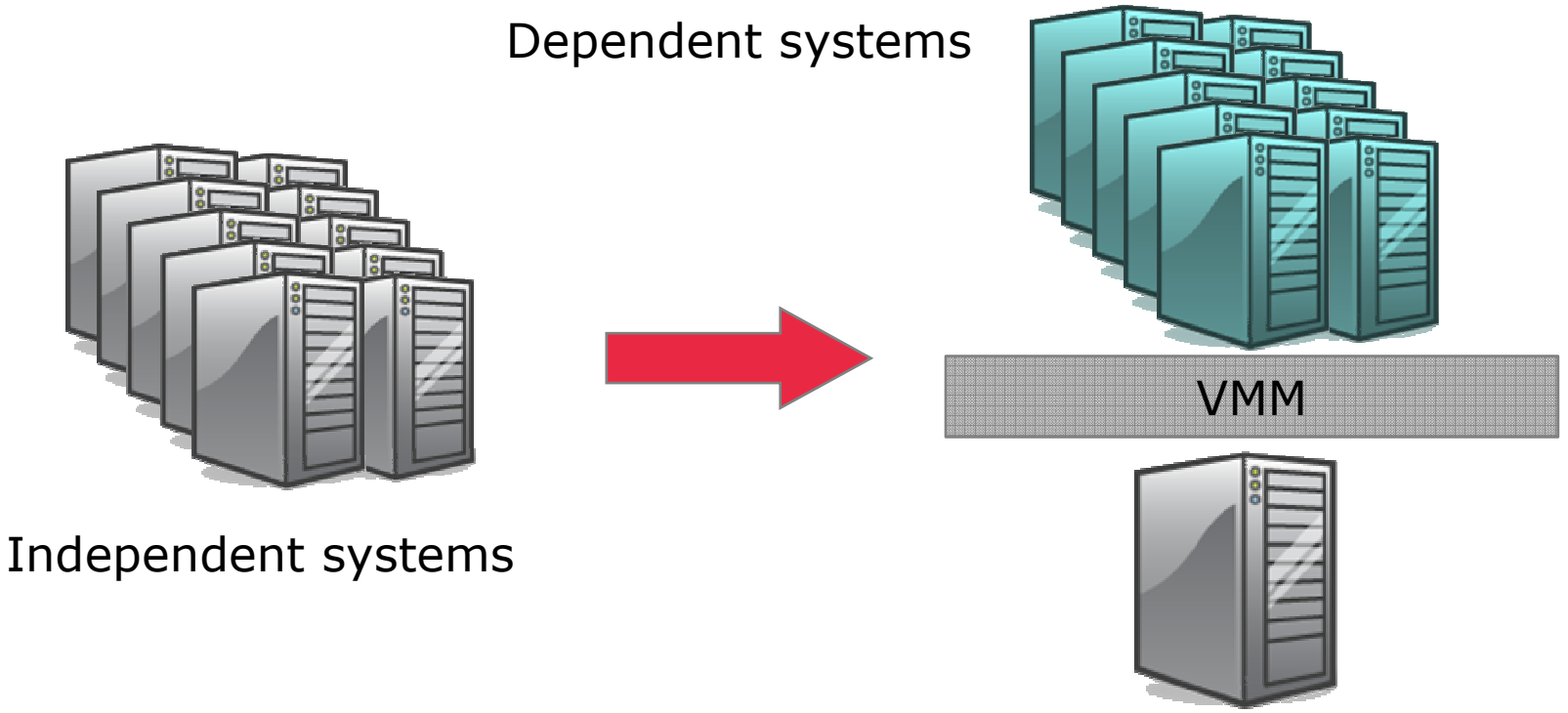


Hypervisor Software Landscape

- VMware is the undisputed leader in the x86 virtualization space
  - Its binary translation technology is currently superior
  - Only uses VT-x on x86-64 because unlike AMD, Intel does not provide long mode segment limits
  - Very mature product
- Xen is an open source hypervisor shipped as part of RedHat and Suse Linux, virtual Iron
  - Uses paravirtualization for modified Linux
  - SVM/VT-x for unmodified guest OS support

