VMware Performance for Gurus

Richard McDougall

Principal Engineer, VMware, Inc

rmc@vmware.com *@richardmcdougll*

Usenix Tutorial, December, 2010

Abstract

- This class teaches the fundamentals of performance and observability for vSphere virtualization technology.
- The objective of the class is to learn how to be a practitioner of performance diagnosis and capacity planning with vSphere.
- We use a combination of introductory vSphere *internals* and performance analysis techniques to expose what's going on under the covers, learn how to interpret metrics, and how to triage performance problems.
- We'll learn how to interpret load measurements, to perform accurate capacity planning.

Credits

- **Thank you to the many contributors of slides and drawings, including:**
  - Ravi Soundararajan – VC and esxtop
  - Andrei Dorofeev – Scheduling
  - Patrick Tullmann – Architecture
  - Bing Tsai – Storage
  - Howie Xu - Networking
  - Scott Drummonds – Performance
  - Devaki Kulkarni - Tuning
  - Jeff Buell – Tuning
  - Irfan Ahmad – Storage & IO
  - Krishna Raj Raja – Performance
  - Kit Colbert – Memory
  - Ole Agesen – Monitor Overview
  - Sreekanth Setty - Networking
  - Ajay Gulati - Storage
  - Wei Zhang - Networking
  - Amar Padmanabhan – Networking

Agenda/Topics

- Introduction
- Performance Monitoring
- CPU
- Memory
- I/O and Storage
- Networking
- Applications

INTRODUCTION TO

VIRTUALIZATION

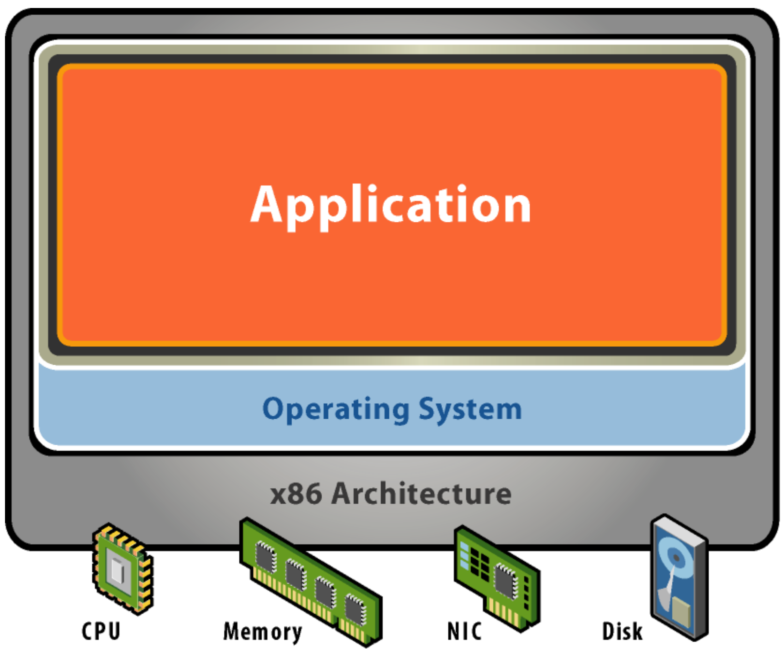
AND

VMWARE VI/ESX

Traditional Architecture

Operating system performs various roles

- Application Runtime Libraries
- Resource Management (CPU, Memory etc)
- Hardware + Driver management



> Performance & Scalability of the OS was paramount

> Performance Observability tools are a feature of the OS

The Virtualized World

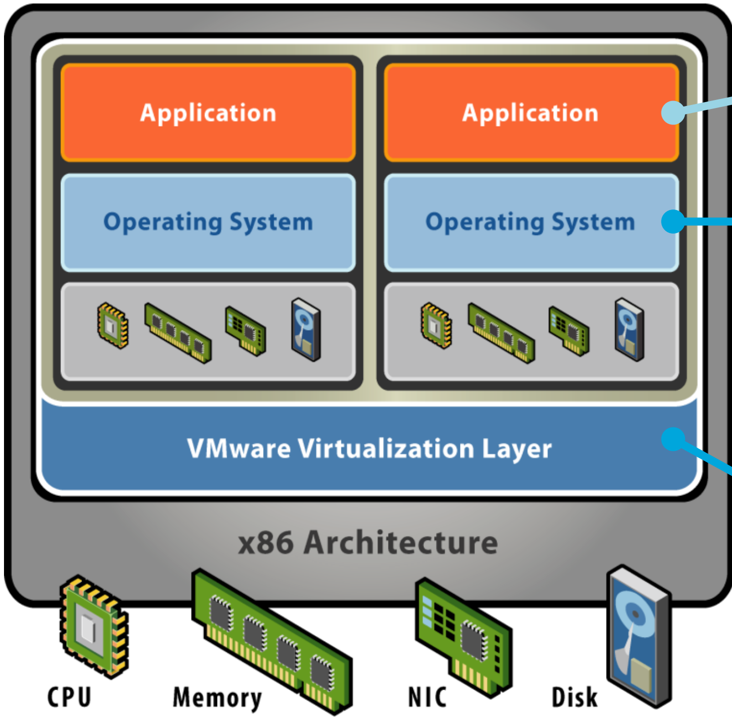
The OS takes on the role of a Library, Virtualization layer grows

Application

Run-time Libraries and Services

Application-Level Service Management

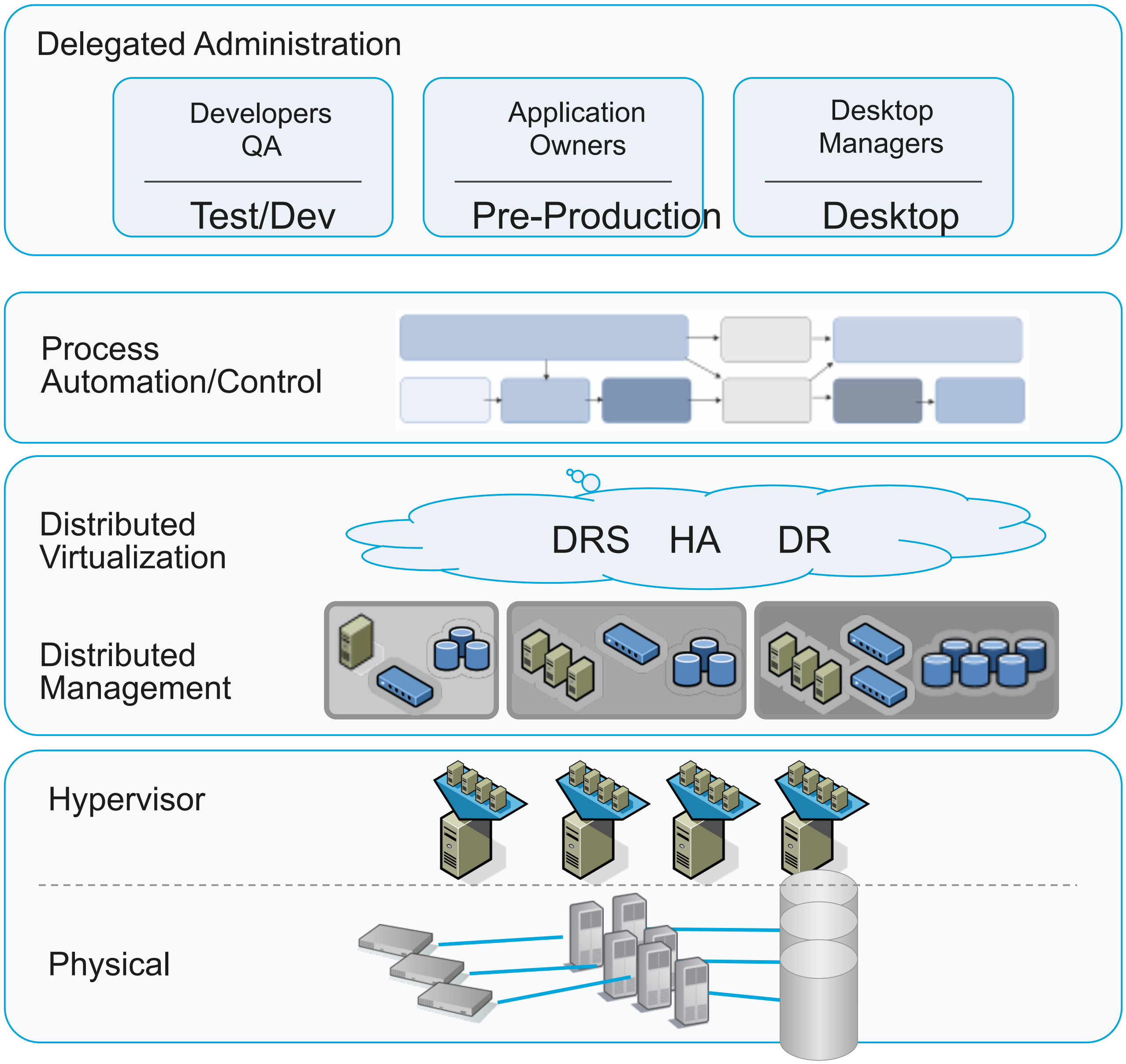
Application-decomposition of performance



Run-time or Deployment OS
Local Scheduling and Memory Management
Local File System

Infrastructure OS (Virtualization Layer)
Scheduling
Resource Management
Device Drivers
I/O Stack
File System
Volume Management
Network QoS
Firewall
Power Management
Fault Management
Performance Observability of System Resources

vShere Platform



DBAs get their
Own per-DB Sandbox

Rapid, Templated
DB Provisioning

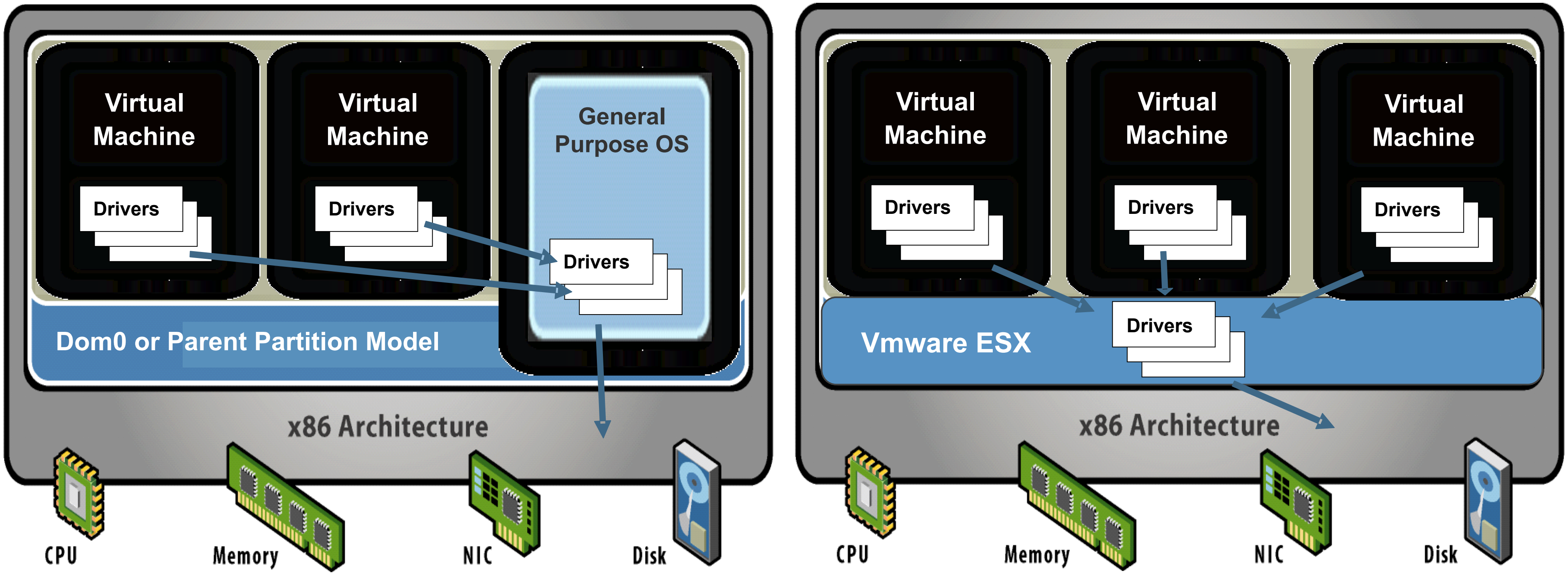
Resource Management
Availability, DR

Virtual, Portable
DB Instances

High Performance
Scalable Consolidation

Storage Virtualization

Hypervisor Architectures



Very Small Hypervisor

General purpose OS in parent partition for I/O and management

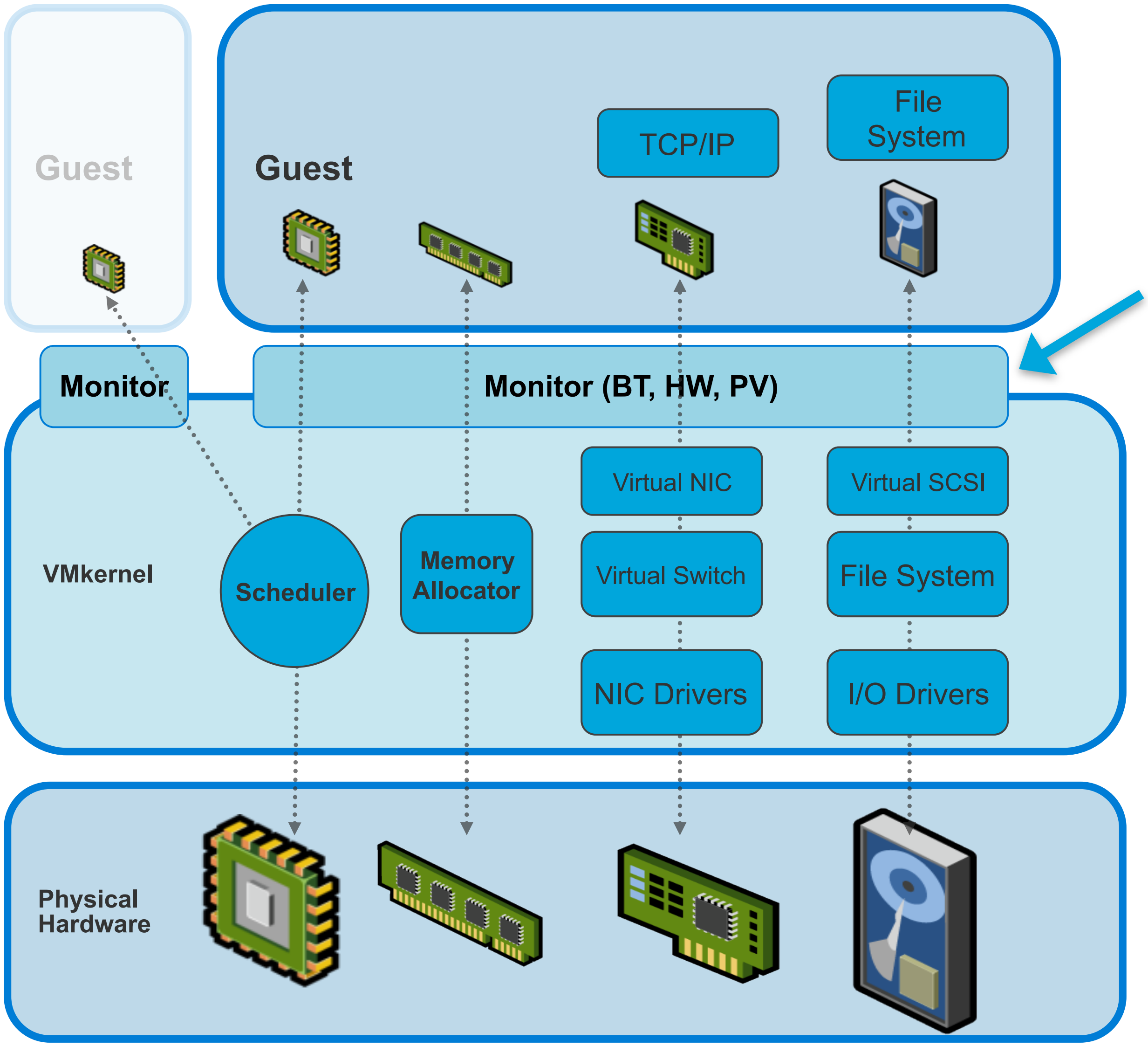
All I/O driver traffic going thru parent OS

Extra Latency, Less control of I/O

ESX Server

- Small Hypervisor < 24 mb
- Specialized Virtualization Kernel
- Direct driver model
- Management VMs
  - Remote CLI, CIM, VI API

VMware ESX Architecture



CPU is controlled by scheduler and virtualized by monitor

Monitor supports:

- BT (Binary Translation)
- HW (Hardware assist)
- PV (Paravirtualization)

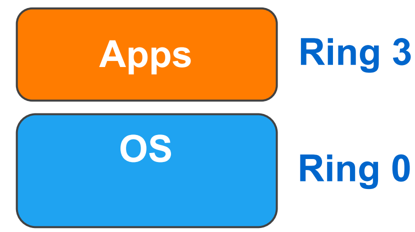
Memory is allocated by the VMkernel and virtualized by the monitor

Network and I/O devices are emulated and proxied though native device drivers

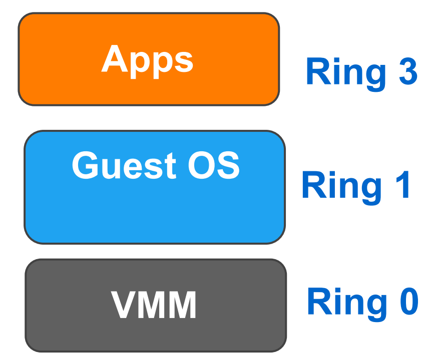
Inside the Monitor: Classical Instruction Virtualization

Trap-and-emulate

- **Nonvirtualized ("native") system**
  - OS runs in privileged mode
  - OS "owns" the hardware
  - Application code has less privilege



- **Virtualized**
  - VMM most privileged (for isolation)
  - Classical "ring compression" or "de-privileging"
    - Run guest OS kernel in Ring 1
    - Privileged instructions trap; emulated by VMM
  - But: does not work for x86 (lack of traps)



Classical VM performance

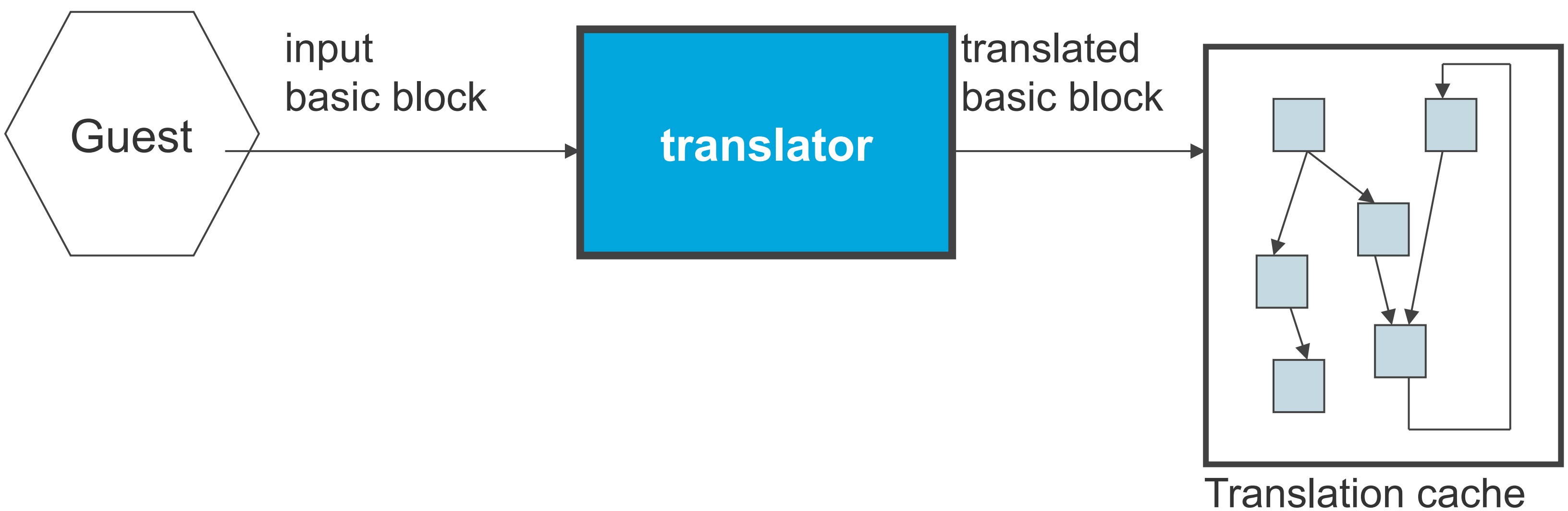
- **Native speed except for traps**
  - Overhead = trap frequency * average trap cost
- **Trap sources:**
  - Privileged instructions
  - Page table updates (to support memory virtualization)
  - Memory-mapped devices
- **Back-of-the-envelope numbers:**
  - Trap cost is high on deeply pipelined CPUs: ~1000 cycles
  - Trap frequency is high for “tough” workloads: 50 kHz or greater
  - Bottom line: substantial overhead

Binary Translation of Guest Code

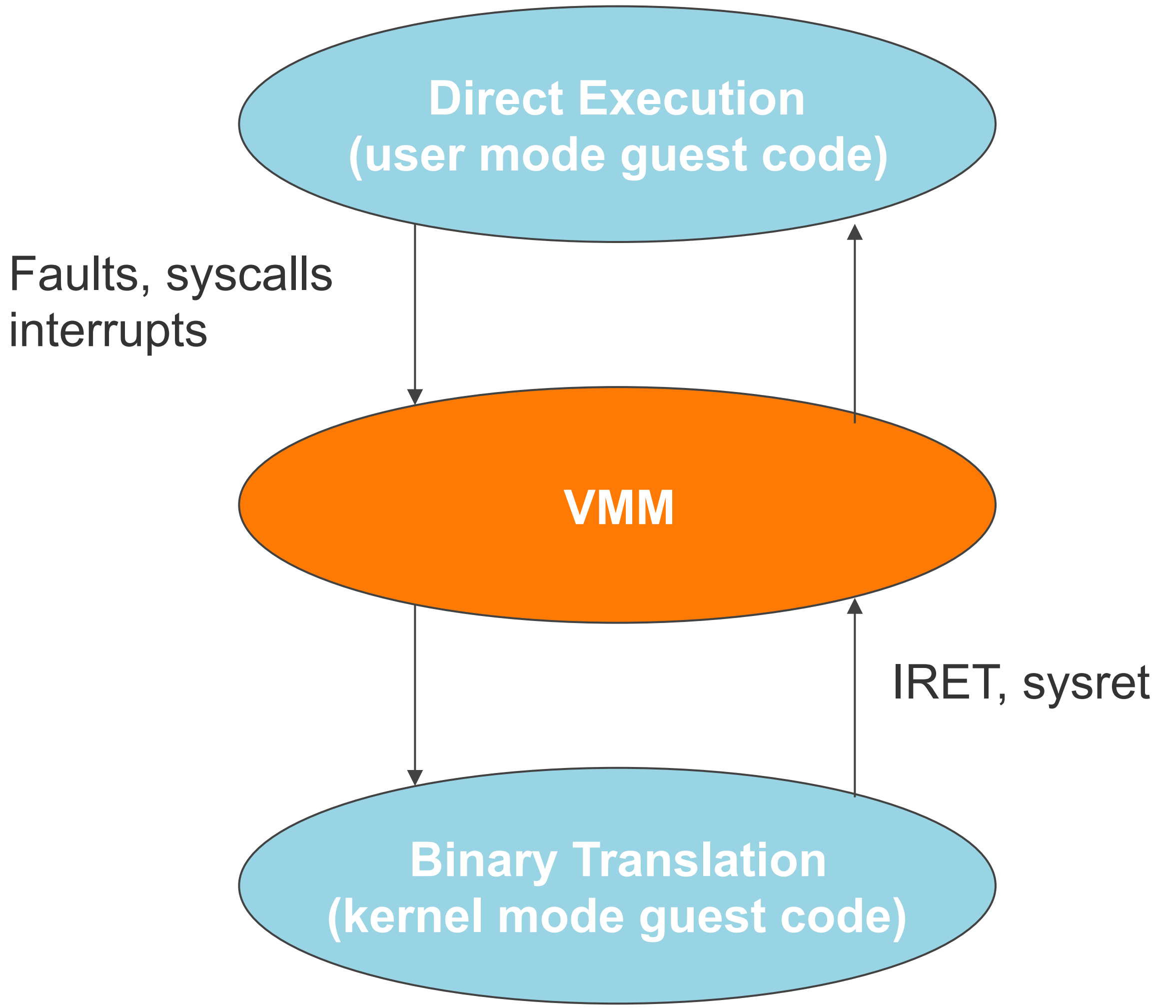
- **Translate guest kernel code**
- **Replace privileged instrs with safe “equivalent” instruction sequences**
- **No need for traps**
- **BT is an extremely powerful technology**
  - Permits *any* unmodified x86 OS to run in a VM
  - Can virtualize *any* instruction set

BT Mechanics

- **Each translator invocation**
  - Consume one input basic block (guest code)
  - Produce one output basic block
- **Store output in translation cache**
  - Future reuse
  - Amortize translation costs
  - Guest-transparent: no patching “in place”



Combining BT and Direct Execution

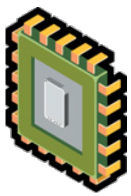
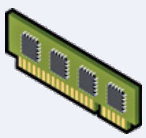
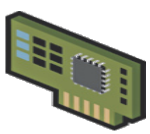


Performance of a BT-based VMM

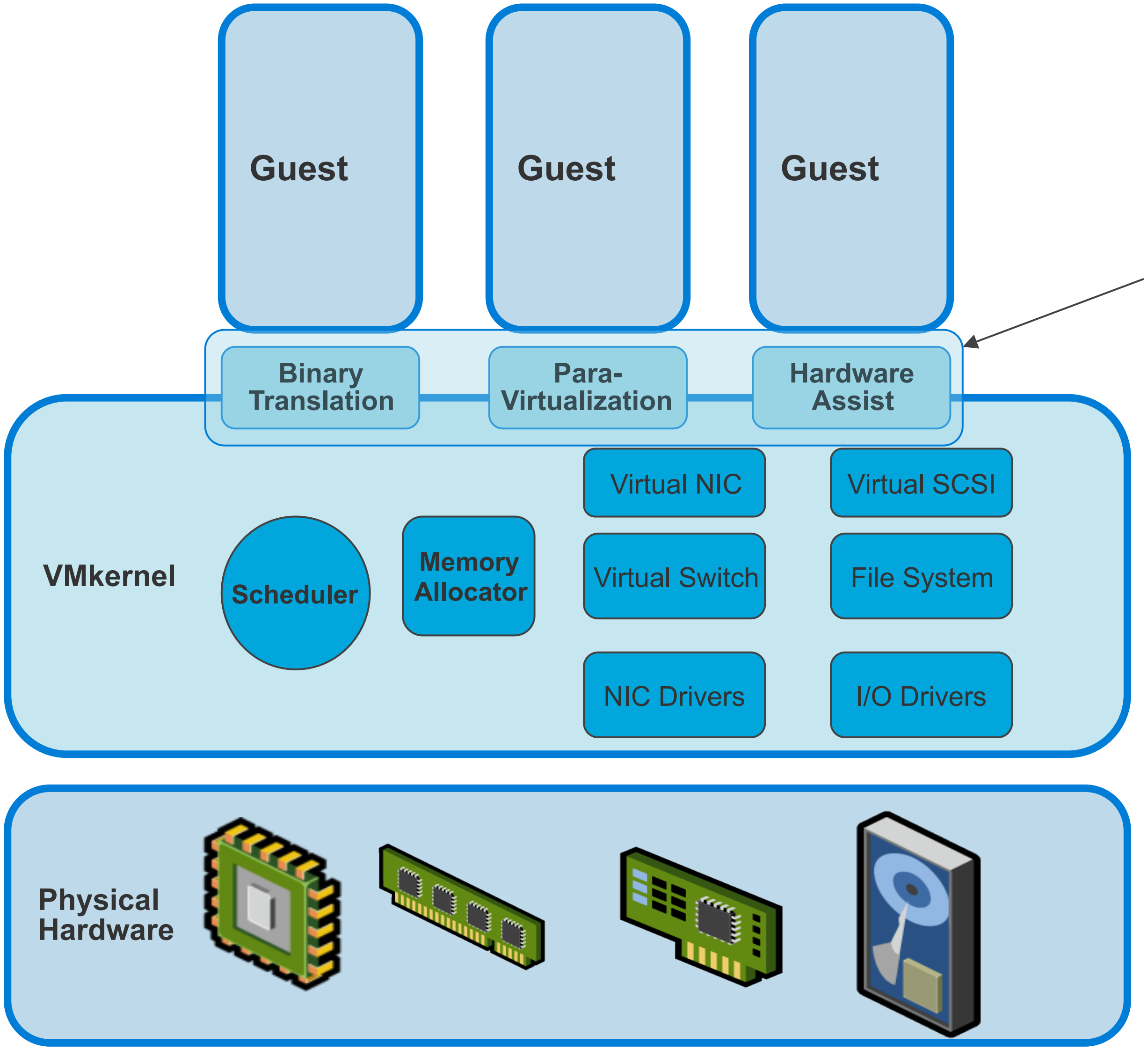
- **Costs**
  - Running the translator
  - Path lengthening: output is sometimes longer than input
  - System call overheads: DE/BT transition
- **Benefits**
  - Avoid costly traps
  - Most instructions need no change (“identical” translation)
  - Adaptation: adjust translation in response to guest behavior
    - Online profile-guided optimization
  - User-mode code runs at full speed (“direct execution”)

Speeding Up Virtualization

Technologies for optimizing performance

	**Privileged instruction virtualization**	Binary Translation, Paravirt. CPU Hardware Virtualization Assist
	**Memory virtualization**	Binary translation Paravirt. Memory Hardware Guest Page Tables
	**Device and I/O virtualization**	Paravirtualized Devices Stateless offload, Direct Mapped I/O

Multi-mode Monitors



There are different types of Monitors for different Workloads and CPU types

VMware ESX provides a dynamic framework to allow the best Monitor for the workload

Let’s look at some of the charactersitics of the different monitors

Virtualization Hardware Assist

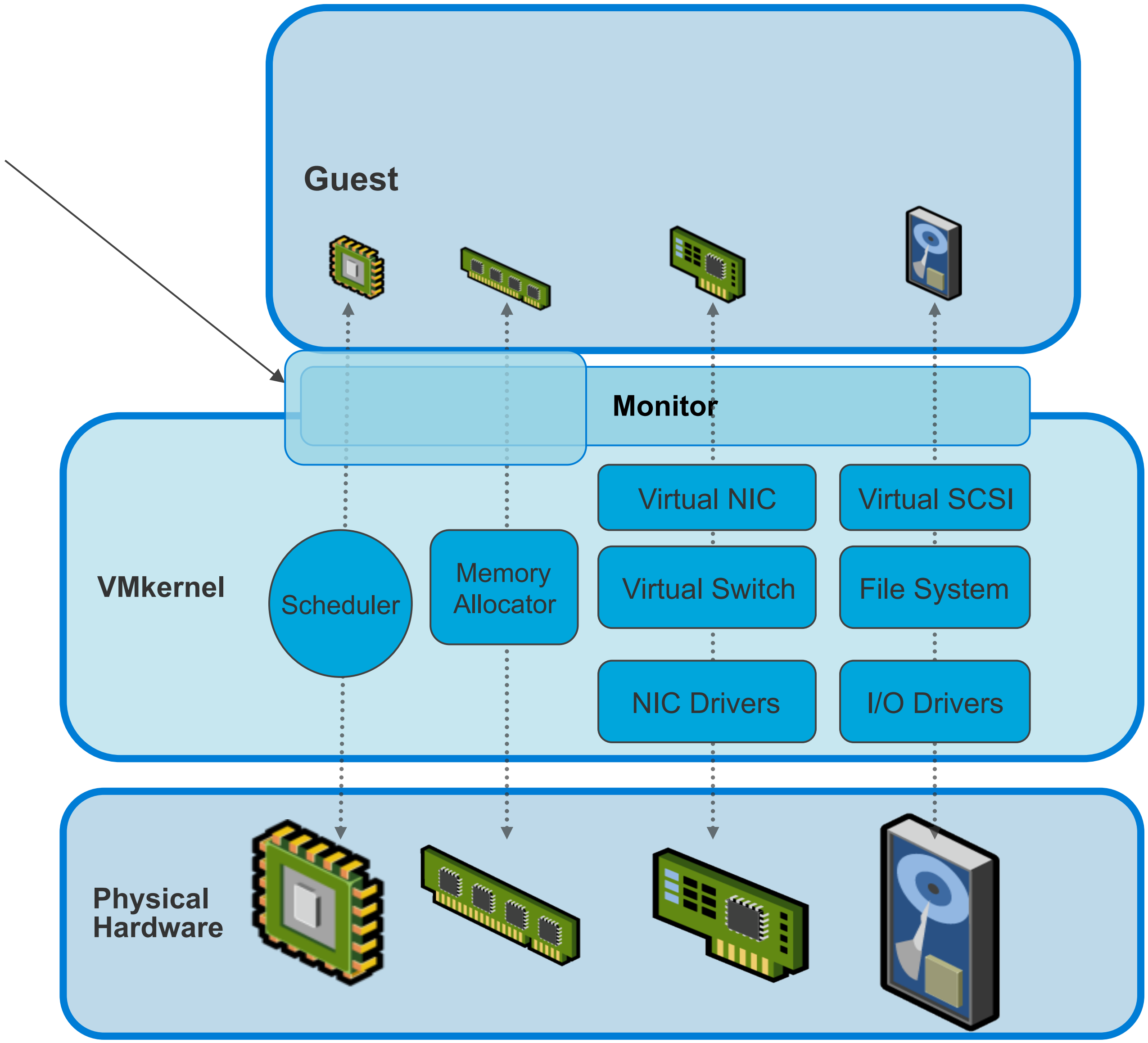
More recent CPUs have features to reduce some of the overhead at the monitor level

1st Gen: Intel VT and AMD-V

- doesn't remove all virtualization overheads: scheduling, memory management and I/O are still virtualized with a software layer

2nd Gen: AMD Barcelona RVI and Intel EPT

- Helps with memory virtualization overheads
- Most workloads run with less than 10% overhead
- EPT provides performance gains of up to 30% for MMU intensive benchmarks (Kernel Compile, Citrix etc)
- EPT provides performance gains of up to 500% for MMU intensive micro-benchmarks
- Far fewer "outlier" workloads

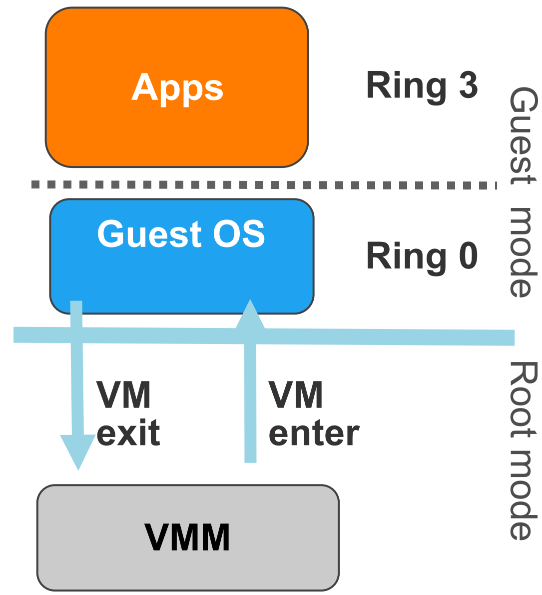


vSphere 4 Monitor Enhancements

- **8-VCPU virtual Machines**
  - Impressive scalability from 1-8 vCPUs
- **Monitor type chosen based on Guest OS and CPU model**
  - UI option to override the default
- **Support for upcoming processors with hardware memory virtualization**
  - Rapid Virtualization Indexing from AMD already supported
  - Extended Page Table from Intel
  - Improvements to software memory virtualization
- **Better Large Page Support (Unique to VMware ESX)**
  - (Includes enhancements in VMkernel)

Intel VT-x / AMD-V: 1st Generation HW Support

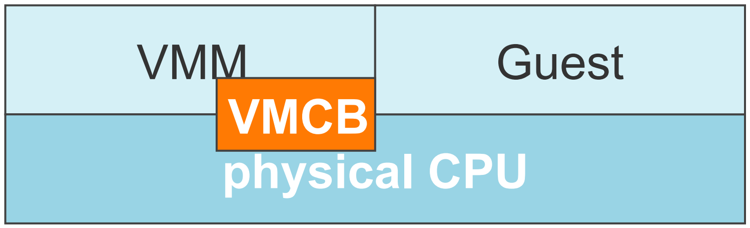
- **Key feature: root vs. guest CPU mode**
  - VMM executes in root mode
  - Guest (OS, apps) execute in guest mode
- **VMM and Guest run as "co-routines"**
  - VM enter
  - Guest runs
  - A while later: VM exit
  - VMM runs
  - ...



vmware

How VMM Controls Guest Execution

- **Hardware-defined structure**
  - Intel: VMCS (virtual machine control structure)
  - AMD: VMCB (virtual machine control block)
- **VMCB/VMCS contains**
  - Guest state
  - Control bits that define conditions for exit
    - Exit on IN, OUT, CPUID, ...
    - Exit on write to control register CR3
    - Exit on page fault, pending interrupt, ...
  - VMM uses control bits to “confine” and observe guest



Performance of a VT-x/AMD-V Based VMM

- **VMM only intervenes to handle exits**
- **Same performance equation as classical trap-and-emulate:**
  - overhead = exit frequency * average exit cost
- **VMCB/VMCS can avoid simple exits (e.g., enable/disable interrupts), but many exits remain**
  - Page table updates
  - Context switches
  - In/out
  - Interrupts

Qualitative Comparison of BT and VT-x/AMD-V

- **BT loses on:**
  - system calls
  - translator overheads
  - path lengthening
  - indirect control flow
- **BT wins on:**
  - page table updates (adaptation)
  - memory-mapped I/O (adapt.)
  - IN/OUT instructions
  - no traps for priv. instructions

- **VT-x/AMD-V loses on:**
  - exits (costlier than “callouts”)
  - no adaptation (cannot elim. exits)
  - page table updates
  - memory-mapped I/O
  - IN/OUT instructions
- **VT-x/AMD-V wins on:**
  - system calls
  - almost all code runs “directly”

Qualitative Comparison of BT and VT-x/AMD-V

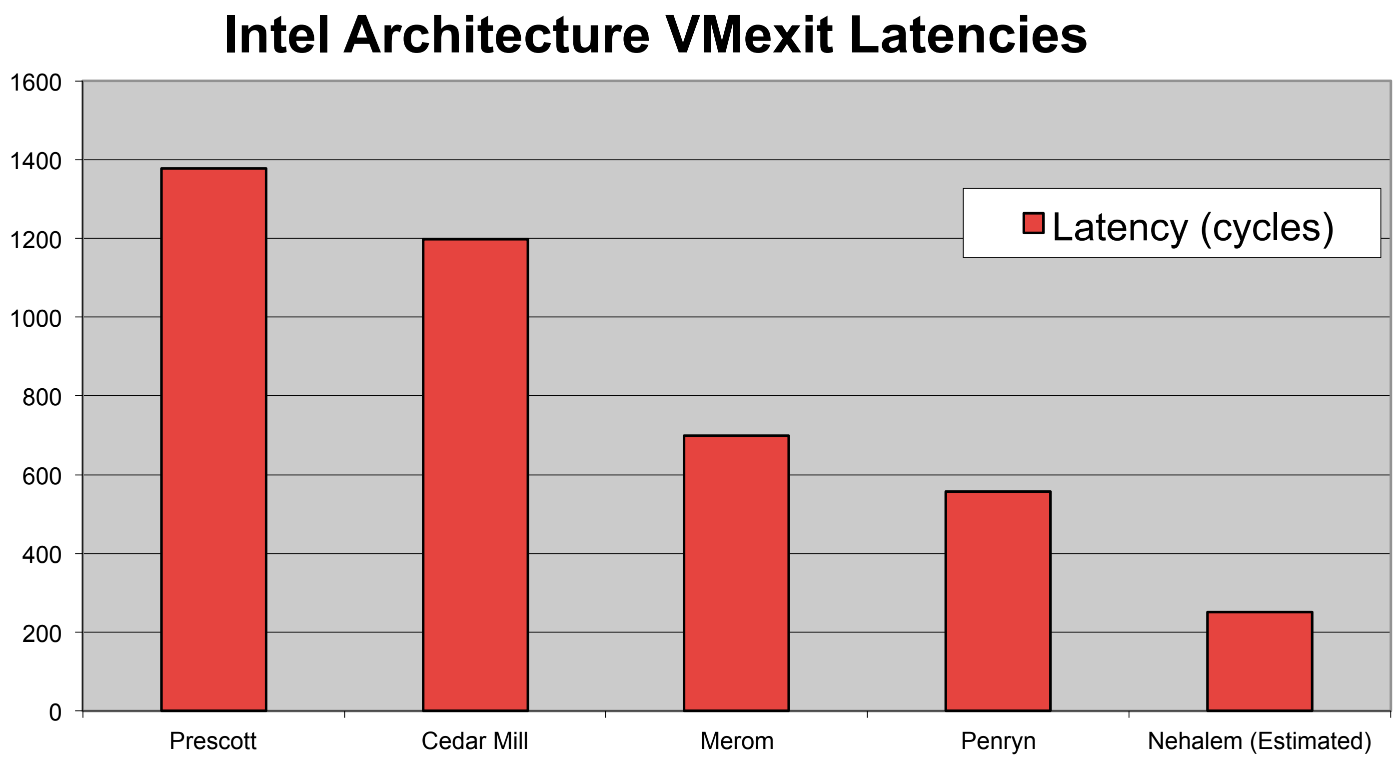
- **BT loses on:**
  - system calls
  - translator overheads
  - path lengthening
  - indirect control flow
- **BT wins on:**
  - page table updates (adaptation)
  - memory-mapped I/O (adapt.)
  - IN/OUT instructions
  - no traps for priv. instructions

- **VT-x/AMD-V loses on:**
  - exits (costlier than “callouts”)
  - no adaptation (cannot elim. exits)
  - page table updates
  - memory-mapped I/O
  - IN/OUT instructions
- **VT-x/AMD-V wins on:**
  - system calls
  - almost all code runs “directly”

Qualitative Comparison of BT and VT-x/AMD-V

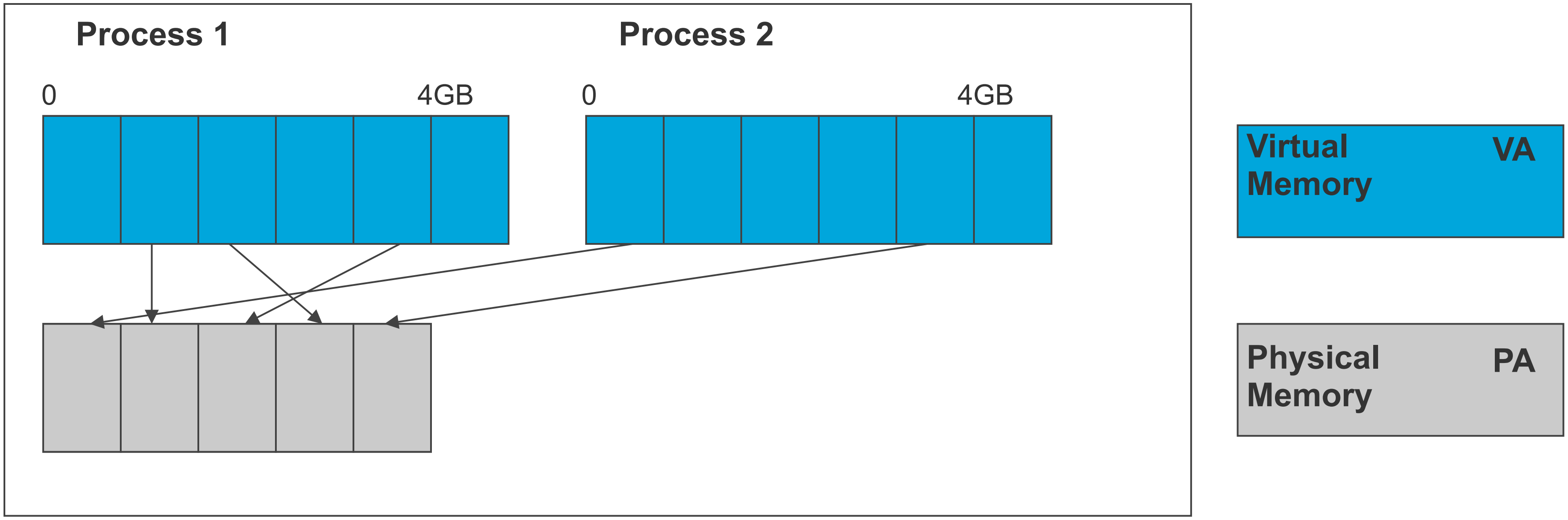
- **BT loses on:**
  - system calls
  - translator overheads
  - path lengthening
  - indirect control flow
- **BT wins on:**
  - page table updates (adaptation)
  - memory-mapped I/O (adapt.)
  - IN/OUT instructions
  - no traps for priv. instructions
- **VT-x/AMD-V loses on:**
  - exits (costlier than “callouts”)
  - no adaptation (cannot elim. exits)
  - page table updates
  - memory-mapped I/O
  - IN/OUT instructions
- **VT-x/AMD-V wins on:**
  - system calls
  - almost all code runs “directly”

VMexit Latencies are getting lower…



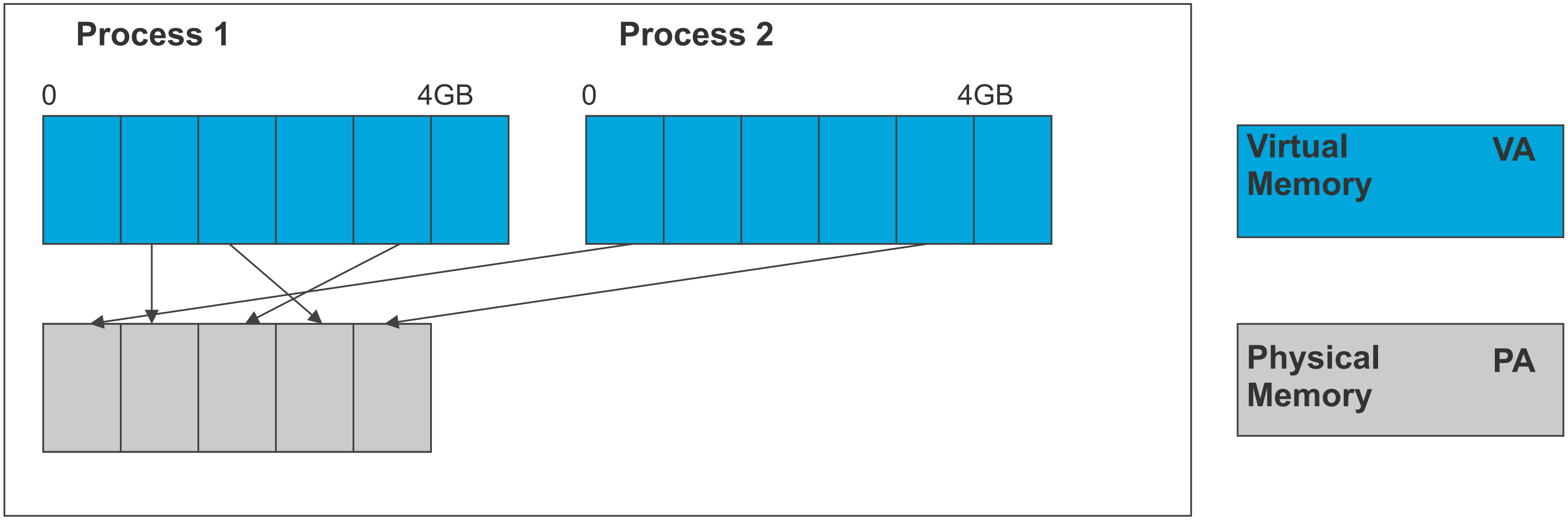
- VMexit performance is critical to hardware assist-based virtualization
- In additional to generational performance improvements, Intel is improving VMexit latencies