  KVM is being shipped as part of RedHat
  - Uses SVM/VT-x
  - Linux module
- Microsoft Viridian
  - Uses SVM/VT-x for CPU virtualization and paravirtualized device drivers
  - Still in development, released 180-days after Longhorn server

Virtualization is not a Panacea



- Increasing utilization through consolidation decreases the reliability
  - Need better hardware reliability, error reporting, and fault tolerance
  - Need better software fault isolation

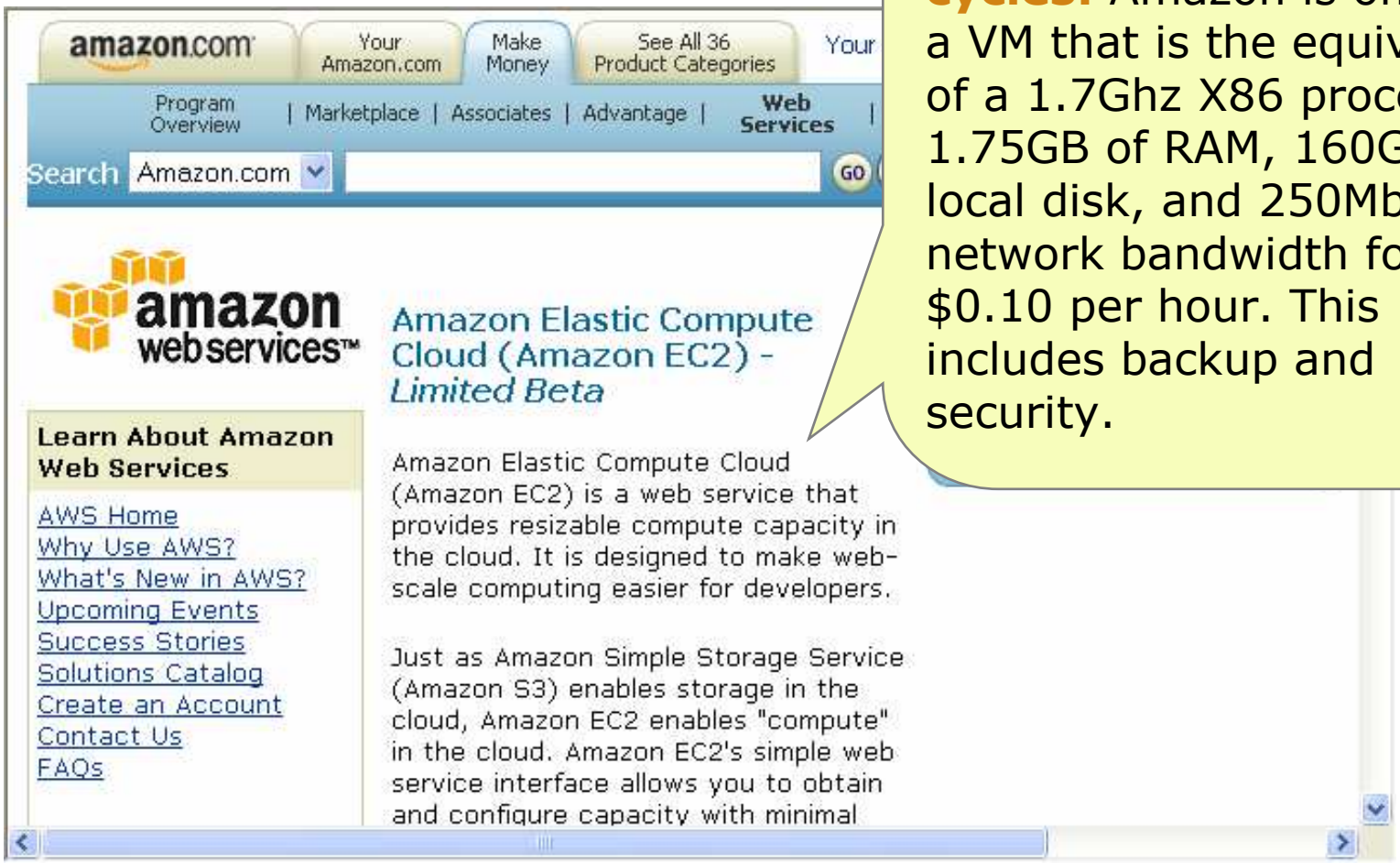
Server Workloads Are Changing!