Virtual Memory in a Native OS

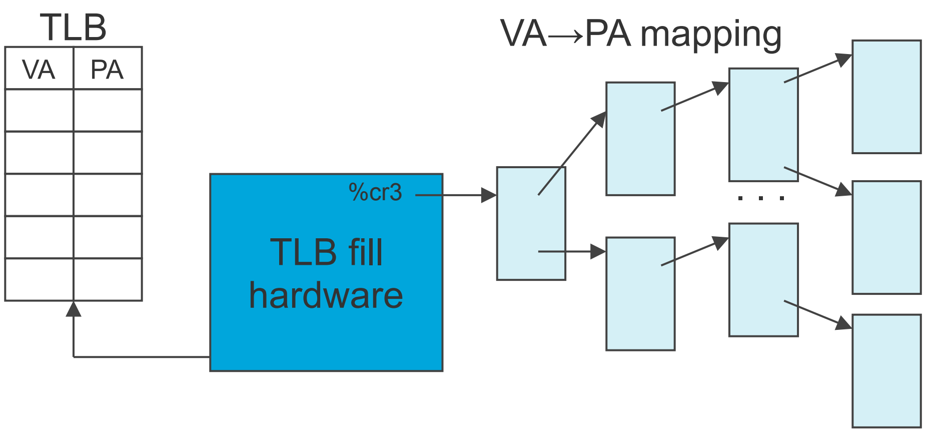


- **Applications see contiguous virtual address space, not physical memory**
- **OS defines VA -> PA mapping**
  - Usually at 4 KB granularity: a *page* at a time
  - Mappings are stored in page tables

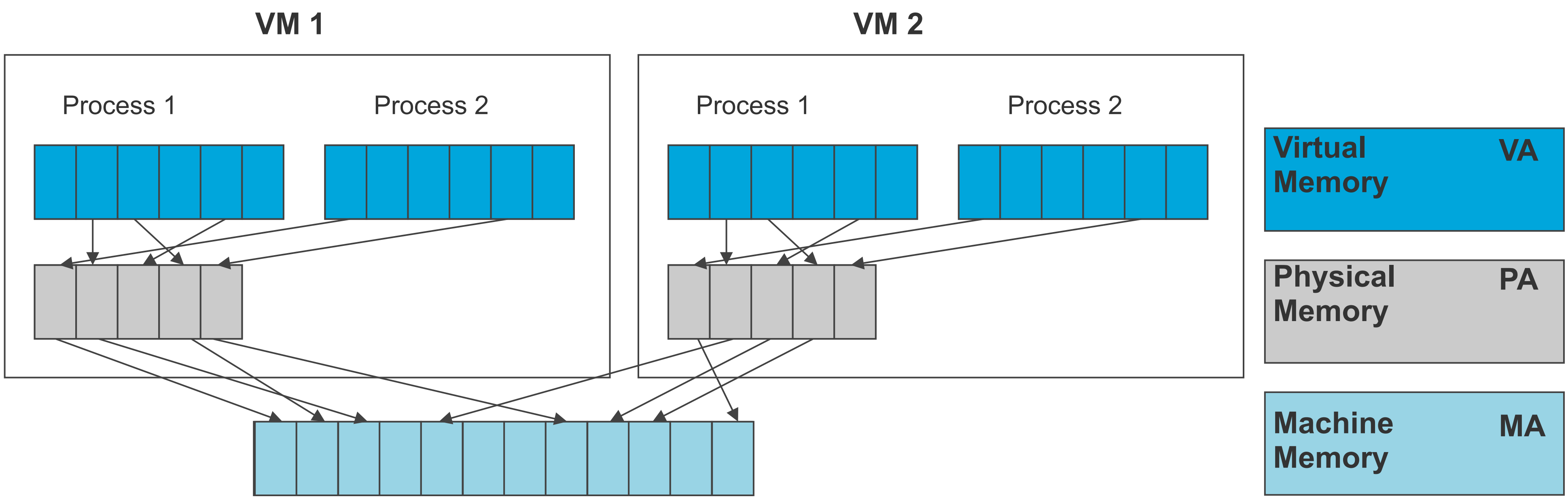
Virtual Memory (ctd)



- **Applications see contiguous virtual address space, not physical memory**
- **OS defines VA -> PA mapping**
  - Usually at 4 KB granularity
  - Mappings are stored in page tables
- **HW memory management unit (MMU)**
  - Page table walker
  - TLB (translation look-aside buffer)



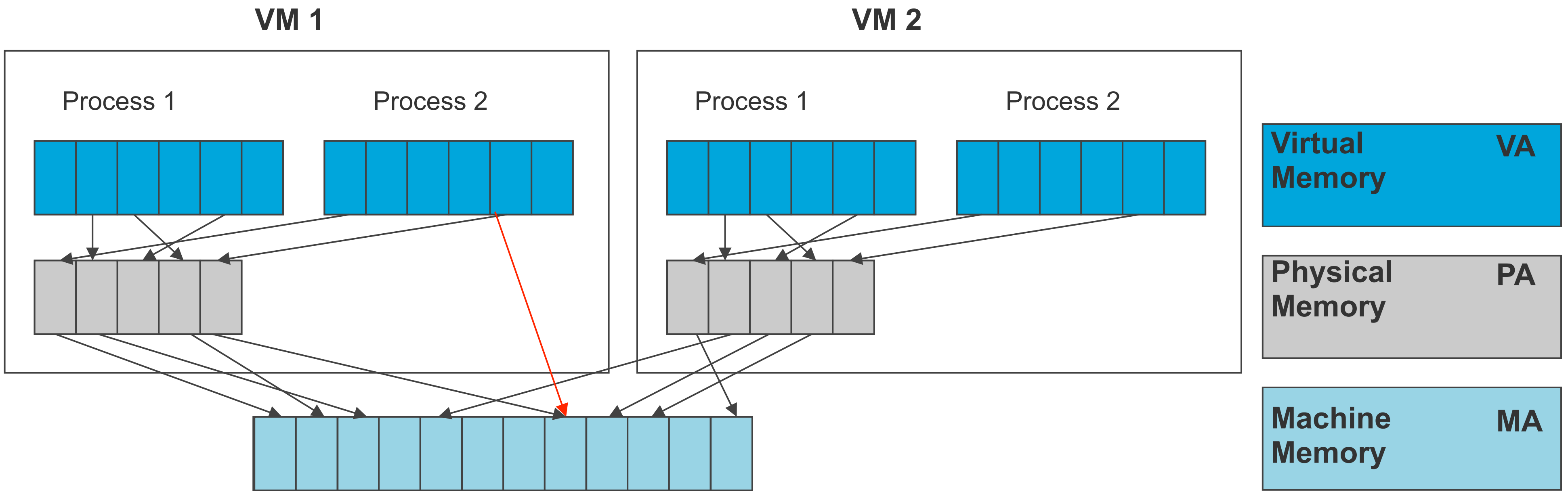
Virtualizing Virtual Memory



- **To run multiple VMs on a single system, another level of memory virtualization must be done**
  - Guest OS still controls virtual to physical mapping: VA -> PA
  - Guest OS has no direct access to machine memory (to enforce isolation)
- **VMM maps guest physical memory to actual machine memory: PA -> MA**

Virtualizing Virtual Memory

Shadow Page Tables



- **VMM builds "shadow page tables" to accelerate the mappings**
  - Shadow directly maps VA -> MA
  - Can avoid doing two levels of translation on every access
  - TLB caches VA->MA mapping
  - Leverage hardware walker for TLB fills (walking shadows)
  - When guest changes VA -> PA, the VMM updates shadow page tables

vmware

3-way Performance Trade-off in Shadow Page Tables

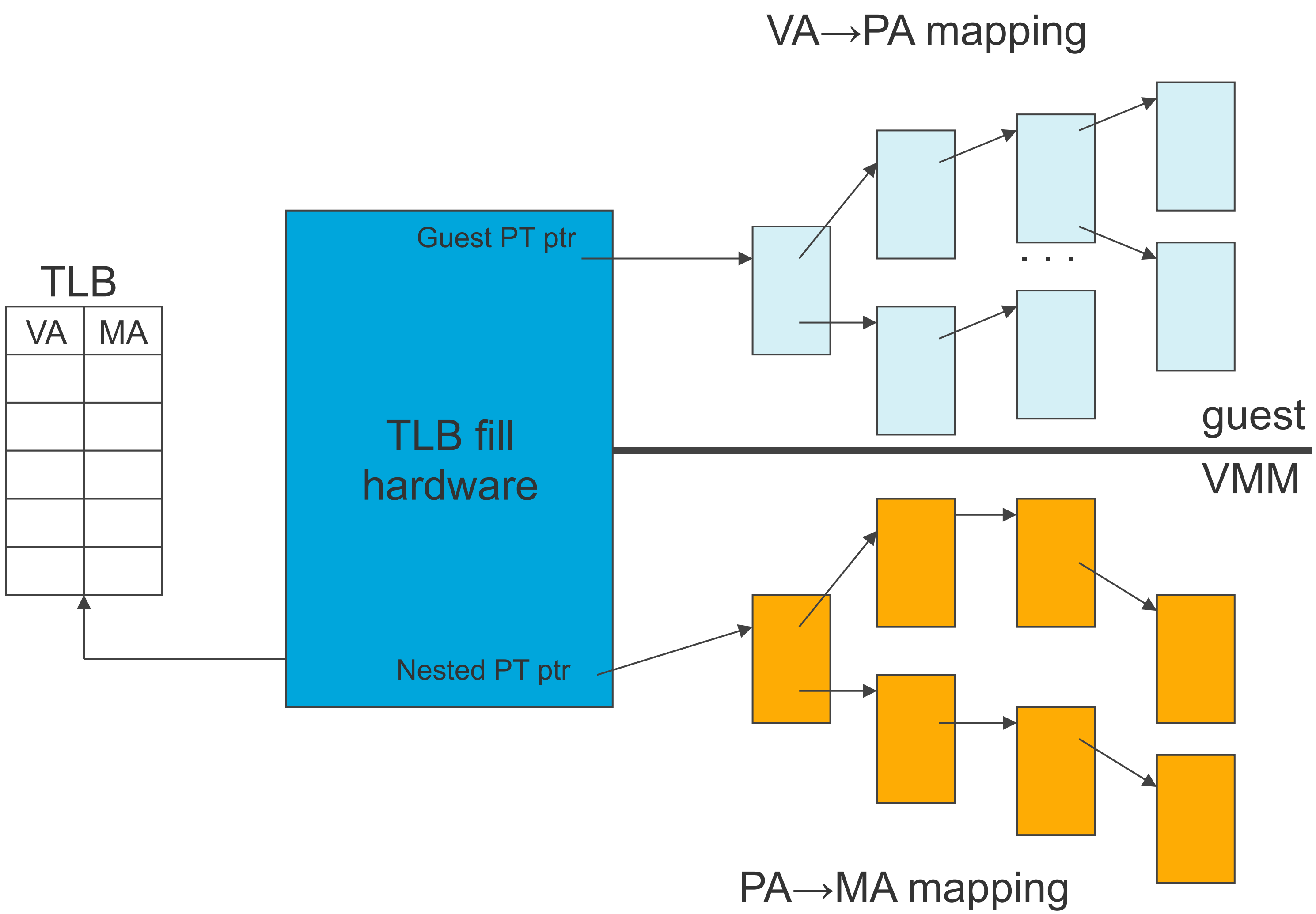
- **1. Trace costs**
  - VMM must intercept Guest writes to primary page tables
  - Propagate change into shadow page table (or invalidate)
- **2. Page fault costs**
  - VMM must intercept page faults
  - Validate shadow page table entry (hidden page fault), or forward fault to Guest (true page fault)
- **3. Context switch costs**
  - VMM must intercept CR3 writes
  - Activate new set of shadow page tables
- **Finding good trade-off is crucial for performance**
- **VMware has 9 years of experience here**

Shadow Page Tables and Scaling to Wide vSMP

- **VMware currently supports up to 4-way vSMP**
- **Problems lurk in scaling to higher numbers of vCPUs**
  - Per-vcpu shadow page tables
    - High memory overhead
    - Process migration costs (cold shadows/lack of shadows)
    - Remote trace events costlier than local events
  - vcpu-shared shadow page tables
    - Higher synchronization costs in VMM
- **Can already see this in extreme cases**
  - forkwait is slower on vSMP than a uniprocessor VM

2nd Generation Hardware Assist

Nested/Extended Page Tables



Analysis of NPT

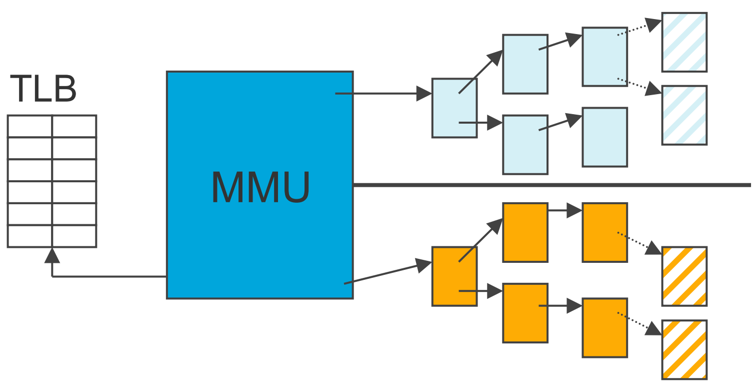
- **MMU composes VA->PA and PA->MA mappings *on the fly* at TLB fill time**
- **Benefits**
  - Significant reduction in “exit frequency”
    - No trace faults (primary page table modifications as fast as native)
    - Page faults require no exits
    - Context switches require no exits
  - No shadow page table memory overhead
  - Better scalability to wider vSMP
    - Aligns with multi-core: performance through parallelism
- **Costs**
  - More expensive TLB misses: $O(n^2)$ cost for page table walk, where n is the depth of the page table tree

Analysis of NPT

- **MMU composes VA->PA and PA->MA mappings *on the fly* at TLB fill time**
- **Benefits**
  - Significant reduction in “exit frequency”
    - No trace faults (primary page table modifications as fast as native)
    - Page faults require no exits
    - Context switches require no exits
  - No shadow page table memory overhead
  - Better scalability to wider vSMP
    - Aligns with multi-core: performance through parallelism
- **Costs**
  - More expensive TLB misses: $O(n^2)$ cost for page table walk, where n is the depth of the page table tree

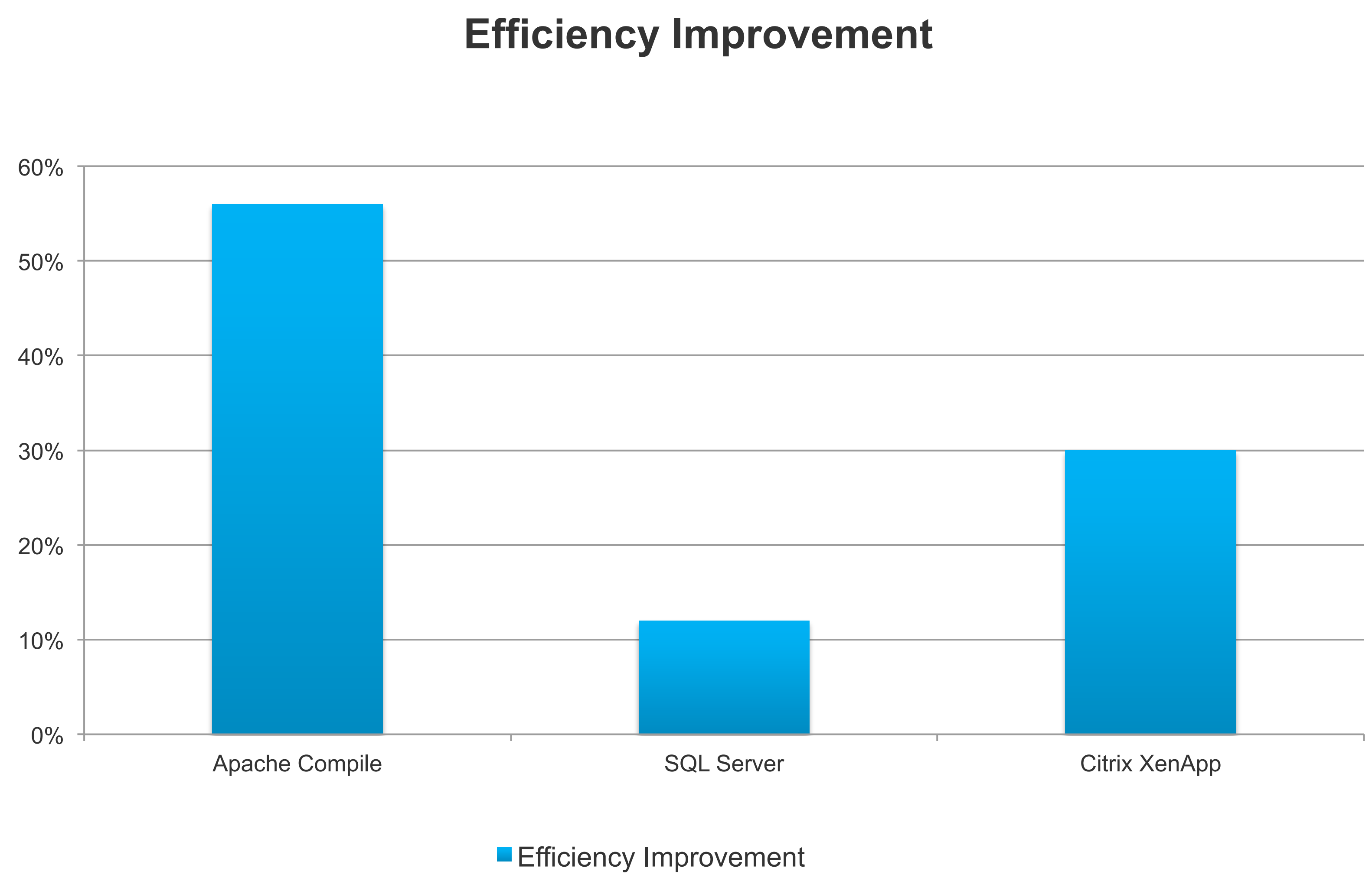
Improving NPT Performance
Large pages

- **2 MB today, 1 GB in the future**
  - In part guest's responsibility: "inner" page tables
    - For most guests/workloads this *requires explicit setup*
  - In part VMM's responsibility: "outer" page tables
    - ESX will take care of it
- **1st benefit: faster page walks (fewer levels to traverse)**
- **2nd benefit: fewer page walks (increased TLB capacity)**



vmware

Hardware-assisted Memory Virtualization



vSphere Monitor Defaults

VM Configuration	Core i7	45nm Core2 with VT-x	65nm Core2 with VT-x and FlexPriority	65nm Core2 with VT-x and No FlexPriority	P4 with VT-x	EM64T without VT-x	No EM64T
FT enabled	VT-x + swMMU	VT-x + swMMU	VT-x + swMMU	VT-x + swMMU	Not runnable	Not runnable	Not runnable
64-bit Guests	VT-x + EPT	VT-x + swMMU	VT-x + swMMU	VT-x + swMMU	VT-x + swMMU	Not Runnable	Not runnable
VMI enabled**	BT + swMMU	BT + swMMU	BT + swMMU	BT + swMMU	BT + swMMU	BT + swMMU	BT + swMMU
OpenServer UnixWare	VT-x + EPT	VT-x + swMMU	VT-x + swMMU	VT-x + swMMU	VT-x + swMMU	BT + swMMU	BT + swMMU
OS/2	VT-x + EPT	VT-x + swMMU	VT-x + swMMU	VT-x + swMMU	VT-x + swMMU	Not Runnable	Not Runnable
32-bit Linux 32-bit FreeBSD	VT-x + EPT	VT-x + swMMU	BT + swMMU (*)	BT + swMMU (*)	BT + swMMU (*)	BT + swMMU	BT + swMMU
32-bit Windows: XP, Vista, Server 2003, Server 2008	VT-x + EPT	VT-x + swMMU	VT-x + swMMU	BT + swMMU (*)	BT + swMMU (*)	BT + swMMU	BT + swMMU
Windows 2000, NT, 95, 98, DOS, Netware, 32-bit Solaris	BT + swMMU (*)	BT + swMMU (*)	BT + swMMU (*)	BT + swMMU (*)	BT + swMMU (*)	BT + swMMU	BT + swMMU
Other 32-bit Guests	VT-x + EPT	VT-x + swMMU	VT-x + swMMU	VT-x + swMMU	VT-x + swMMU	BT + swMMU	BT + swMMU

Performance Help from the Hypervisor

- **Take advantage of new Hardware**
  - Utilize multi-core systems easily without changing the app or OS
  - Leverage 64-bit memory hardware sizes with existing 32-bit VMs
  - Take advantage of newer high performance I/O + networking asynchronously from guest-OS changes/revs.
- **More flexible Storage**
  - More options for distributed, reliable boot
  - Leverage low-cost, high performance NFS, iSCSI I/O for boot or data without changing the guest OS
- **Distributed Resource Management**
  - Manage Linux, Solaris, Windows with one set of metrics and tools
  - Manage horizontal apps with cluster-aware resource management

CPU and Memory Paravirtualization

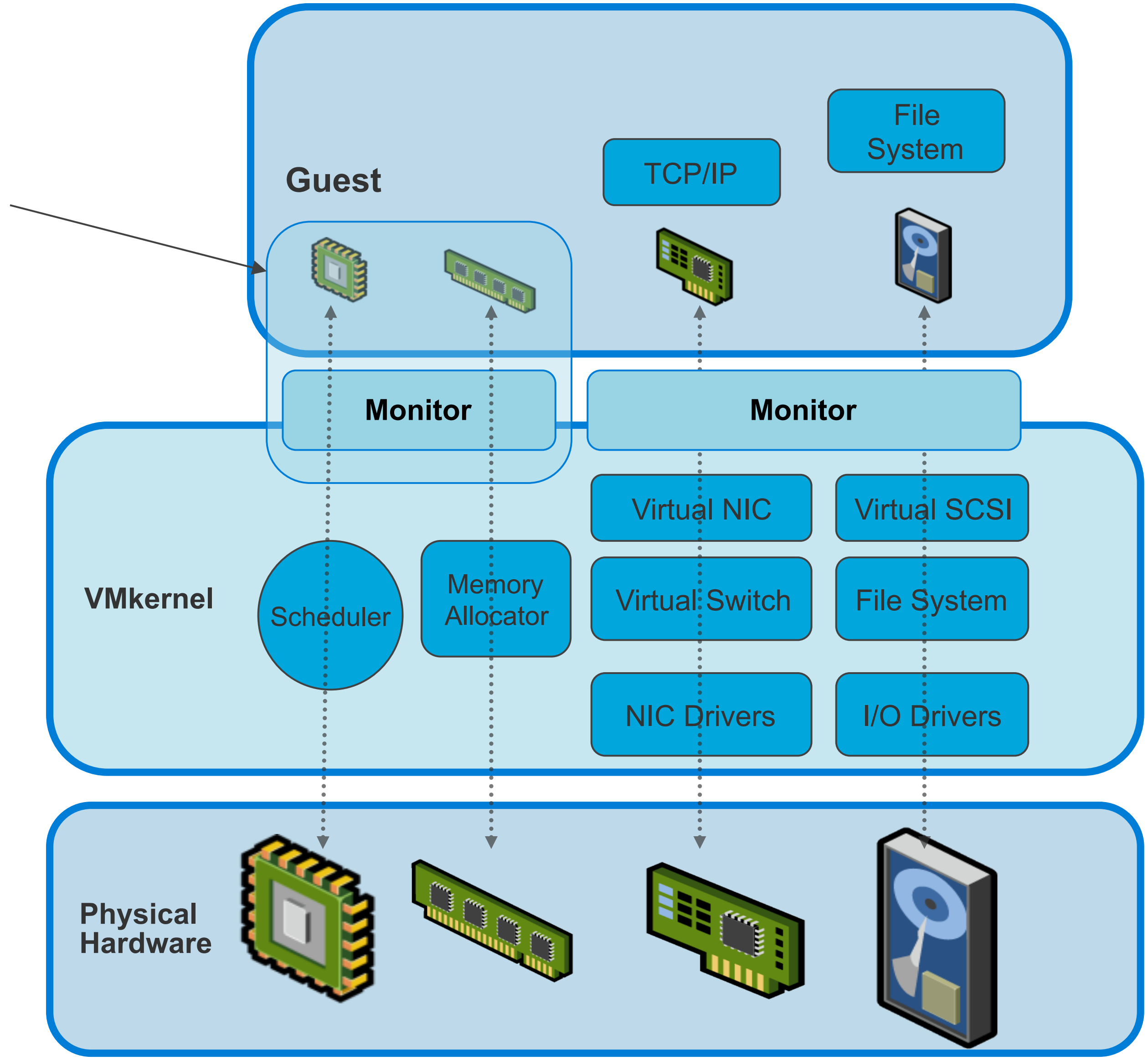
Paravirtualization extends the guest to allow direct interaction with the underlying hypervisor

Paravirtualization reduces the monitor cost including memory and System call operations.

Gains from paravirtualization are workload specific

Hardware virtualization mitigates the need for some of the paravirtualization calls

VMware approach:
VMI and paravirt-ops



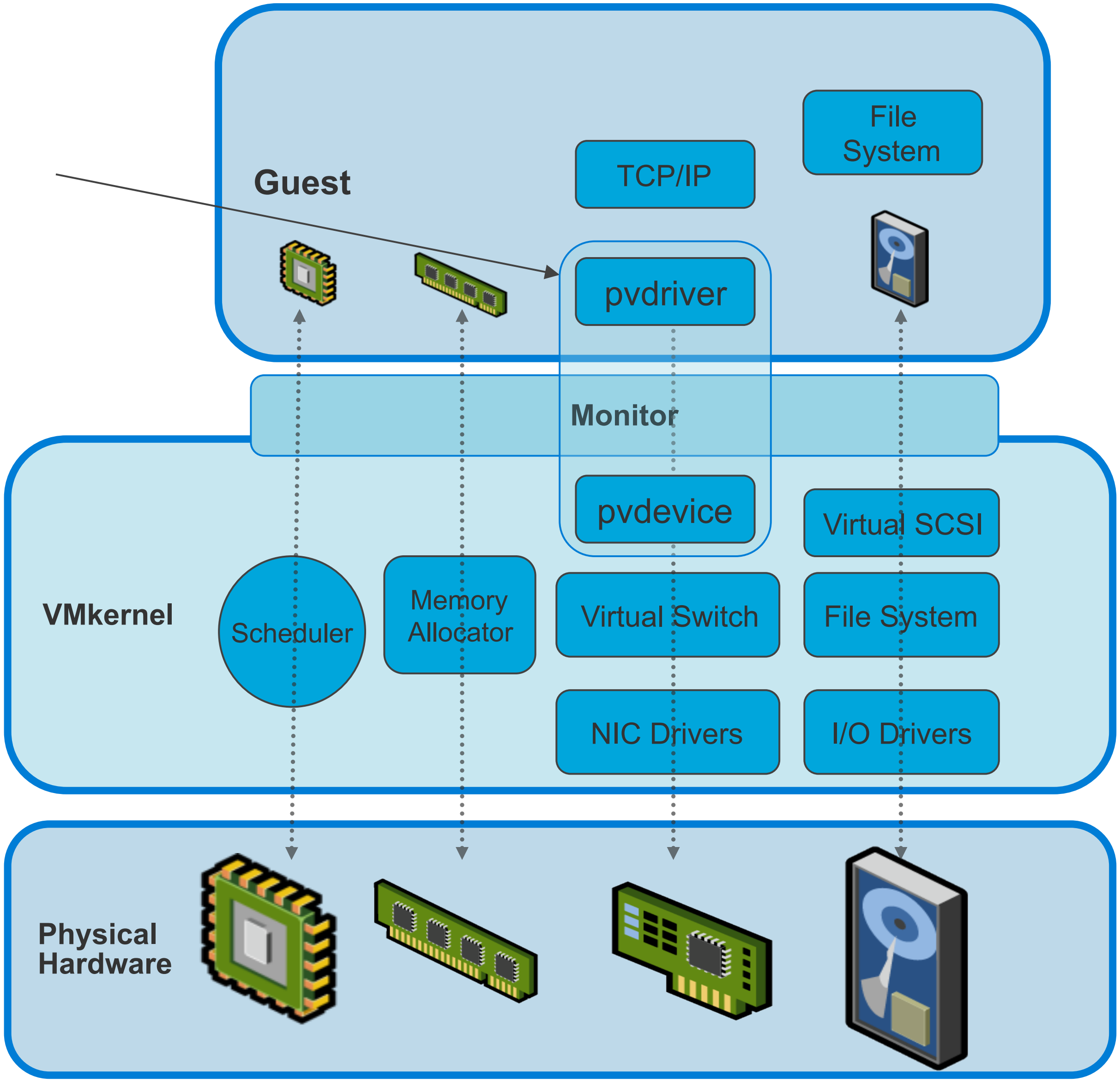
Device Paravirtualization

Device Paravirtualization places A high performance virtualization-Aware device driver into the guest

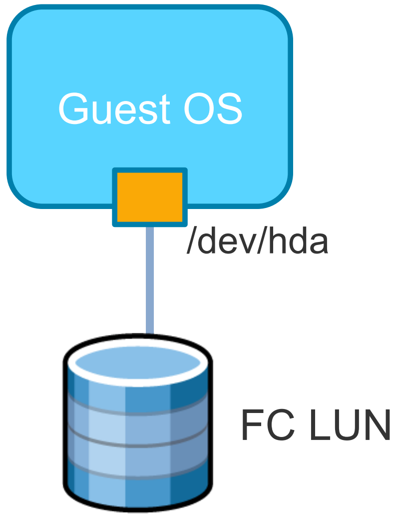
Paravirtualized drivers are more CPU efficient (less CPU over-head for virtualization)

Paravirtualized drivers can also take advantage of HW features, like partial offload (checksum, large-segment)

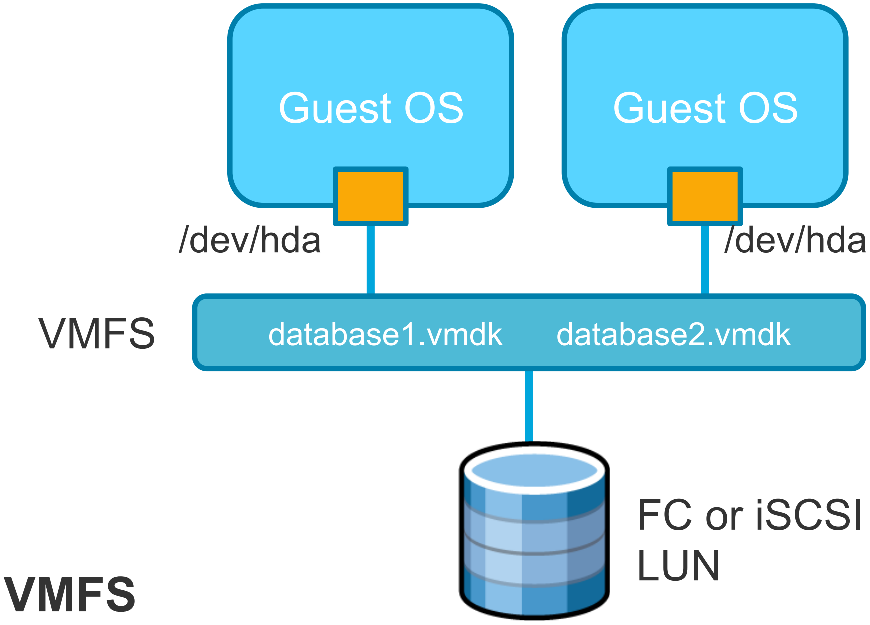
VMware ESX uses para-virtualized network drivers



Storage – Fully virtualized via VMFS and Raw Paths

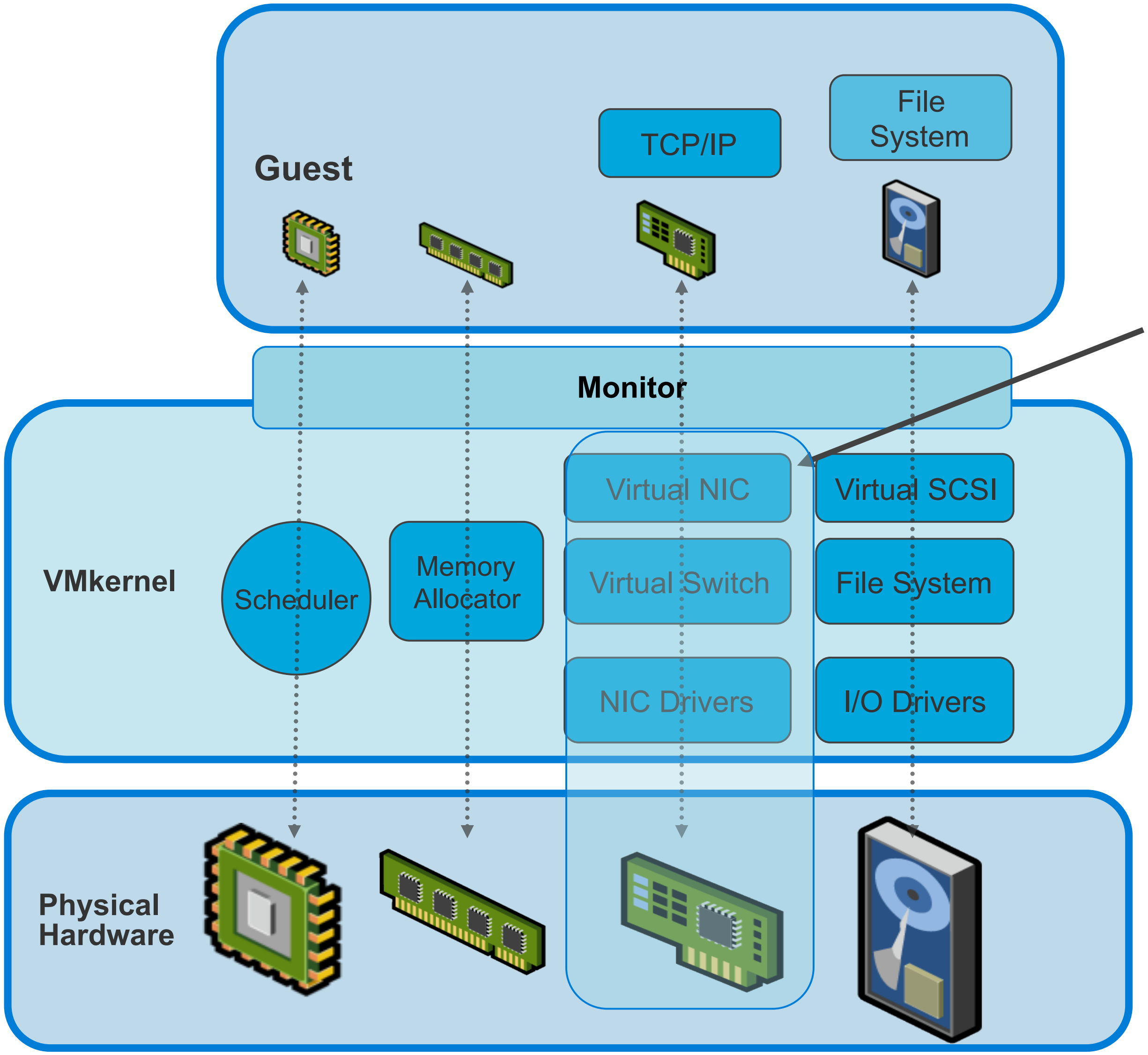


- **RAW**
- **RAW provides direct access to a LUN from within the VM**
- **Allows portability between physical and virtual**
- **RAW means more LUNs**
  - More provisioning time
- **Advanced features still work**



- **VMFS**
- **Leverage templates and quick provisioning**
- **Fewer LUNs means you don’t have to watch Heap**
- **Scales better with Consolidated Backup**
- **Preferred Method**

Optimized Network Performance



Network stack and drivers ere implemented in ESX layer (not in the guest)

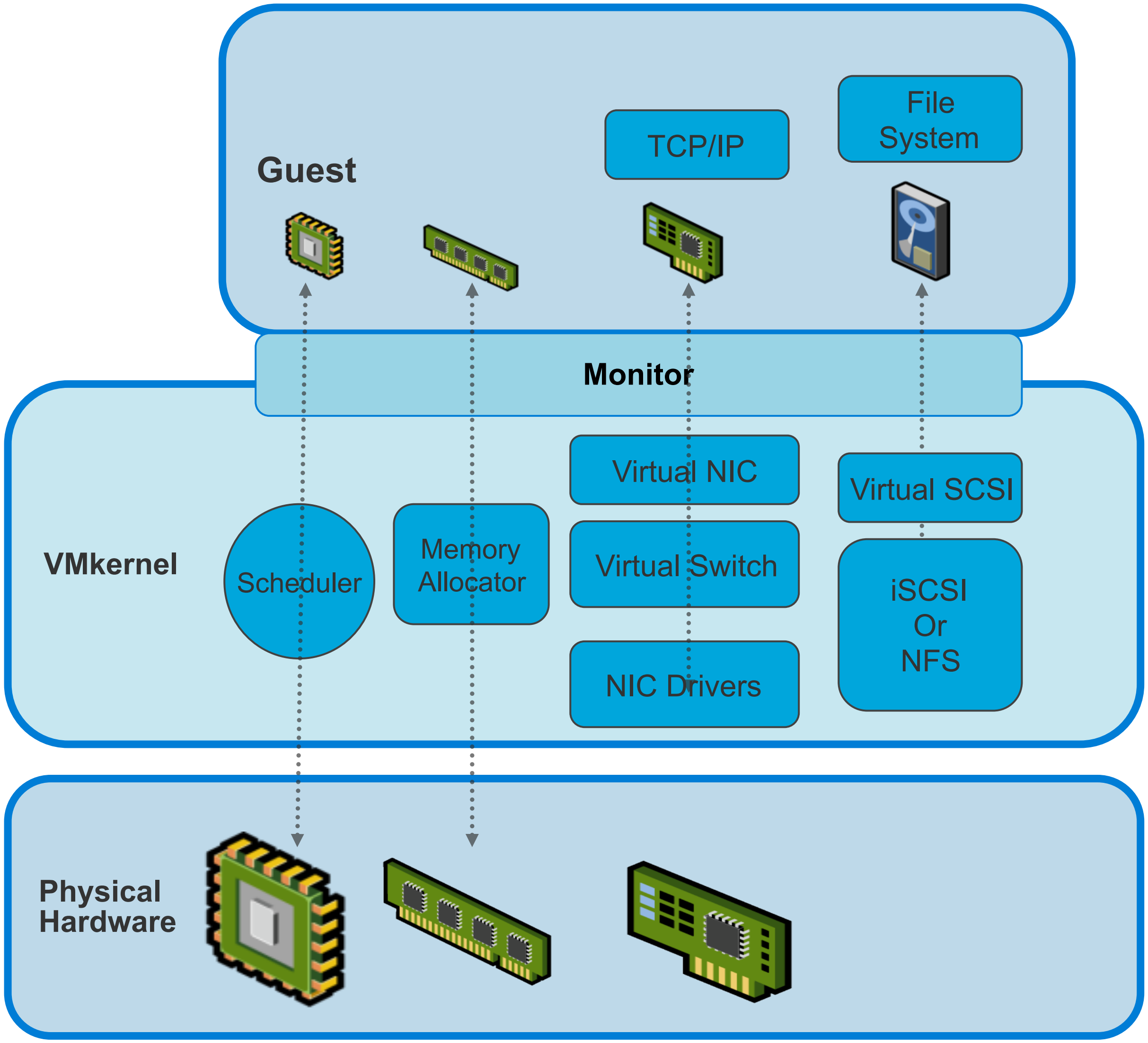
VMware's strategy is to optimize the network stack in the ESX layer, and keep the guest 100% agnostic of the underlying hardware

This enables full-virtualization capabilities (vmotion etc)

ESX Stack is heavily Performance optimized

ESX Focus: stateless offload; including LSO (large segment Offload), Checksum offload, 10Gbe perf, Multi-ring NICs

Guest-Transparent NFS and iSCSI
iSCSI and NFS Virtualization in VMware ESX



iSCSI and NFS are growing To be popular, due to their low port/switch/fabric costs

Virtualization provides the ideal mechanism to transparently adopt iSCSI/NFS

Guests don’t need iSCSI/NFS Drivers: they continue to see SCSI

VMware ESX 3 provides high Performance NFS and iSCSI Stacks

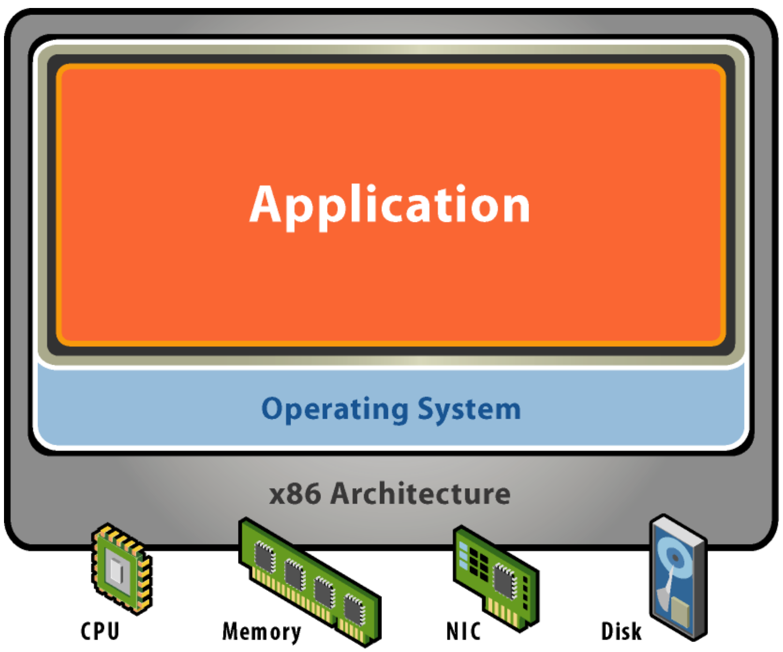
Futher emphasis on 1Gbe/ 10Gbe performance

INTRODUCTION TO PERFORMANCE MONITORING

Traditional Architecture

Operating system performs various roles

- Application Runtime Libraries
- Resource Management (CPU, Memory etc)
- Hardware + Driver management

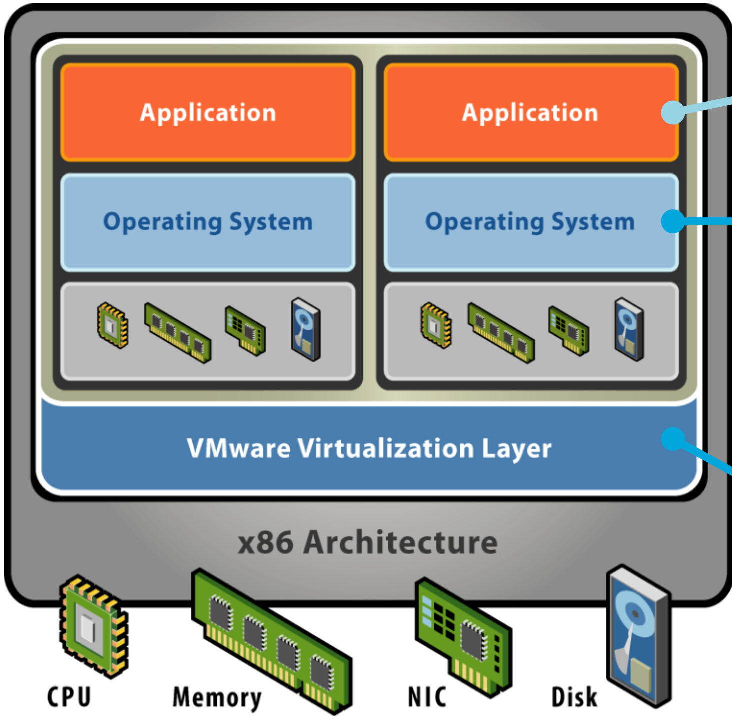


> Performance & Scalability of the OS was paramount

> Performance Observability tools are a feature of the OS

Performance in a Virtualized World

The OS takes on the role of a Library, Virtualization layer grows



Application

Run-time Libraries and Services

Application-Level Service Management

Application-decomposition of performance

Run-time or Deployment OS

Local Scheduling and Memory Management

Local File System

Infrastructure OS (Virtualization Layer)

Scheduling

Resource Management

Device Drivers

I/O Stack

File System

Volume Management

Network QoS

Firewall

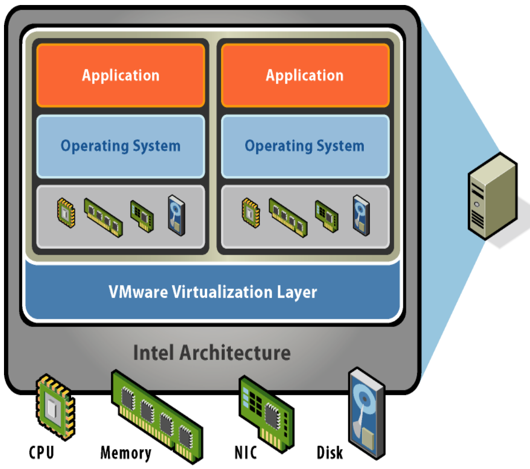
Power Management

Fault Management

Performance Observability of System Resources

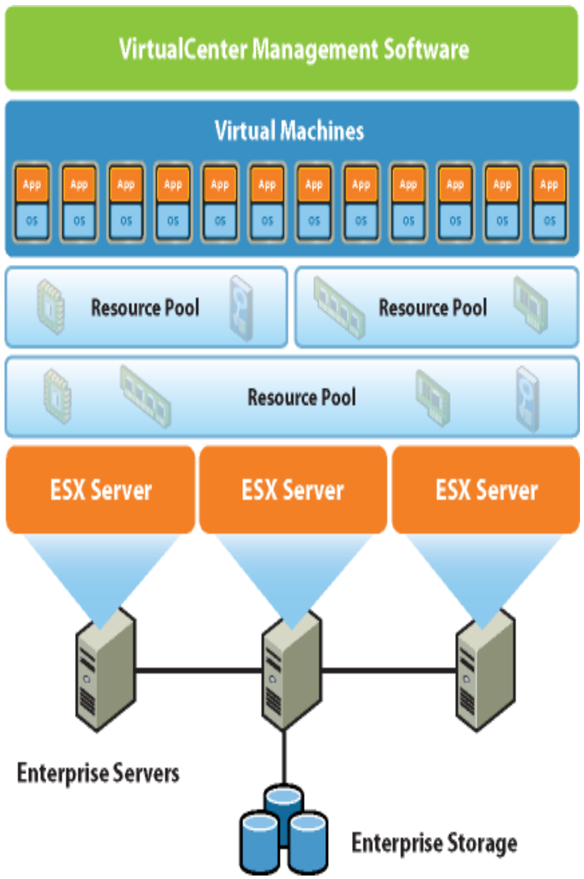
Performance Management Trends

Partitioning



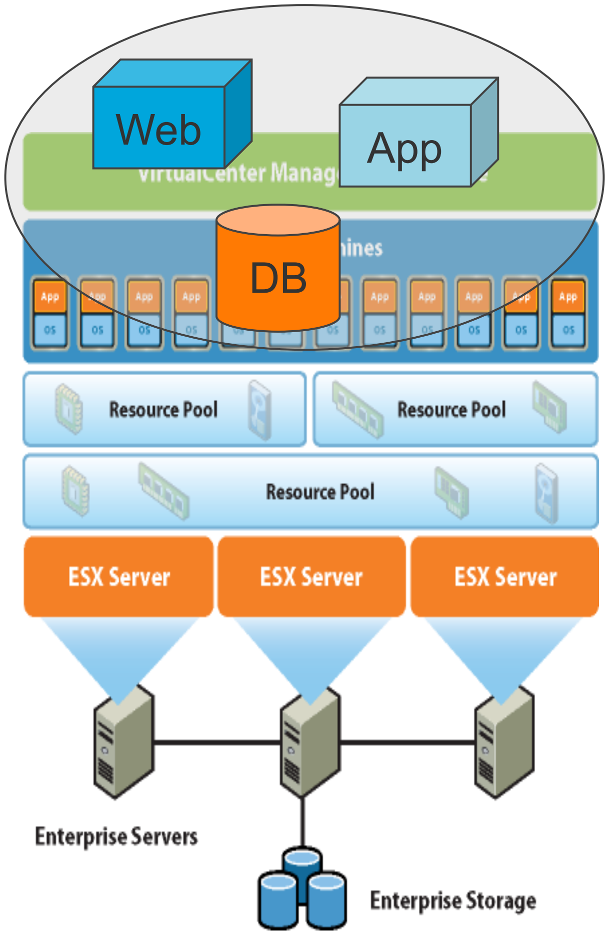
ESX 1.x

Distributed Resource Management



vSphere

Service-Oriented/ Service-Level Driven



PaaS, Appspeed

Performance Measurement

- **Three basic performance measurement metrics:**
  - Throughput: Transactions per/Sec, Instructions Retired per sec, MB/sec, IOPS, etc, …
  - Latency: How long does it take
    - e.g., Response time

  - Utilization: How much resource is consumed to perform a unit of work
- **Latency and throughput are often inter-related, latency becomes important for smaller jobs**

Throughput, Queues and Latency

Arriving Customers
(arrivals per minute)

Queue
(how many people in queue)

Checkout
Utilization = percentage of time busy serving customers

Customers Serviced
(throughput is customers service per minute)

queue time

service time

response time

vmware

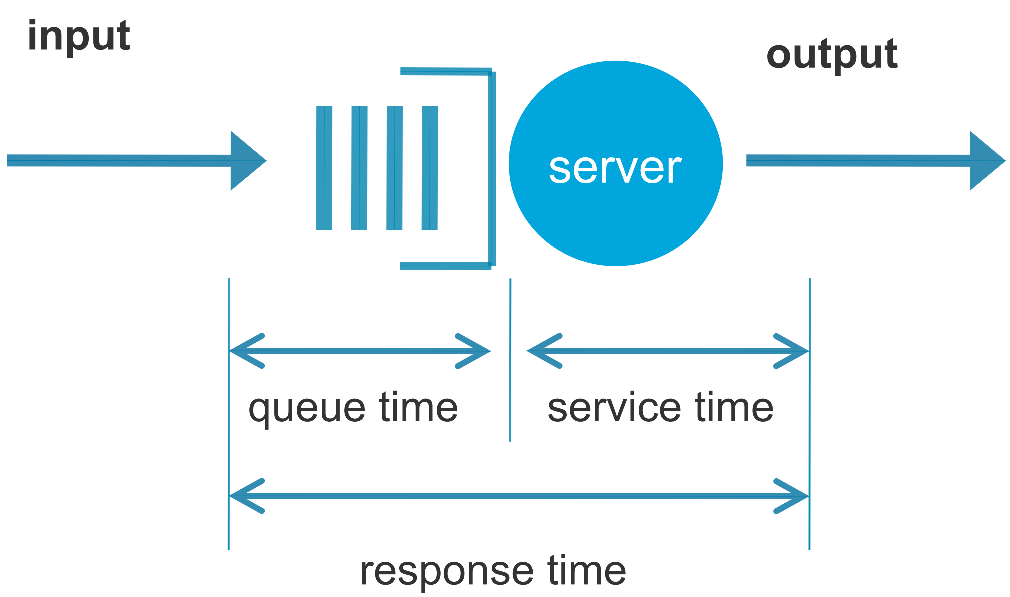
Mathematical Representation, terms

Arriving Customers

Queue

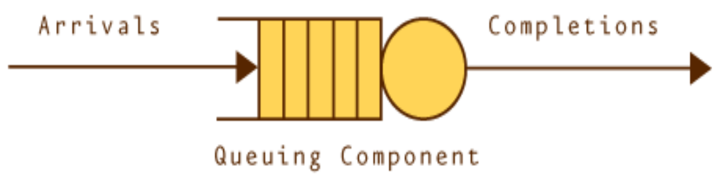
Checkout

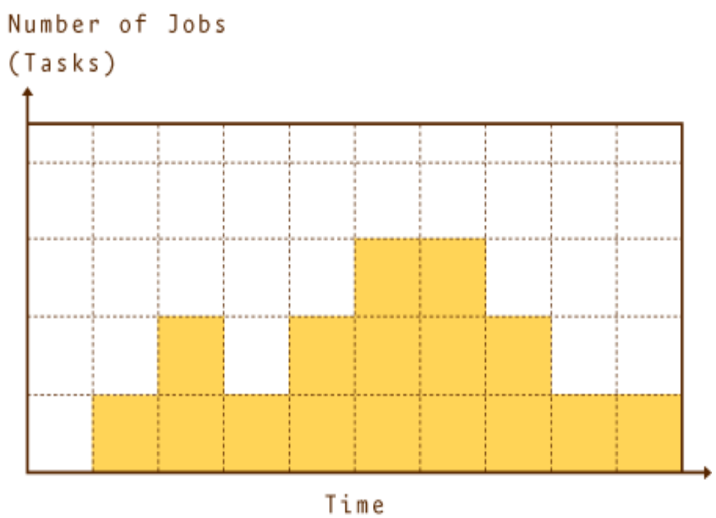
Utilization = busy-time at server / time elapsed



Throughput,Utilization and Response time are connected

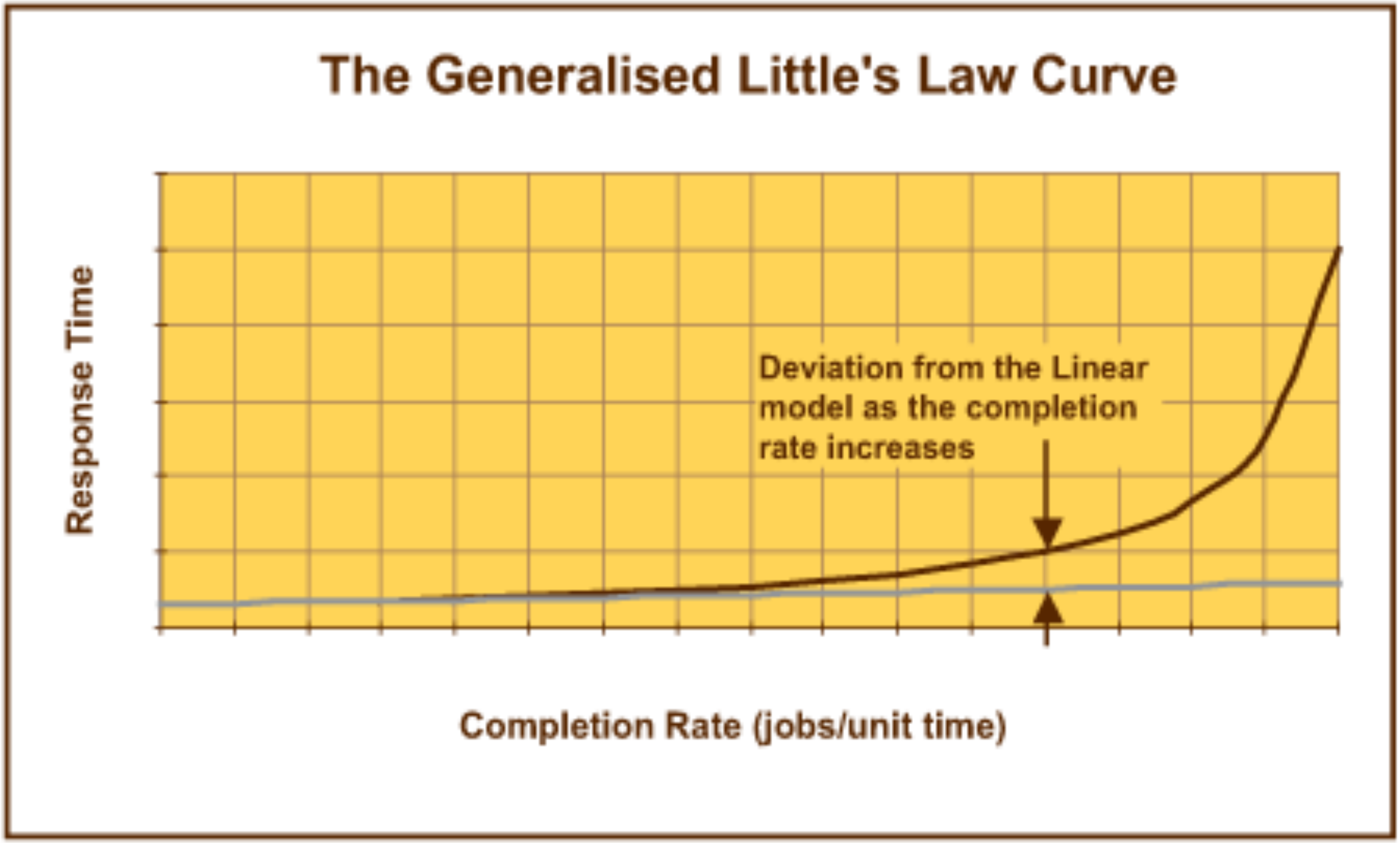
The Buzen and Denning Method





Metric	Symbol	Definition
Length of Observation	T	Total number of time units over which the observation has been made.
Arrivals	N	Total number of Arrivals over the length of observation.
Completions	C	Total number of Completions over the length of the observation.
Busy Time	B	The number of time units where the number of messages in the system exceeds zero.
Utilisation	U	The calculated value: $U = \frac{B}{T}$
Throughput	X	The calculated value: $X = \frac{C}{T}$
Mean Service Time	S	The calculated value: $S = \frac{B}{C}$
Execution Distribution	A	The calculated value: $A = \sum_{t=0}^{T}(Messages_t)$
Mean Queue Length	L	The calculated value: $L = \frac{A}{T}$
Residence Time	RT	The calculated value: $RT = \frac{A}{C}$
Queuing Time	Q	The calculated value: RT – S

Relationship between Utilization and Response Time



Summary of Queuing and Measurements

- **Utilization is a measure of the resources, not quality of service**
  - We can measure utilization (e.g. CPU), but don't assume good response time
  - Measuring service time and queuing (Latency) is much more important
- **Throughput shows how much work is completed only**
  - Quality of service (response time) may be compromised if there is queuing or slow service times.
- **Make sure your key measurement indicators represent what constitutes good performance for your users**
  - Measure end-user latency of users
  - Measure throughput and latency of a system
- **Common mistakes**
  - Measure something which has little to do with end-user happiness/performance
  - Measure utilization only
  - Measure throughput of an overloaded system with a simple benchmark, resulting in artificially high results since response times are bad

Potential Impacts to Performance

- **Virtual Machine Contributors Latency:**
  - CPU Overhead can contribute to latency
  - Scheduling latency (VM runnable, but waiting…)
  - Waiting for a global memory paging operation
  - Disk Reads/Writes taking longer
- **Virtual machine impacts to Throughput:**
  - Longer latency, but only if the application is thread-limited
  - Sub-systems not scaling (e.g. I/O)
- **Virtual machine Utilization:**
  - Longer latency, but only if the application is thread-limited

Comparing Native to Virtualized Performance

- **Pick the key measure**
  - Not always Utilization
  - User response-time and throughput might be more important
- **It's sometimes possible to get better virtual performance**
  - Higher throughput: Can use multiple-VMs to scale up higher than native
  - Memory sharing can reduce total memory footprint
- **Pick the right benchmark**
  - The best one is your real application
  - Avoid micro-benchmarks: they often emphasize the wrong metric
    - especially in virtualized environments

Performance Tricks and Catches

- **Can trade-off utilization for latency**
  - Offloading to other CPUs can improve latency of running job at the cost of more utilization
  - A good thing in light of multi-core
- **Latency and Throughput may be skewed by time**
  - If the time measurement is inaccurate, so will be the latency or throughput measurements
  - Ensure that latency and throughput are measured from a stable time source

Time keeping in Native World

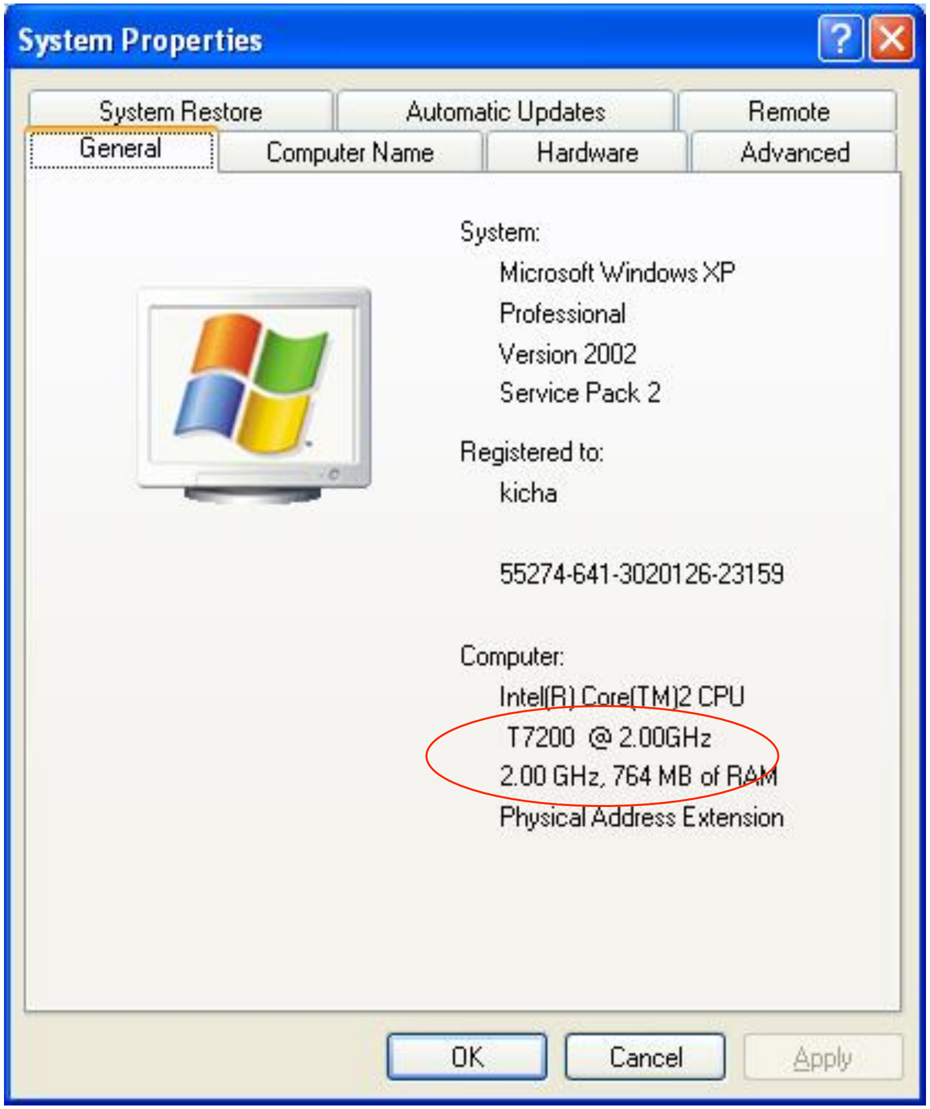
- **OS time keeping**
  - OS programs the timer hardware to deliver timer interrupts at specified frequency
  - Time tracked by counting timer interrupts
  - Interrupts are masked in critical section of the OS code
  - Time loss is inevitable however rate of progress of time is nearly constant
- **Hardware time keeping**
  - TSC: Processor maintains Time Stamp Counter. Applications can query TSC (RDTSC instruction) for high precision time
    - Not accurate when processor frequency varies (e.g. Intel's Speedstep)

Time keeping in Virtualized World

- **OS time keeping**
  - Time progresses in the guest with the delivery of virtual timer interrupts
  - Under CPU over commitment timer interrupts may not be delivered to the guest at the requested rate
  - Lost ticks are compensated with fast delivery of timer interrupts
    - Rate of progress of time is not constant (Time sync does not address this issue)
- **Hardware time keeping**
  - TSC: Guest OSes see pseudo-TSC that is based on physical CPU TSC
  - TSC's may not be synchronized between physical CPUs
  - RDTSC is unreliable if the VM migrates between physical CPUs or across host (Vmotion)

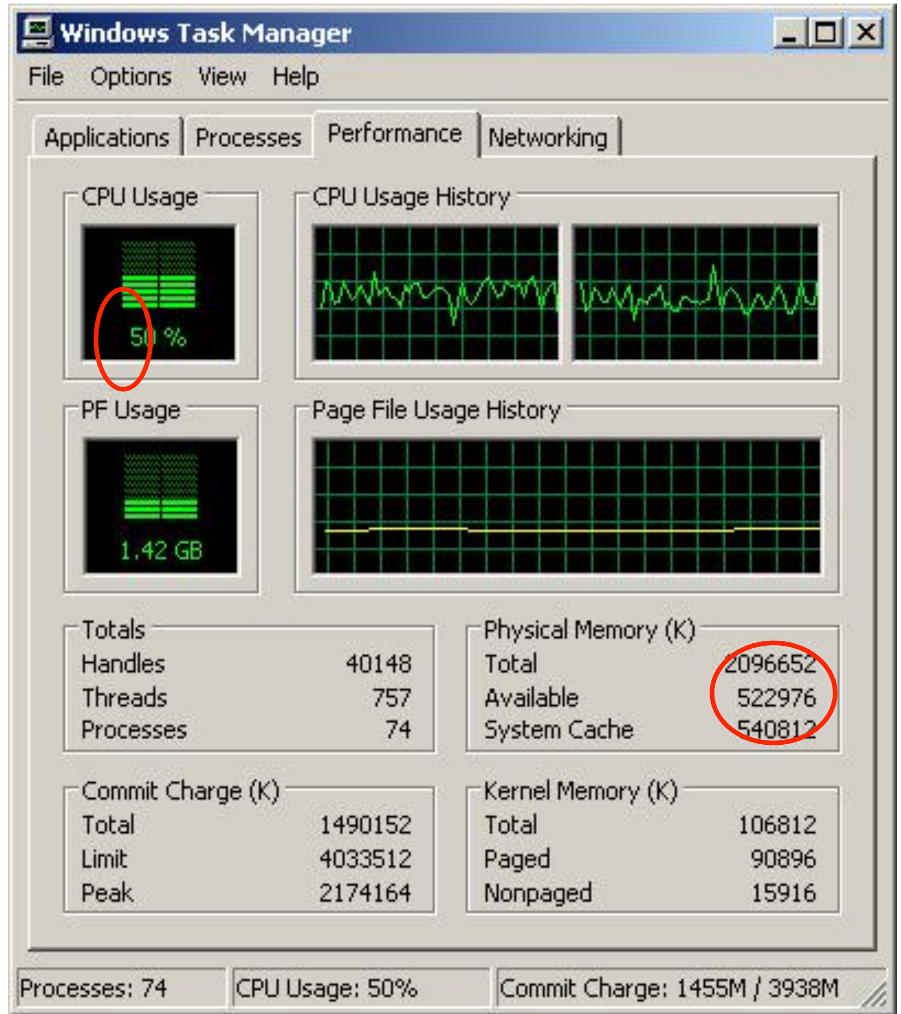
Native-VM Comparison Pitfalls (1 of 3)

- **Guest reports clock speed of the underlying physical processor**
  - Resource pool settings may limit the CPU clock cycles
  - Guest may not get to use the CPU all the time under contention with other virtual machines
- **Guest reports total memory allocated by the user**
  - This doesn't have to correspond to the actual memory currently allocated by the hypervisor



Native-VM Comparison Pitfalls (2 of 3)

- **Processor Utilization accounting**
  - Single threaded application can ping pong between CPUs
  - CPU utilization reported in task manager is normalized per CPU
  - Windows does not account idle loop spinning



- **Available Memory**
  - Available memory inside the guest may come from swap on the host

vmware

Native-VM Comparison Pitfalls (3 of 3)

- **Hardware setup and configuration differences**
  - Processor: Architecture, cache, clock speed
    - Performance difference between different architecture is quite substantial
    - L2, L3 cache size impacts performance of some workload
    - Clock speed becomes relevant only when the architecture is the same
  - Disk : Local dedicated disk versus shared SAN
    - Incorrect SAN configuration could impact performance
  - File system: Local file system versus Distributed VMFS
    - Distributed file systems (VMFS) have locking overhead for metadata updates
  - Network: NIC adapter class, driver, speed/duplex

→ Slower hardware can outperform powerful hardware when the latter shares resources with more than one OS/Application

Virtualized World Implications

- **Guest OS metrics**
  - Performance metrics in the guest could be skewed when the rate of progress of time is skewed
  - Guest OS resource availability can give incorrect picture
- **Resource availability**
  - Resources are shared, hypervisors control the allocation
  - Virtual machines may not get all the hardware resources
- **Performance Profiling**
  - Hardware performance counters are not virtualized
  - Applications cannot use hardware performance counters for performance profiling in the guest
- **Virtualization moves performance measurement and management to the hypervisor layer**

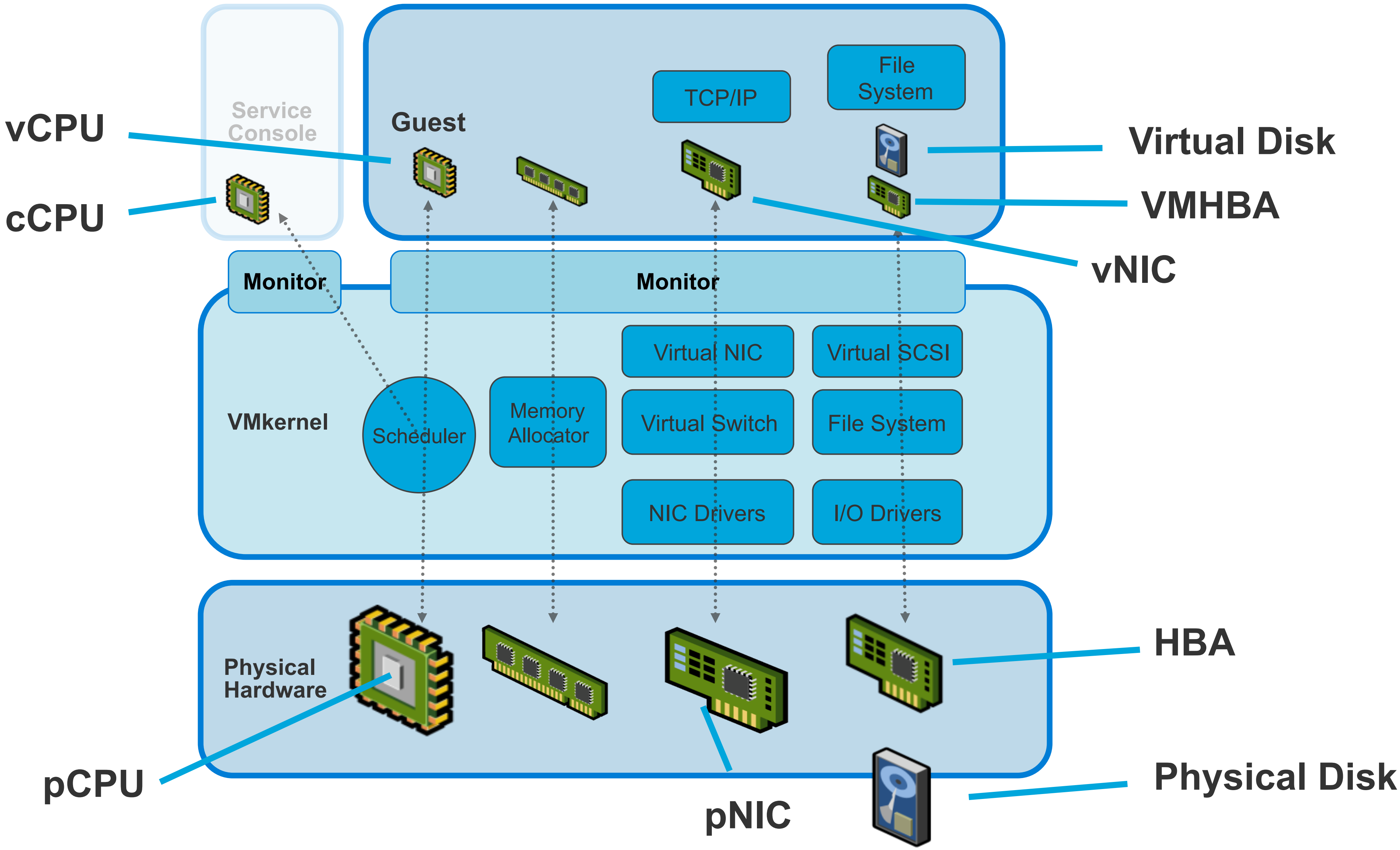
Approaching Performance Issues

- Make sure it is an apples-to-apples comparison
- Check guest tools & guest processes
- Check host configurations & host processes
- Check VirtualCenter client for resource issues
- Check esxtop for obvious resource issues
- Examine log files for errors
- If no suspects, run microbenchmarks (e.g., Iometer, netperf) to narrow scope
- Once you have suspects, check relevant configurations
- If all else fails…discuss on the Performance Forum

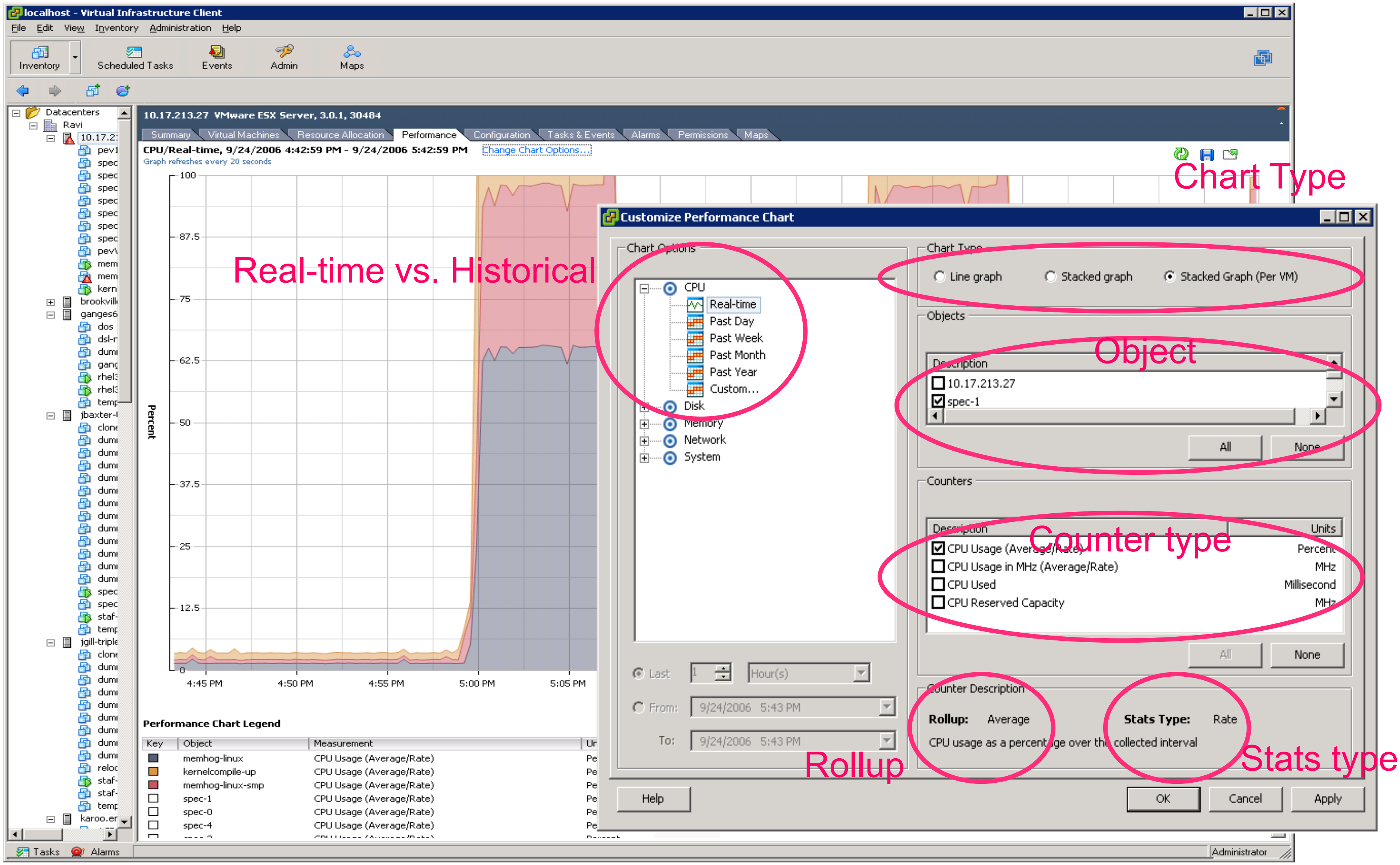
Tools for Performance Analysis

- **VirtualCenter client (VI client):**
  - Per-host and per-cluster stats
  - Graphical Interface
  - Historical and Real-time data
- **esxtop: per-host statistics**
  - Command-line tool found in the console-OS
- **SDK**
  - Allows you to collect only the statistics they want
- **All tools use same mechanism to retrieve data (special vmkernel calls)**

Important Terminology



VI Client



Real-time vs. Historical
Chart Type
Object
Counter type
Rollup
Stats type

VI Client

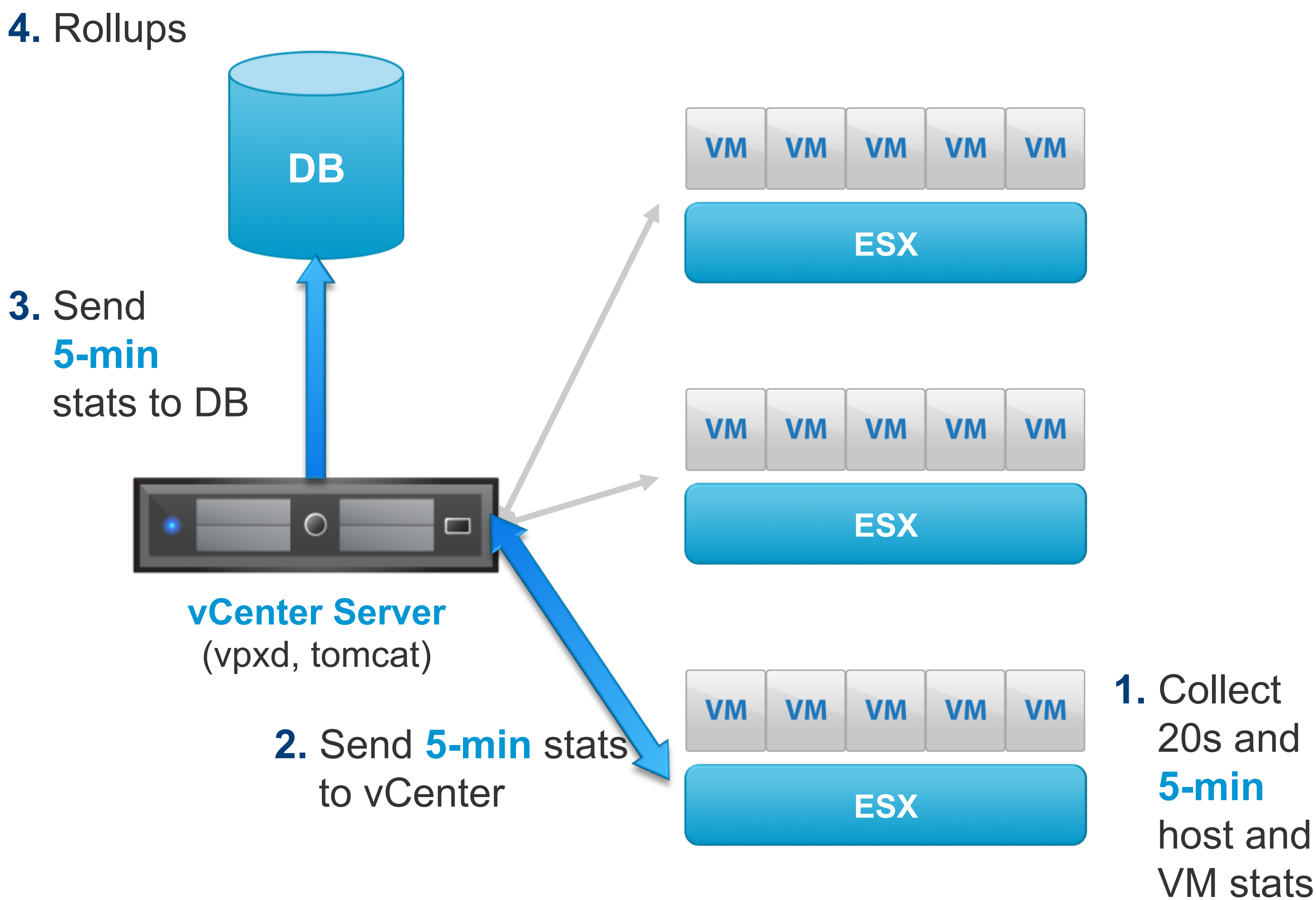
- Real-time vs. archived statistics (past hour vs. past day)
- Rollup: representing different stats intervals
- Stats Type: rate vs. number
- Objects (e.g., vCPU0, vCPU1, all CPUs)
- Counters (e.g., which stats to collect for a given device)
- Stacked vs. Line charts

Real-time vs. Historical stats

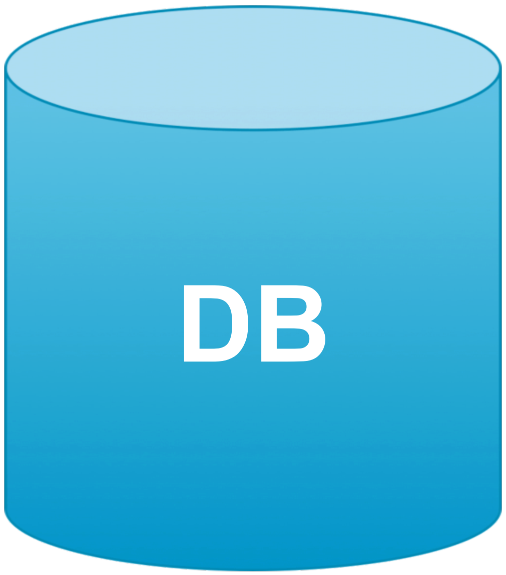
- **VirtualCenter stores statistics at different granularities**

Time Interval	Data frequency	Number of samples
Past Hour (real-time)	20s	180
Past Day	5 minutes	288
Past Week	15 minutes	672
Past Month	1 hour	720
Past Year	1 day	365

Stats Infrastructure in vSphere



Rollups

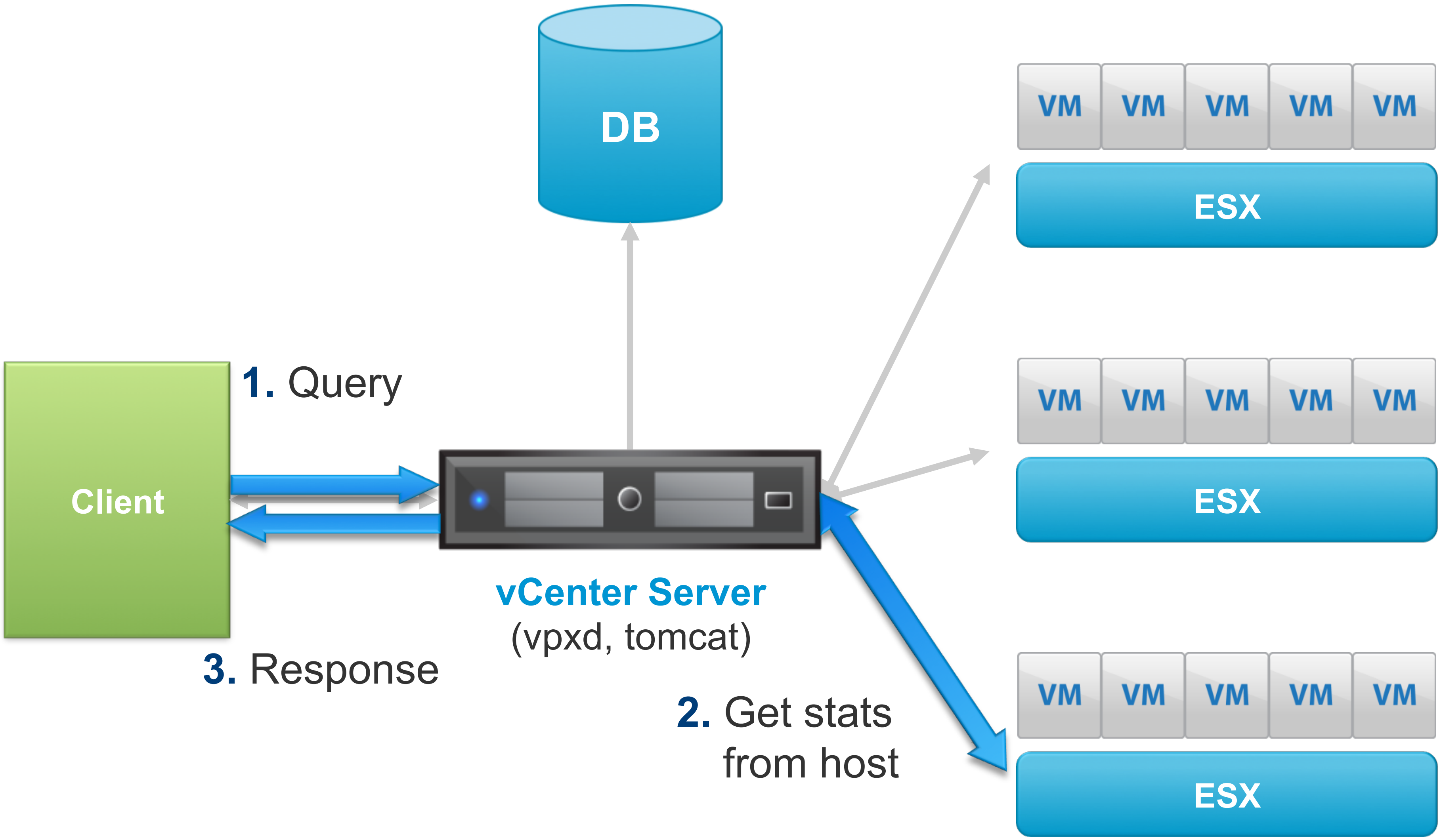


1. Past-Day (5-minutes) → Past-Week
2. Past-Week (30-minutes) → Past-Month
3. Past-Month (2-hours) → Past-Year
4. (Past-Year = 1 data point per day)

DB only archives historical data

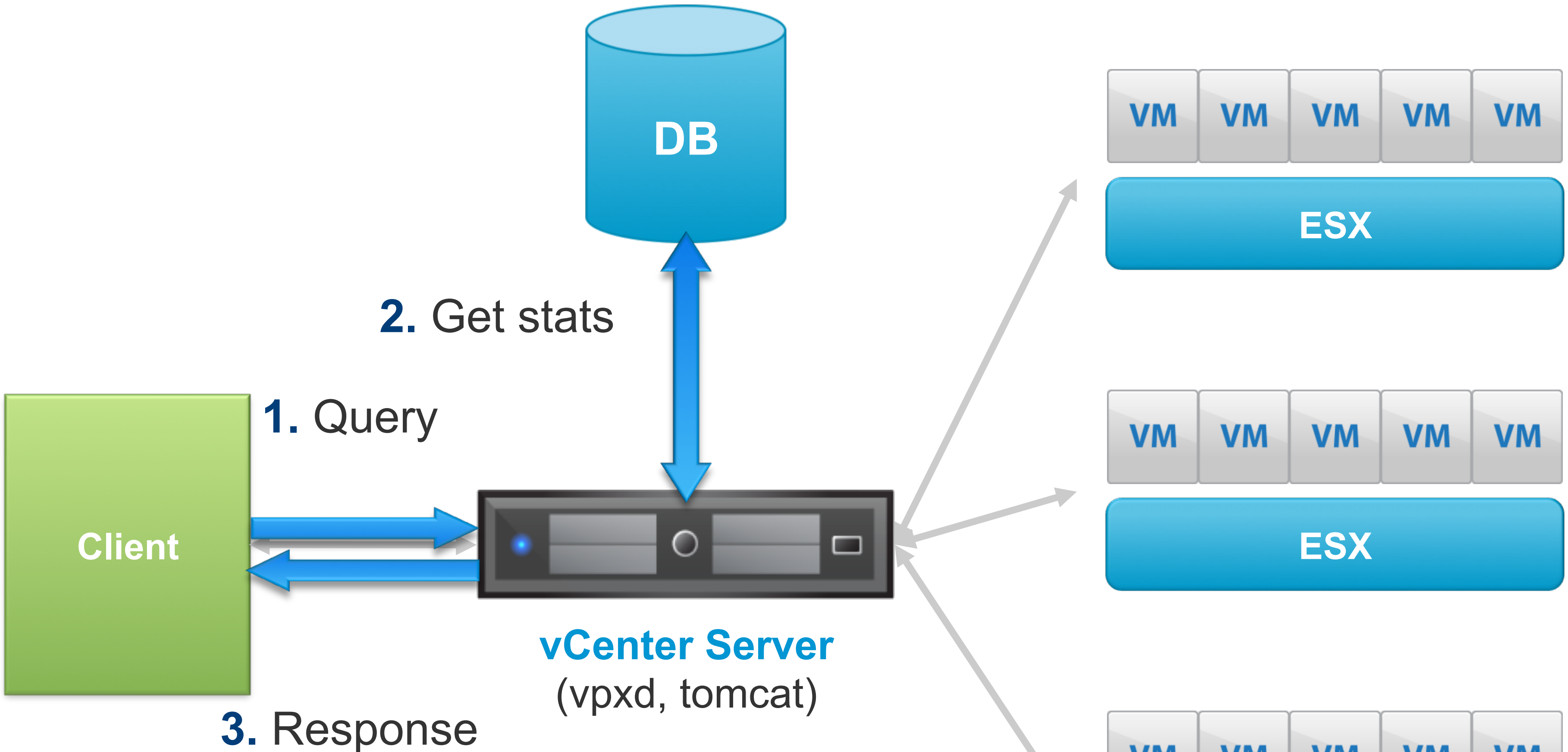
- Real-time (i.e., Past hour) **NOT** archived at DB
- Past-day, Past-week, etc. → Stats Interval
- Stats Levels **ONLY APPLY TO HISTORICAL DATA**

Anatomy of a Stats Query: Past-Hour ("RealTime") Stats



No calls to DB
Note: Same code path for past-day stats within last 30 minutes

Anatomy of a Stats Query: Archived Stats



No calls to ESX host (caveats apply)
Stats Level = Store this stat in the DB

Stats type

- **Statistics type: rate vs. delta vs. absolute**

Statistics type	Description	Example
Rate	Value over the current interval	CPU Usage (MHz)
Delta	Change from previous interval	CPU Ready time
Absolute	Absolute value (independent of interval)	Memory Active

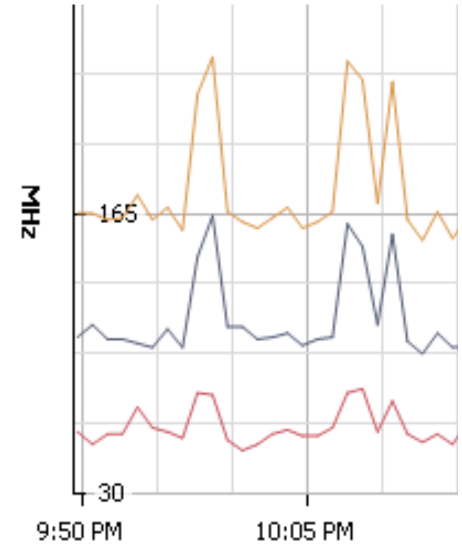
Objects and Counters

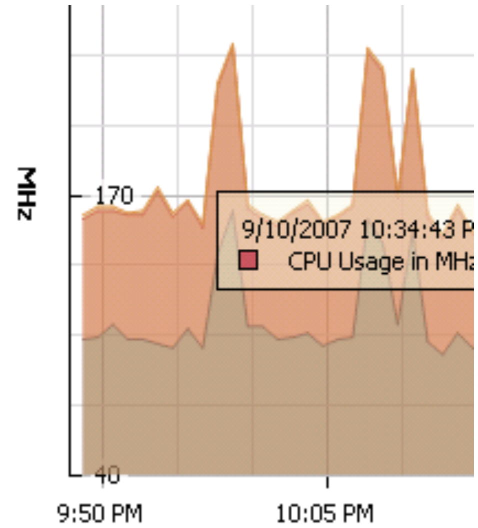
- **Objects: instances or aggregations of devices**
  - Examples: VCPU0, VCPU1, vmhba1:1:2, aggregate over all NICs

- **Counters: which stats to collect**
  - Examples:
    - CPU: used time, ready time, usage (%)
    - NIC: network packets received
    - Memory: memory swapped

Stacked vs. Line charts

- **Line**
  - Each instance shown separately
- **Stacked**
  - Graphs are stacked on top of each other
  - Only applies to certain kinds of charts, e.g.:
    - Breakdown of Host CPU MHz by Virtual Machine
    - Breakdown of Virtual Machine CPU by VCPU





esxtop

- **What is esxtop ?**
  - Performance troubleshooting tool for ESX host
  - Displays performance statistics in rows and column format

Fields

```
10:55:46am up 43 days 23:51, 61 worlds; CPU load average: 0.01, 0.01, 0.01
PCPU(%):    2.54,    1.70,    1.82,    1.16 ;  used total:    1.80
CCPU(%):    0 us,    0 sy,   97 id,    2 wa ;        cs/sec:     77

     ID      GID NAME                  NWLD   %USED    %RUN    %SYS   %WAIT    %RDY   %IDLE  %OVR
      1        1 idle                     4  395.54  395.97    0.00    0.00    6.71    0.00    0.
      2        2 system                   6    0.01    0.01    0.00  600.00    0.00    0.00    0.
      6        6 helper                  22    0.01    0.01    0.00 2200.00    0.01    0.00    0.
      7        7 drivers                 11    0.01    0.01    0.00 1100.00    0.00    0.00    0.
      9        9 console                  1    1.07    1.08    0.00   99.00    0.60   98.98    0.
     14       14 vmkapimod                2    0.00    0.00    0.00  200.00    0.00    0.00    0.
     15       15 vmware-vmkauthd          1    0.00    0.00    0.00  100.00    0.00    0.00    0.
     16       16 Windows 2003 SP          7    4.28    4.28    0.01  699.85    0.54  196.53    0.
     17       17 SQL2005                  7    1.41    1.41    0.01  700.00    0.27  199.79    0.
```

Entities -running worlds in this case

esxtop FAQ

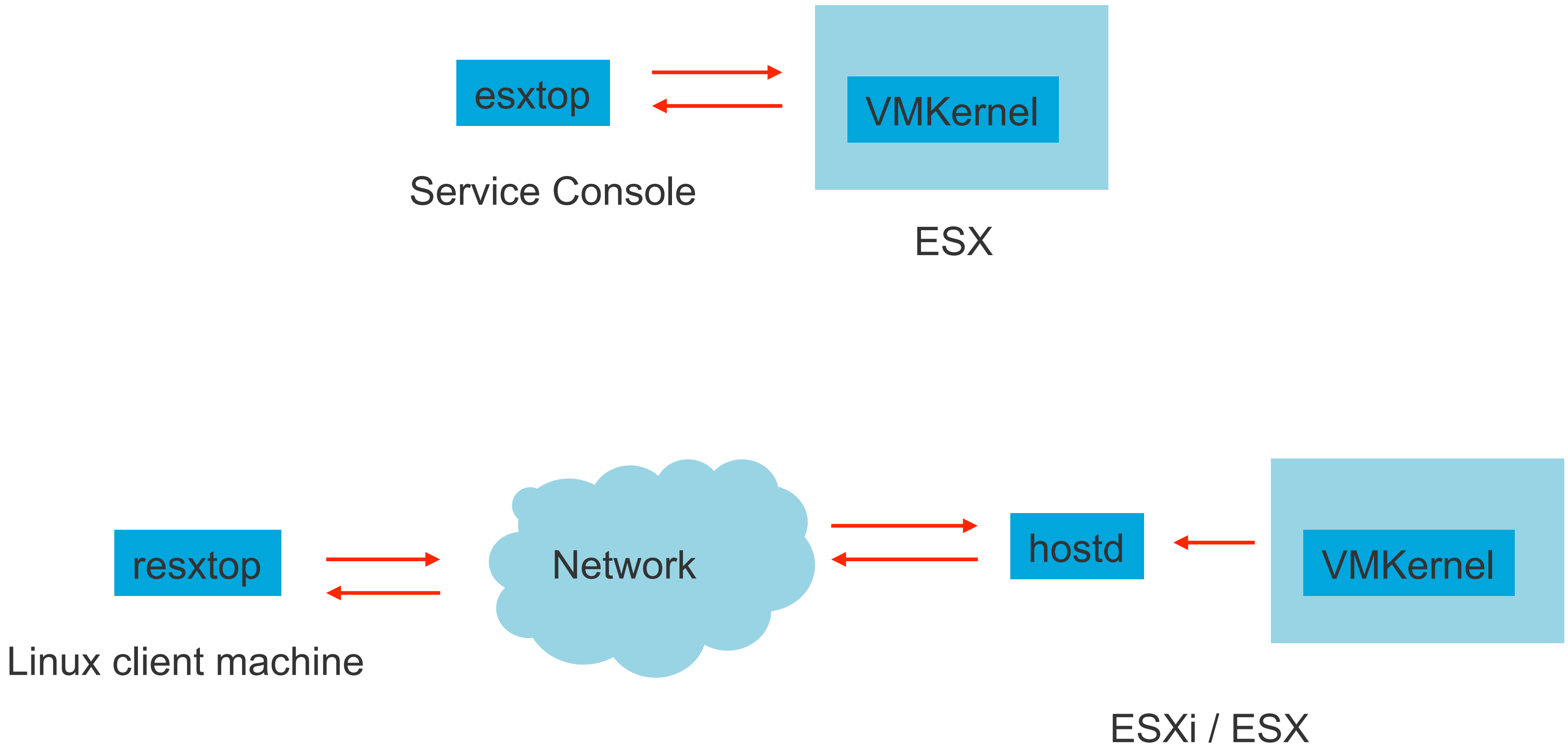
- **Where to get it?**
  - Comes pre-installed with ESX service console
  - Remote version of esxtop (resxtop) ships with the Remote Command Line interface (RCLI) package

- **What are its intended use cases?**
  - Get a quick overview of the system
  - Spot performance bottlenecks