- Utility computing is a disruptive business model
  - Very attractive for small and medium businesses
  - Managed security, backups and hardware upgrades
  - Heavily depends on virtualization

- Open issues
  - Improve platform reliability (RAS)
  - Improve software reliability (fault isolation)
  - Add per VM QoS guarantees and billing capabilities
  - How to scale the number of VMs significantly?

    World switch times, direct device access, number of cached VMCBs, over commit resources, …

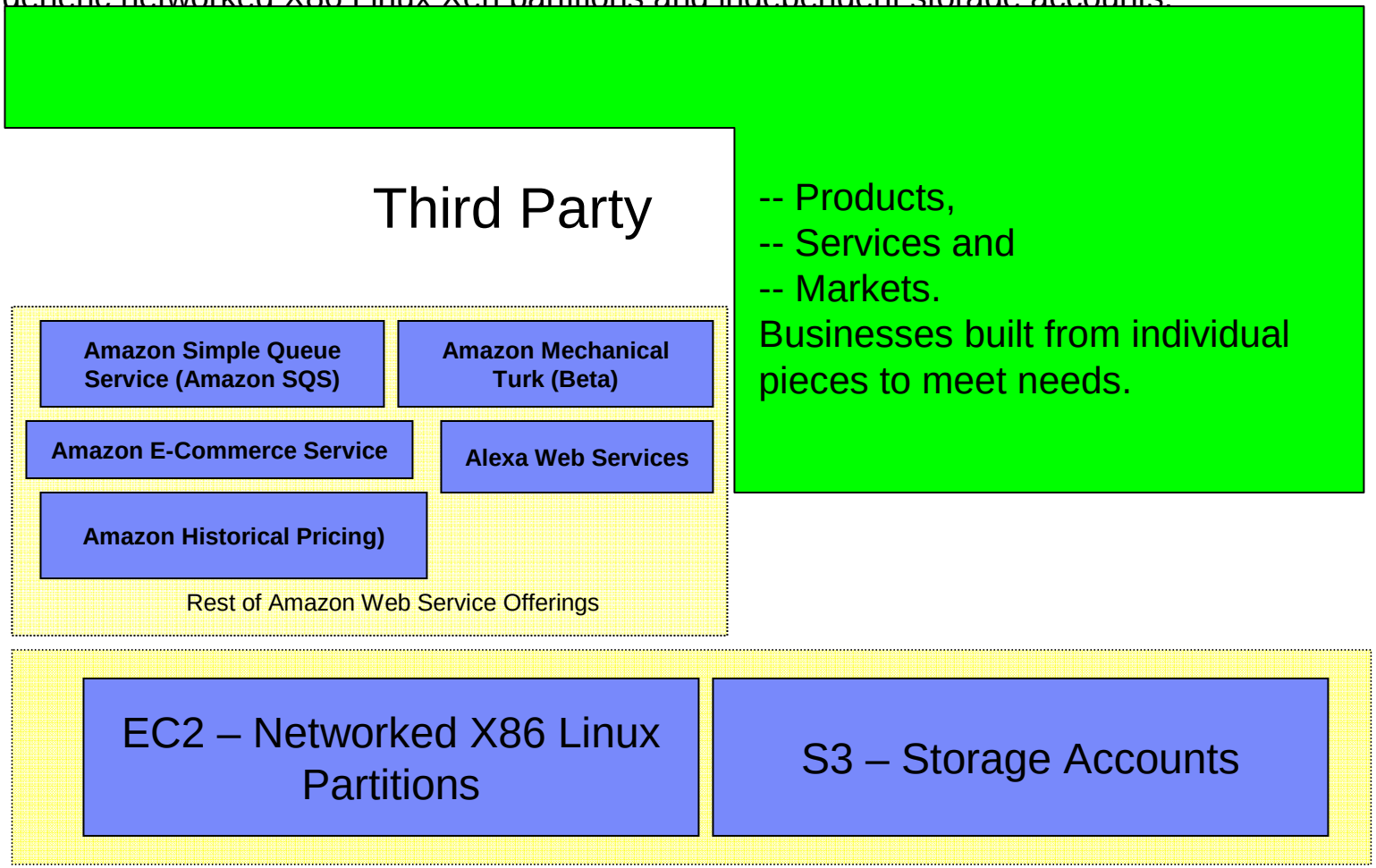
Example: Utility Computing