- **What it is not meant for ?**
  - Not meant for long term performance monitoring, data mining, reporting, alerting etc. Use VI client or the SDK for those use cases

esxtop FAQ

- **What is the difference between esxtop and resxtop**



Introduction to esxtop

- **Performance statistics**
  - Some are static and don't change during runtime, for example MEMSZ (memsize), VM Name etc
  - Some are computed dynamically, for example CPU load average, memory over-commitment load average etc
  - Some are calculated from the delta between two successive snapshots. Refresh interval (-d) determines the time between successive snapshots
    - for example %CPU used = (CPU used time at snapshot 2 - CPU used time at snapshot 1) / time elapsed between snapshots

esxtop modes

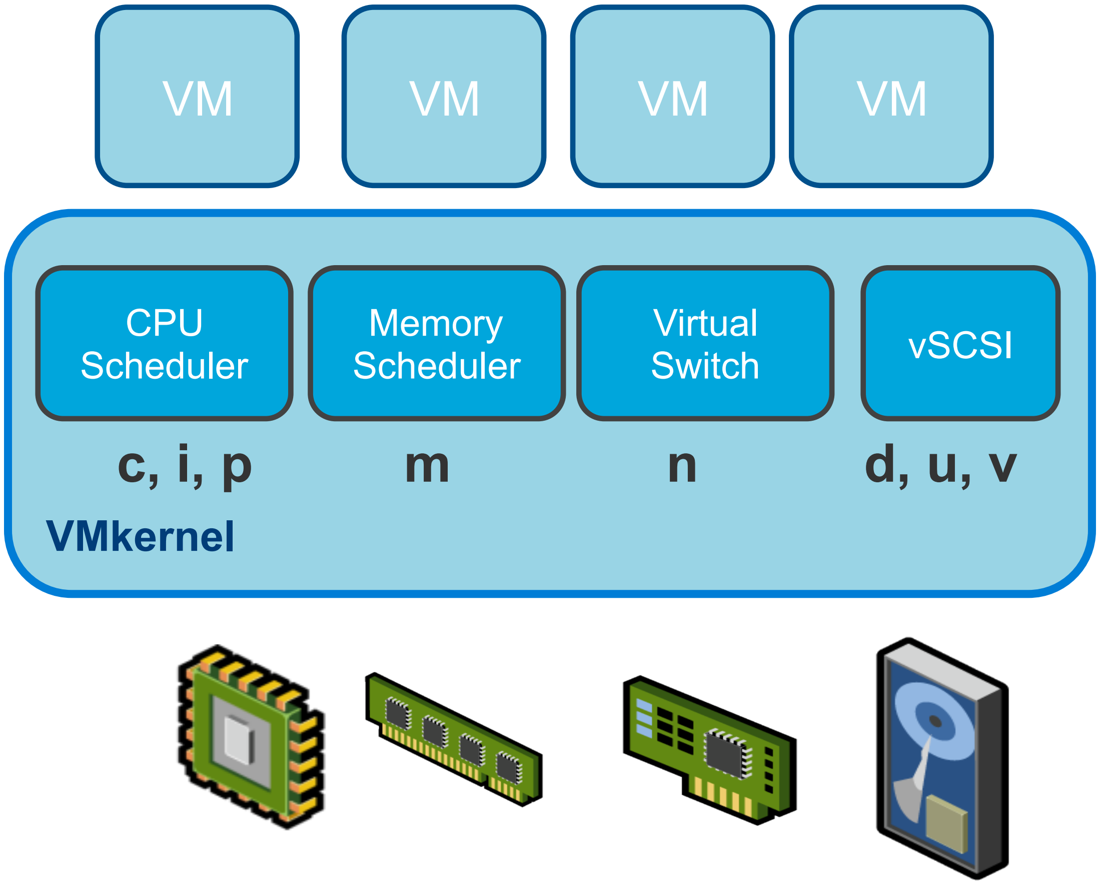
- **Interactive mode (default)**
  - Shows data in the screen and accepts keystrokes
  - Requires TERM=xterm
- **Batch mode (-b)**
  - Dumps data to stdout in CSV format
  - Dumps default fields or fields stored in the configuration file
- **Replay mode (-R)**
  - Replays data from vm-support performance snapshot

esxtop interactive mode

- **Global commands**
  - space - update display
  - s - set refresh interval (default 5 secs)
  - f - select fields (context sensitive)
  - W - save configuration file (~/.esxtop3rc)
  - V - view VM only
  - oO - Change the order of displayed fields (context sensitive)
  - ? - help (context sensitive)
  - ^L - redraw screen
  - q - quit

esxtop screens

- **Screens**
  - c: cpu (default)
  - m: memory
  - n: network
  - d: disk adapter
  - u: disk device (added in ESX 3.5)
  - v: disk VM (added in ESX 3.5)
  - i: Interrupts (new in ESX 4.0)
  - p: power management (new in ESX 4.1)



Using screen

Time Uptime running worlds

```
 8:34:56am up 47 days 21:30, 80 worlds; CPU load average: 0.03, 0.03, 0.11
PCPU(%):   3.78,   2.31,   1.89,   4.63 ;  used total:   3.15
CCPU(%):   0 us,   0 sy, 100 id,   0 wa ;       cs/sec:    75
```

ID	GID	NAME	NWLD	%USED	%RUN	%SYS	%WAIT	%RDY	%IDLE	%OVRLP	%CSTP	%ML
1	1	idle	4	390.25	390.81	0.00	0.00	11.86	0.00	0.00	0.00	0
2	2	system	6	0.00	0.00	0.00	600.00	0.00	0.00	0.00	0.00	0
6	6	helper	22	0.01	0.01	0.00	2200.00	0.01	0.00	0.00	0.00	0
7	7	drivers	11	0.01	0.01	0.00	1100.00	0.00	0.00	0.00	0.00	0
9	9	console	1	0.56	0.56	0.00	99.93	0.18	99.92	0.04	0.00	0
14	14	vmkapimod	2	0.00	0.00	0.00	200.00	0.00	0.00	0.00	0.00	0
15	15	vmware-vmkauthd	1	0.00	0.00	0.00	100.00	0.00	0.00	0.00	0.00	0
16	16	Windows 2003 SP	7	2.91	2.90	0.01	700.00	0.64	198.16	0.17	0.00	0
17	17	SQL2005	7	1.58	1.57	0.02	700.00	0.50	199.59	0.13	0.00	0

fields hidden from the view…

- Worlds = VMKernel processes
- ID = world identifier
- GID = world group identifier
- NWLD = number of worlds

Using screen - expanding groups

press ‘e’ key

```
Group to expand/rollup (gid): 17
```

ID	GID	NAME	NWLD	%USED	%RUN	%SYS	%WAIT	%RDY	%IDLE	%OVRLP	%CSTP	%ML
1	1	idle	4	386.38	387.21	0.00	0.00	12.14	0.00	0.00	0.00	0
2	2	system	6	0.01	0.01	0.00	599.02	0.00	0.00	0.00	0.00	0
6	6	helper	22	0.00	0.00	0.00	2196.41	0.00	0.00	0.00	0.00	0
7	7	drivers	11	0.01	0.01	0.00	1098.20	0.00	0.00	0.00	0.00	0
9	9	console	1	0.47	0.51	0.00	99.21	0.12	99.21	0.05	0.00	0
14	14	vmkapimod	2	0.00	0.00	0.00	199.67	0.00	0.00	0.00	0.00	0
15	15	vmware-vmkauthd	1	0.00	0.00	0.00	99.84	0.00	0.00	0.00	0.00	0
16	16	Windows 2003 SP	7	3.81	3.81	0.01	694.29	0.76	195.39	0.13	0.00	0
1078	17	vmware-vmx	1	0.05	0.05	0.00	99.77	0.01	0.00	0.00	0.00	0
1079	17	vmm0:SQL2005	1	0.74	0.74	0.01	98.96	0.14	98.87	0.04	0.00	0
1080	17	vmm1:SQL2005	1	0.44	0.44	0.00	99.14	0.26	99.04	0.03	0.00	0
1081	17	vmware-vmx	1	0.00	0.00	0.00	99.83	0.00	0.00	0.00	0.00	0
1082	17	mks:SQL2005	1	0.13	0.13	0.00	99.60	0.11	0.00	0.03	0.00	0
1083	17	vcpu-0:SQL2005	1	0.01	0.01	0.00	99.83	0.00	0.00	0.00	0.00	0
1084	17	vcpu-1:SQL2005	1	0.01	0.01	0.00	99.83	0.00	0.00	0.00	0.00	0
18	18	vc server	7	1.08	1.08	0.00	697.45	0.33	198.40	0.16	0.00	0

•In rolled up view stats are cumulative of all the worlds in the group
•Expanded view gives breakdown per world
•VM group consists of mks, vcpu, vmx worlds. SMP VMs have additional vcpu and vmm worlds
•vmm0, vmm1 = Virtual machine monitors for vCPU0 and vCPU1 respectively

esxtop replay mode

- **To record esxtop data**
  - vm-support -S -d <duration>

- **To replay**
  - tar xvzf vm-support-dump.tgz
  - cd vm-support-*/
  - esxtop -R ./ (esxtop version should match)

esxtop replay mode

Current time

```
 4:40:47am up 47 days 19:37, 80 worlds; CPU load average: 0.04, 0.04, 0.04
PCPU(%):  10.94,   3.13,   3.48,   2.76 ;   used total:   5.08
CCPU(%):    0 us,    6 sy,   91 id,    3 wa ;        cs/sec:    199
```

ID	GID	NAME	NWLD	%USED	%RUN	%SYS	%WAIT	%RDY
1	1	idle	4	379.00	379.69	0.00	0.00	19.43
2	2	system	6	0.01	0.01	0.00	598.89	0.00
6	6	helper	22	0.16	0.16	0.00	2194.82	0.16
7	7	drivers	11	0.01	0.01	0.00	1098.32	0.00
9	9	console	1	9.17	9.23	0.01	90.11	0.43
14	14	vmkapimod	2	0.00	0.00	0.00	199.94	0.00
15	15	vmware-vmkauthd	1	0.00	0.00	0.00	99.97	0.00
16	16	Windows 2003 SP	7	3.39	3.39	0.02	695.68	0.72
17	17	SQL2005	7	1.33	1.32	0.01	698.04	0.40
18	18	vc server	7	0.98	0.97	0.01	698.57	0.25
19	19	Conductor	5	2.74	2.72	0.03	496.34	0.79
20	20	fakeDB	7	1.30	1.30	0.00	698.14	0.33

```
*** Read stats from ./snapshots/vsi/vsi.2 ***
```

esxtop batch mode

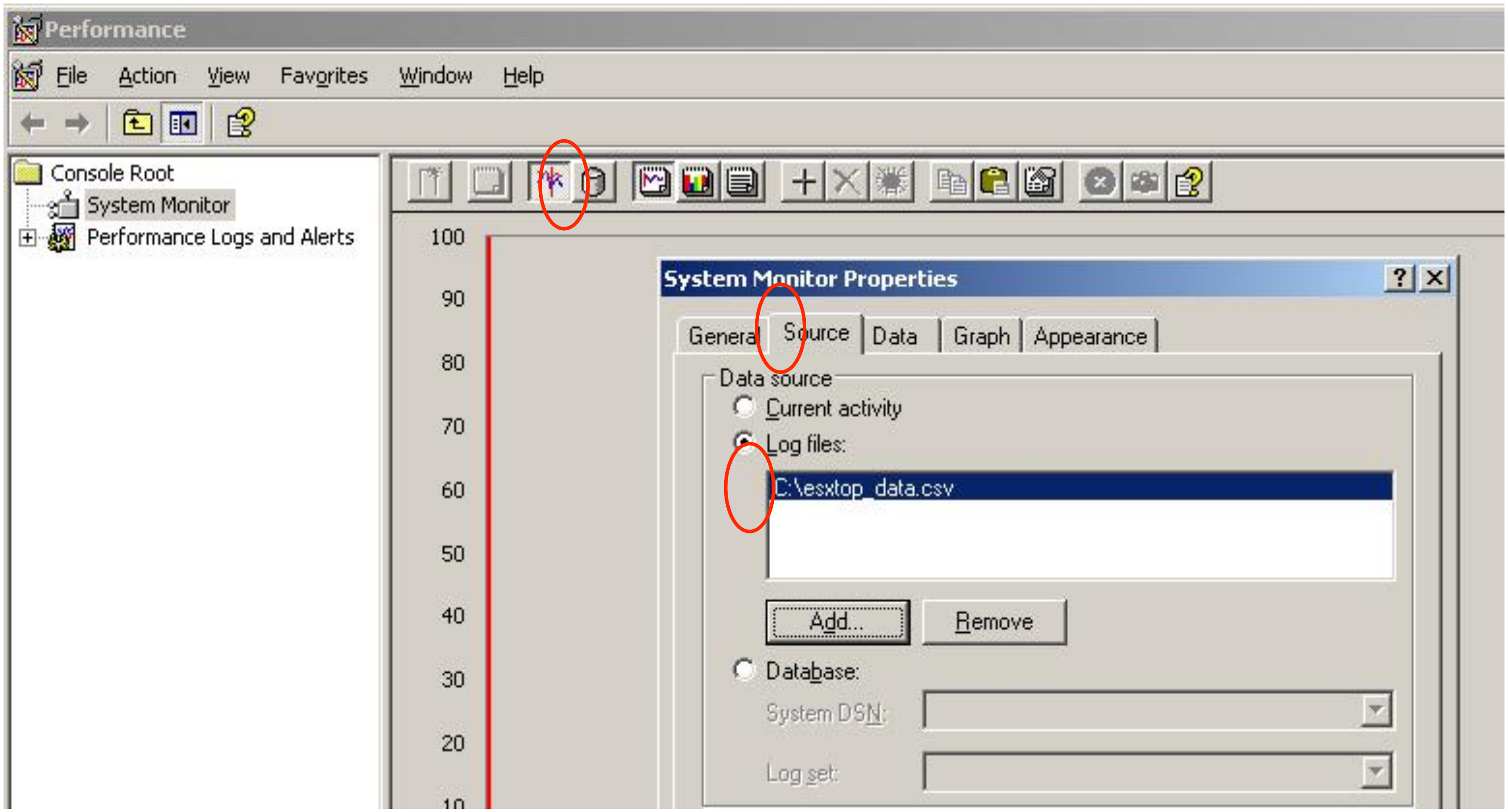
- **Batch mode (-b)**
  - Produces windows perfmon compatible CSV file
  - CSV file compatibility requires fixed number of columns on every row - statistics of VMs/worlds instances that appear after starting the batch mode are not collected because of this reason
  - Only counters that are specified in the configuration file are collected, (-a) option collects all counters
  - Counters are named slightly differently

vmware®

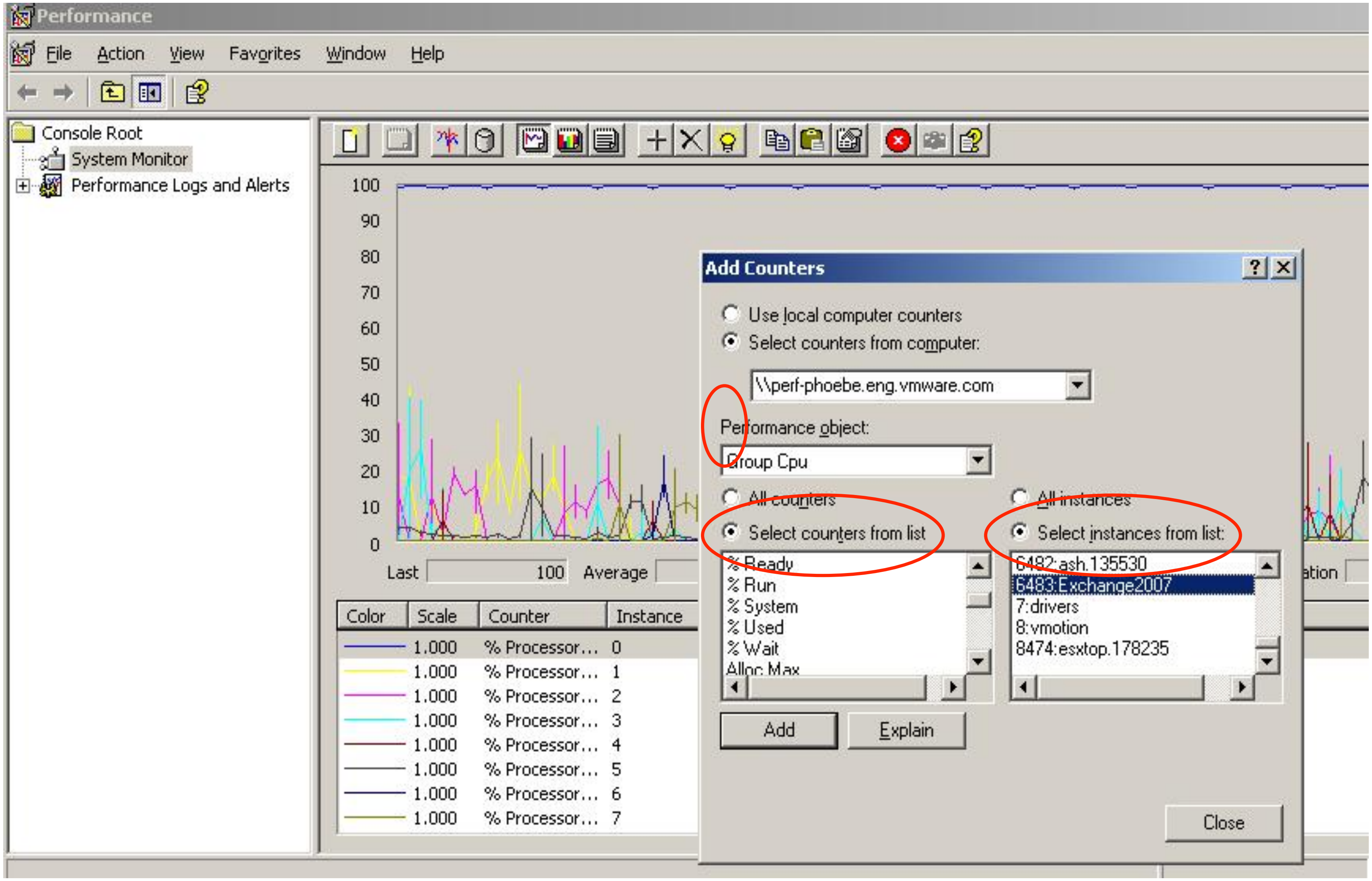
esxtop batch mode

- **To use batch mode**
  - esxtop -b > esxtop_output.csv
- **To select fields**
  - Run esxtop in interactive mode
  - Select the fields
  - Save configuration file (‘w’ key)
- **To dump all fields**
  - esxtop -b -a > esxtop_output.csv

esxtop batch mode – importing data into perfmon



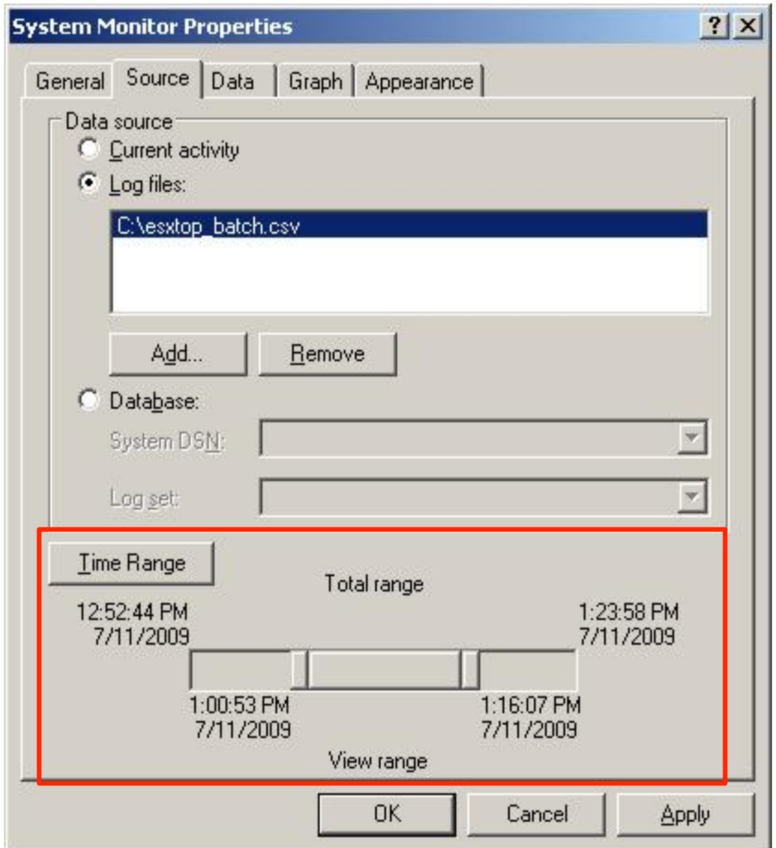
esxtop batch mode – viewing data in perfmon



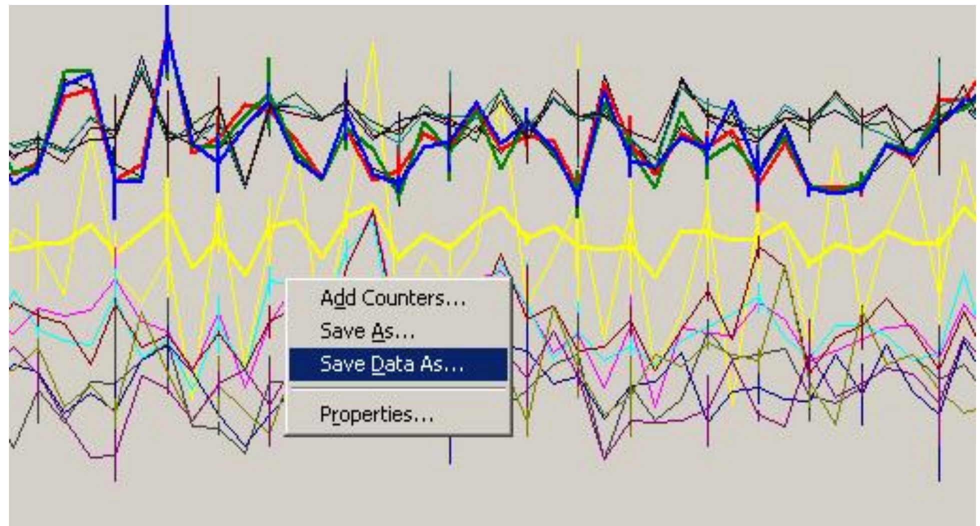
vmware

esxtop batch mode – trimming data

Trimming data



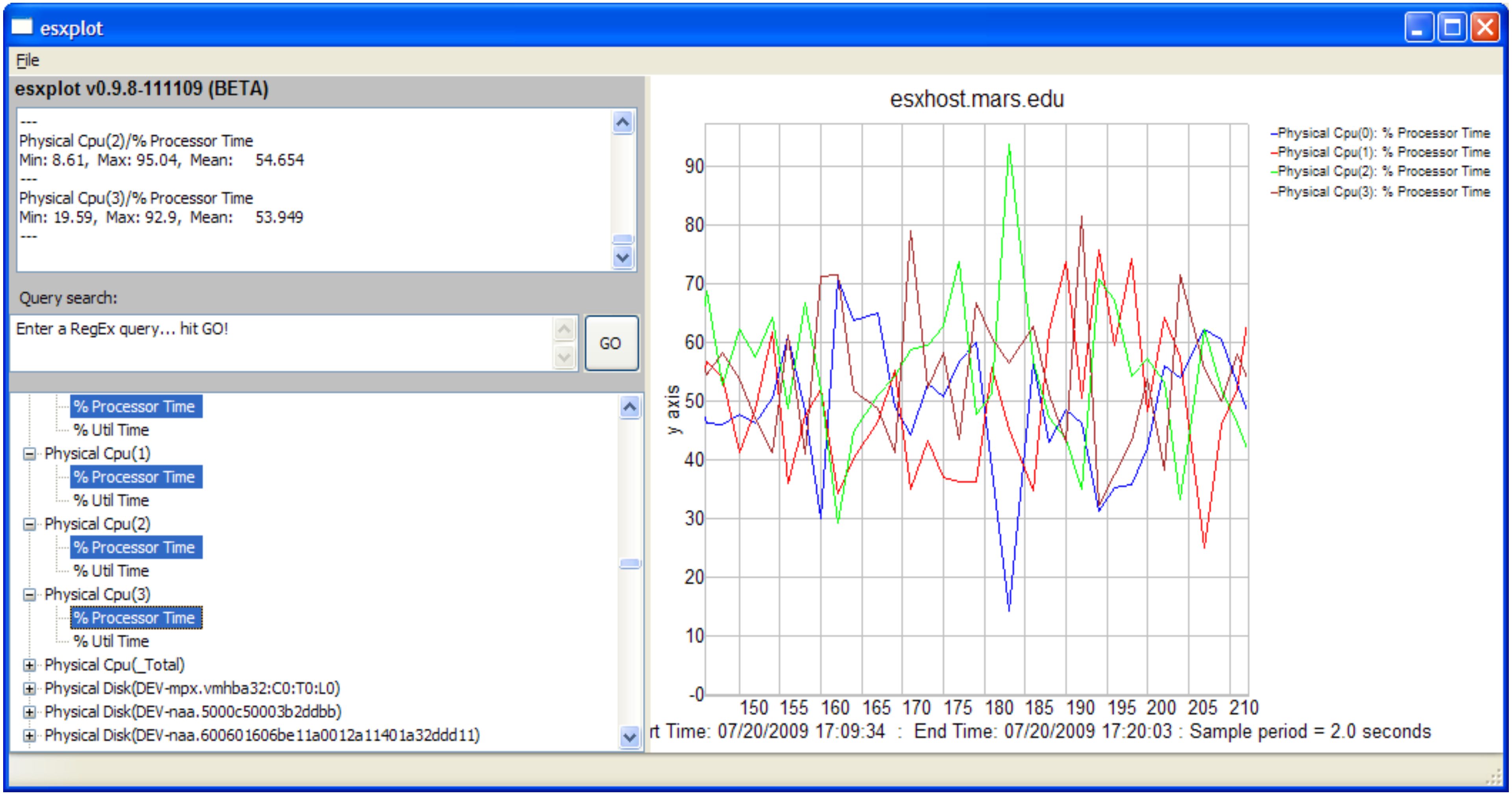
Saving data after trim



vmware

esxplot

- http://labs.vmware.com/flings/esxplot



SDK

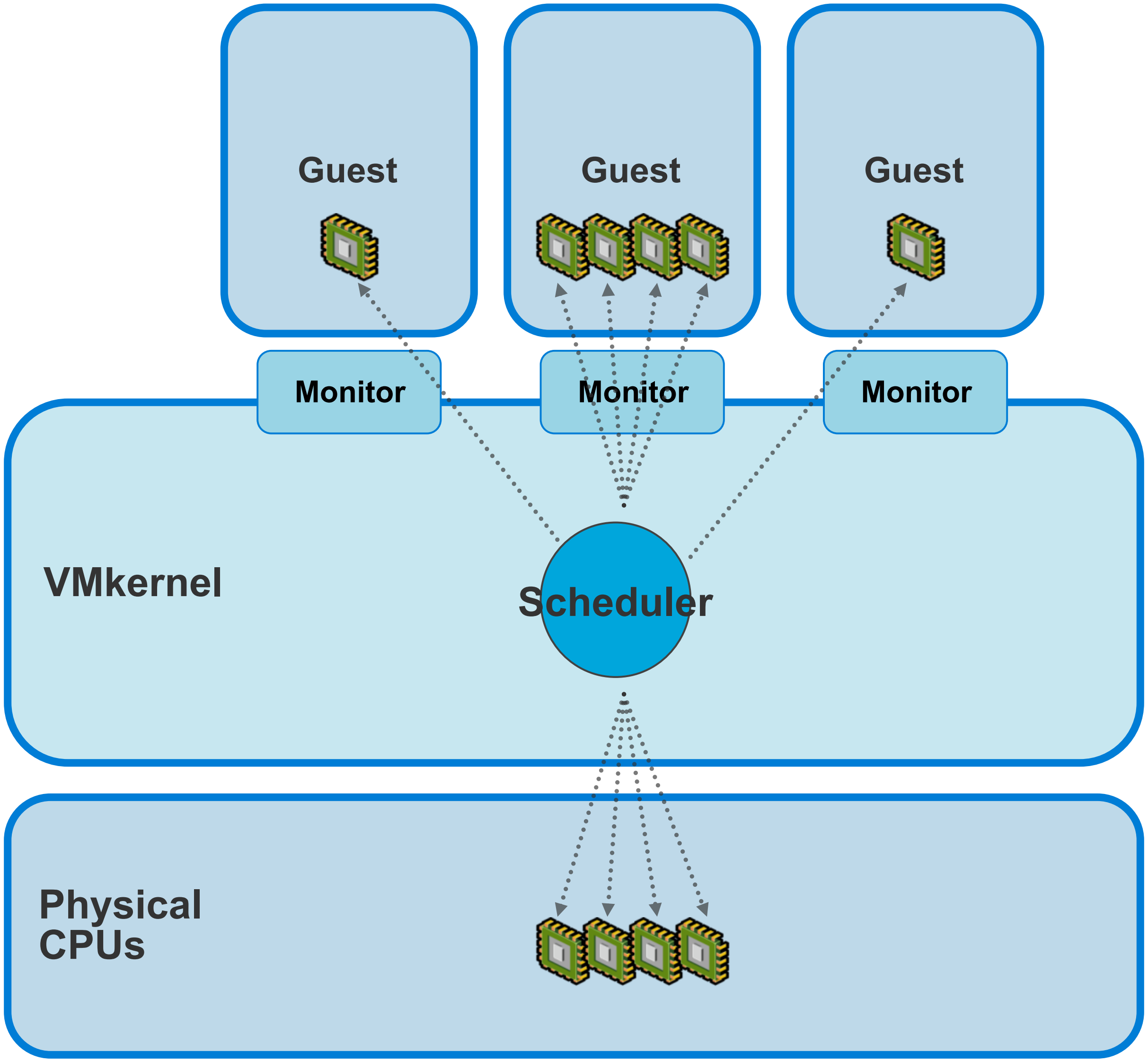
- Use the VIM API to access statistics relevant to a particular user
- Can only access statistics that are exported by the VIM API (and thus are accessible via esxtop/VI client)

Conclusions

- **Always Analyze with a Latency approach**
  - Response time of user
  - Queuing for resources in the guest
  - Queuing for resources in vSphere
  - Queing for resources outside of the host (SAN, NAS etc)
- **These tools are useful in different contexts**
  - Real-time data: esxtop
  - Historical data: VirtualCenter
  - Coarse-grained resource/cluster usage: VirtualCenter
  - Fine-grained resource usage: esxtop

CPU

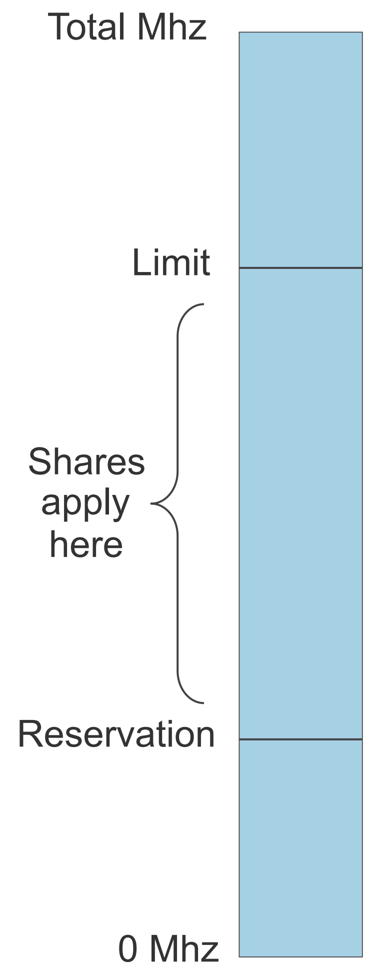
CPUs and Scheduling



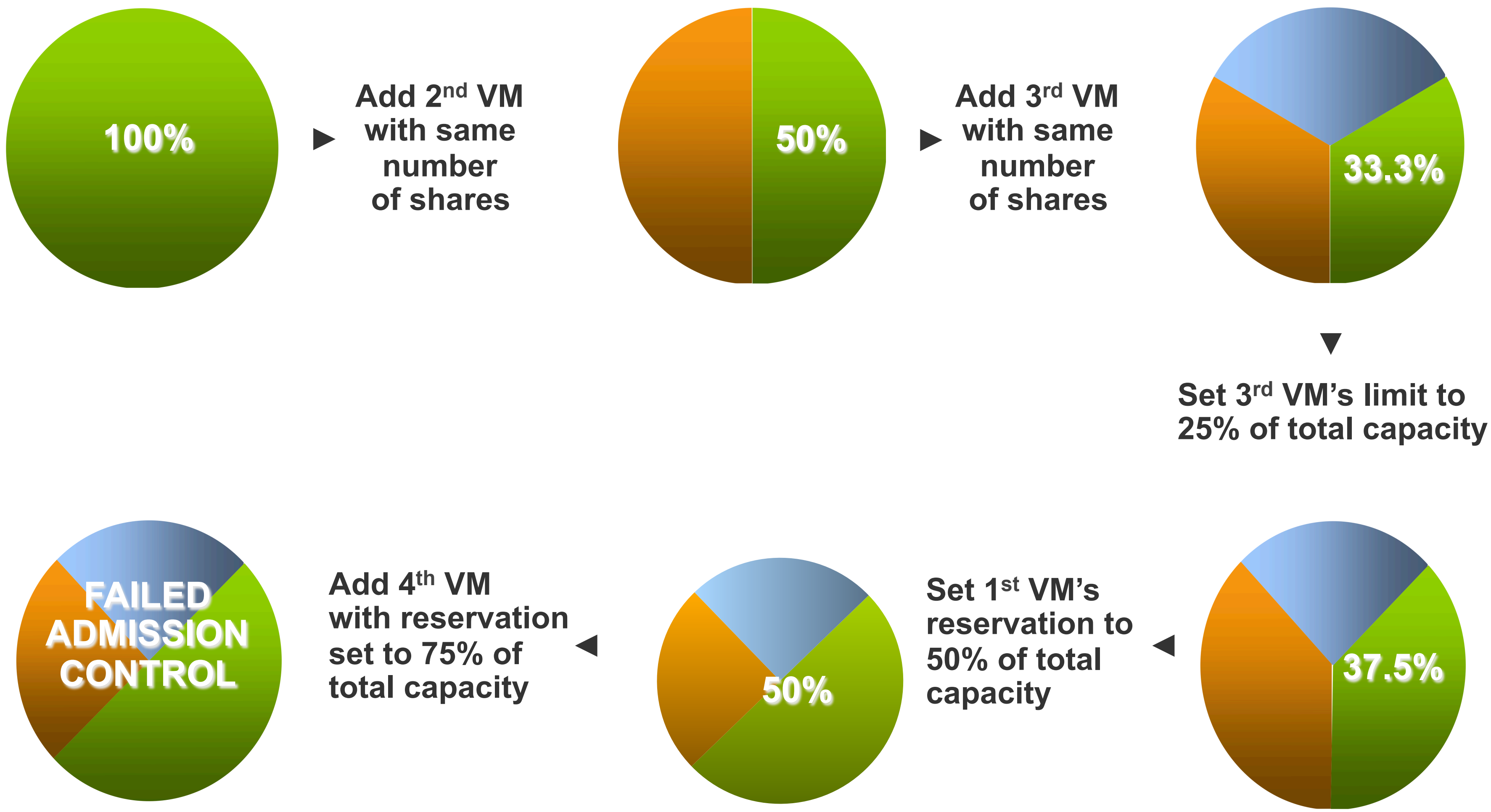
- Schedule virtual CPUs on physical CPUs
- Virtual time based proportional-share CPU scheduler
- Flexible and accurate rate-based controls over CPU time allocations
- NUMA/processor/cache topology aware
- Provide graceful degradation in over-commitment situations
- High scalability with low scheduling latencies
- Fine-grain built-in accounting for workload observability
- Support for VSMP virtual machines

Resource Controls

- **Reservation**
  - Minimum service level guarantee (in MHz)
  - Even when system is overcommitted
  - Needs to pass admission control
- **Shares**
  - CPU entitlement is directly proportional to VM's shares and depends on the total number of shares issued
  - Abstract number, only ratio matters
- **Limit**
  - Absolute upper bound on CPU entitlement (in MHz)
  - Even when system is not overcommitted

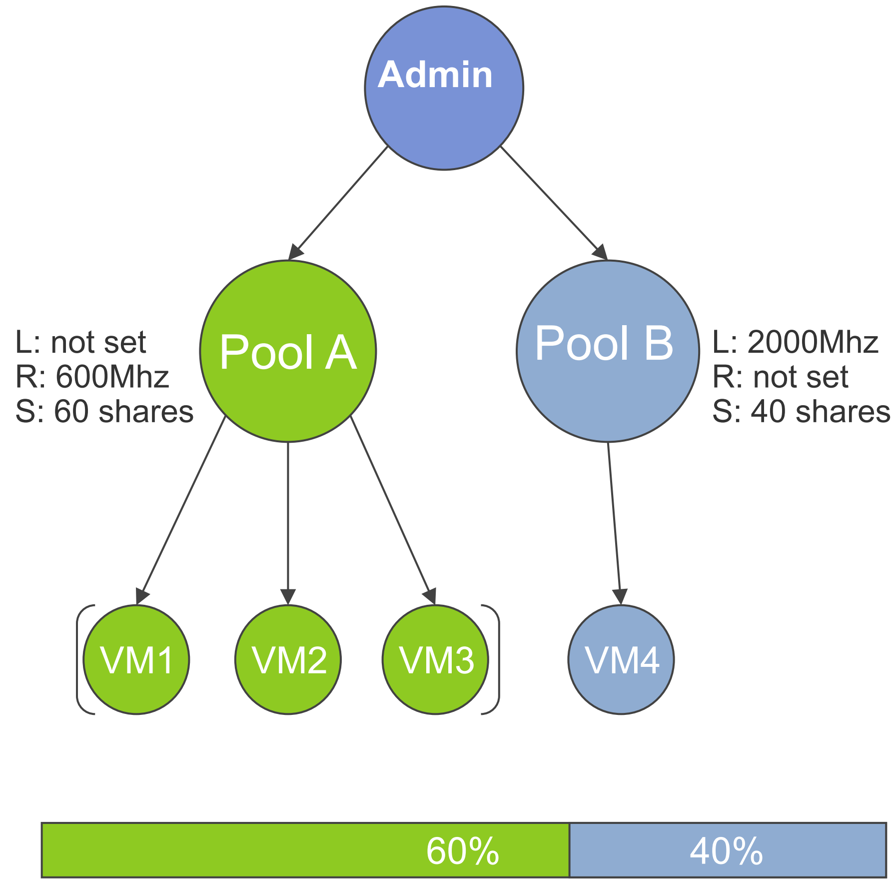


Resource Control Example

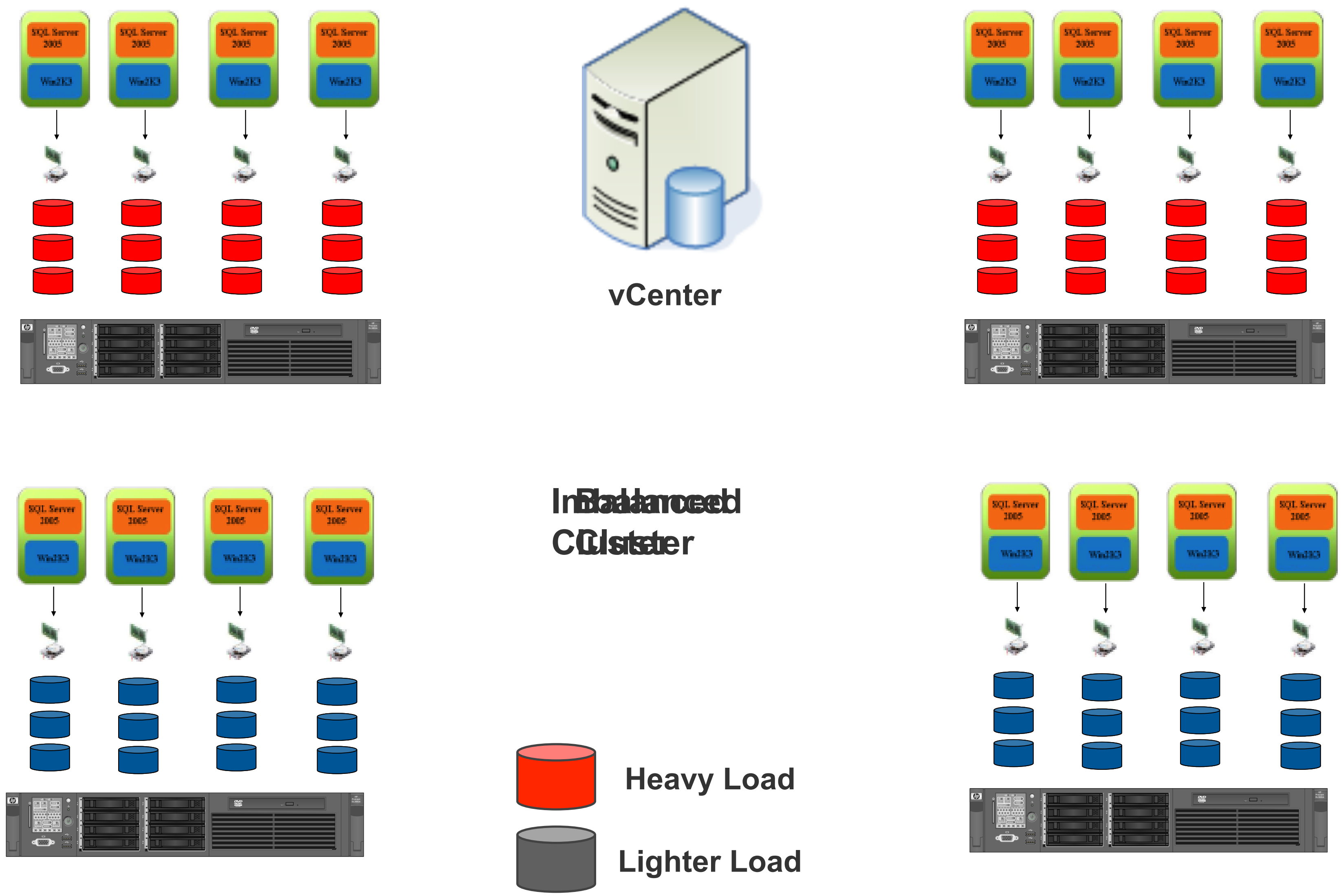


Resource Pools

- **Motivation**
  - Allocate aggregate resources for sets of VMs
  - Isolation between pools, sharing within pools
  - Flexible hierarchical organization
  - Access control and delegation
- **What is a resource pool?**
  - Abstract object with permissions
  - Reservation, limit, and shares
  - Parent pool, child pools and VMs
  - Can be used on a stand-alone host or in a cluster (group of hosts)

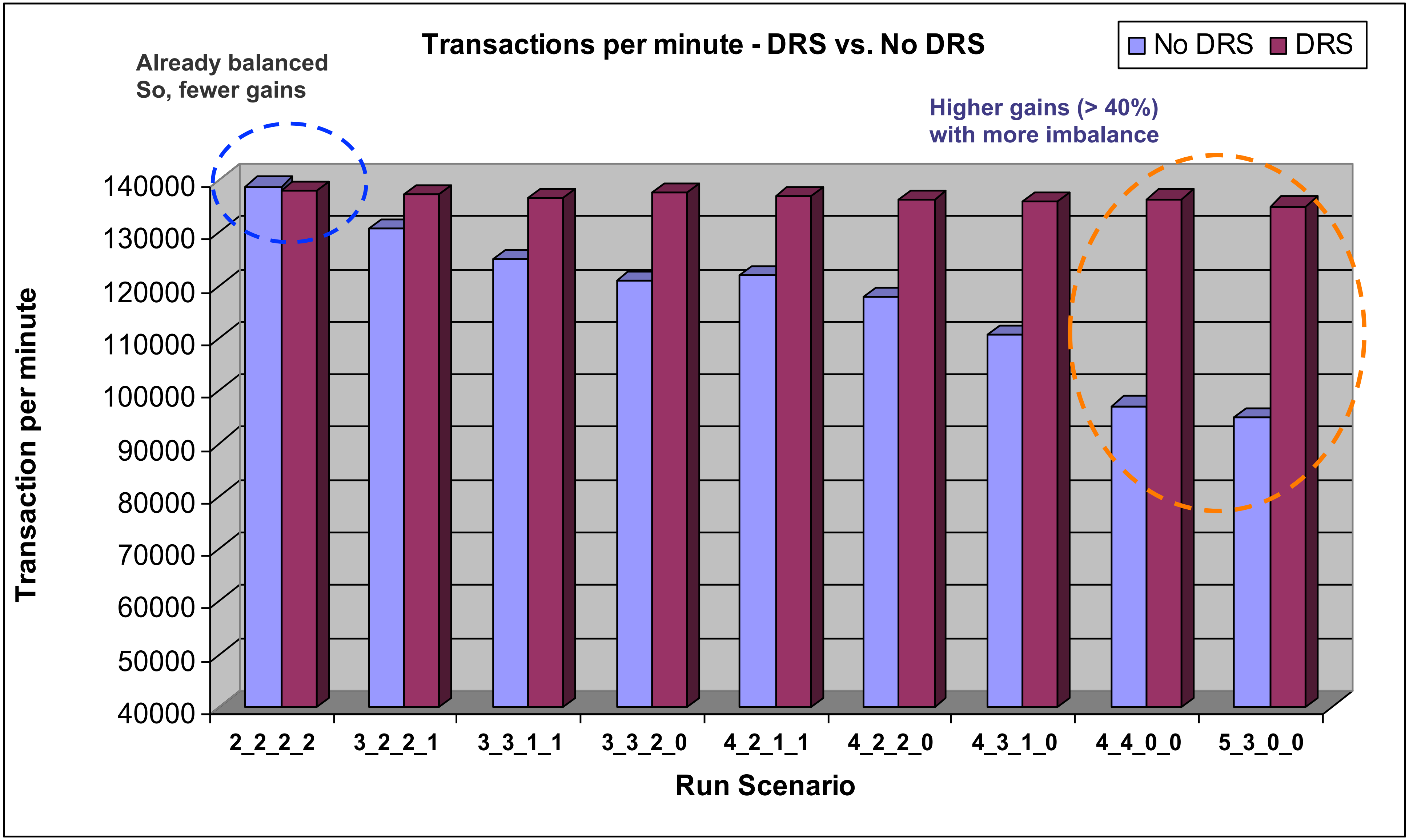


Example migration scenario 4_4_0_0 with DRS



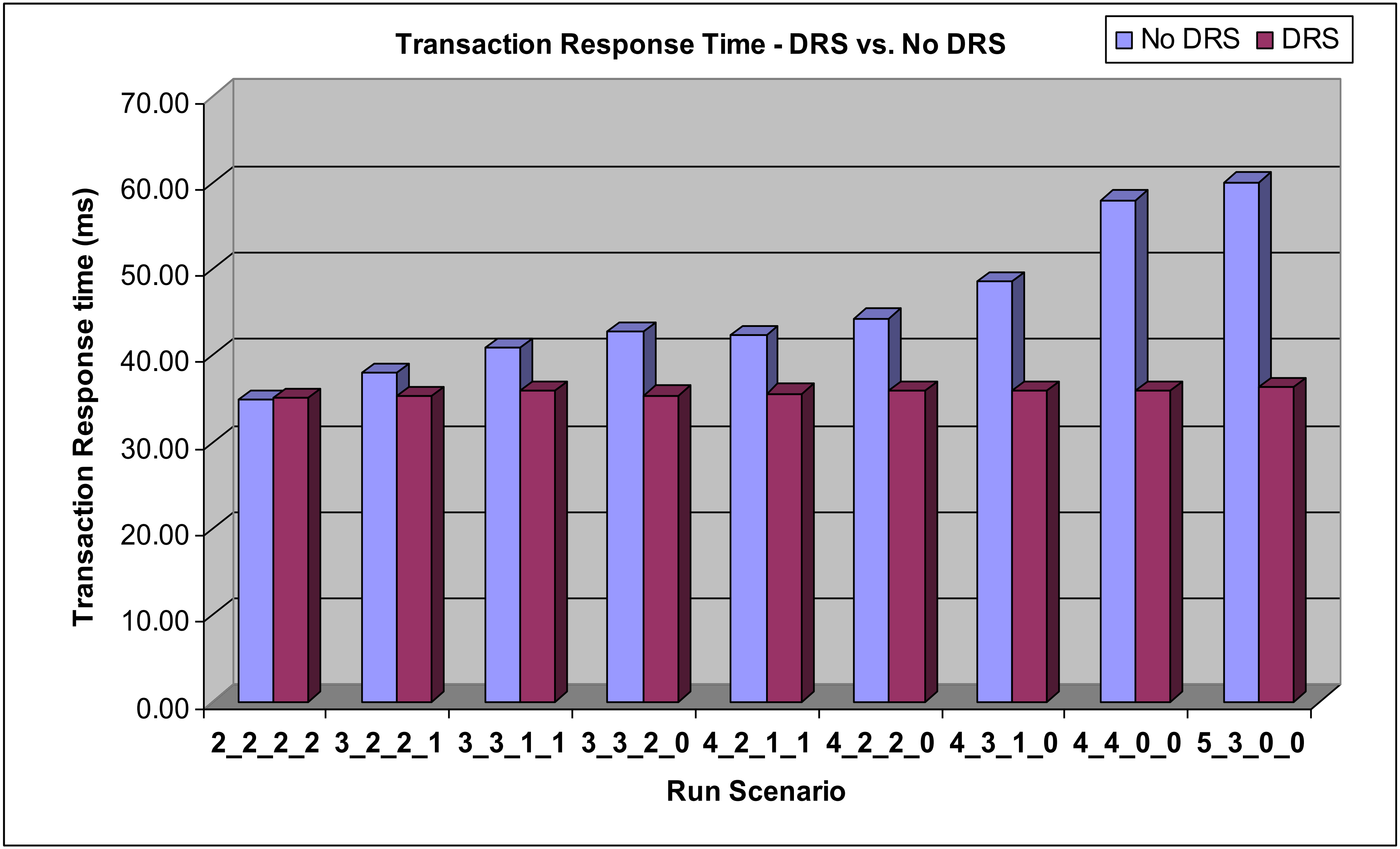
DRS Scalability – Transactions per minute