Google for computing cycles: Amazon is offering a VM that is the equivalent of a 1.7Ghz X86 processor, 1.75GB of RAM, 160GB of local disk, and 250Mb/s of network bandwidth for $0.10 per hour. This includes backup and security.

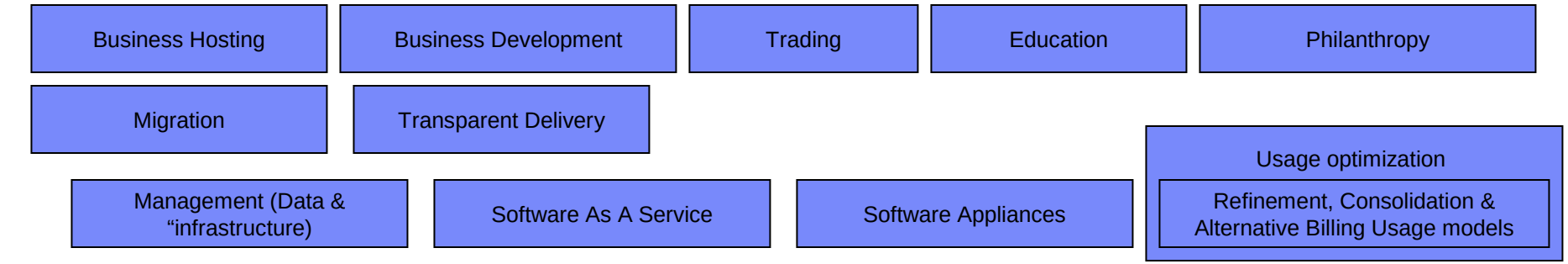
What Amazon is Offering

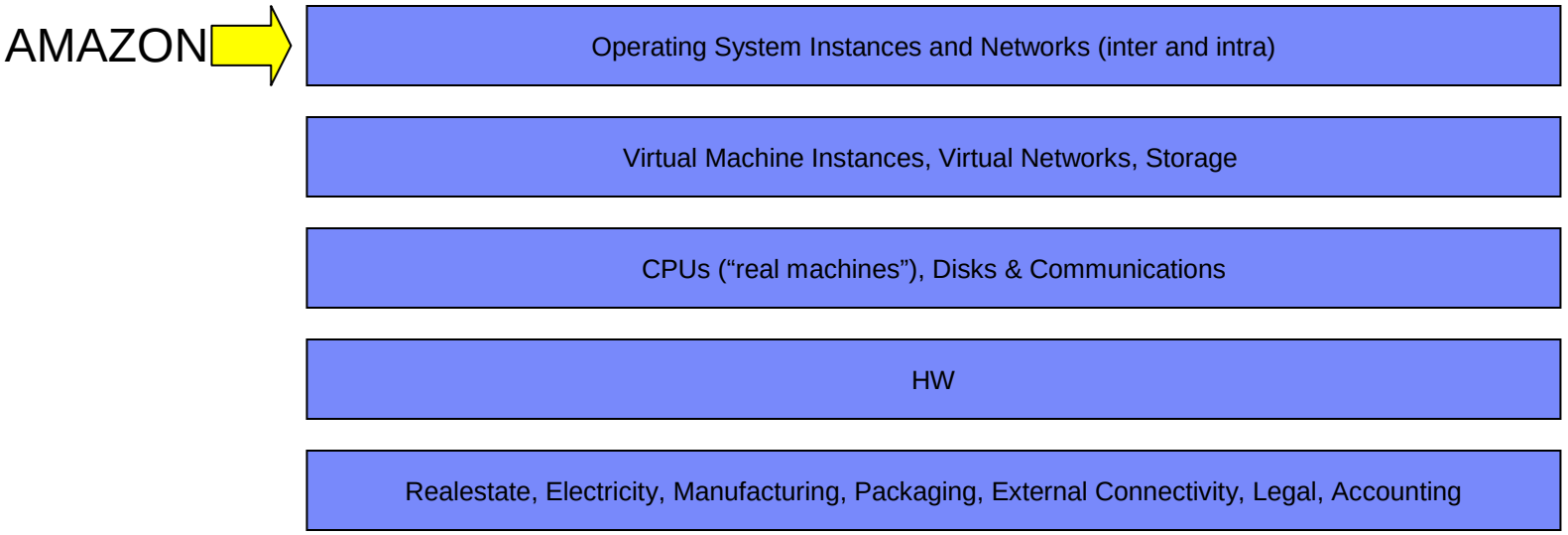
A constellation of independent Amazon products (building blocks) for constructing and running businesses on top of an Amazon provided compute, communications and storage capacity packaged as generic networked X86 Linux Xen partitions and independent storage accounts.



Key to Success : Fine grain Decomposition of products and services leading to fine grain decomposition of value and commitments.

The Phenomena – Less is More



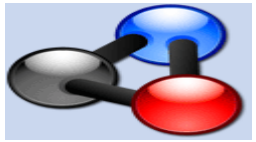


The Amazon "Beta" Emerging Market

Unlike the Google-verse the Amazon-verse is emergent, self-sustaining, competitive and market driven. Others are refining and reselling EC2 thus driving up Amazon's acceptance and revenue. Even Universities! All that is necessary is the provisioning of the minimal building block that others can refine. Enable ingenuity – many people have good ideas but all of them requires resources to realize! And those that are successful need to scale instantaneously.

Eswap.com

quizical ALPHA



Gumiyo CONNECTING BUYERS AND SELLERS

m dialogue

CROSSCHECK networks

SECOND LIFE Your World. Your Imagination.

Webmail.us

distributedpotential PAY-PER-USE GRID COMPUTING CAPACITY

openfount

BaseJumpr

cruxy

RightScale

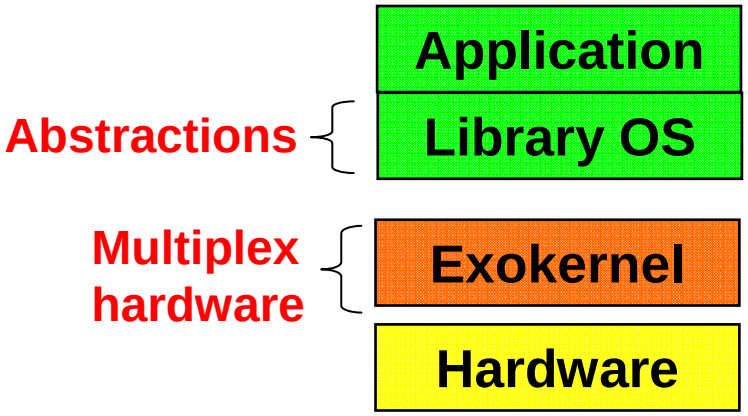
wisnet.com HIGH PERFORMANCE WEBSITES

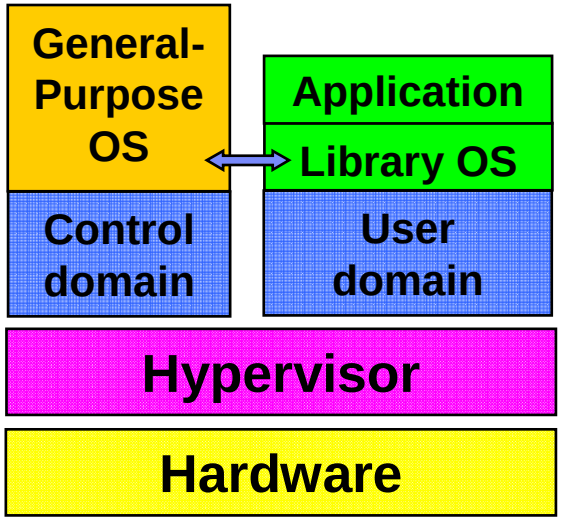
PODMAILING.COM BETA

A little web surfing produced the above… this by no means is complete and some of these are portals for large usage models