(Higher the better)



DRS Scalability – Application Response Time

(Lower the better)

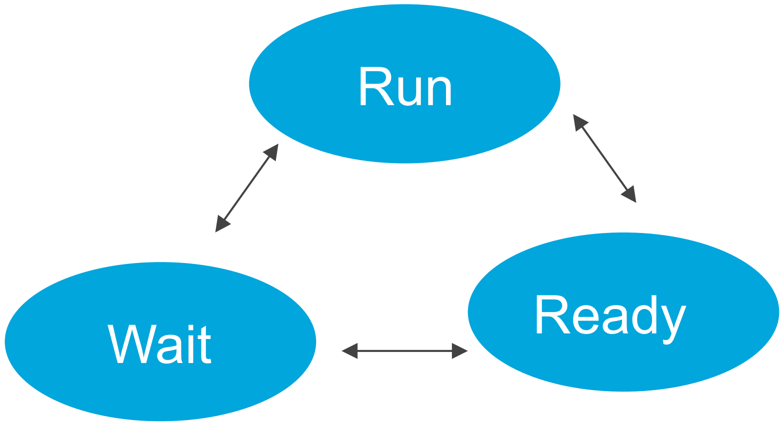


ESX CPU Scheduling States

- **World states (simplified view):**
  - ready = ready-to-run but no physical CPU free
  - run = currently active and running
  - wait = blocked on I/O
- **Multi-CPU Virtual Machines => gang scheduling**
  - Co-run (latency to get vCPUs running)
  - Co-stop (time in “stopped” state)

Ready Time (1 of 2)

- **VM state**
  - running (%used)
  - waiting (%twait)
  - ready to run (%ready)



- **When does a VM go to “ready to run” state**
  - Guest wants to run or need to be woken up (to deliver an interrupt)
  - CPU unavailable for scheduling the VM

Ready Time (2 of 2)

- **Factors affecting CPU availability**
  - CPU overcommitment
    - Even Idle VMs have to be scheduled periodically to deliver timer interrupts
  - NUMA constraints
    - NUMA node locality gives better performance
  - Burstiness – Inter-related workloads
    - Tip: Use host anti affinity rules to place inter related workloads on different hosts
  - Co-scheduling constraints
  - CPU affinity restrictions

Fact: Ready time could exist even when CPU usage is low

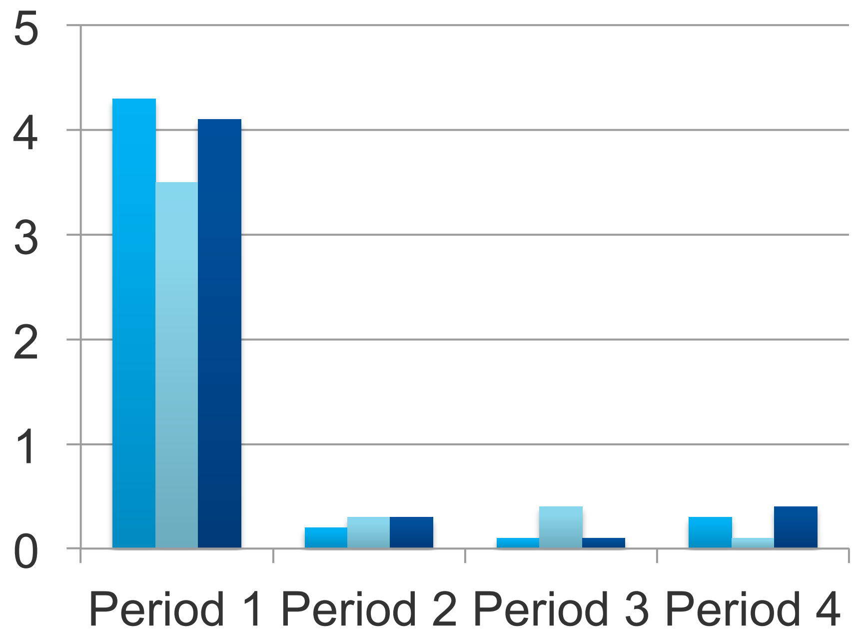
Different Metrics for Different Reasons

- **Problem Indication**
  - Response Times, Latency contributors
  - Queuing
- **Headroom Calculation**
  - Measure Utilization, predict headroom
- **Capacity Prediction**
  - If I have n users today, how much resource is needed in the future?
- **Service Level Prediction**
  - Predict the effect of response time changes
  - Resource or Load changes

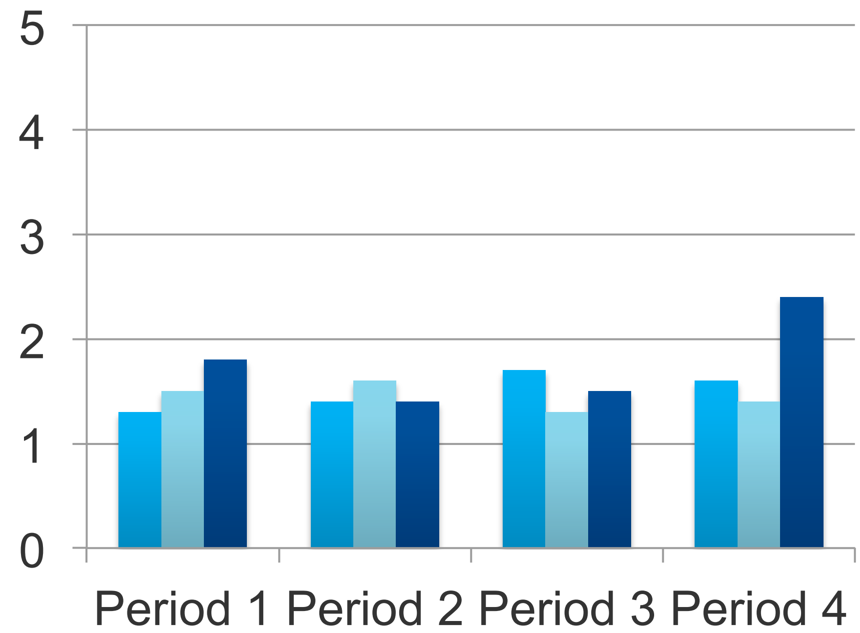
Myths and Fallacies

- **High CPU utilization is an indicator of a problem**
  - Not always: Single threaded compute intensive jobs operate quite happily at 100%
- **Less than 100% CPU means service is good (false)**
  - Not always: Bursty transaction oriented workloads follow littles-law curve, which limits effective utilization to a lower number

Consider these two workloads

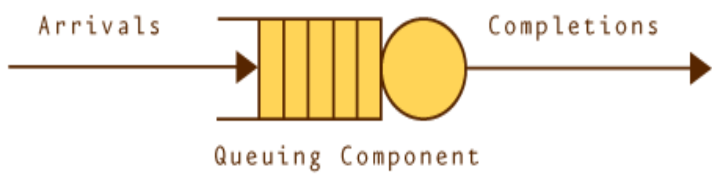


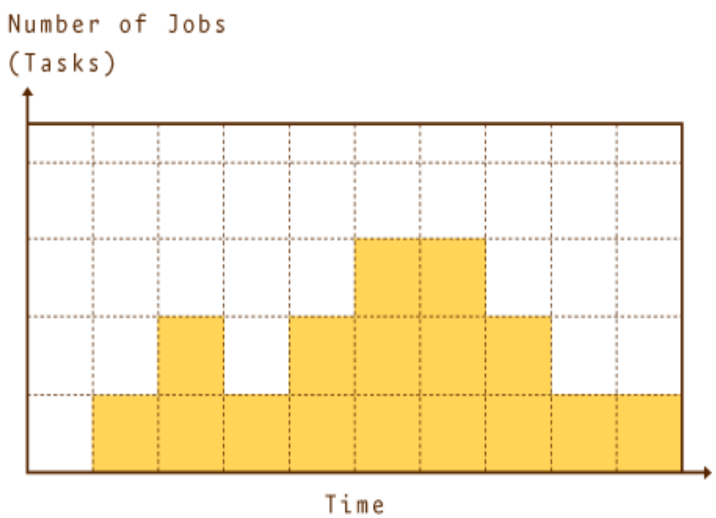
Utilization is 25%
Average Response time is high



Utilization is 25%
Average Response time is low

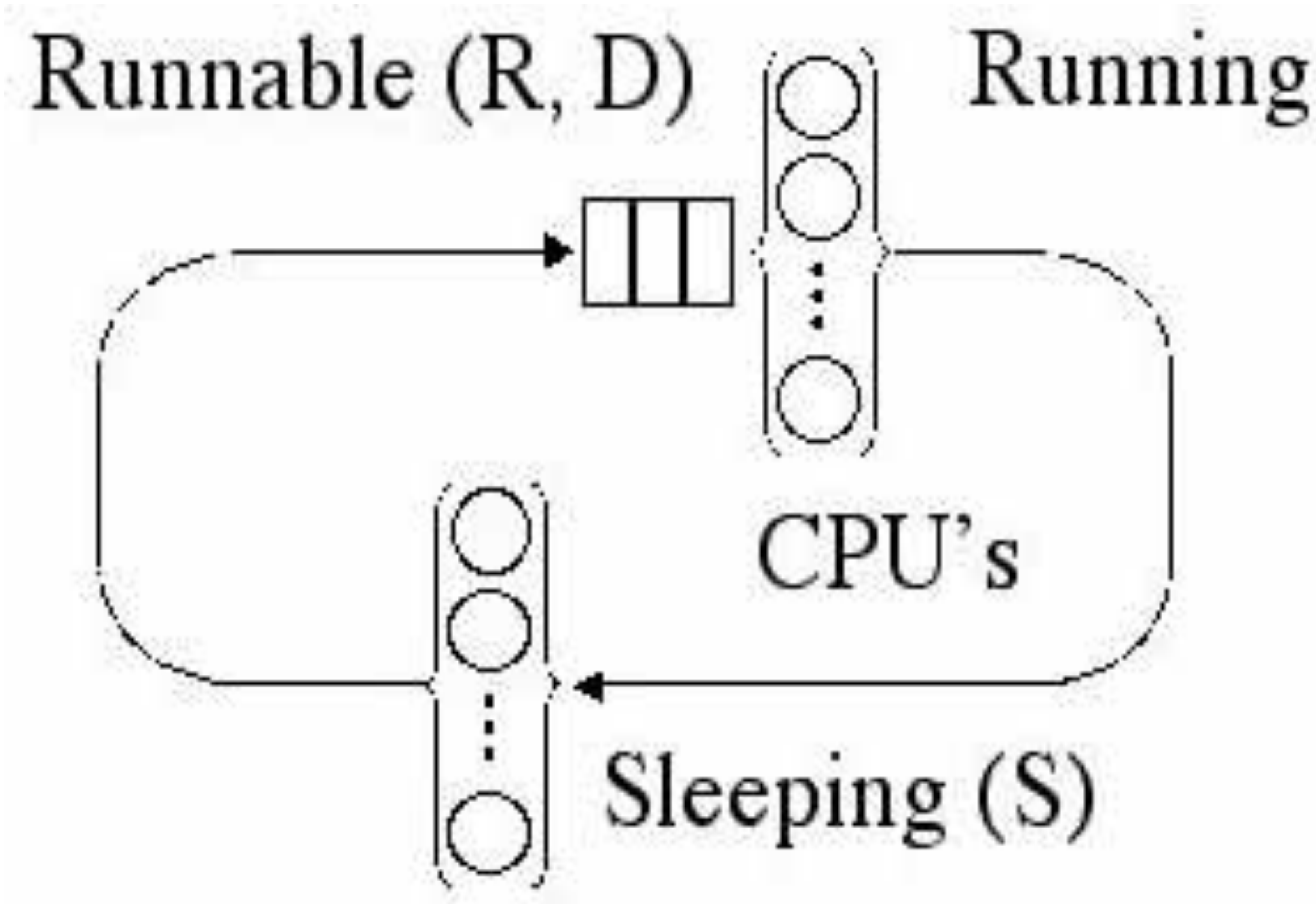
The Buzen and Denning Method





Metric	Symbol	Definition
Length of Observation	T	Total number of time units over which the observation has been made.
Arrivals	N	Total number of Arrivals over the length of observation.
Completions	C	Total number of Completions over the length of the observation.
Busy Time	B	The number of time units where the number of messages in the system exceeds zero.
Utilisation	U	The calculated value: $U = \frac{B}{T}$
Throughput	X	The calculated value: $X = \frac{C}{T}$
Mean Service Time	S	The calculated value: $S = \frac{B}{C}$
Execution Distribution	A	The calculated value: $A = \sum_{t=0}^{T} (Messages_t)$
Mean Queue Length	L	The calculated value: $L = \frac{A}{T}$
Residence Time	RT	The calculated value: $RT = \frac{A}{C}$
Queuing Time	Q	The calculated value: RT – S

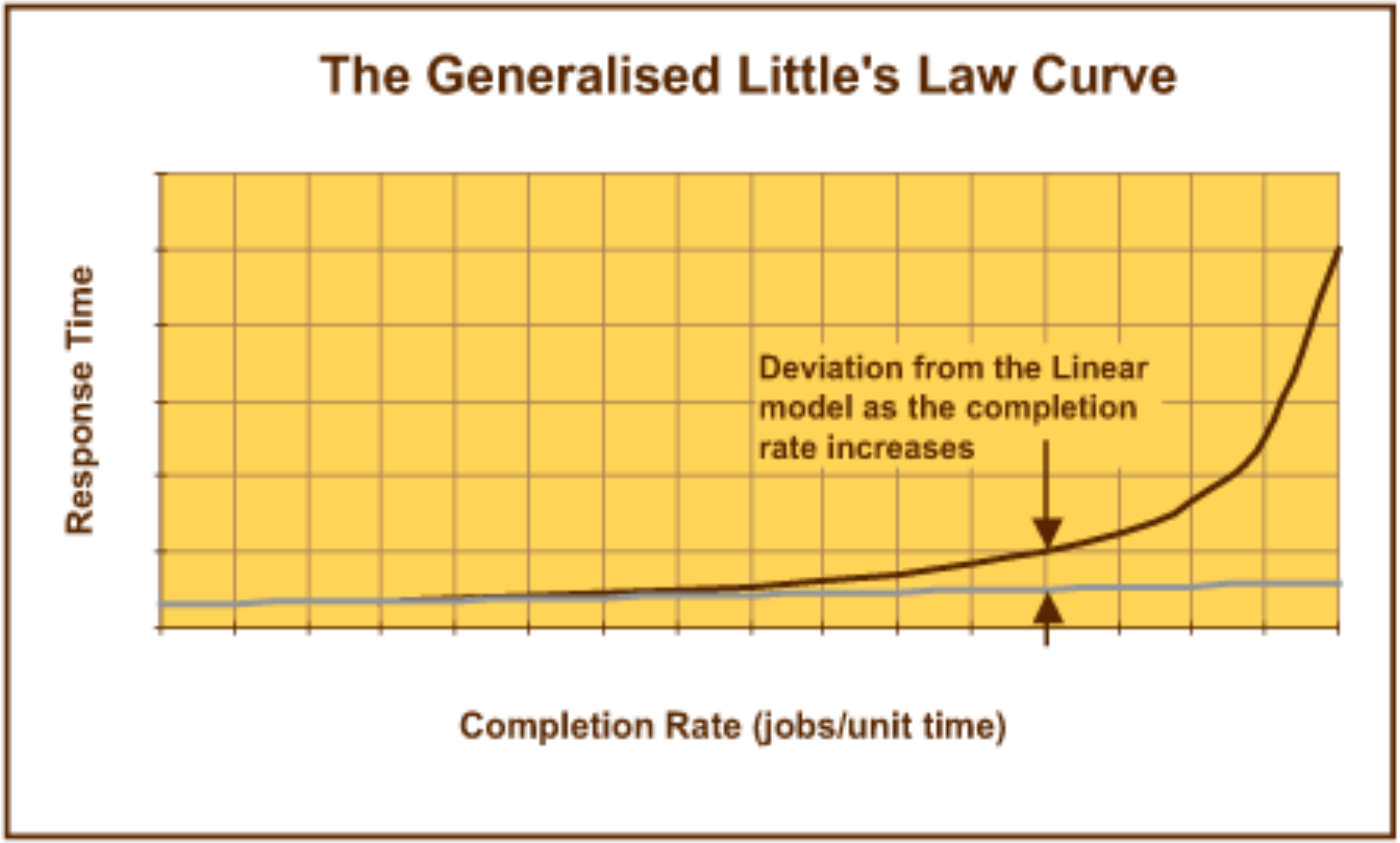
Simple model of the Scheduler



CPU and Queuing Metrics

- **How much CPU is too much?**
  - It’s workload dependent.
  - The only reliable metrics is to calculate how much time a workload waits in a queue for CPU
  - This must be a measure of guest-level threads (not VMkernel)
- **Which is better – a faster CPU or more CPUs?**
  - Typical question in the physical world
  - Question for us: will additional vCPUs help?

Relationship between Utilization and Response Time



Tools for diagnosing CPU performance: VI Client

- **Basic stuff**
  - CPU usage (percent)
  - CPU ready time (but ready time by itself can be misleading)
- **Advanced stuff**
  - CPU wait time: time spent blocked on IO
  - CPU extra time: time given to virtual machine over reservation
  - CPU guaranteed: min CPU for virtual machine
- **Cluster-level statistics**
  - Percent of entitled resources delivered
  - Utilization percent
  - Effective CPU resources: MHz for cluster

CPU capacity

How do we know we are maxed out?

- If VMs are waiting for CPU time, maybe we need more CPUs.
- To measure this, look at CPU *ready time*.

What exactly am I looking for?

- For each host, collect *ready time* for each VM
- Compute *%ready time* for each VM (*ready time/sampling interval*)
- If average *%ready time* > 50%, probe further

Possible options

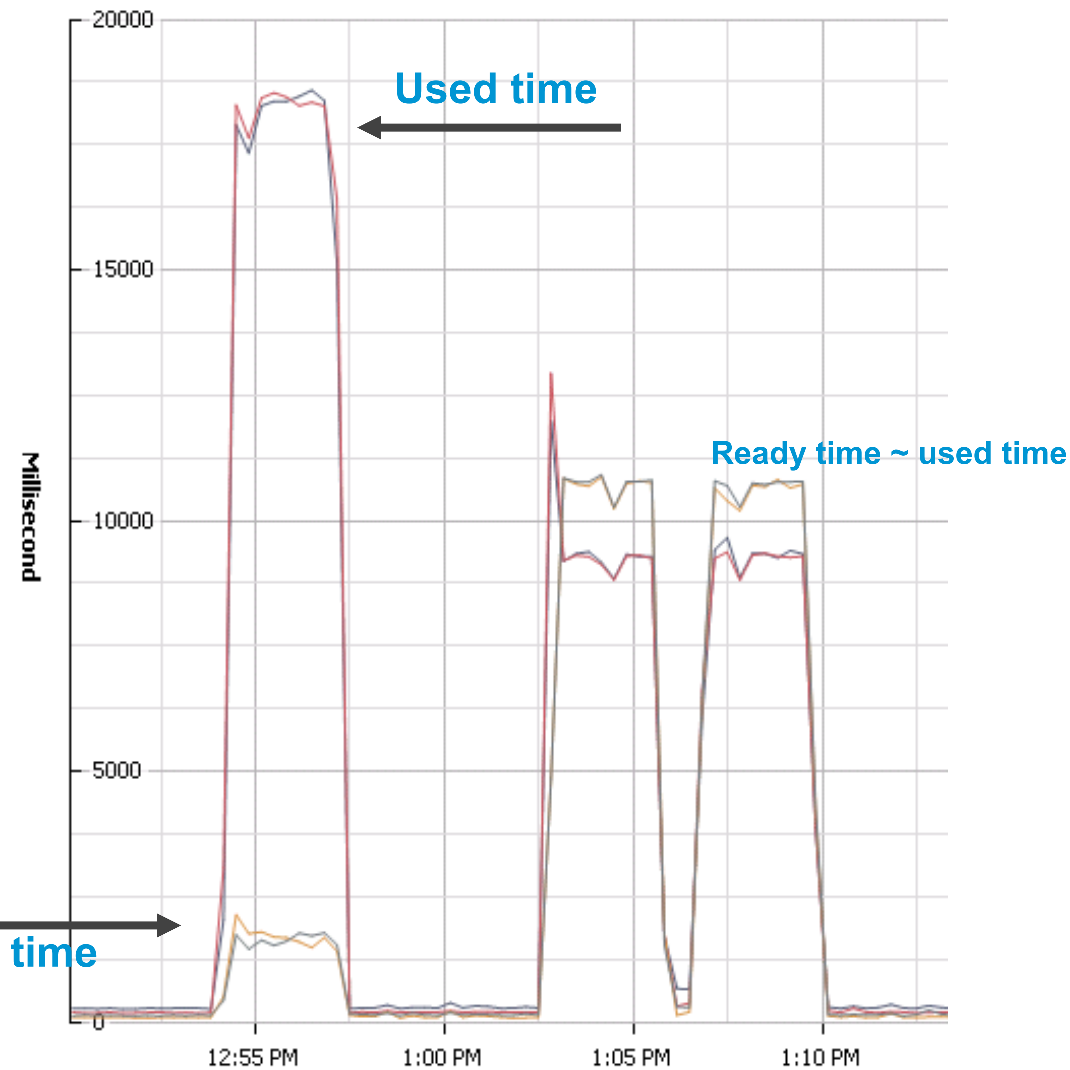
- DRS could help optimize resources
- Change share allocations to de-prioritize less important VMs
- More CPUs may be the solution

CPU capacity

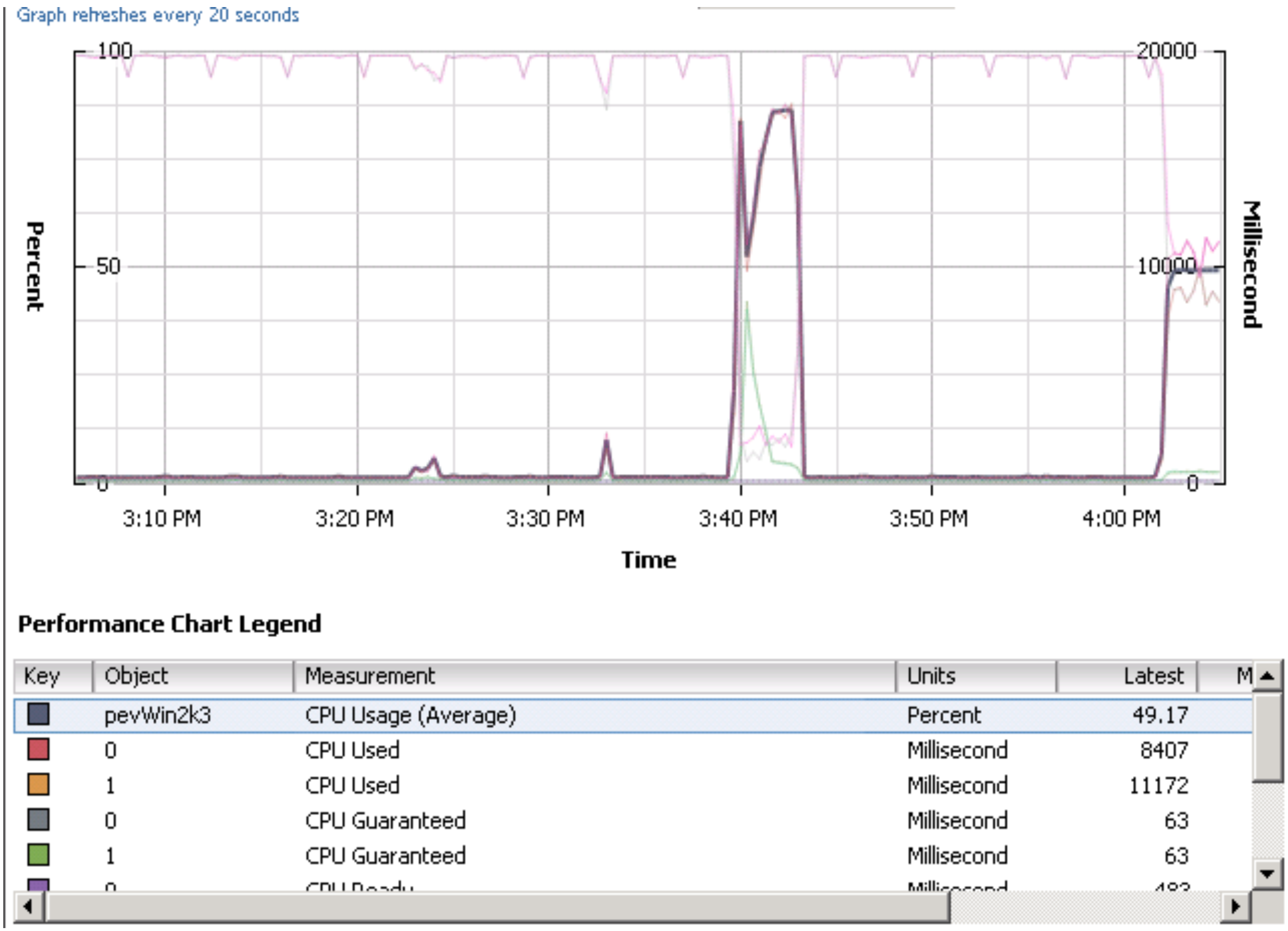
(screenshot from VI Client)

Some caveats on ready time

- Used time ~ ready time: may signal contention. However, might not be overcommitted due to workload variability
- In this example, we have periods of activity and idle periods: CPU isn't overcommitted all the time

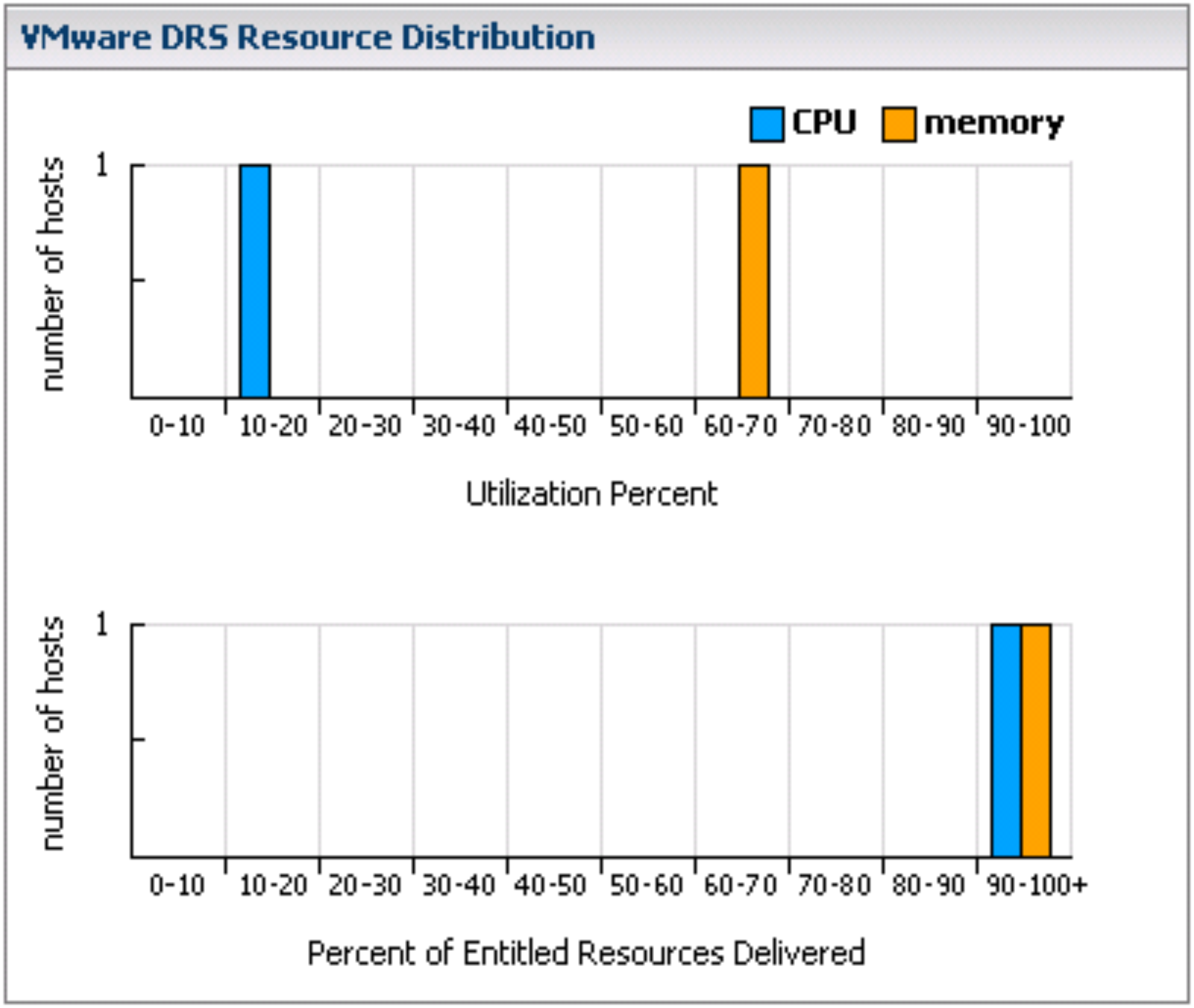


VI Client CPU screenshot



Note CPU milliseconds and percent are on the same chart but use different axes

Cluster-level information in the VI Client



- **Utilization % describes available capacity on hosts (here: CPU usage low, memory usage medium)**

  - % Entitled resources delivered: best if all 90-100+.

CPU performance analysis: esxtop

- **PCPU(%): CPU utilization**
- **Per-group stats breakdown**
  - %USED: Utilization
  - %RDY: Ready Time
  - %TWAIT: Wait and idling time
- **Co-Scheduling stats (multi-CPU Virtual Machines)**
  - %CRUN: Co-run state
  - %CSTOP: Co-stop state
- **Nmem: each member can consume 100% (expand to see breakdown)**
- **Affinity**
- **HTSharing**

esxtop CPU screen (c)

```
10:55:46am up 43 days 23:51, 61 worlds; CPU load average: 0.01, 0.01, 0.01
PCPU(%):     2.54,    1.70,    1.82,    1.16 ;   used total:   1.80
CCPU(%):     0 us,    0 sy,   97 id,    2 wa ;        cs/sec:    77

      ID      GID NAME             NWLD   %USED    %RUN    %SYS   %WAIT    %RDY   %IDLE  %OVR
       1        1 idle                4  395.54  395.97    0.00    0.00    6.71    0.00    0.
       2        2 system              6    0.01    0.01    0.00  600.00    0.00    0.00    0.
       6        6 helper             22    0.01    0.01    0.00 2200.00    0.01    0.00    0.
       7        7 drivers            11    0.01    0.01    0.00 1100.00    0.00    0.00    0.
       9        9 console             1    1.07    1.08    0.00   99.00    0.60   98.98    0.
      14       14 vmkapimod           2    0.00    0.00    0.00  200.00    0.00    0.00    0.
      15       15 vmware-vmkauthd     1    0.00    0.00    0.00  100.00    0.00    0.00    0.
      16       16 Windows 2003 SP     7    4.28    4.28    0.01  699.85    0.54  196.53    0.
      17       17 SQL2005             7    1.41    1.41    0.01  700.00    0.27  199.79    0.
```

PCPU = Physical CPU

CCPU = Console CPU (CPU 0)

Press ‘f’ key to choose fields

```
Current Field order: ABCDEfgh

* A:  ID = Id
* B:  GID = Group Id
* C:  NAME = Name
* D:  NWLD = Num Members
* E:  %STATE TIMES = CPU State Times
  F:  EVENT COUNTS/s = CPU Event Counts
  G:  CPU ALLOC = CPU Allocations
  H:  SUMMARY STATS = CPU Summary Stats

Toggle fields with a-h, any other key to return:
```

New metrics in CPU screen

```
PCPU USED(%):  54  54 0.7 0.1 0.9 0.1 0.1 0.1 0.9 0.3 0.2
PCPU UTIL(%): 100 100 0.7 0.1 0.8 0.1 0.3 0.1 1.1 0.4 0.5
CORE UTIL(%): 100     0.7     0.9     0.3     1.4     0.6

     ID NAME    %LAT_C  %LAT_M %DMD   EMIN TIMER/s
 385754 KC1       32.0     0.0   68   9767  200.00
 385931 KC2       32.0     0.0   68   9767  200.00
```

%LAT_C : %time the VM was not scheduled due to CPU resource issue

%LAT_M : %time the VM was not scheduled due to memory resource issue

%DMD : Moving CPU utilization average in the last one minute

EMIN : Minimum CPU resources in MHZ that the VM is guaranteed to get when there is CPU contention

Troubleshooting CPU related problems

- **CPU constrained**

```
12:43:56pm up 53 days  1:39, 103 worlds; CPU load aver
PCPU(%):  40.87,  26.84,  88.10,  73.86 ;   used total
CCPU(%):   0 us,   0 sy, 100 id,   0 wa ;        cs/sec

      ID      GID NAME              NWLD   %USED    %RUN
      23       23 cpuBurn-CLONE        7  210.72  211.30
      16       16 Windows 2003 SP      7    5.18    5.19
```

SMP VM

High CPU utilization

```
12:45:02pm up 53 days  1:40, 103 worlds; CPU load ave
PCPU(%):  20.28,  31.85,  92.27,  84.18 ;   used tota
CCPU(%):   0 us,   0 sy,  99 id,   1 wa ;        cs/se

      ID      GID NAME              NWLD   %USED    %RUN
    1118       23 vmware-vmx           1    0.06    0.06
    1119       23 vmm0:cpuBurn-CL      1  105.05  105.23
    1120       23 vmm1:cpuBurn-CL      1  105.83  105.98
    1121       23 vmware-vmx           1    0.00    0.00
    1122       23 mks:cpuBurn-CLO      1    0.20    0.20
    1123       23 vcpu-0:cpuBurn-      1    0.01    0.01
    1124       23 vcpu-1:cpuBurn-      1    0.00    0.00
      16       16 Windows 2003 SP      7    5.04    5.08
```

Both the virtual CPUs CPU constrained

Troubleshooting CPU related problems

- **CPU limit**

```
 1:10:48pm up 55 days  2:06, 101 worlds; CPU load average: 0.64, 0.48, 0.34
PCPU(%):  61.62,  59.58,  64.95,  93.86 ;   used total:  70.00
CCPU(%):   0 us,   0 sy,  99 id,   1 wa ;        cs/sec:    111

NAME                 NWLD   %USED     %RUN     %SYS    %WAIT     %RDY   %IDLE  %OVRLP   %CSTP  %MLMTD
cpuBurn-CLONE1          7  206.19   206.78     0.02   529.94     7.86    0.00    0.59    0.11    0.00
cpuBurn-CLONE           7   53.39    53.60     0.05   525.18   162.75    0.00    0.27    3.13  150.80
```

Max Limited

```
 1:12:00pm up 55 days  2:07, 101 worlds; CPU load average: 0.71, 0.48, 0.40
PCPU(%):  54.36,  58.40,  75.21,  63.15 ;   used total:  62.78
CCPU(%):   0 us,   1 sy,  92 id,   7 wa ;        cs/sec:     95

   GID  NAME               NWLD          AMIN        AMAX  ASHRS      AMLMT   AUNITS
    30  cpuBurn-CLONE1        7             0          -1     -3         -1      mhz
    32  cpuBurn-CLONE         7             0        1000     -3         -1      mhz
```

CPU Limit

AMAX = -1 : Unlimited

Troubleshooting CPU related problems

- **CPU contention**

```
12:40:10pm up 53 days  1:35, 103 worlds; CPU load average: 0.88, 0.53, 0.40
PCPU(%): 100.00, 100.00, 100.00, 100.00 ;   used total: 100.00
CCPU(%):    1 us,    1 sy,   98 id,    1 wa ;        cs/sec:     637

       ID      GID NAME                NWLD    %USED     %RUN     %SYS    %WAIT     %RDY
       23       23 cpuBurn-CLONE          7   148.67   149.02     0.00   532.96    62.42
       25       25 cpuBurn-CLONE2         7   128.25   128.58     0.01   542.29    73.46
       24       24 cpuBurn-CLONE1         7   127.69   128.15     0.01   541.34    74.94
```

4 CPUs, all at 100%

3 SMP VMs

VMs don't get to run all the time

%ready accumulates

Further ready time examination

```
 2:01:53pm up 4 days 29 min, 87 worlds; CPU load average: 0.16, 0.16, 0.09
PCPU(%):  13.20,  15.55,  10.71,  23.06 ;   used total:  15.63
CCPU(%):    0 us,    0 sy,  99 id,    0 wa ;        cs/sec:      98
```

ID	GID	NAME	NWLD	%USED	%RUN	%SYS	%WAIT	%RDY	%MLMTD
1	1	idle	4	337.41	338.34	0.00	0.00	61.66	0.00
2	2	system	6	0.02	0.02	0.00	599.97	0.00	0.00
6	6	helper	22	0.02	0.02	0.00	2199.96	0.11	0.00
7	7	drivers	11	0.01	0.01	0.00	1099.98	0.00	0.00
9	9	console	1	0.92	0.93	0.00	98.27	0.80	0.00
14	14	vmkapimod	2	0.00	0.00	0.00	200.00	0.00	0.00
15	15	vmware-vmkauthd	1	0.00	0.00	0.00	100.00	0.00	0.00
16	16	cpuBurn-CLONE	7	50.11	50.03	0.00	495.74	154.20	152.44
17	17	fakeDB	7	1.43	1.44	0.00	697.67	0.89	0.00
18	18	Windows 2003 SP	7	5.10	5.12	0.00	693.01	1.87	0.00
19	19	SQL2005	7	1.63	1.59	0.03	697.16	1.24	0.00
20	20	memhog-linux-CL	5	1.17	1.16	0.02	498.29	0.55	0.00
21	21	cpuBurn-CLONE2	7	1.31	1.29	0.03	698.01	0.70	0.00

High Ready Time

High MLMTD: there is a limit on this VM…

→High ready time not always because of overcommitment
→When you see high ready time, double-check if limit is set

Troubleshooting CPU related problems

- **SMP VM running UP HAL/Kernel**

```
 4:38:19am up 53 days 17:33, 110 worlds; CPU load average: 0.15, 0.12, 0.06
PCPU(%):  17.82,  15.09,  13.71,  25.65 ;   used total:  18.07
CCPU(%):   1 us,   2 sy,  91 id,   6 wa ;        cs/sec:    623
```

ID	GID	NAME	NWLD	%USED	%RUN	%SYS	%WAIT	%RDY
1139	26	vmware-vmx	1	0.14	0.14	0.00	95.50	0.10
1140	26	vmm0:win2k	1	52.10	52.16	0.35	42.07	1.52
1141	26	vmm1:win2k	1	0.22	0.22	0.00	94.95	0.58
1142	26	vmware-vmx	1	0.00	0.00	0.00	95.75	0.00
1143	26	mks:win2k	1	0.36	0.33	0.02	94.87	0.55
1144	26	vcpu-0:win2k	1	0.11	0.11	0.00	95.62	0.02
1145	26	vcpu-1:win[illegible]	1	0.00	0.00	0.00	95.73	0.01

vCPU 1 not used by the VM

It is also possible that you are running a single threaded application in a SMP VM

vmware

Troubleshooting CPU related problems

- **High CPU activity in the Service Console**

```
 5:37:37am up 52 days 18:33, 94 worlds; CPU l
PCPU(%):   94.66,    7.66,    5.98,    5.47 ;    u
CCPU(%):   99 us,    1 sy,    0 id,    0 wa ;

      ID      GID NAME                NWLD    %USED
       1        1 idle                   4   268.52
       2        2 system                 6     0.01
       6        6 helper                22     0.41
       7        7 drivers               11     0.01
       9        9 console                1    95.61
```

Some process in the service console is hogging CPU

```
 5:27:02am up 52 days 18:22, 94 worlds; CPU l
PCPU(%):   73.69,   23.68,   29.23,   28.43 ;    u
CCPU(%):    1 us,    5 sy,   84 id,   10 wa ;

      ID      GID NAME                NWLD    %USED
       1        1 idle                   4   270.45
       2        2 system                 6     0.13
       6        6 helper                22     0.06
       7        7 drivers               11     0.01
       9        9 console                1    30.49
```

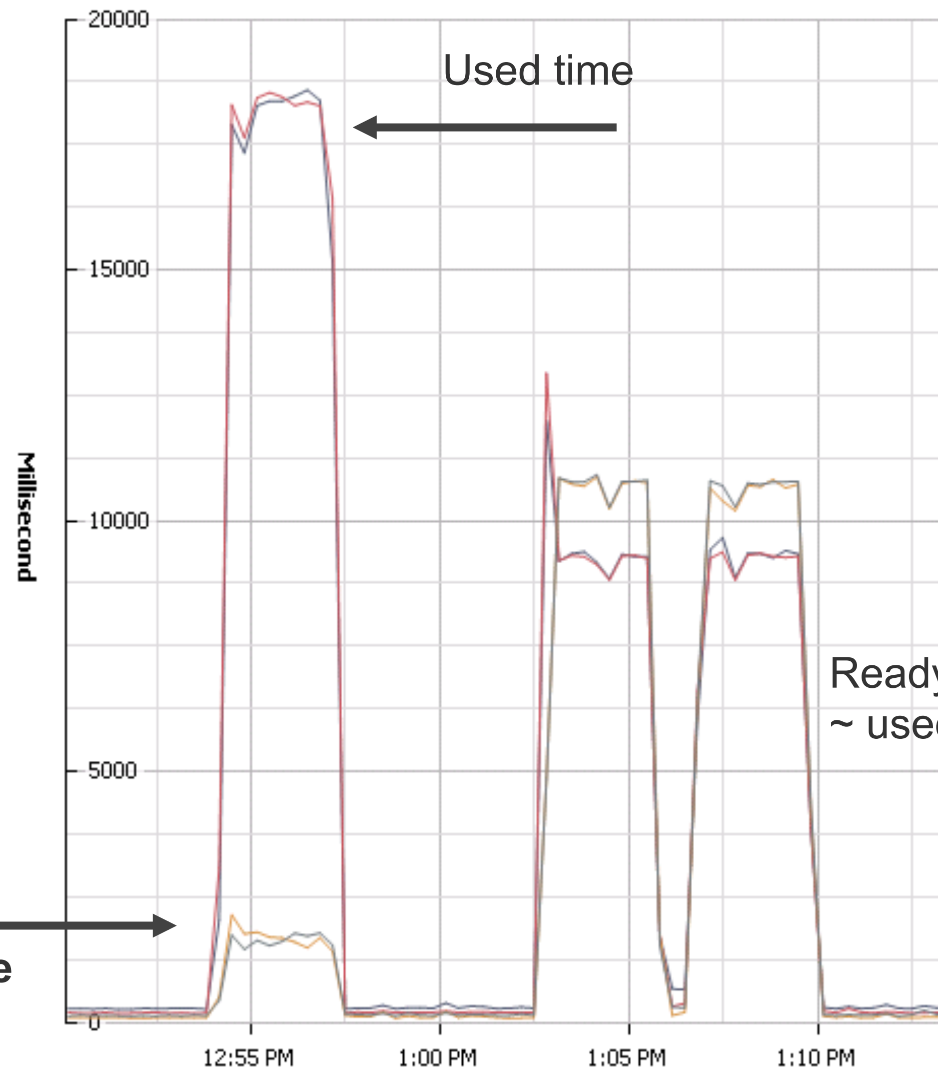
Not much activity in the service console

VMKernel is doing some activity on behalf of the console OS - cloning in this case

vmware

VI Client and Ready Time

- Used time ~ ready time: may signal contention. However, might not be overcommitted due to workload variability
- In this example, we have periods of activity and idle periods: CPU isn't overcommitted all the time



Ready time ~ used time

Ready time < used time

CPU Performance

- **vSphere supports eight virtual processors per VM**
  - Use UP VMs for single-threaded applications
    - Use UP HAL or UP kernel
  - For SMP VMs, configure only as many VCPUs as needed
  - Unused VCPUs in SMP VMs:
    - Impose unnecessary scheduling constraints on ESX Server
    - Waste system resources (idle looping, process migrations, etc.)