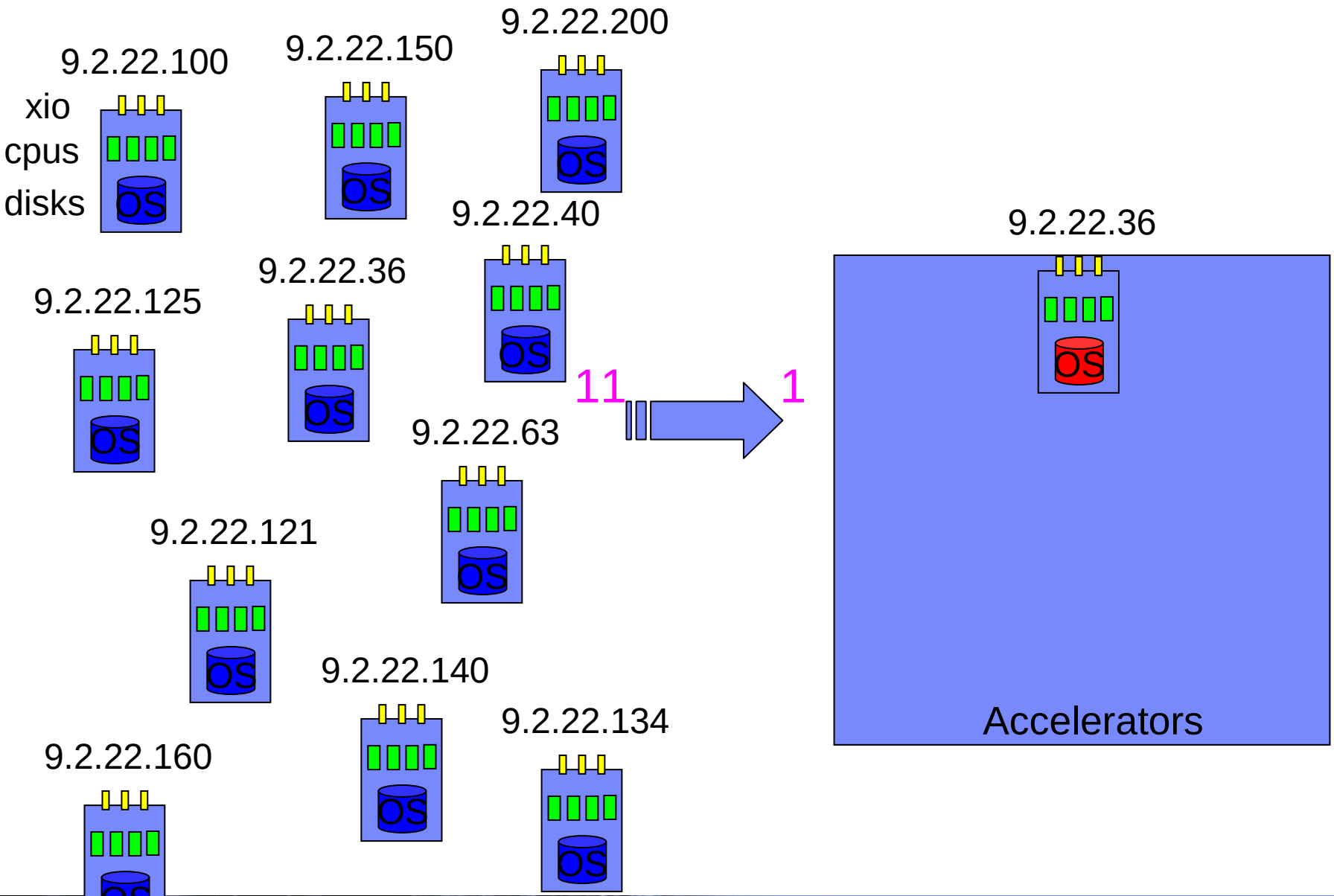
Current Exploration of Virtualization : library OS approach