CPU Performance

- **For threads/processes that migrate often between VCPUs**
  - Pin the guest thread/process to a particular VCPU
  - Pinning guest VCPUs to PCPUs rarely needed
- **Guest OS timer interrupt rate**
  - Most Windows, Linux 2.4: 100 Hz
  - Most Linux 2.6: 1000 Hz
  - Recent Linux: 250 Hz
  - Upgrade to newer distro, or rebuild kernel with lower rate

Performance Tips

- **Idling VMs**
  - Consider overhead of delivering guest timer interrupts
  - Lowering guest periodic timer interrupt rate should help
- **VM CPU Affinity**
  - Constrains the scheduler: can cause imbalances
  - Reservations may not be met – use on your own risk
- **Multi-core processors with shared caches**
  - Performance characteristics heavily depend on the workload
  - Constructive/destructive cache interference

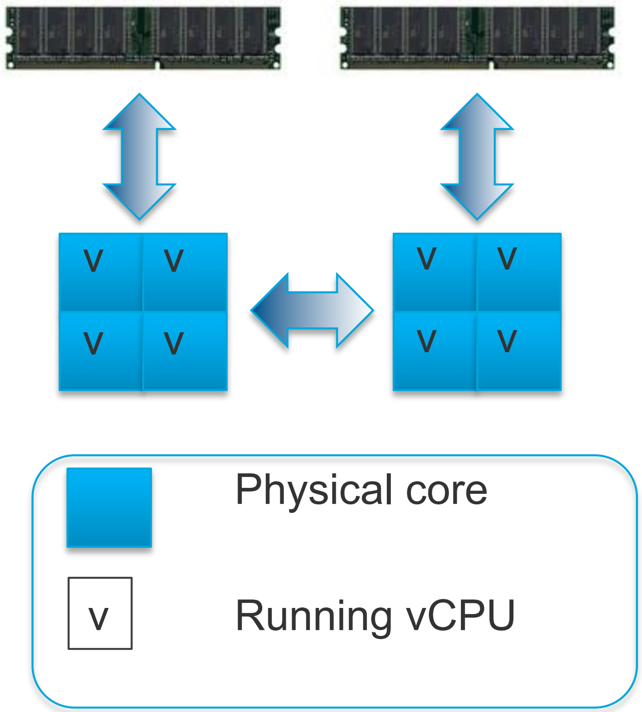
Performance Tips

- **SMP VMs**
  - Use as few virtual CPUs as possible
  - Consider timer interrupt overhead of idling CPUs
  - Co-scheduling overhead increases with more VCPUs
  - Use SMP kernels in SMP VMs
  - Pinning guest threads to VCPUs may help to reduce migrations for some workloads
- **Interactive Workloads (VDI, etc)**
  - Assign more shares, increase reservations to achieve faster response times

vSphere Scheduler and HT

- **Intel Hyper-threading provides the appearance of two logical cores for each physical core**
  - They are somewhat faster than one core but not as fast as two
- **Threads sharing cores less CPU than threads with their own cores**
- **Threads accessing common memory will benefit from running on the same socket**
- **So, 5+ vCPU VMs must choose between more CPU and faster memory**

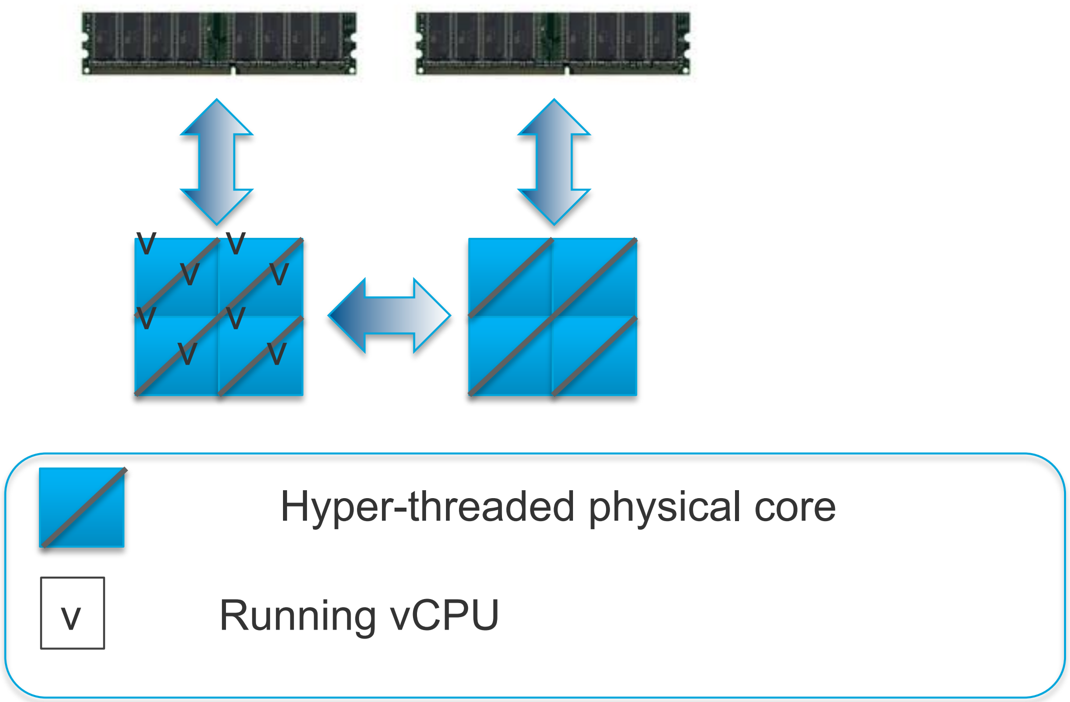
The default: more CPU



Optimizing the Scheduler for Large VMs

- On some virtual machines, memory latency is more important than CPU
- If VM has more vCPUs than there are cores in a single socket, it will run faster if forced to a single socket
- Done with Advanced Settings: NUMA.preferHT

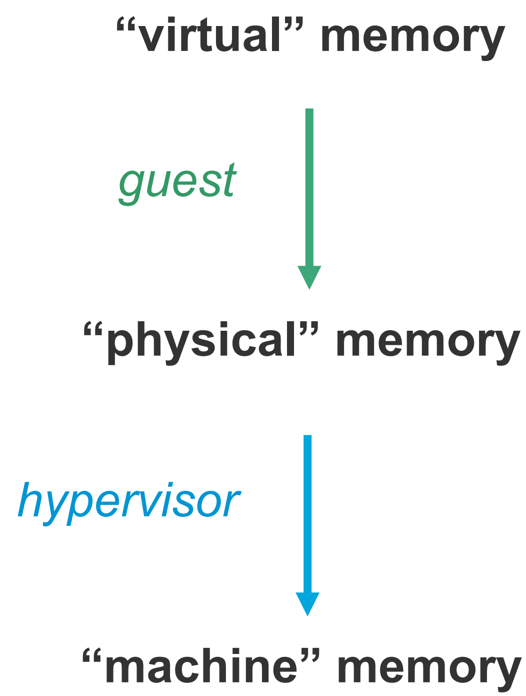
preferHT



MEMORY

Virtual Memory

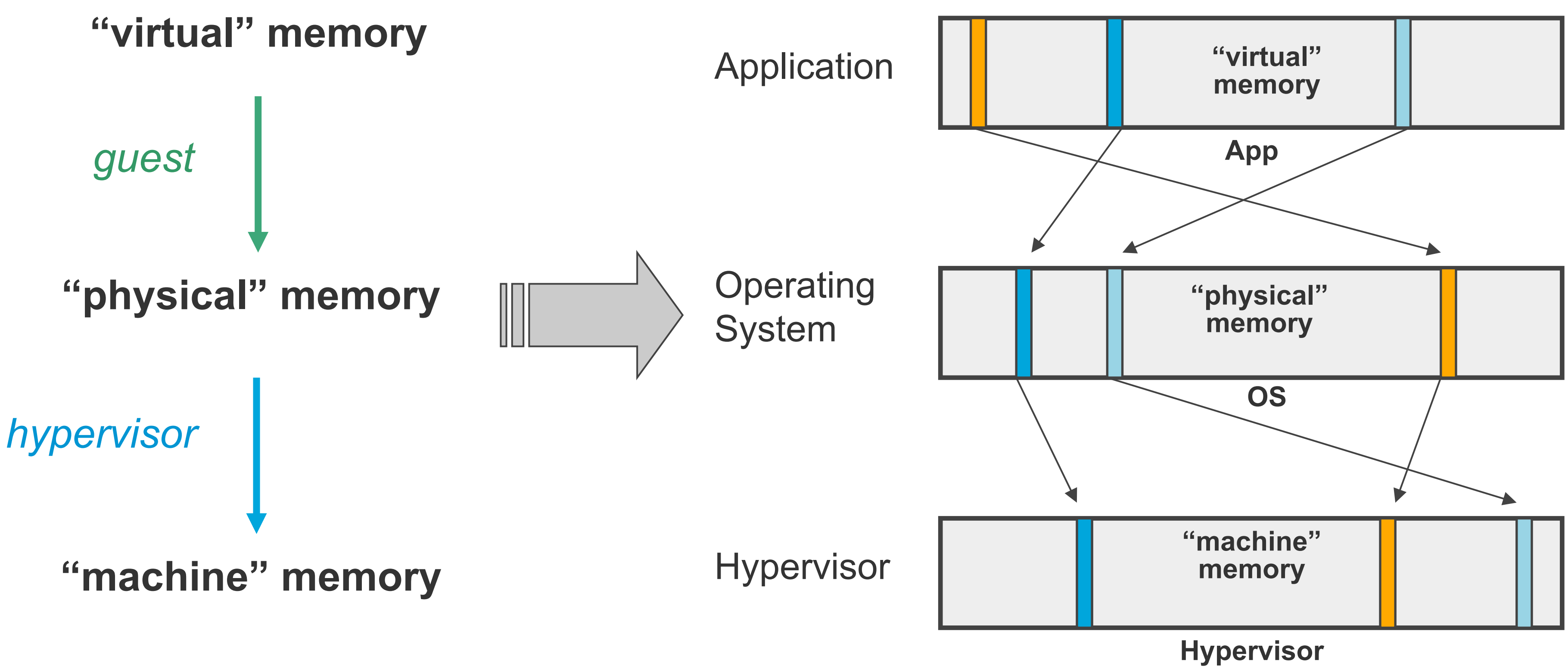
- **Creates uniform memory address space**
  - Operating system maps application virtual addresses to physical addresses
  - Gives operating system memory management abilities transparent to application



Hypervisor adds extra level of indirection

- Maps guest’s physical addresses to machine addresses
- Gives hypervisor memory management abilities transparent to guest

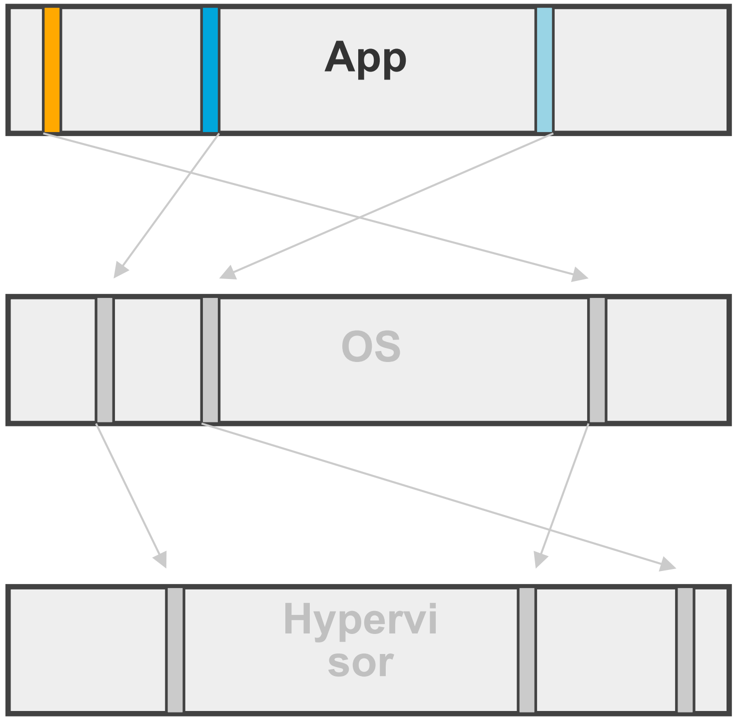
Virtual Memory



vmware

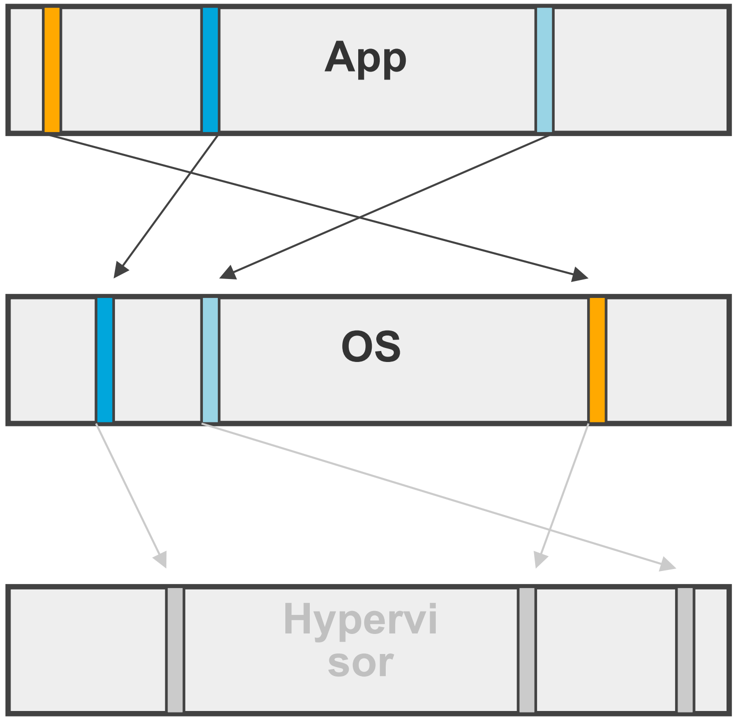
Application Memory Management

- Starts with no memory
- Allocates memory through syscall to operating system
- Often frees memory voluntarily through syscall
- Explicit memory allocation interface with operating system



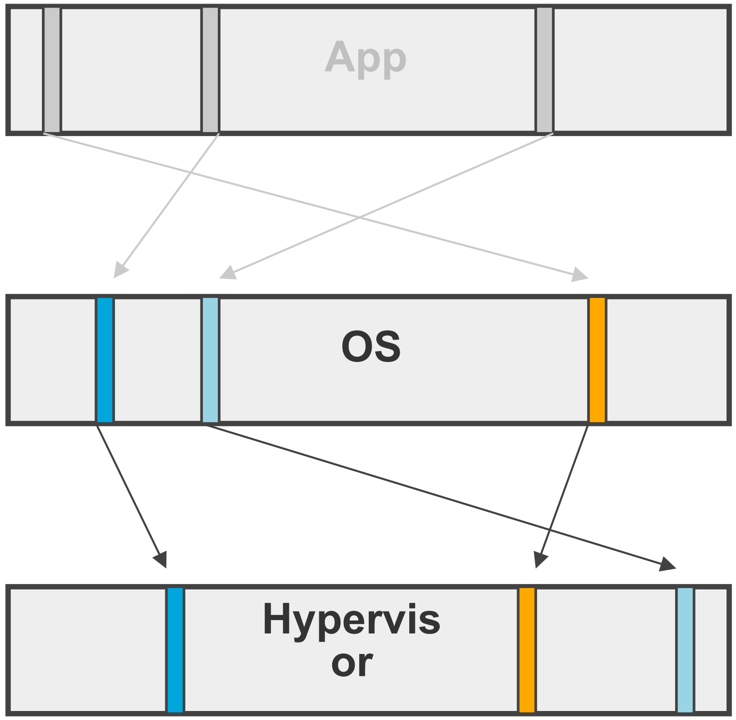
Operating System Memory Management

- Assumes it owns all physical memory
- No memory allocation interface with hardware
  - Does not explicitly allocate or free physical memory
- Defines semantics of “allocated” and “free” memory
  - Maintains “free” list and “allocated” lists of physical memory
  - Memory is “free” or “allocated” depending on which list it resides



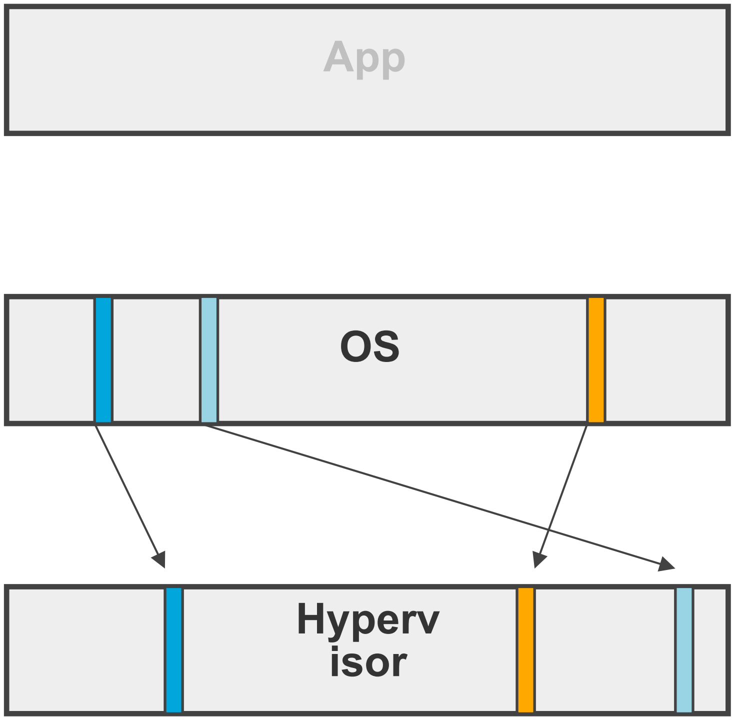
Hypervisor Memory Management

- Very similar to operating system memory management
  - Assumes it owns all machine memory
  - No memory allocation interface with hardware
  - Maintains lists of “free” and “allocated” memory



VM Memory Allocation

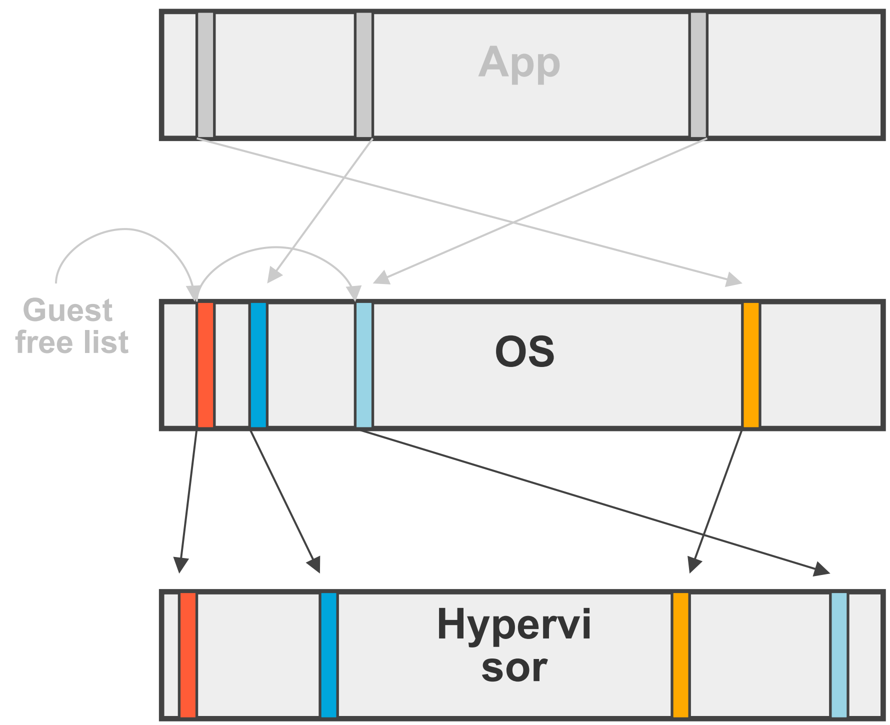
- VM starts with no physical memory allocated to it
- Physical memory allocated on demand
  - Guest OS will not explicitly allocate
  - Allocate on first VM access to memory (read or write)



vmware

VM Memory Reclamation

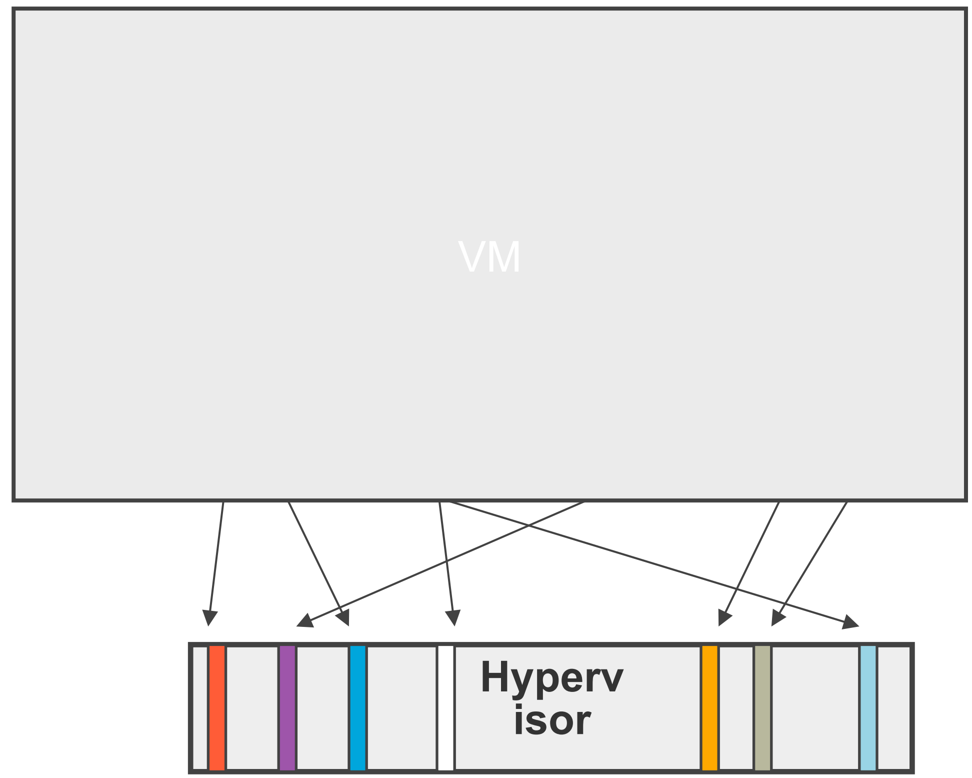
- Guest physical memory not “freed” in typical sense
  - Guest OS moves memory to its “free” list
  - Data in “freed” memory may not have been modified



- **Hypervisor isn’t aware when guest frees memory**
  - Freed memory state unchanged
  - No access to guest’s “free” list
  - Unsure when to reclaim “freed” guest memory

VM Memory Reclamation Cont'd

- **Guest OS (inside the VM)**
  - Allocates and frees…
  - And allocates and frees…
  - And allocates and frees…



- **VM**
  - Allocates…
  - And allocates…
  - And allocates…

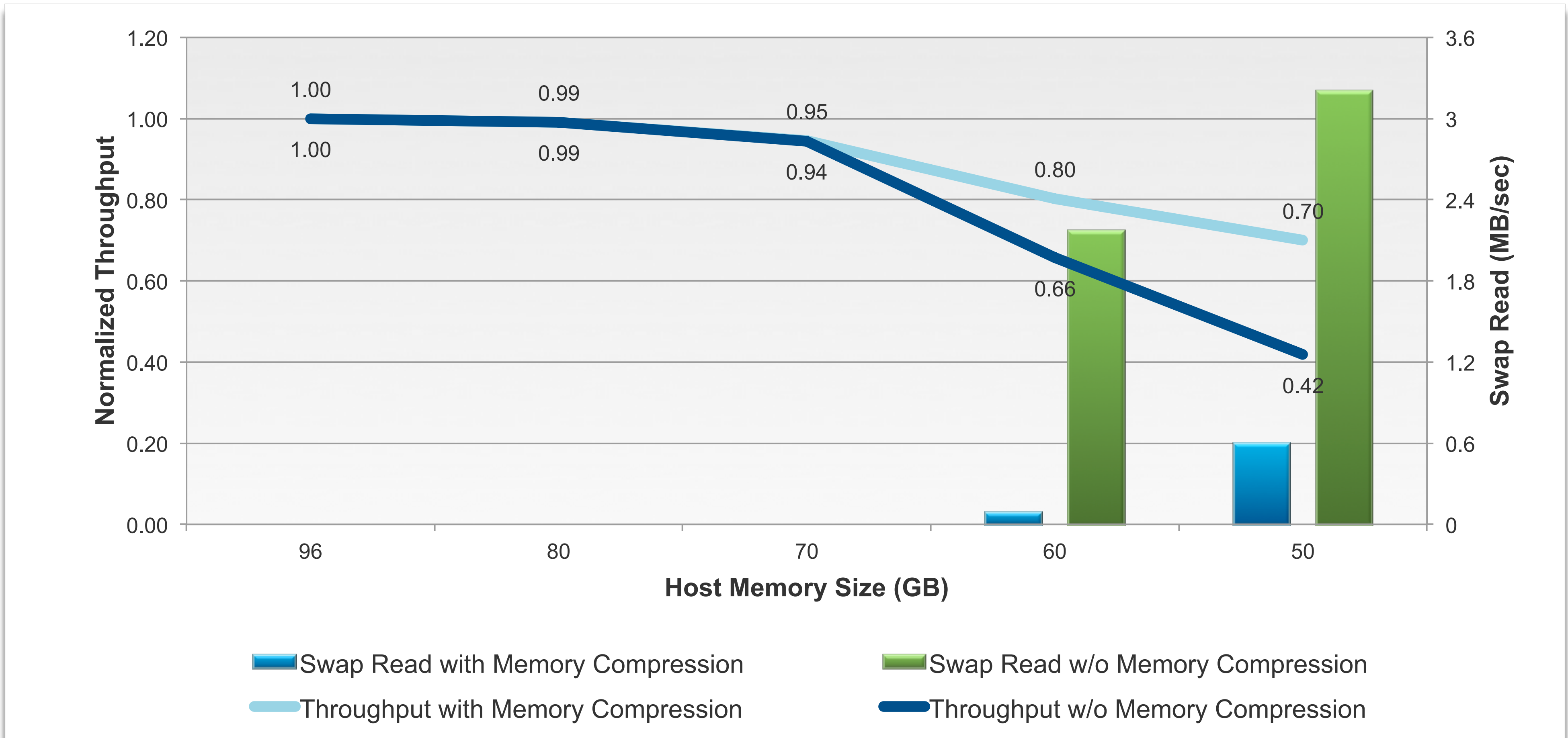
- **Hypervisor needs some way of reclaiming memory!**

Memory Resource Management

- **ESX must balance memory usage**
  - Page sharing to reduce memory footprint of Virtual Machines
  - Ballooning to relieve memory pressure in a graceful way
  - Host swapping to relieve memory pressure when ballooning insufficient
  - Compression to relieve memory pressure without host-level swapping
- **ESX allows overcommitment of memory**
  - Sum of configured memory sizes of virtual machines can be greater than physical memory if working sets fit
- **Memory also has limits, shares, and reservations**
- **Host swapping can cause performance degradation**

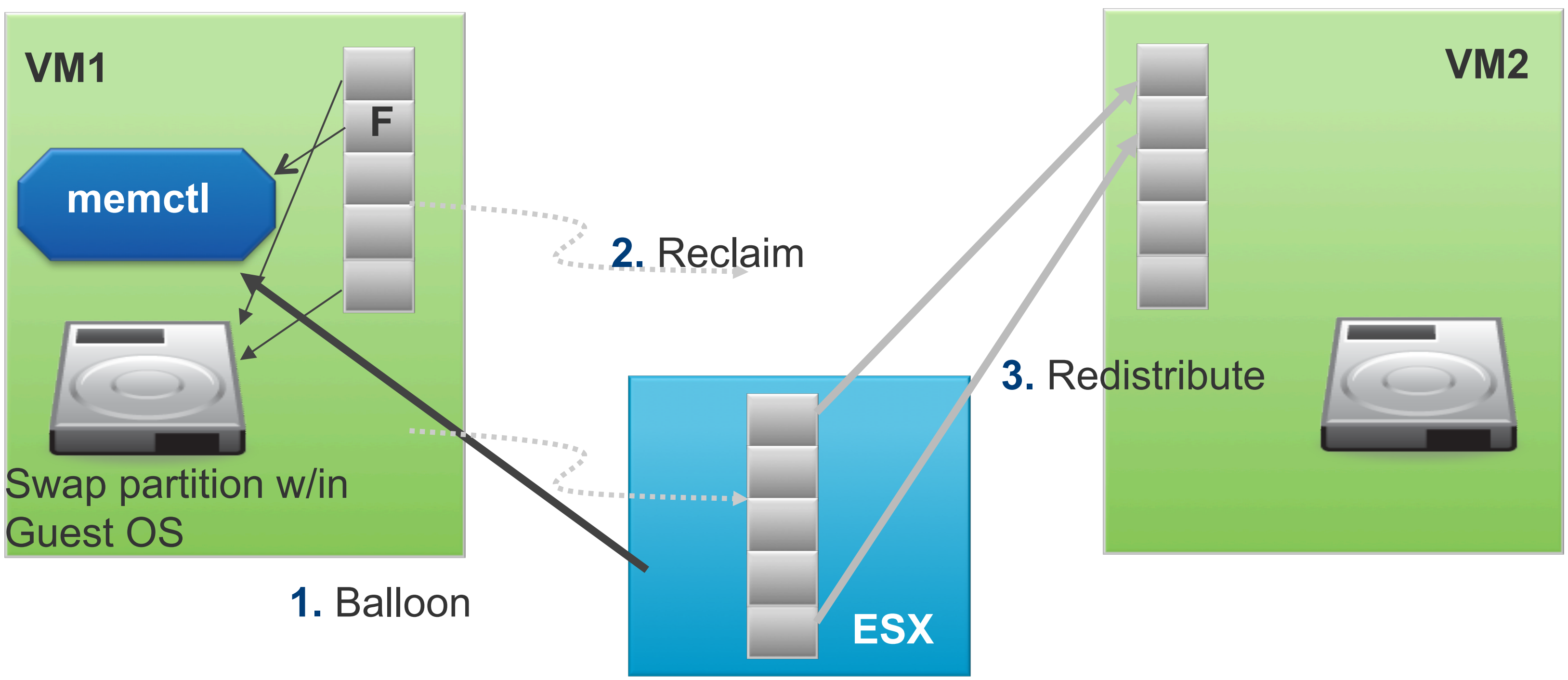
New in vSphere 4.1 – Memory Compression

- Compress memory as a last resort before swapping
- Kicks in after ballooning has failed to maintain free memory
- Reclaims part of the performance lost when ESX is forced to induce swapping



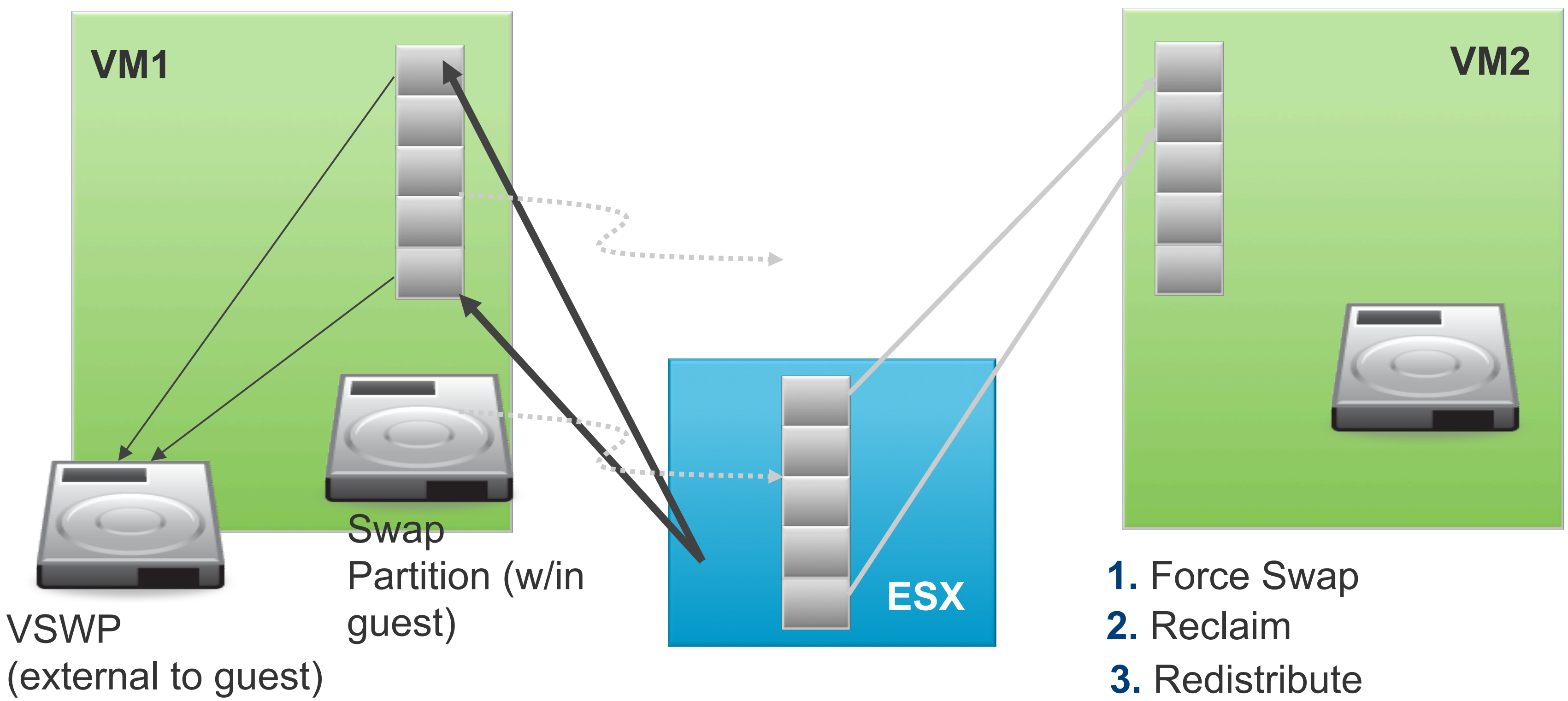
Ballooning, Compression, and Swapping (1)

- **Ballooning: Memctl driver grabs pages and gives to ESX**
  - Guest OS choose pages to give to memctl (avoids “hot” pages if possible): either free pages or pages to swap
    - Unused pages are given directly to memctl
    - Pages to be swapped are first written to swap partition within guest OS and then given to memctl



Ballooning, Swapping, and Compression (2)

- **Swapping: ESX reclaims pages forcibly**
  - Guest doesn't pick pages…ESX may inadvertently pick "hot" pages (→possible VM performance implications)
  - Pages written to VM swap file

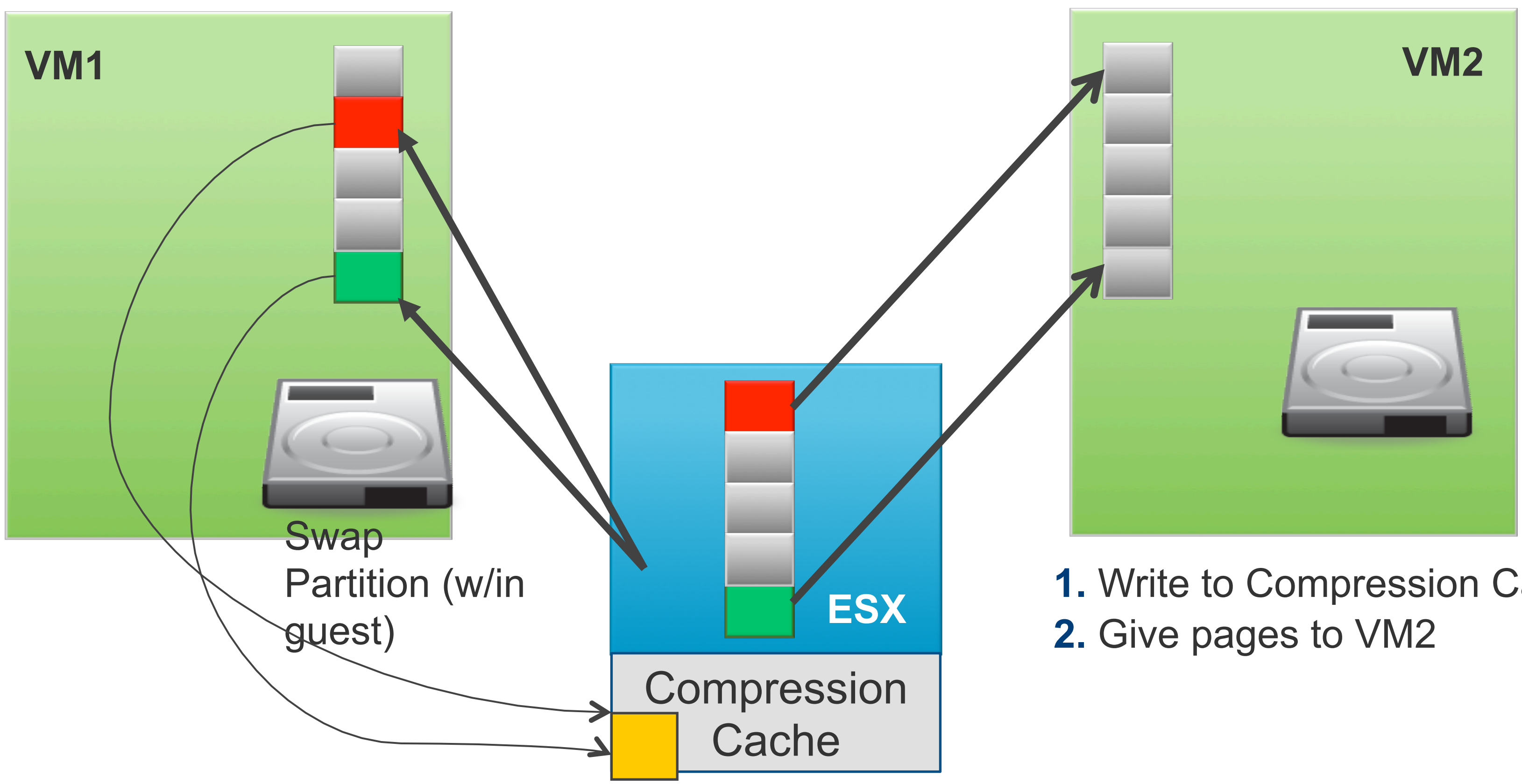


1. Force Swap
2. Reclaim
3. Redistribute

vmware

Ballooning, Swapping and Compression (3)

- **Compression: ESX reclaims pages, writes to in-memory cache**
  - Guest doesn't pick pages…ESX may inadvertently pick "hot" pages (→possible VM performance implications)
  - Pages written in-memory cache → faster than host-level swapping



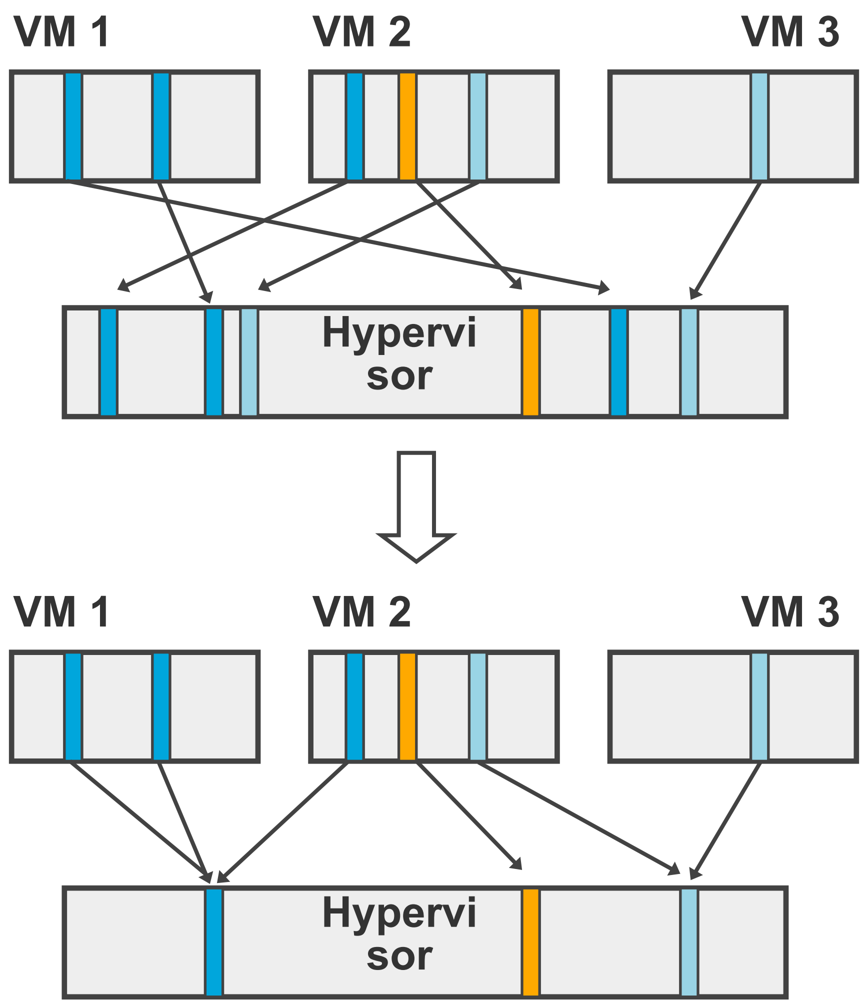
1. Write to Compression Cache
2. Give pages to VM2

Ballooning, Swapping, and Compression (4)

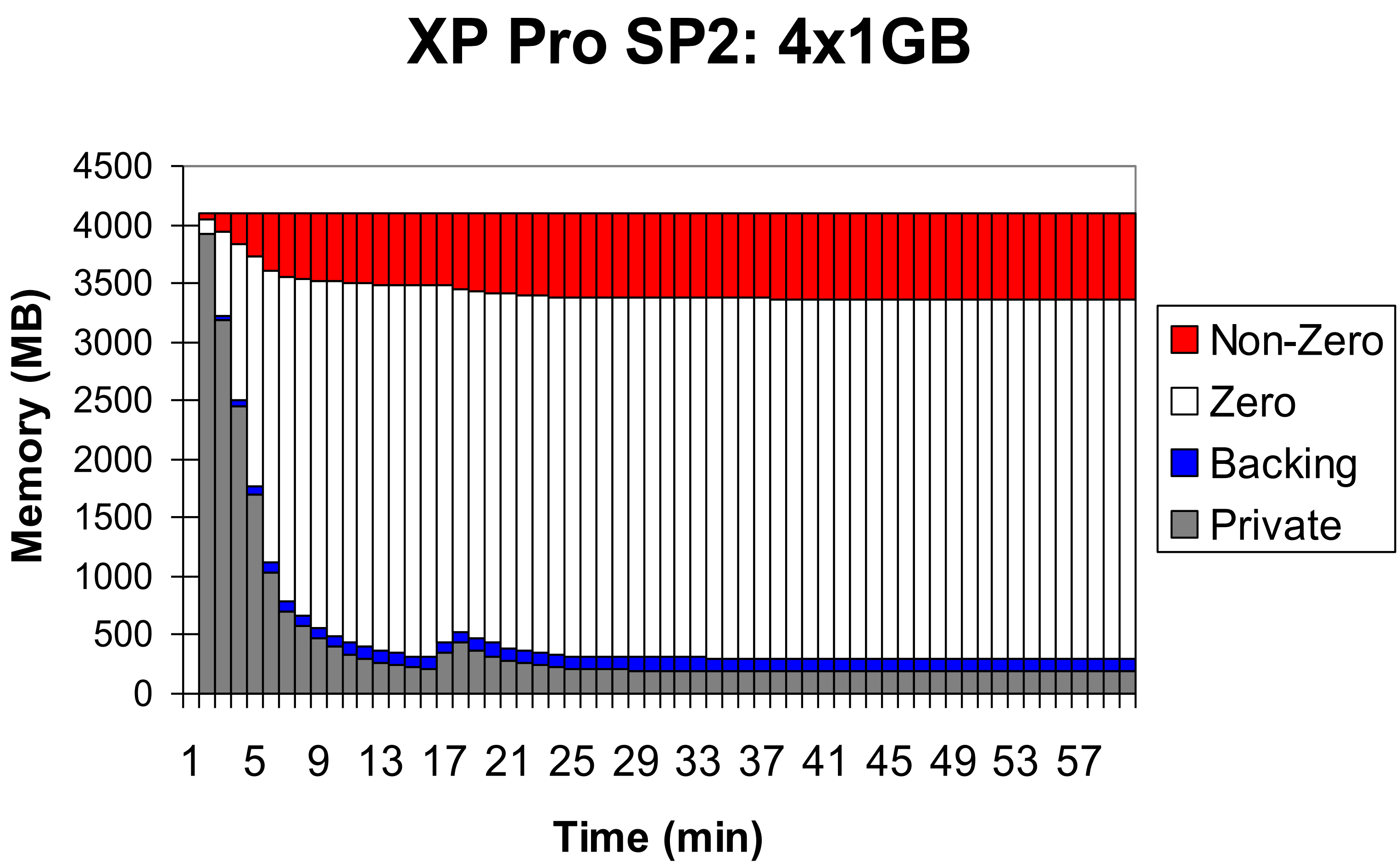
- **Bottom line:**
  - Ballooning may occur even when no memory pressure just to keep memory proportions under control
  - *Ballooning is preferable to compression and vastly preferable to swapping*
    - Guest can surrender unused/free pages
      - With host swapping, ESX cannot tell which pages are unused or free and may accidentally pick “hot” pages
    - Even if balloon driver has to swap to satisfy the balloon request, guest chooses what to swap
      - Can avoid swapping “hot” pages within guest
    - Compression: reading from compression cache is faster than reading from disk

Transparent Page Sharing

- **Simple idea: why maintain many copies of the same thing?**
  - If 4 Windows VMs running, there are 4 copies of Windows code
  - Only one copy needed
- **Share memory between VMs when possible**
  - Background hypervisor thread identifies identical sets of memory
  - Points all VMs at one set of memory, frees the others
  - VMs unaware of change

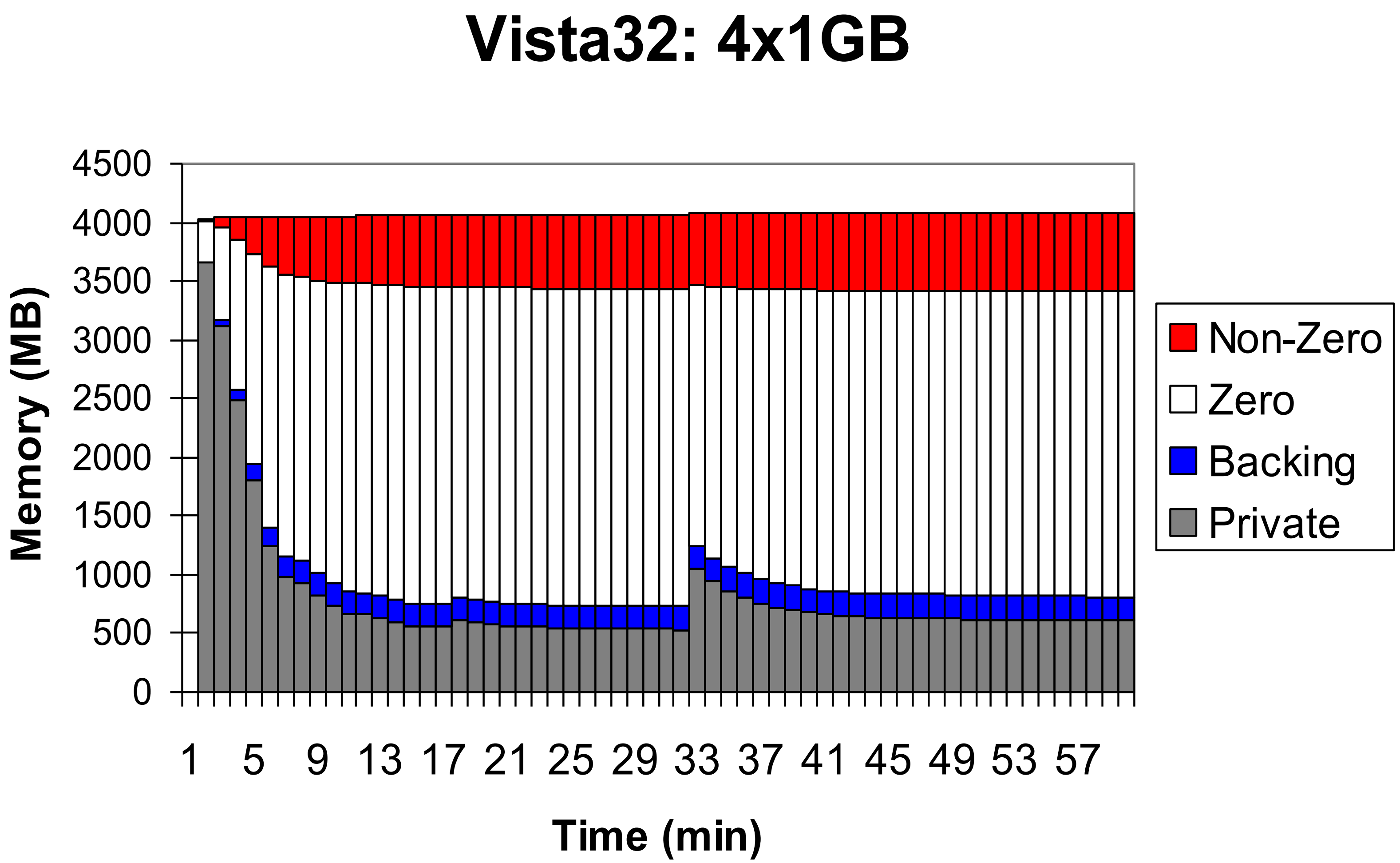


Page Sharing in XP



Memory footprint of four idle VMs quickly decreased to 300MB due to aggressive page sharing.

Page Sharing in Vista



Memory footprint of four idle VMs quickly decreased to 800MB.
(Vista has larger memory footprint.)

Memory capacity

How do we identify host memory contention?

- Host-level swapping (e.g., robbing VM A to satify VM B).
- Active memory for all VMs > physical memory on host
  This could mean possible memory over-commitment

What do I do?

- Check *swapin* (cumulative), *swapout* (cumulative) and *swapused* (“instantaneous”) for the host. Ballooning (vmmemctl) is also useful.
- If *swapin* and *swapout* are increasing, it means that there is possible memory over-commitment
- Another possibility: sum up active memory for each VM. See if it exceeds host physical memory.

Memory Terminology

memory size
total amount of memory presented to guest

allocated memory
memory assigned to applications

unallocated memory
memory not assigned

active memory
allocated memory recently accessed or used by applications

inactive memory
allocated memory not recently accessed or used

Host memory usage measures this, sorta… (allocated memory)

Guest memory usage measures this (active memory)

Differences Between Memory Statistics

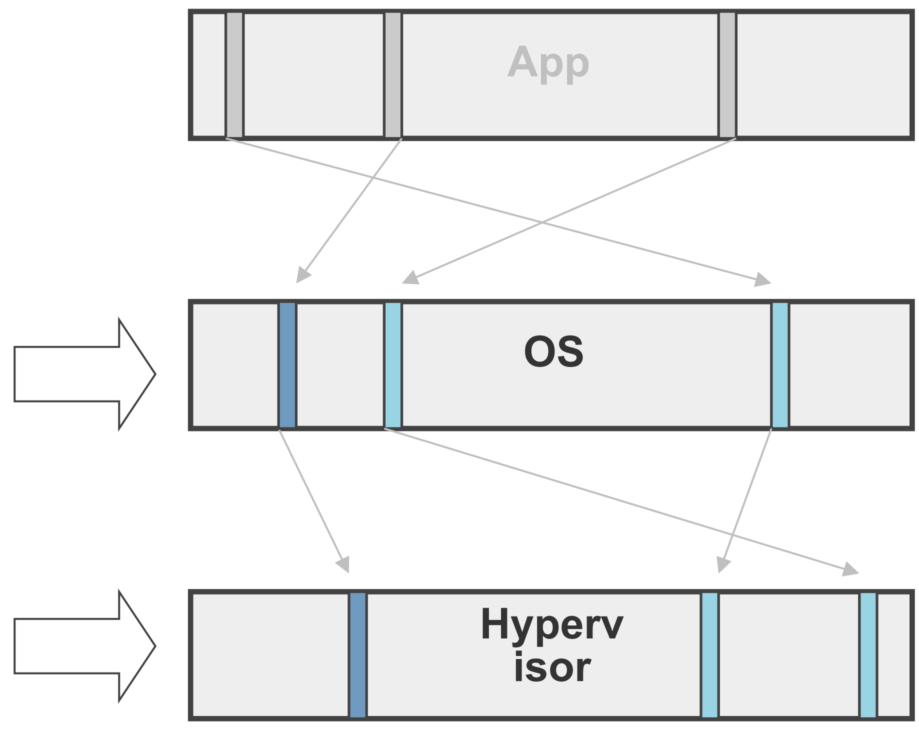
- **Biggest difference is physical memory vs. machine memory**
  - Accounting very different between the two layers!

Physical memory statistics

> Active, Balloon, Granted, Shared, Swapped, Usage

Machine memory statistics

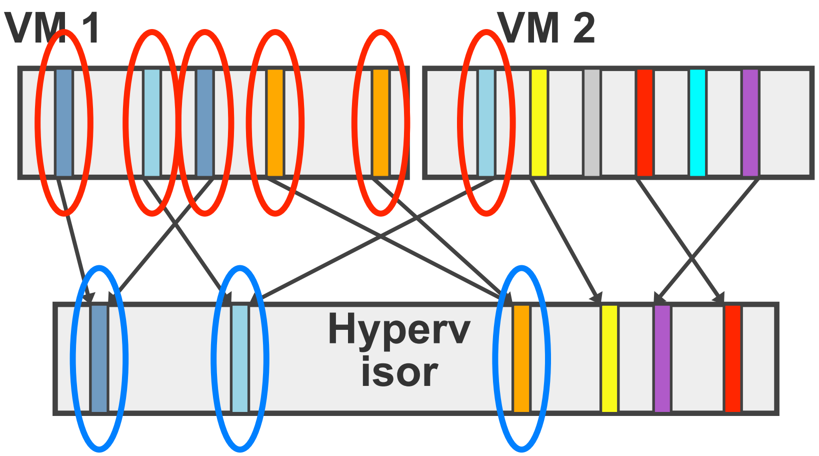
> Consumed, Overhead, Shared Common



Memory Shared vs. Shared Common

▪ Memory Shared

- Amount of physical memory whose mapped machine memory has multiple pieces of physical memory mapped to it
- 6 pieces of memory (VM 1 & 2)



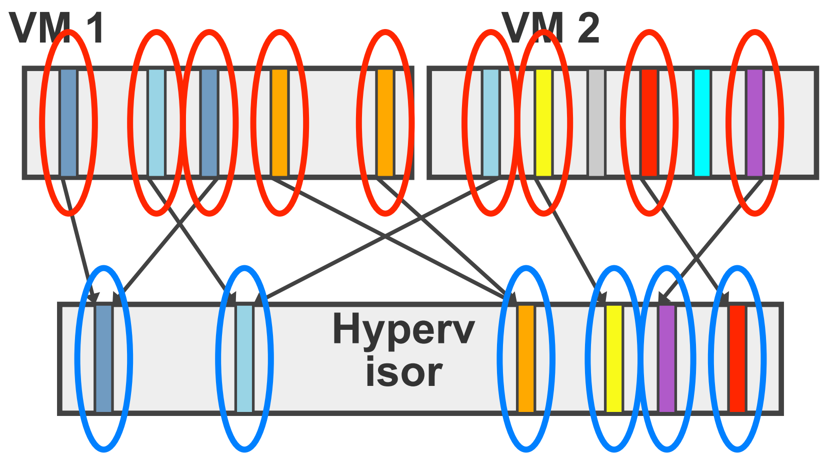
Memory Shared Common

> Amount of machine memory with multiple pieces of physical memory mapped to it

> 3 pieces of memory

Memory Granted vs. Consumed

- **Memory Granted**
  - Amount of physical memory mapped to machine memory
  - 9 pieces of memory (VM 1 & 2)



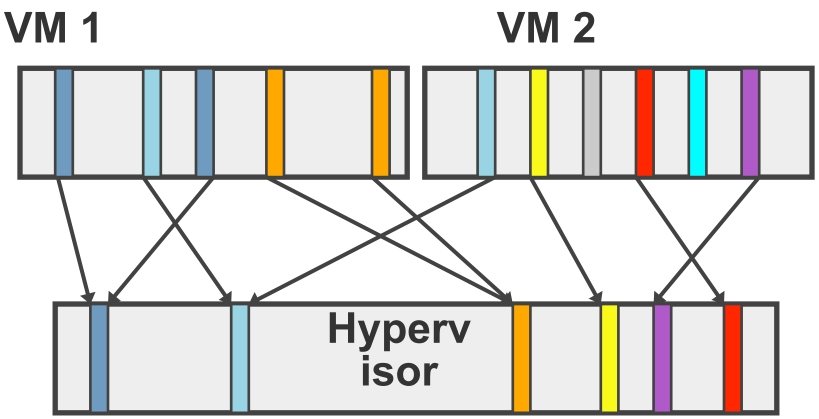
Memory Consumed

- Amount of machine memory that has physical memory mapped to it
- 6 pieces of memory

Difference due to page sharing!

Memory Active vs. Host Memory

- **Memory Active/Consumed/Shared**
  - All measure **physical** memory



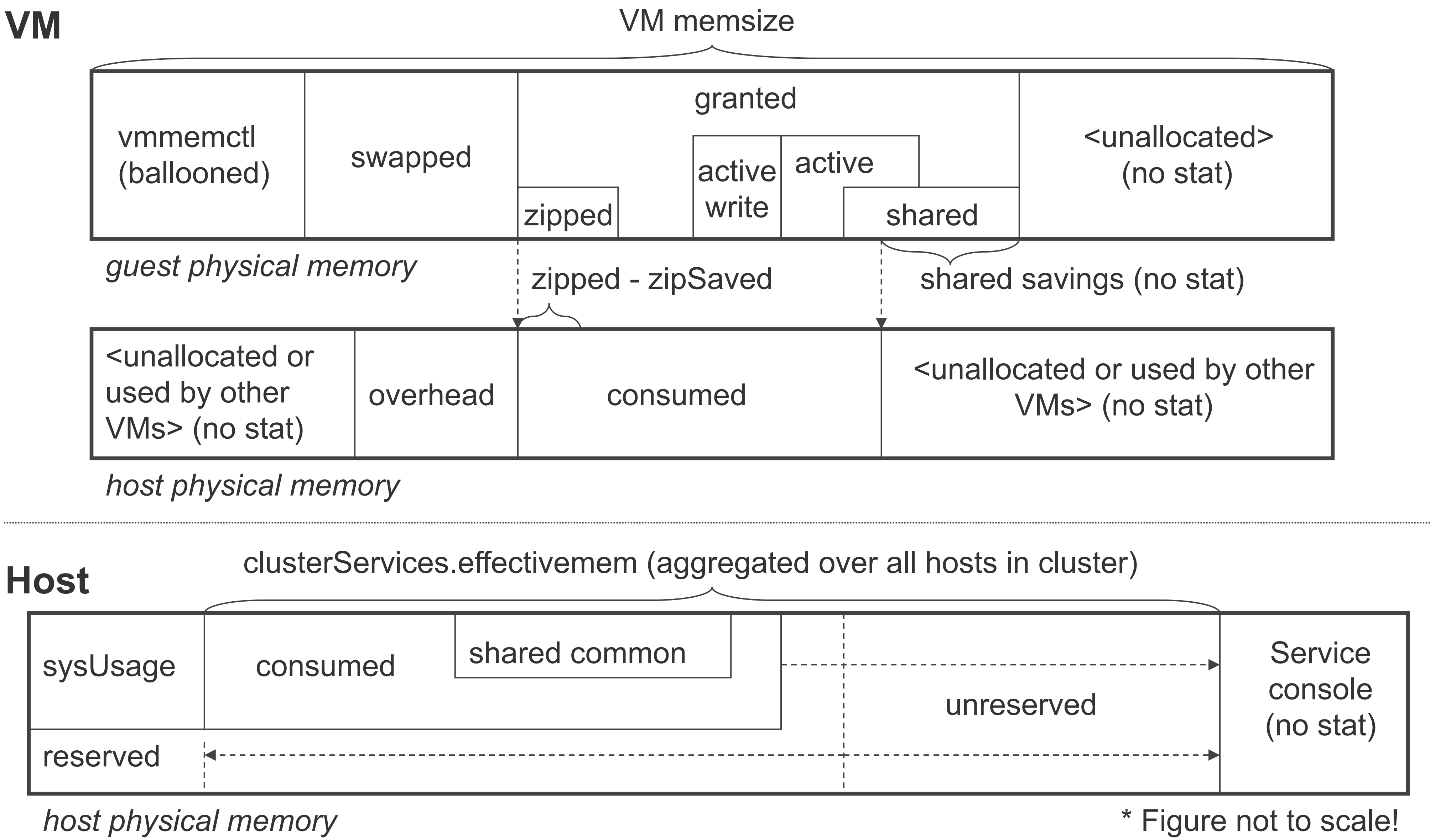
Host Memory

> Total **machine** memory on host

Be careful to not mismatch physical and machine statistics!

> Guest physical memory can/will be greater than machine memory due to memory overcommitment and page sharing

Memory Metric Diagram *

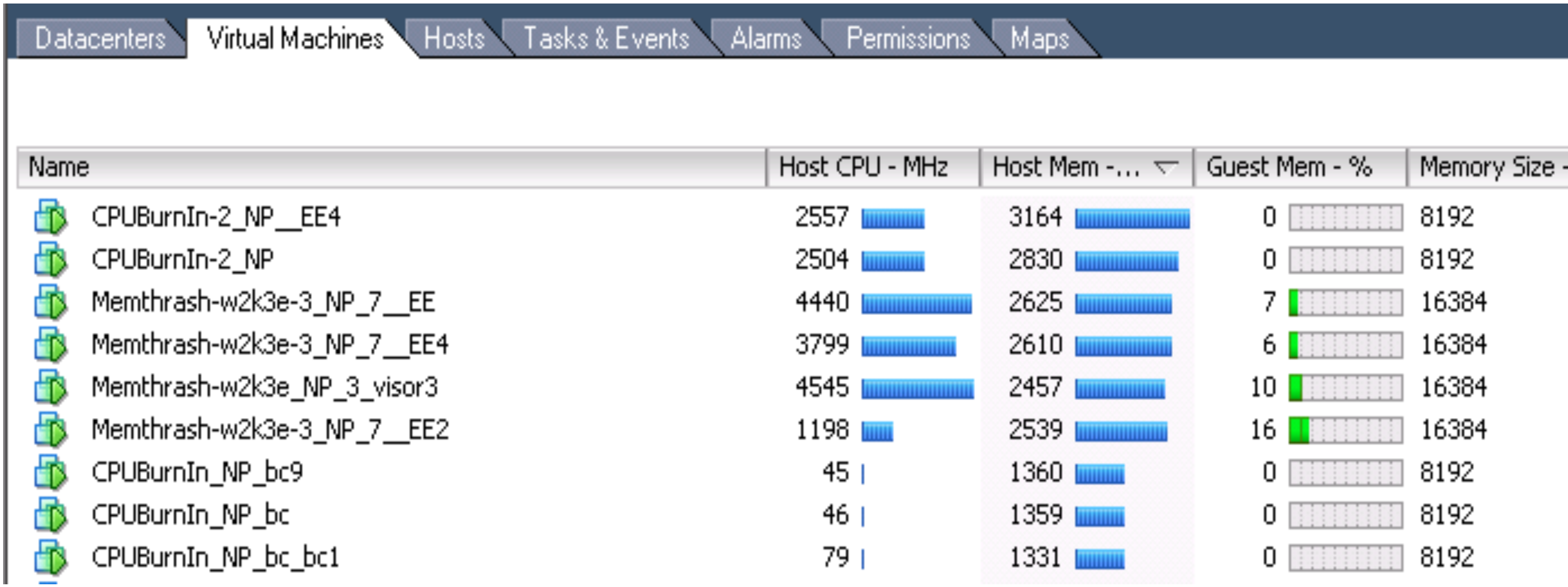


* Figure not to scale!

Using Host and Guest Memory Usage

- **Useful for quickly analyzing VM's status**
  - Coarse-grained information
  - Important for prompting further investigation
- **Requires understanding of memory management concepts**
  - Many aspects of host/guest memory interaction not obvious

VI Client: VM list summary



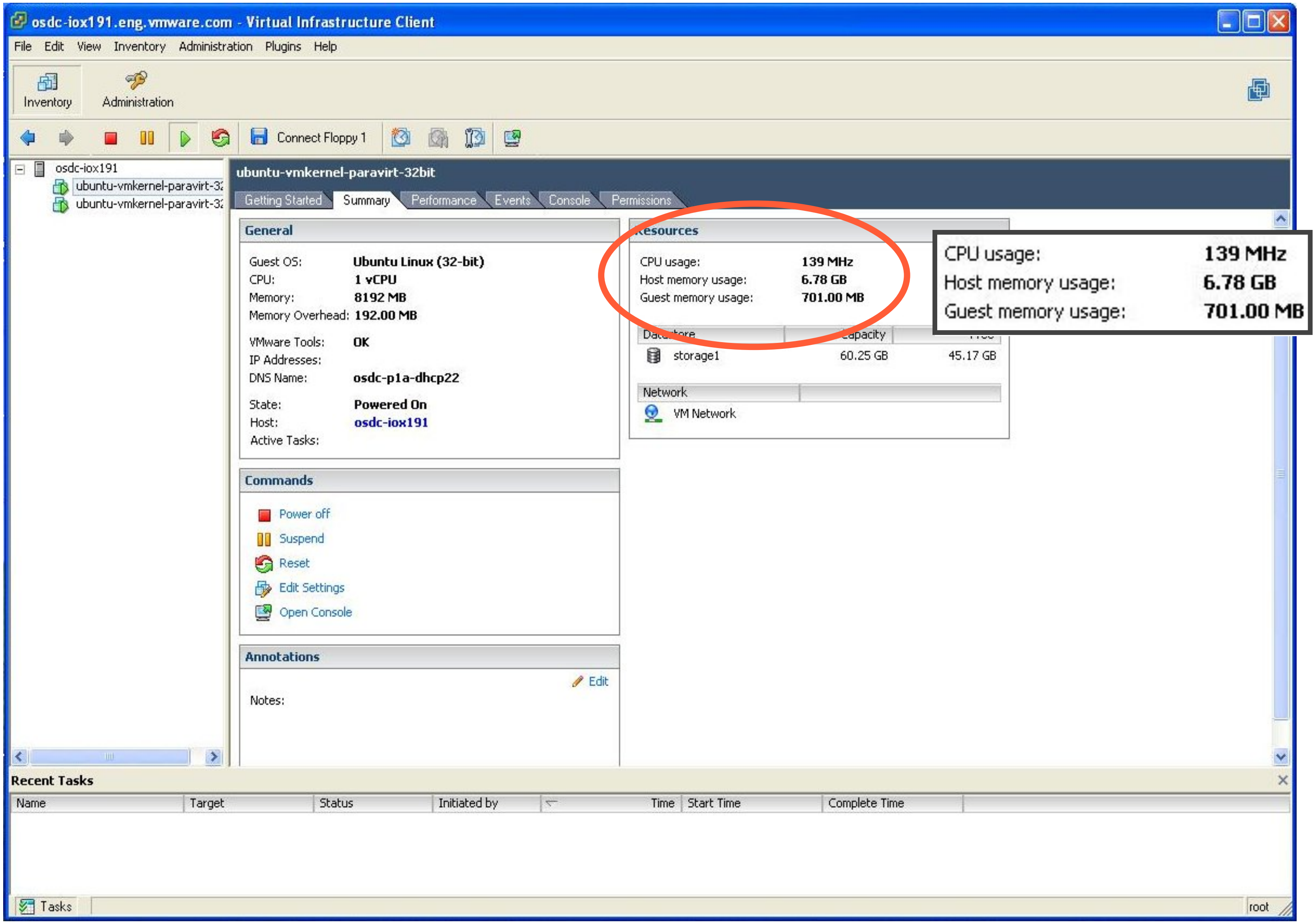
Name	Host CPU - MHz	Host Mem -…	Guest Mem - %	Memory Size -
CPUBurnIn-2_NP__EE4	2557	3164	0	8192
CPUBurnIn-2_NP	2504	2830	0	8192
Memthrash-w2k3e-3_NP_7__EE	4440	2625	7	16384
Memthrash-w2k3e-3_NP_7__EE4	3799	2610	6	16384
Memthrash-w2k3e_NP_3_visor3	4545	2457	10	16384
Memthrash-w2k3e-3_NP_7__EE2	1198	2539	16	16384
CPUBurnIn_NP_bc9	45	1360	0	8192
CPUBurnIn_NP_bc	46	1359	0	8192
CPUBurnIn_NP_bc_bc1	79	1331	0	8192

Host CPU: avg. CPU utilization for Virtual Machine

Host Memory: consumed memory for Virtual Machine

Guest Memory: active memory for guest

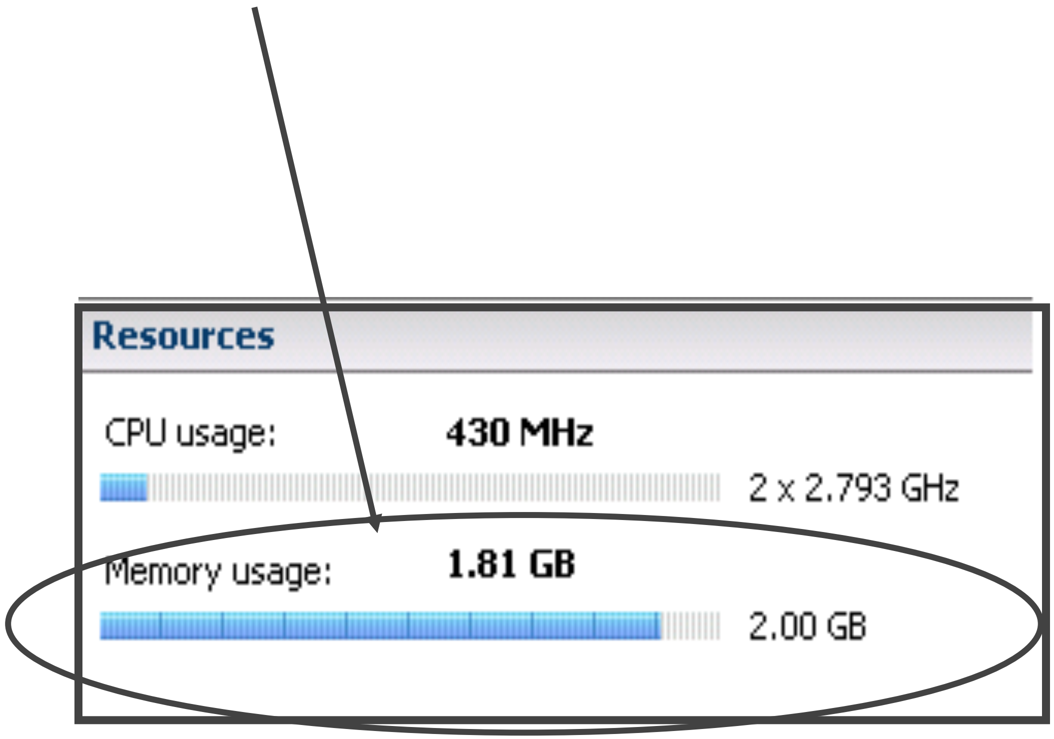
Host and Guest Memory Usage



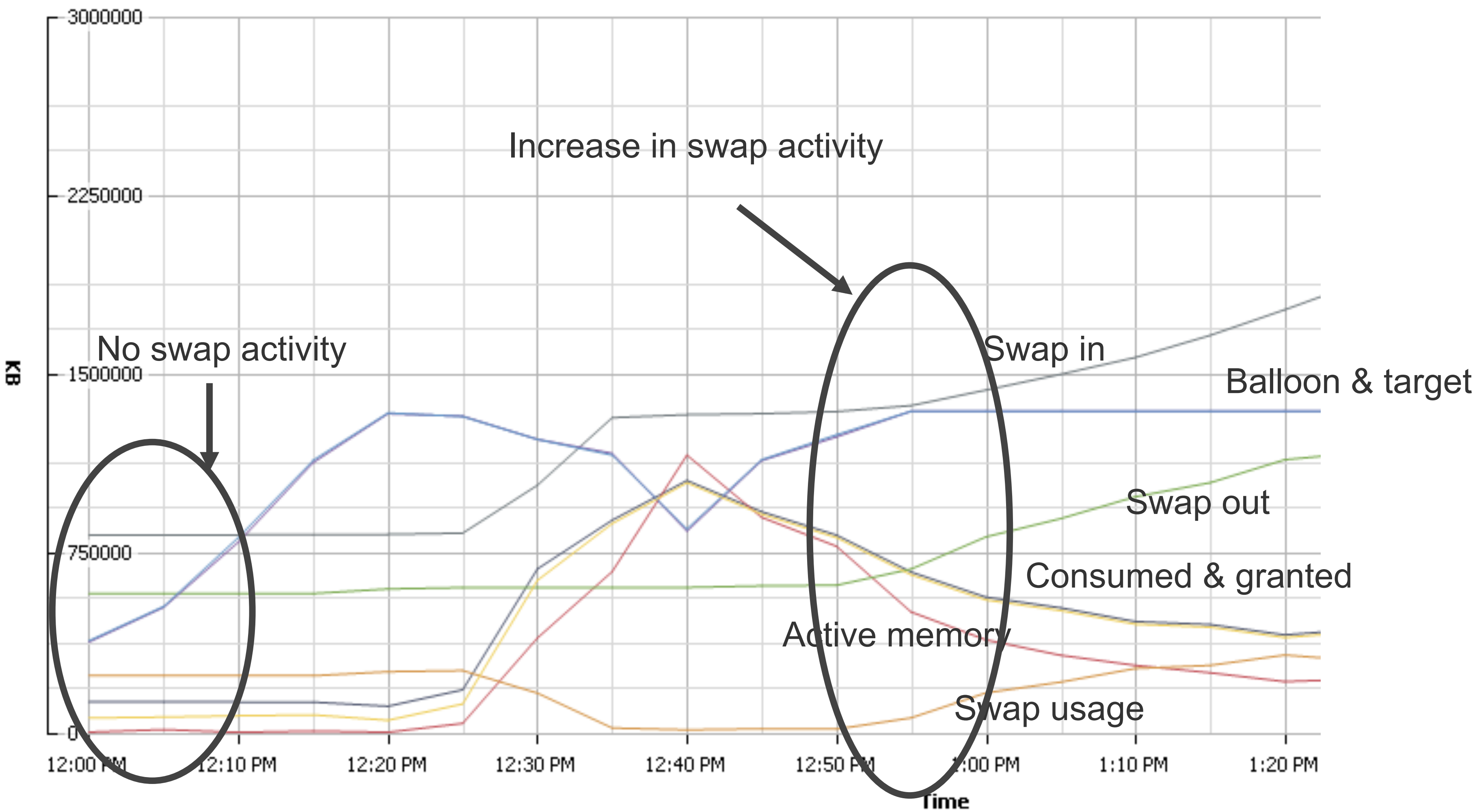
vmware

VI Client

- **Main page shows "consumed" memory (formerly "active" memory)**
- **Performance charts show important statistics for virtual machines**
  - Consumed memory
  - Granted memory
  - Ballooned memory
  - Shared memory
  - Swapped memory
    - Swap in
    - Swap out



VI Client: Memory example for Virtual Machine

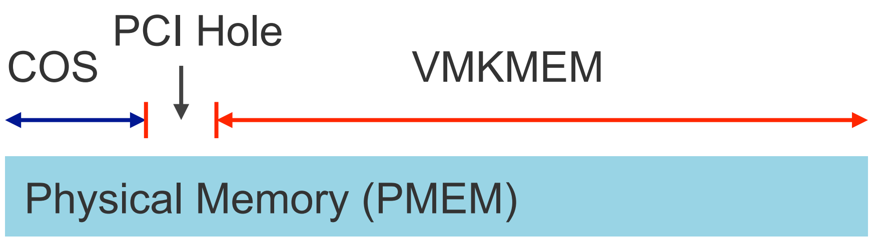


esxtop memory screen (m)

Possible states:
High,
Soft, hard and
low

```
10:55:29am up 43 days 23:50, 61 worlds; MEM overcommit avg: 0.00, 0.00, 0.00
PMEM  /MB:  4095    total:  272      cos,   171 vmk,     847 other,   2805 free
VMKMEM/MB:  3735  managed:  224  minfree,   496 rsvd,   3132 ursvd,  high state
COSMEM/MB:     5     free:  541   swap_t,   541 swap_f:   0.00 r/s,   0.00 w/s
PSHARE/MB:  2403   shared,   35   common:  2368 saving
SWAP  /MB:     0     curr,    0   target:                 0.00 r/s,   0.00 w/s
MEMCTL/MB:     0     curr,    0   target,  1996 max

  GID NAME                 NWLD    MEMSZ    SZTGT    SWCUR    SWTGT    SWR/s    SWW/s   OVHDUW     OVHD  OVHDMAX
   15 vmware-vmkauthd         1     5.46     5.46     0.00     0.00     0.00     0.00     0.00     0.00     0.00
   16 Windows 2003 SP         7  1024.00   380.20     0.00     0.00     0.00     0.00    30.41    62.86   121.87
   17 SQL2005                 7  2048.00   591.30     0.00     0.00     0.00     0.00    47.45    78.74   145.46
```

```
Current Field order: aBCDefGhiJklMno

  A:  ID = Id
* B:  GID = Group Id
* C:  NAME = Name
* D:  NWLD = Num Members
  E:  MEM ALLOC = MEM Allocations
  F:  NUMA STATS = Numa Statistics
* G:  SIZE = MEM Size (MB)
  H:  ACTV = MEM Active (MB)
  I:  MCTL = MEM Ctl  (MB)
* J:  SWAP STATS = Swap Statistics (MB)
  K:  CPT = MEM Checkpoint (MB)
  L:  COW = MEM Cow (MB)
* M:  OVHD = MEM Overhead (MB)
  N:  CMT = MEM Committed (MB)
  O:  RESP? = MEM Responsive?

Toggle fields with a-o, any other key to return:
```



VMKMEM - Memory managed by VMKernel
COSMEM - Memory used by Service Console

esxtop memory screen (m)

```
10:55:29am up 43 days 23:50, 61 worlds; MEM overcommit avg: 0.00, 0.00, 0.00
PMEM   /MB:  4095    total:   272       cos,    171 vmk,     847 other,   2805 free
VMKMEM/MB:  3735 managed:   224 minfree,   496 rsvd,   3132 ursvd,  high state
COSMEM/MB:     5     free:   541  swap_t,   541 swap_f:  0.00 r/s,   0.00 w/s
PSHARE/MB:  2403   shared,    35  common:  2368 saving
SWAP   /MB:     0     curr,     0  target:               0.00 r/s,   0.00 w/s
MEMCTL/MB:     0     curr,     0  target,  1996 max

   GID  NAME                NWLD    MEMSZ    SZTGT    SWCUR    SWTGT   SWR/s   SWW/s  OVHDUW    OVHD  OVHDMAX
    15  vmware-vmkauthd        1     5.46     5.46     0.00     0.00    0.00    0.00    0.00    0.00     0.00
    16  Windows 2003 SP        7  1024.00   380.20     0.00     0.00    0.00    0.00   30.41   62.86   121.87
    17  SQL2005                7  2048.00   591.30     0.00     0.00    0.00    0.00   47.45   78.74   145.46
```

Swapping activity in Service Console

VMKernel Swapping activity

SZTGT : determined by reservation, limit and memory shares
SWCUR = 0 : no swapping in the past
SWTGT = 0 : no swapping pressure
SWR/S, SWR/W = 0 : No swapping activity currently

SZTGT = Size target
SWTGT = Swap target
SWCUR = Currently swapped
MEMCTL = Balloon driver
SWR/S = Swap read /sec
SWW/S = Swap write /sec

Compression stats (new for 4.1)

```
 9:46:10am up 38 min, 300 worlds; MEM overcommit avg: 2.13, 2.12, 1.71
PMEM   /MB: 19834   total:  1214     vmk, 17252 other,   1367 free
VMKMEM/MB: 19737 managed:  1184 minfree, 17595 rsvd,   2142 ursvd,  high state
PSHARE/MB:  3537  shared,   757  common:  2780 saving
SWAP   /MB:  1163    curr,  3115 rclmtgt:                2.99 r/s,   1.51 w/s
ZIP    /MB:  3458  zipped,  2014   saved
MEMCTL/MB: 38641    curr, 38641  target, 38641 max
```

NAME	SWCUR	SWTGT	SWR/s	SWW/s	ZERO	SHRD	SHRDSVD	COWH	CACHESZ	CACHEUSD	ZIP/s	UNZIP/s
linux-vm1-swing	86.64	265.96	0.20	0.16	22.49	292.32	222.19	91.54	128.14	127.66	0.70	0.78
linux-vm2-swing	78.48	243.46	0.20	0.11	24.17	309.55	249.12	117.39	122.91	122.62	0.69	0.60
linux-vm3-swing	163.55	273.43	0.42	0.00	26.46	270.03	218.65	75.95	113.11	113.05	0.00	0.60
linux-vm4-swing	88.36	256.22	0.19	0.11	24.12	300.68	235.86	89.93	121.47	120.73	0.71	0.84
linux-vm5-swing	71.60	245.20	0.17	0.14	21.69	317.78	246.88	111.80	129.13	128.83	0.55	0.66
linux-vm6-swing	55.66	237.05	0.20	0.16	23.03	322.99	255.54	124.95	132.64	132.28	0.57	0.62
linux-vm7-swing	132.75	279.92	0.44	0.18	24.80	267.16	212.66	80.04	102.43	102.05	1.35	0.98
linux-vm8-swing	78.23	257.79	0.16	0.15	22.80	286.76	231.27	105.32	130.21	129.09	0.74	0.82
linux-vm9-swing	100.01	264.39	0.22	0.15	23.82	281.05	223.23	95.72	117.99	117.43	0.96	0.93
linux-vm10-swin	134.38	293.99	0.43	0.16	24.72	259.42	198.09	62.30	110.43	110.01	1.18	0.77
linux-vm11-swin	104.31	261.22	0.18	0.11	24.14	292.48	230.86	92.01	105.42	104.97	0.77	0.59
linux-vm0-swing	63.08	237.23	0.17	0.10	23.97	322.36	254.85	122.49	128.89	127.88	0.63	0.58

COWH : Copy on Write Pages hints – amount of memory in MB that are potentially shareable

CACHESZ: Compression Cache size

CACHEUSD: Compression Cache currently used

ZIP/s, UNZIP/s: Memory compression/decompression rate

Troubleshooting memory related problems (using 4.1 latencies)

```
PCPU USED(%):  54  54 0.7 0.1 0.9 0.1 0.1 0.1 0.9 0.3 0.2
PCPU UTIL(%): 100 100 0.7 0.1 0.8 0.1 0.3 0.1 1.1 0.4 0.5
CORE UTIL(%): 100     0.7     0.9     0.3     1.4     0.6

    ID NAME   %LAT_C %LAT_M %DMD  EMIN TIMER/s
385754 KC1      32.0    0.0   68  9767  200.00
385931 KC2      32.0    0.0   68  9767  200.00
```

%LAT_C : %time the VM was not scheduled due to CPU resource issue

%LAT_M : %time the VM was not scheduled due to memory resource issue

%DMD : Moving CPU utilization average in the last one minute

EMIN : Minimum CPU resources in MHZ that the VM is guaranteed to get when there is CPU contention

Troubleshooting memory related problems

- **Swapping**

```
 6:54:20am up 53 days 19:49, 87 worlds; MEM overcommit avg: 0.98, 1.15, 1.51
PMEM  /MB:  4095   total:   272     cos,   175 vmk,   1461 other,   2186 free
VMKMEM/MB:  3735 managed:   224 minfree,   976 rsvd,  2648 ursvd,  high state
COSMEM/MB:     9    free:   541  swap_t,   541 swap_f:   0.00 r/s,   0.00 w/s
PSHARE/MB:  5338  shared,   184  common:  5154 saving
SWAP  /MB:  1295    curr,   677  target:                 0.75 r/s,   0.01 w/s
MEMCTL/MB:   652    curr,   652  target,  4645 max
```

NAME	MEMSZ	SZTGT	MCTL?	MCTLSZ	MCTLTGT	MCTLMAX	SWCUR	SWTGT
Windows 2003 SP	1024.00	385.53	Y	0.00	0.00	665.60	119.82	0.00
SQL2005	2048.00	456.73	Y	0.00	0.00	1331.20	215.91	0.00
vc server	1024.00	284.19	Y	0.00	0.00	665.60	78.65	0.00
fakeDB	2048.00	483.75	Y	0.00	0.00	1331.20	203.79	0.00
memhog-linux-sm	1024.00	376.21	N	0.00	0.00	0.00	618.47	620.38
memhog-linux-CL	1024.00	347.39	Y	652.21	652.21	652.21	55.10	57.07

Memory Hog VMs

MCTL: N - Balloon driver not active, tools probably not installed

VM with Balloon driver swaps less

Swapped in the past but not actively swapping now

Swap target is more for the VM without the balloon driver

vmware

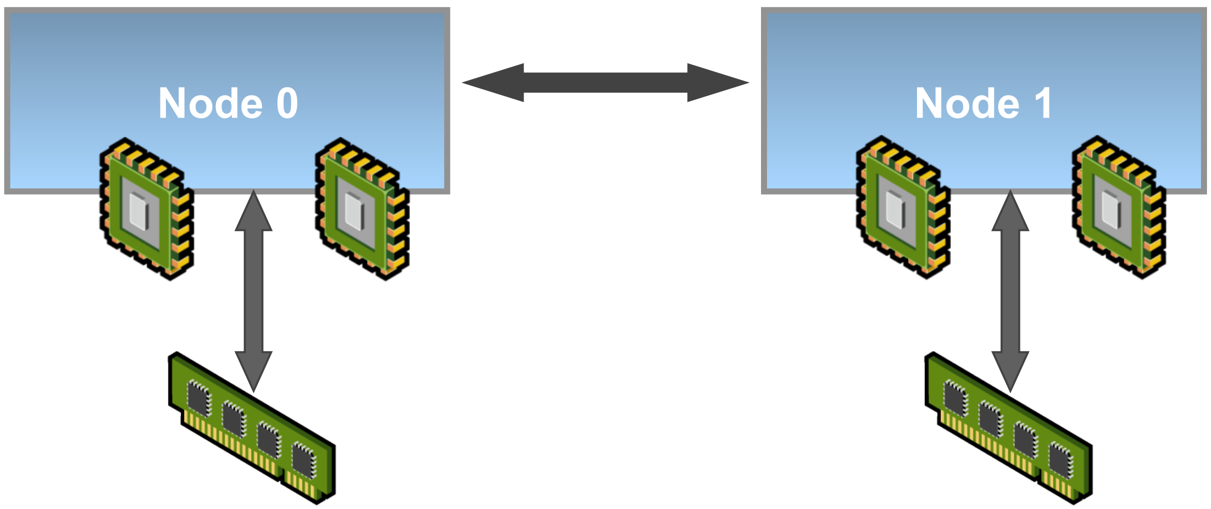
Additional Diagnostic Screens for ESXTOP

- **CPU Screen**
  - PCPU USED(%) – the CPU utilization per physical core or SMT
  - PCPU UTIL(%) – the CPU utilization per physical core or SMT thread
  - CORE UTIL(%) - GRANT (MB): Amount of guest physical memory mapped to a resource pool or virtual machine. Only used when hyperthreading is enabled.
  - SWPWT (%) - Percentage of time the Resource Pool/World was waiting for the ESX VMKernel swapping memory. The %SWPWT (swap wait) time is included in the %WAIT time.
- **Memory Screen**
  - GRANT (MB) - Amount of guest physical memory mapped to a resource pool or virtual machine. The consumed host machine memory can be computed as "GRANT - SHRDSVD".
- **Interrupt Screen (new)**
  - Interrupt statistics for physical devices

vmware

Memory Performance

- **Increasing a VM's memory on a NUMA machine**
  - Will eventually force some memory to be allocated from a remote node, which will decrease performance
  - Try to size the VM so both CPU and memory fit on one node



Memory Performance

- **NUMA scheduling and memory placement policies in ESX 3 manages all VMs transparently**
  - No need to manually balance virtual machines between nodes
  - NUMA optimizations available when node interleaving is disabled
- **Manual override controls available**
  - Memory placement: 'use memory from nodes'
  - Processor utilization: 'run on processors'
  - Not generally recommended
- **For best performance of VMs on NUMA systems**
  - # of VCPUs + 1 <= # of cores per node
  - VM memory <= memory of one node

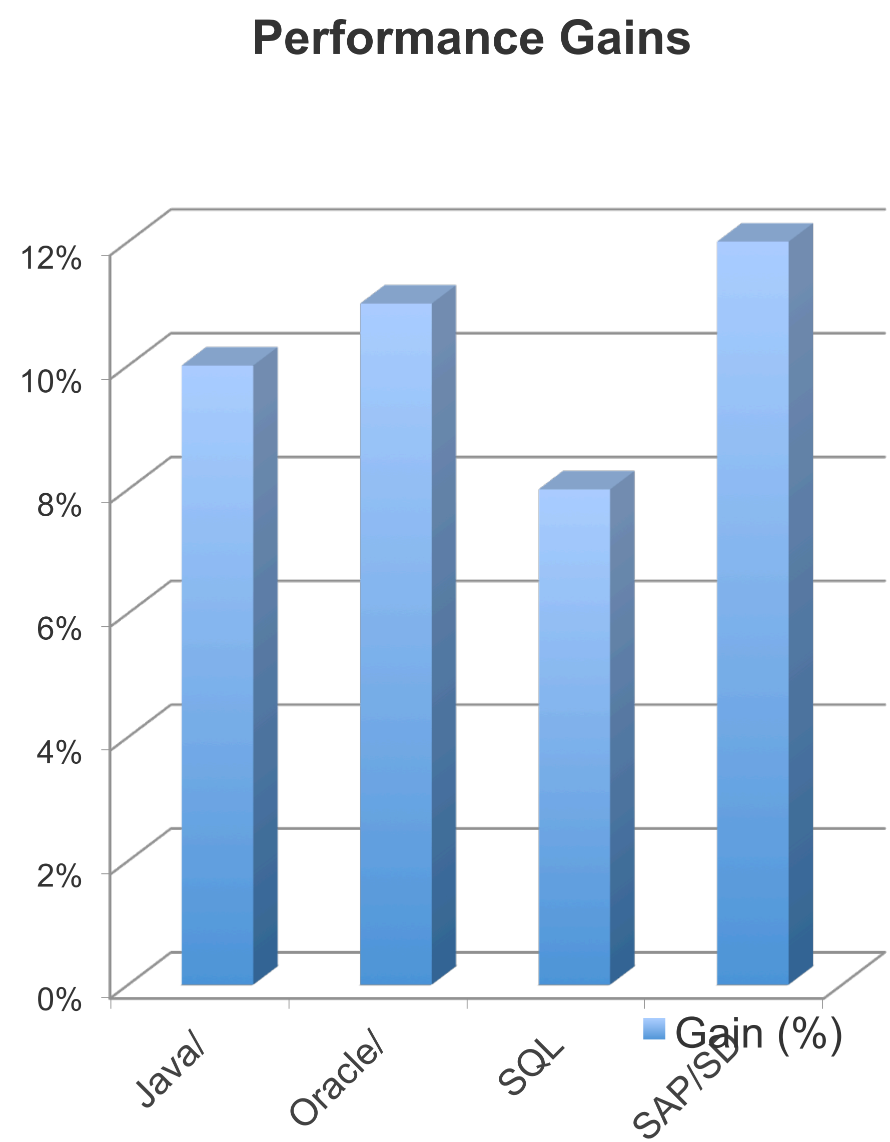
Memory Performance

- **Page tables**
  - ESX cannot use guest page tables
    - ESX Server maintains shadow page tables
    - Translate memory addresses from virtual to machine
    - Per process, per VCPU
  - VMM maintains physical (per VM) to machine maps
  - No overhead from “ordinary” memory references

- **Overhead**
  - Page table initialization and updates
  - Guest OS context switching

Large Pages

- **Increases TLB memory coverage**
  - Removes TLB misses, improves efficiency
- **Improves performance of applications that are sensitive to TLB miss costs**
- **Configure OS and application to leverage large pages**
  - LP will not be enabled by default



vmware

Large Pages and ESX Version

- **ESX 3.5: Large pages enabled manually for guest operations only**
- **ESX 4.0:**
  - With EPT/RVI: all memory backed by large pages
  - Without EPT/RVI: manually enabled, liked ESX 3.5

	Host Small Pages	Host Large Pages
Guest Small Pages	Baseline Performance	Efficient kernel operations, improved TLB for guest operations
Guest Large Pages	Improved page table performance	Improved page table, improved TLB

Memory Performance

- **ESX memory space overhead**
  - Service Console: 272 MB
  - VMkernel: 100 MB+
  - Per-VM memory space overhead increases with:
    - Number of VCPUs
    - Size of guest memory
    - 32 or 64 bit guest OS
- **ESX memory space reclamation**
  - Page sharing
  - Ballooning

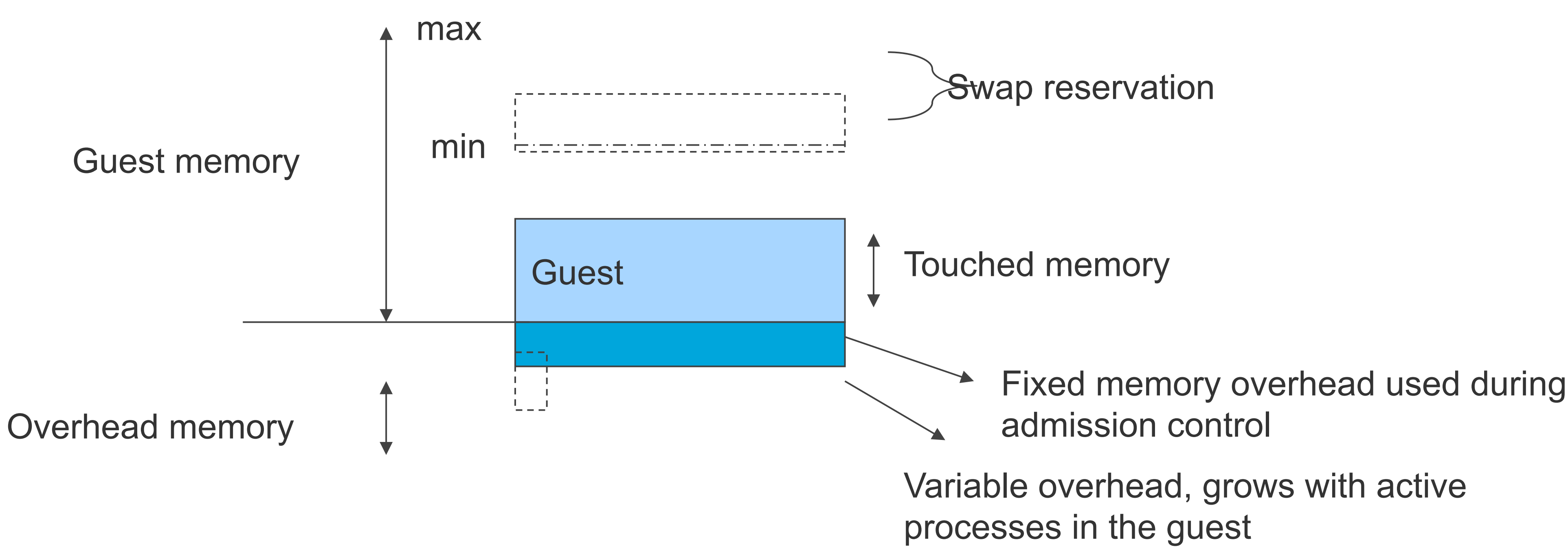
Memory Performance

- **Avoid high active host memory over-commitment**
  - Total memory demand = active working sets of all VMs
    + memory overhead
    – page sharing
  - No ESX swapping: total memory demand < physical memory
- **Right-size guest memory**
  - Define adequate guest memory to avoid guest swapping
  - Per-VM memory space overhead grows with guest memory

vmware

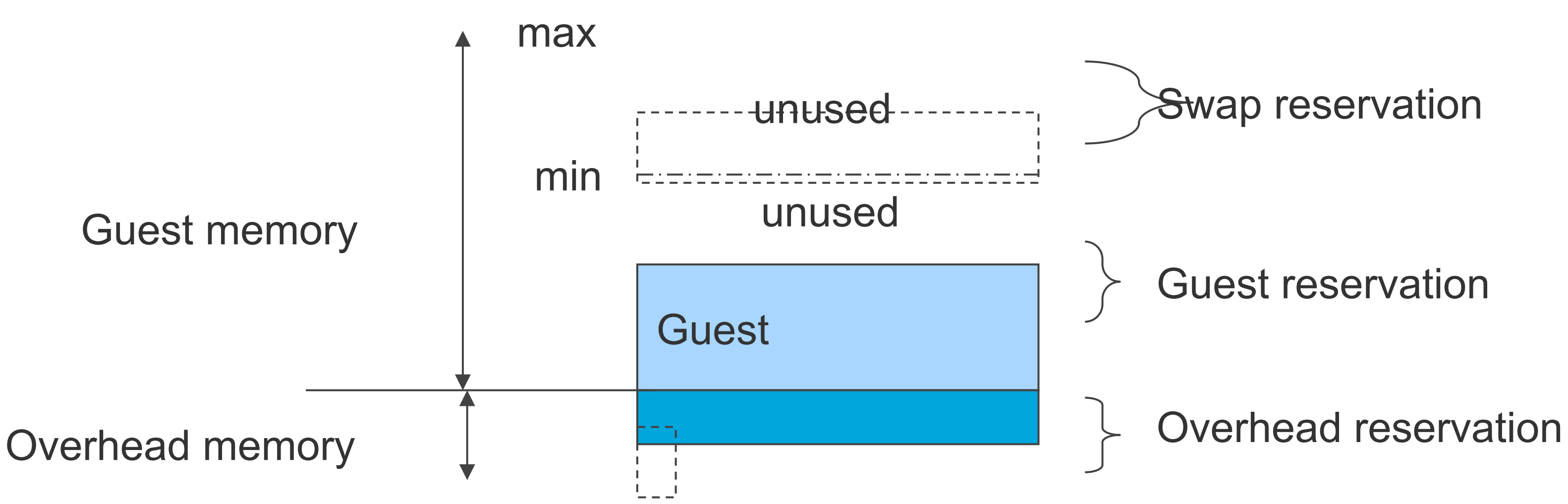
Memory Space Overhead

- **Additional memory required to run a guest**
  - Increases with guest memory size
  - Increases with the virtual CPU count
  - Increases with the number of running processes inside the guest



Memory Space Overhead: Reservation

- **Memory Reservation**
  - Reservation guarantees that memory is not swapped
  - Overhead memory is non-swappable and therefore it is reserved
  - Unused guest reservation cannot be used for another reservation
  - Larger guest memory reservation could restrict overhead memory growth
    - Performance could be impacted when overhead memory is restricted



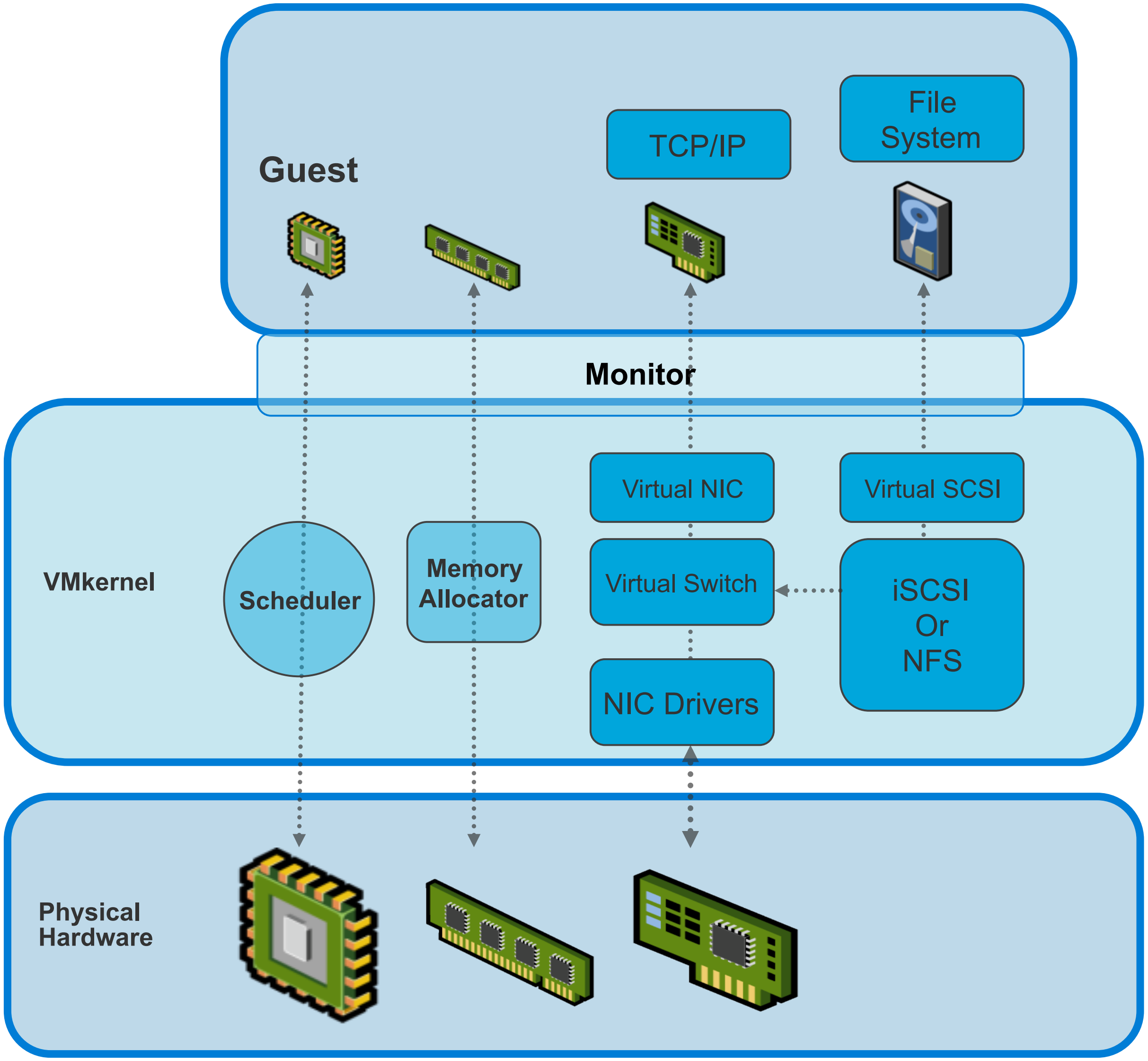
Reducing Memory Virtualization Overhead

- **Basic idea**
  - Smaller is faster (but do not undersize the VM) ☺

- **Recommendations**
  - Right size VM
    - avoids overhead of accessing HIGHMEM (>786M) and PAE pages (>4G) in 32-bit VMs
    - Smaller memory overhead provides room for variable memory overhead growth
  - UP VM
    - Memory virtualization overhead is generally lesser
    - Smaller memory space overhead
  - Tune Guest OS/applications
    - Prevent/reduce application soft/hard page faults
    - Pre-allocate memory for applications if possible