- Customized operating system support for applications
- Previous approaches
  - SPIN, Vino, Scout, K42
  - Exokernel
- Virtualization – new opportunity



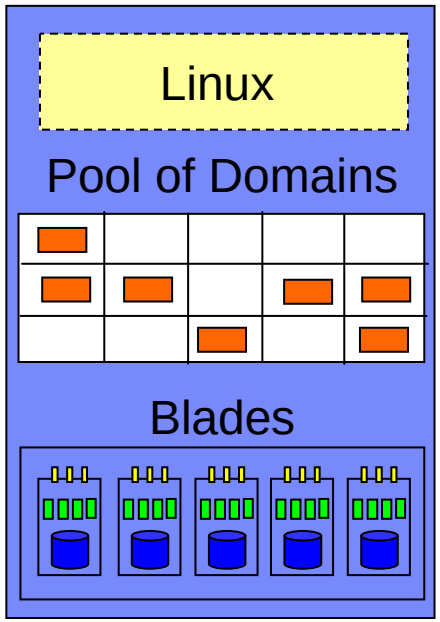


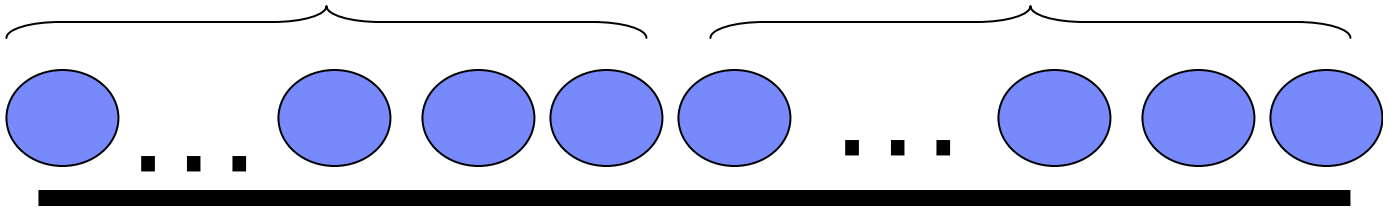
Libra: a library OS for JVMs


9.2.22.100
xio
cpus
disks
9.2.22.150
9.2.22.200
9.2.22.40
9.2.22.36
9.2.22.125
9.2.22.63
11
1
9.2.22.36
9.2.22.121
9.2.22.140
9.2.22.134
9.2.22.160
Accelerators

```
$ ssh chassis0
chassis0 > java HelloWorld
```

Virtual Chassis 0





Terminology: Virtualization Models

- Virtual Environments
  - **Solaris Containers; AIX Corrals/WPAR; Linux VServers, FreeVPS, OpenVZ**
- Full Virtualization
  - **VMware; Parallels; Microsoft; zVM; Xen, KVM**
- Para Virtualization
  - **VMware VMI; PHYP; Xen, KVM, Para-virt; Microsoft-Xen partnership**
- Enlightened guest OS (Microsoft terminology)

Take Away Points

1. Workloads are changing and we do not have good insight into how (especially true for servers)
   - What happens when you run at 100% utilization all the time?
   - What to cache?
   - What are the right bandwidths?
2. Further adoption of virtualization requires improved platform reliability (RAS)
   - Platform consolidation reduces overall reliability
3. How to scale the number of VMs per core?
   - Reduce the cost or eliminate world-switches
   - Over-commit memory resources

Acknowledgements

- Jimi Xenidis (IBM, XenPPC leader)
- Orran Krieger (ex-IBM, now VMware)
- Leendert Van Doorn (ex-IBM, now AMD)