I/O AND STORAGE

Introduction



iSCSI and NFS are growing To be popular, due to their low port/switch/fabric costs

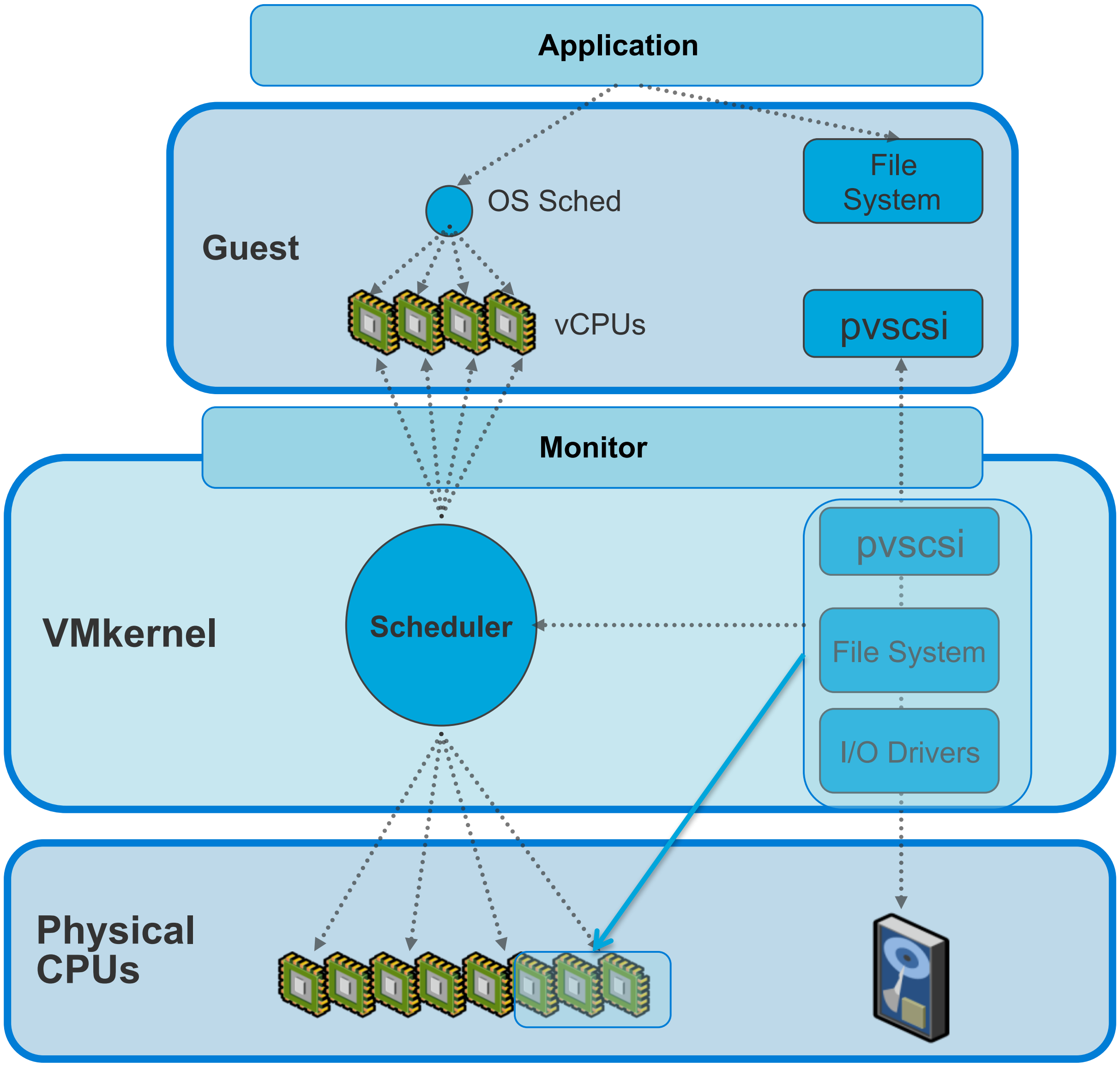
Virtualization provides the ideal mechanism to transparently adopt iSCSI/NFS

Guests don't need iSCSI/NFS Drivers: they continue to see SCSI

VMware ESX 3 provides high Performance NFS and iSCSI Stacks

Futher emphasis on 1Gbe/ 10Gbe performance

Asynchronous I/O (4.0)



On-loads I/O processing to additional cores

Guest VM issues I/O and continues to run immediately

VMware ESX asynchronously issues I/Os and notifies the VM upon completion

VMware ESX can process Multiple I/Os in parallel on separate cpus

Significantly Improves IOPs and CPU efficiency

Device Paravirtualization (4.0)

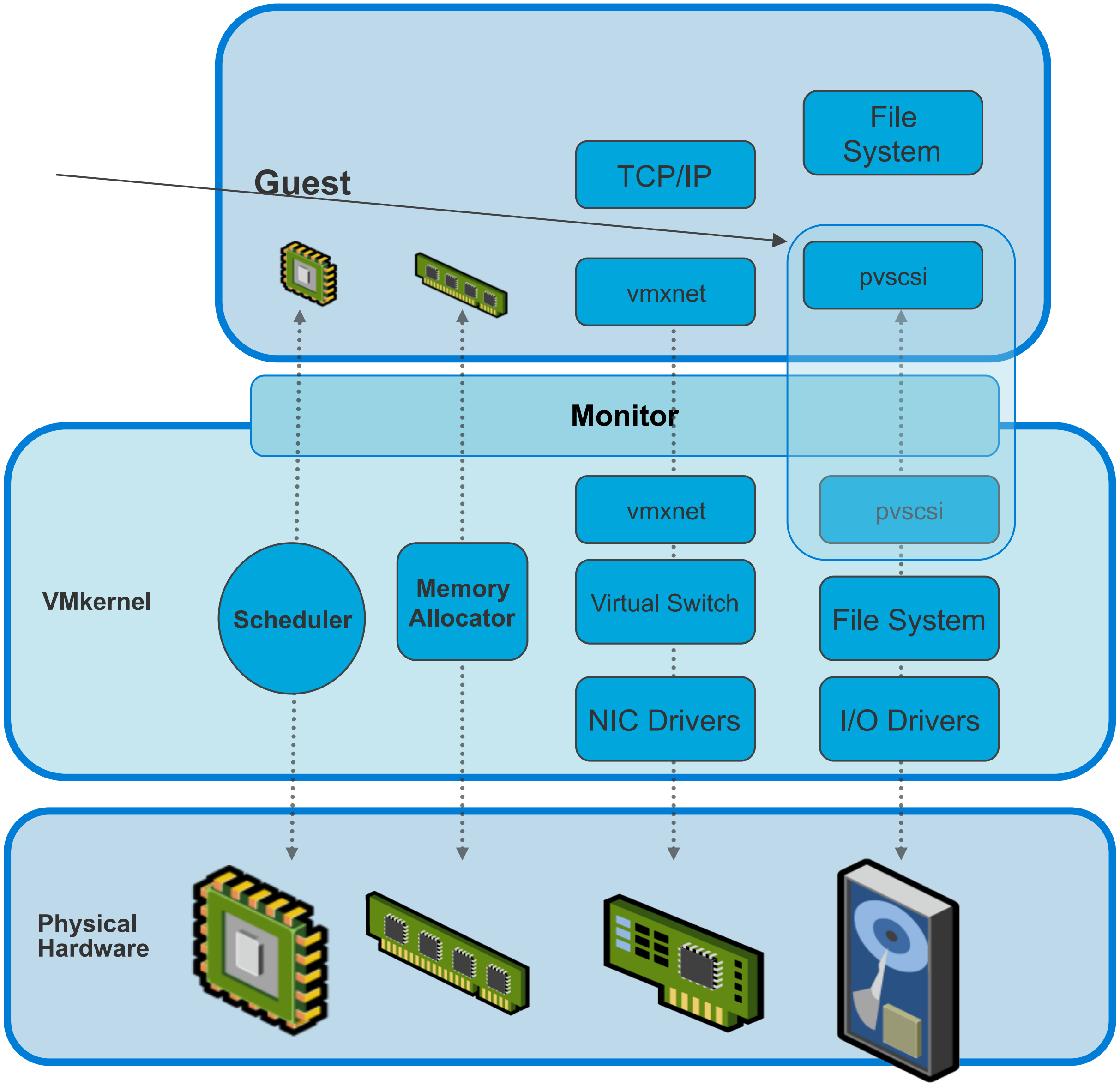
Device Paravirtualization places A high performance virtualization-Aware device driver into the guest

Paravirtualized drivers are more CPU efficient (less CPU over-head for virtualization)

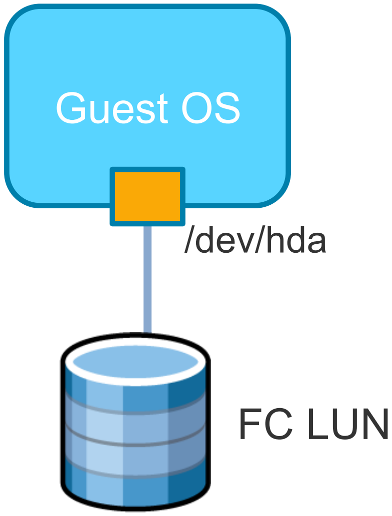
Paravirtualized drivers can also take advantage of HW features, like partial offload (checksum, large-segment)

VMware ESX uses para-virtualized network drivers

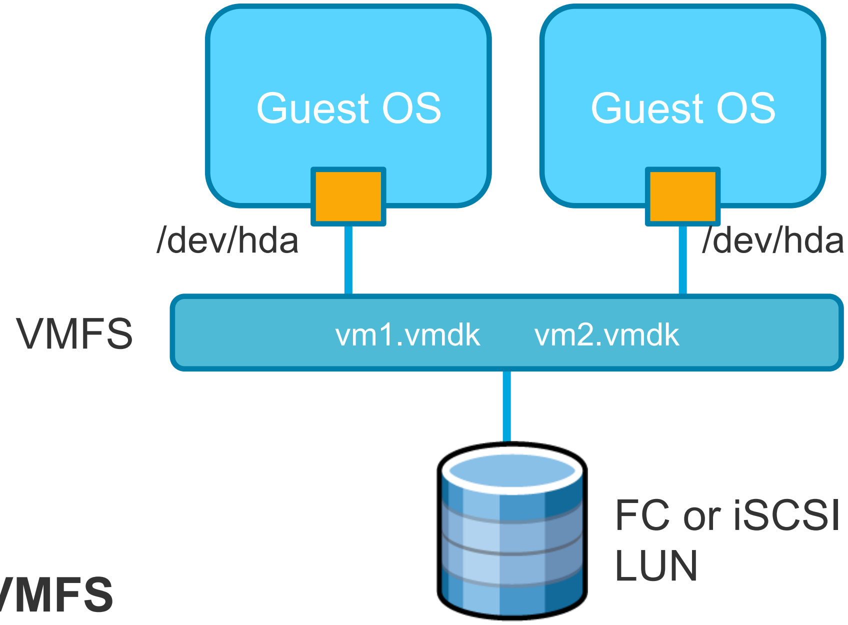
vSphere 4 now provides *pvscsi*



Storage – Fully virtualized via VMFS and Raw Paths

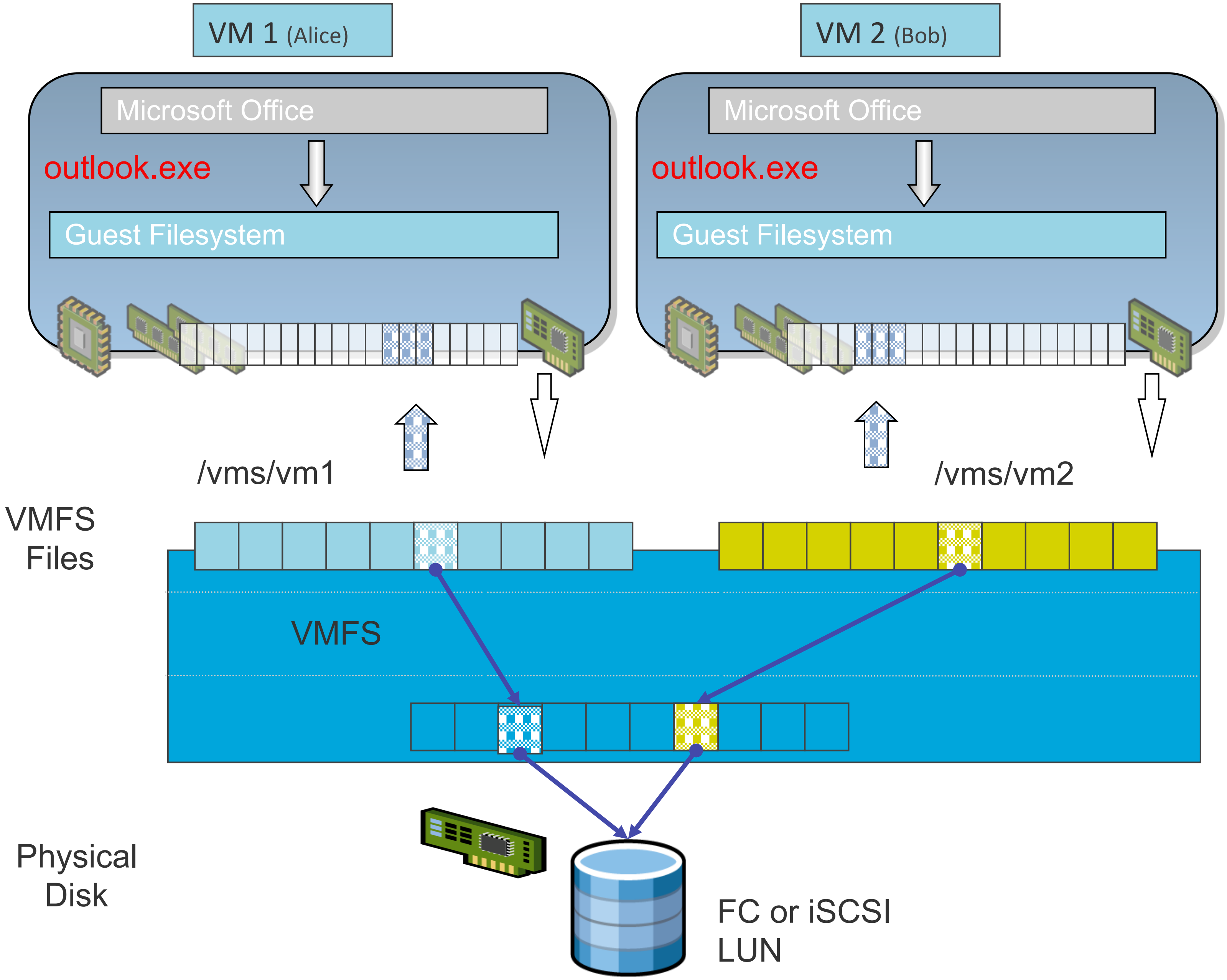


- **RAW**
- **RAW provides direct access to a LUN from within the VM**
- **Allows portability between physical and virtual**
- **RAW means more LUNs**
  - More provisioning time
- **Advanced features still work**

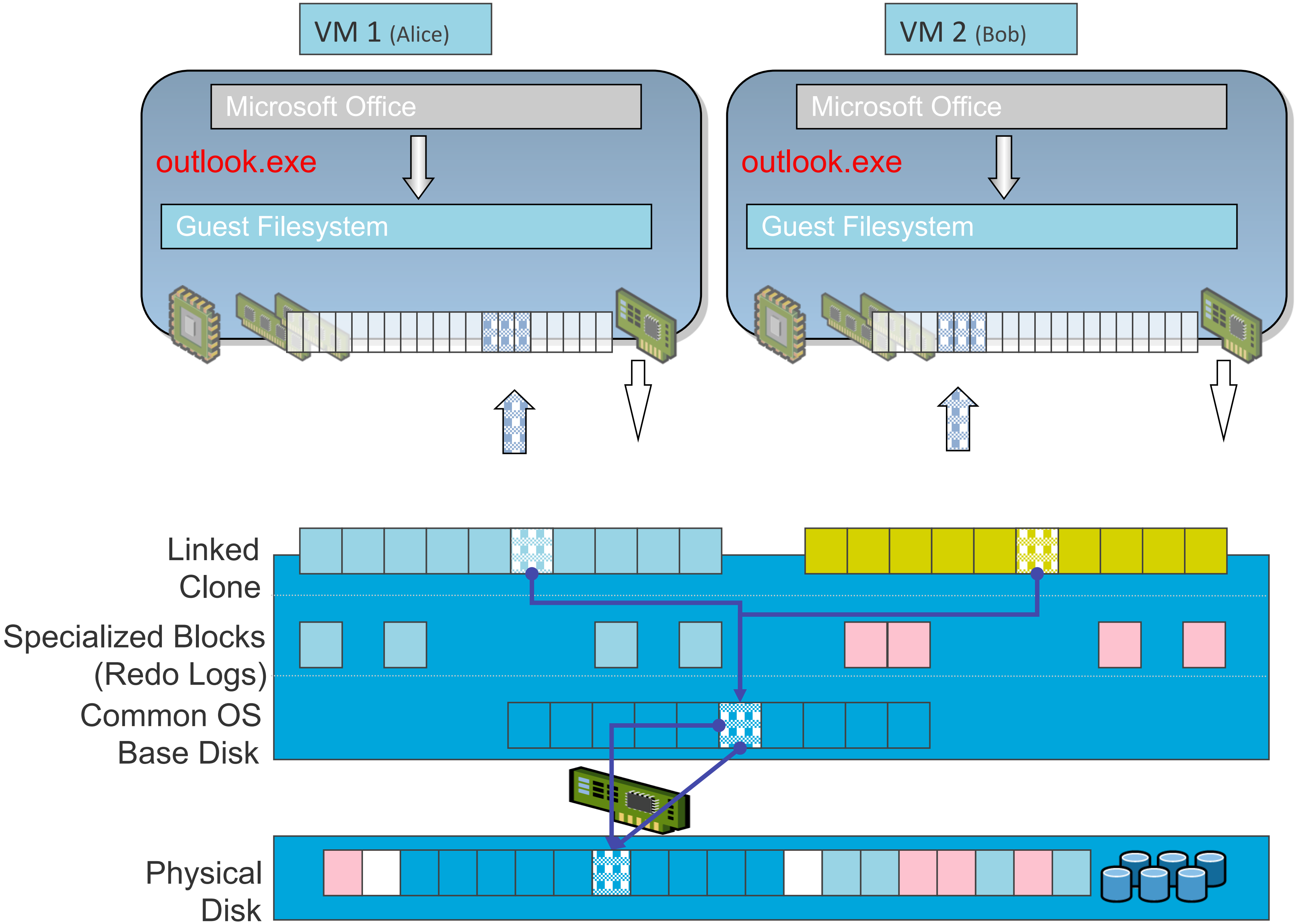


- **VMFS**
- **Easier provisioning**
- **Snapshots, clones possible**
- **Leverage templates and quick provisioning**
- **Scales better with Consolidated Backup**
- **Preferred Method**

How VMFS Works



VMFS Clones and Snapshots



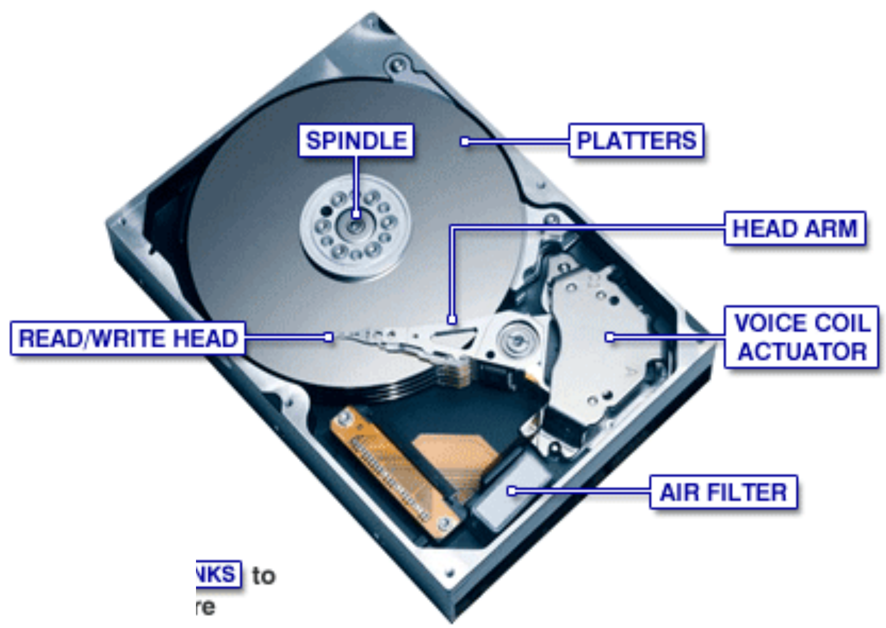
I/O Performance

- **Disk performance is dependent on many factors:**
  - Filesystem performance
  - Disk subsystem configuration (SAN, NAS, iSCSI, local disk)
  - Disk caching
  - Disk formats (thick, sparse, thin)

- **ESX is tuned for Virtual Machine I/O**
- **VMFS clustered filesystem => keeping consistency imposes some overheads**

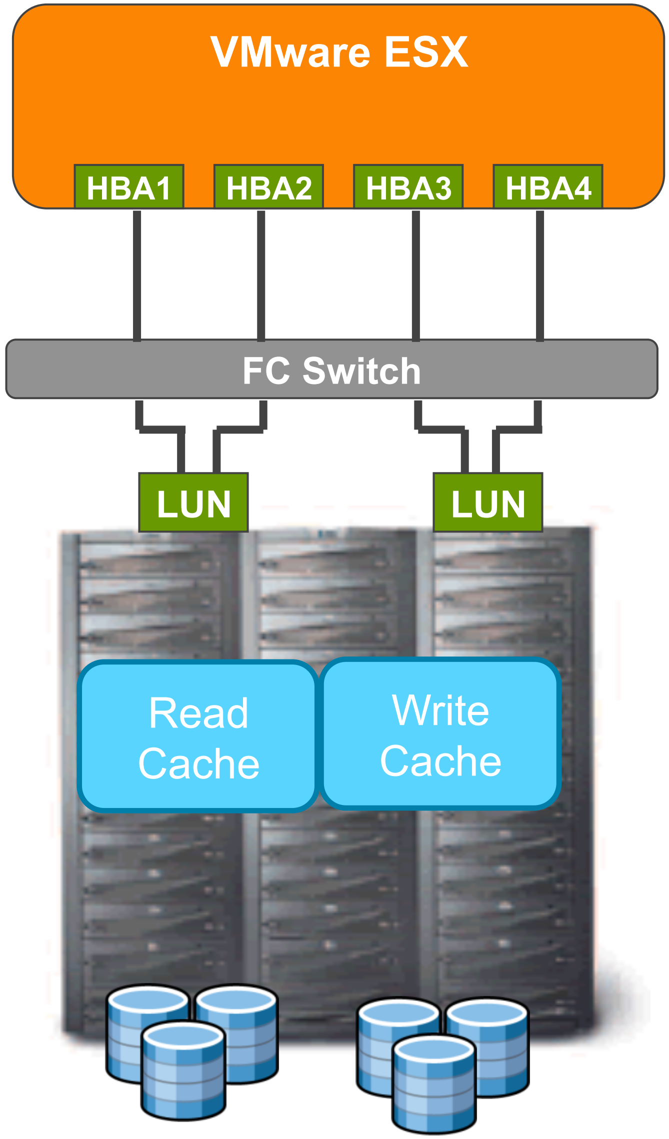
Disk Fundamentals

- **Disk performance is impacted by Bandwidth and I/O demands**
- **Sequential accesses to disk are bandwidth limited**
  - ~70MBytes/sec for a SATA disk
  - ~150Mbytes/sec for a 15k RPM FC disk
- **Random Accesses to disk are dominated by seek/rotate**
  - 10k RPM Disks: 150 IOPS max, ~80 IOPS Nominal
  - 15k RPM Disks: 250 IOPS max, ~120 IOPS Nominal
- **Typically hidden behind an array**
  - ESX sees LUN latency
  - Exception is local-disk

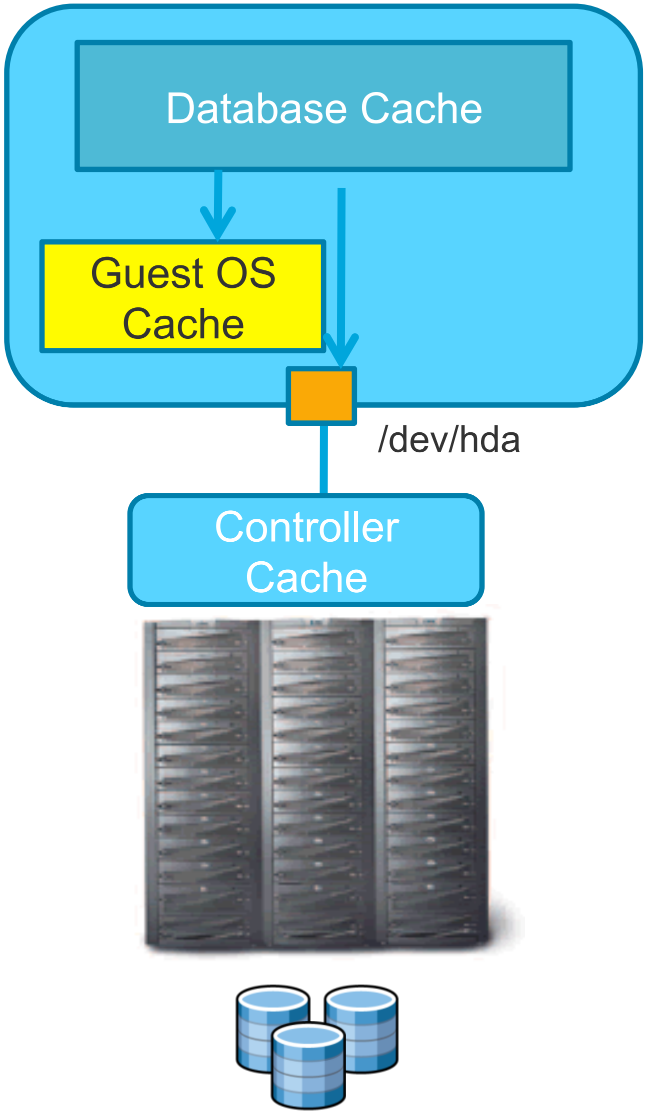


Disk Arrays

- **Lowest level resource is disk**
  - 150 IOPS, 70-150MByte/sec
- **Disks are aggregated into LUNS**
  - Increase performance and availability
- **LUNS can be (should be) cached**
  - Read caches or write caches
  - Write caches hide *wait-for-write*
- **Disk arrays share FC Connections**
  - Typically 200 or 400MBytes/sec

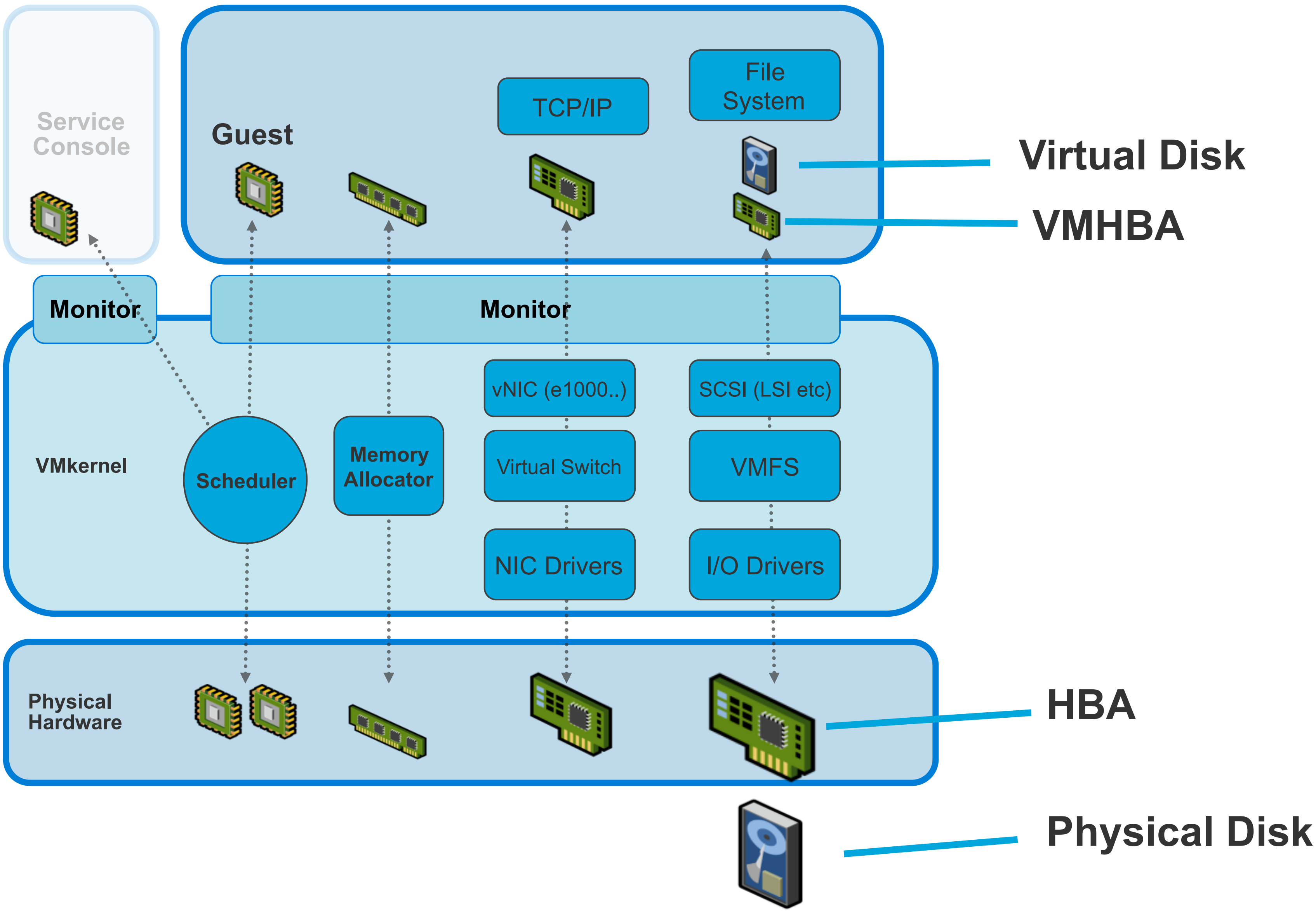


It's important to understand caches when observing I/O

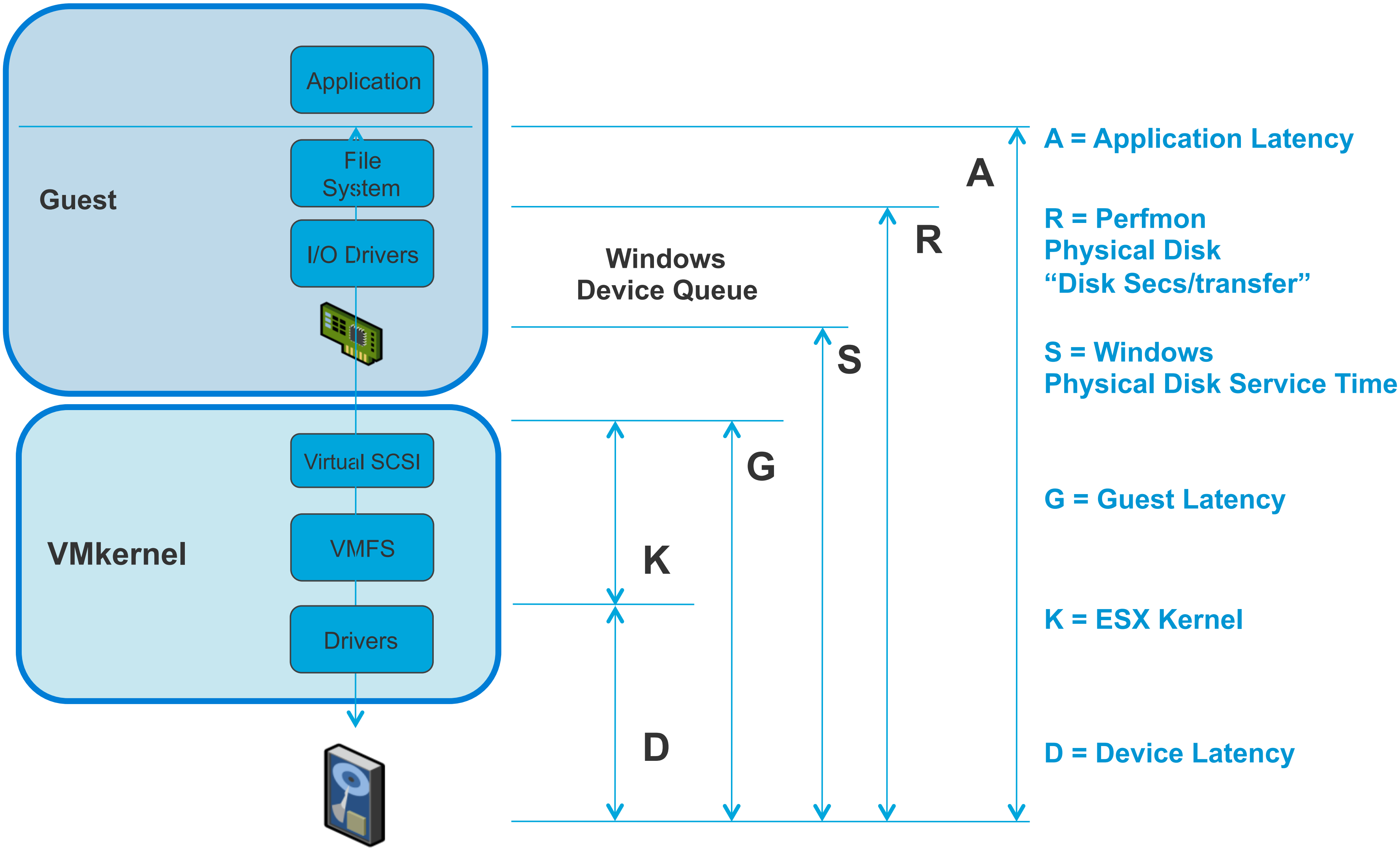


- Caches attempt to eliminate I/Os
  - The best I/O is the one you don't do
- Caches are at multiple layers:
  - Application
  - Guest-OS
  - Disk-array
- Q: What's the impact on the number of disks if we improve cache hit rates from 90% to 95%?
  - 10 in 100 => 5 in 100…
  - #of disks reduced by 2x!

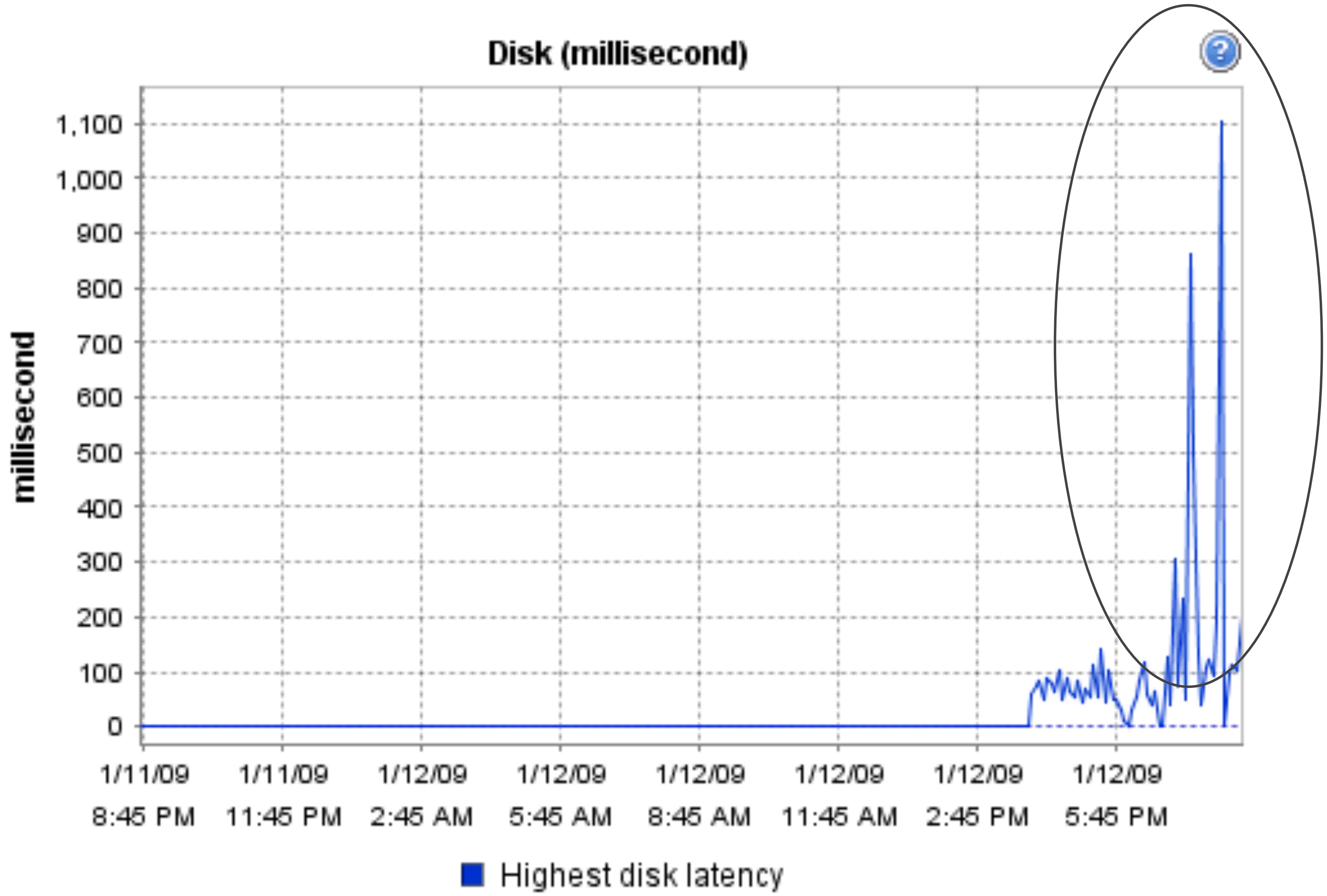
Observing I/O Performance: Important I/O Terminology



Disk Latencies Explained



Let’s look at the vSphere client…



Rule of thumb: latency > 20ms is Bad.

Here:

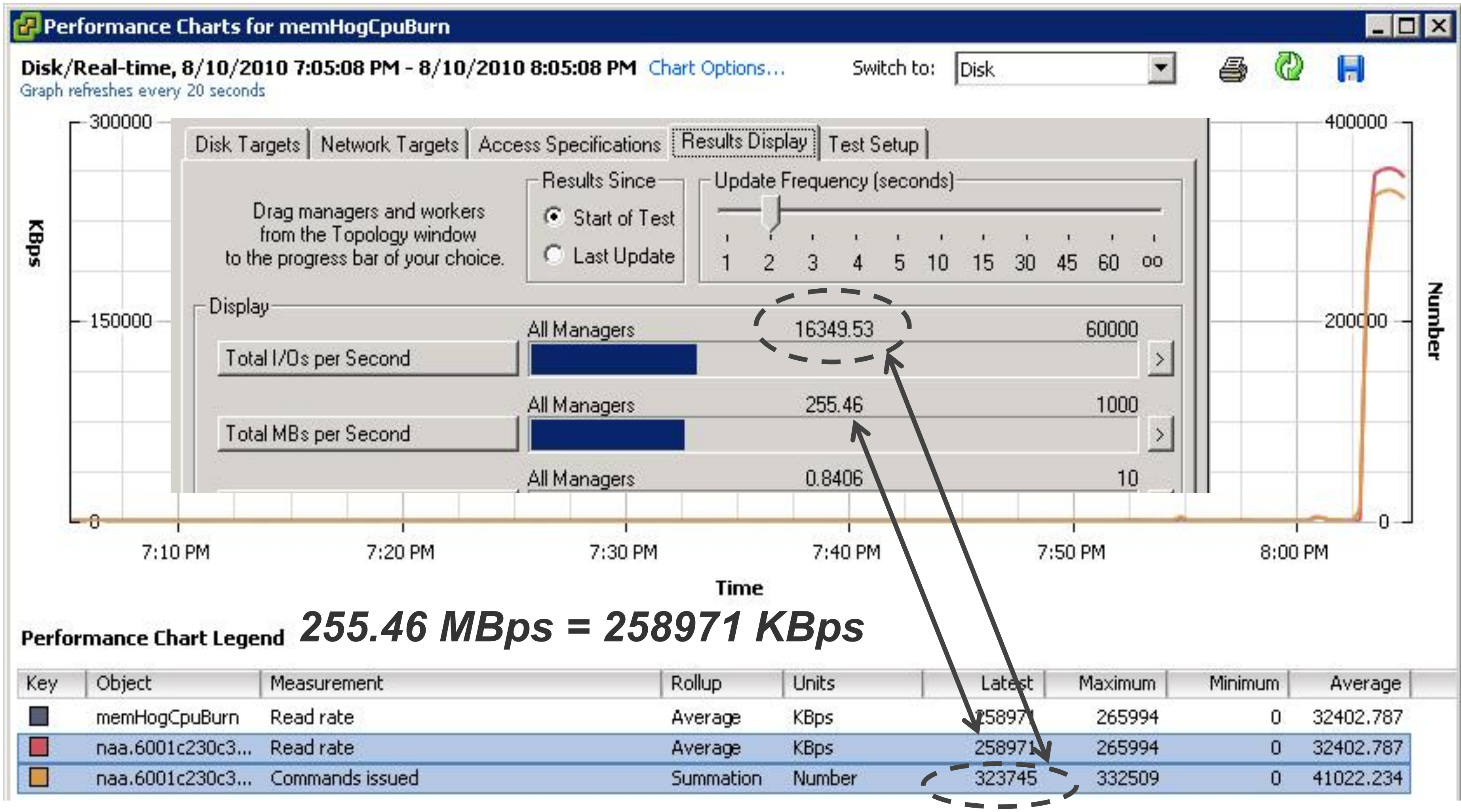
1,100ms

REALLY BAD!!!

A Word About Units in vSphere

Operation throughput: commands per refresh interval (not IOPS)

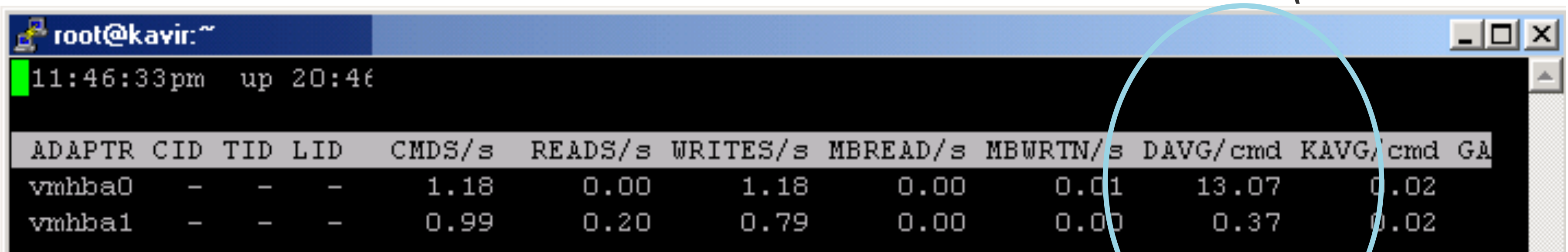
Bandwidth in KBps (not MBps)



Real-time chart: refresh 20s. 16349 IOPS = 323745 commands/20s

Disk Latencies

(screenshot of esxtop)

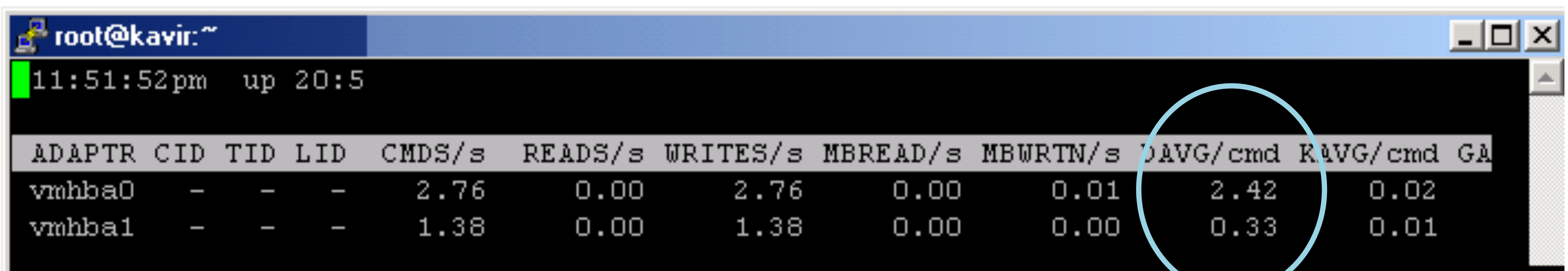


root@kavir:~

11:46:33pm up 20:46

ADAPTR	CID	TID	LID	CMDS/s	READS/s	WRITES/s	MBREAD/s	MBWRTN/s	DAVG/cmd	KAVG/cmd	GA
vmhba0	-	-	-	1.18	0.00	1.18	0.00	0.01	13.07	0.02	
vmhba1	-	-	-	0.99	0.20	0.79	0.00	0.00	0.37	0.02	

Latency seems high



root@kavir:~

11:51:52pm up 20:5

ADAPTR	CID	TID	LID	CMDS/s	READS/s	WRITES/s	MBREAD/s	MBWRTN/s	DAVG/cmd	KAVG/cmd	GA
vmhba0	-	-	-	2.76	0.00	2.76	0.00	0.01	2.42	0.02	
vmhba1	-	-	-	1.38	0.00	1.38	0.00	0.00	0.33	0.01	

After enabling cache, latency is much better

esxtop disk adapter screen (d)

10:46:28am up 2 days 3:16, 77 worlds; CPU load average: 0.32, 0.31, 0.32

ADAPTR	CMDS/s	READS/s	WRITES/s	MBREAD/s	MBWRTN/s	DAVG/cmd	KAVG/cmd	GAVG/cmd	QAVG/cmd
vmhba0	5.24	0.00	5.24	0.00	0.09	46.11	29.87	75.98	0.00
vmhba1	5.06	0.00	5.06	0.00	0.02	1.10	0.01	1.11	0.00
vmhba2	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Host bus adapters (HBAs) - includes SCSI, iSCSI, RAID, and FC-HBA adapters

Latency stats from the Device, Kernel and the Guest

DAVG/cmd - Average latency (ms) from the Device (LUN)

KAVG/cmd - Average latency (ms) in the VMKernel

GAVG/cmd - Average latency (ms) in the Guest

esxtop disk device screen (u)

```
10:23:35am up 2 days  2:53, 77 worlds; CPU load average: 0.30, 0.33, 0.37

   DEVICE       CMDS/s  READS/s WRITES/s MBREAD/s MBWRTN/s DAVG/cmd KAVG/cmd GAVG/cmd QAVG/cmd
  vmhba0:0:0      4.68     0.00     4.68     0.00     0.08    47.80    32.74    80.53    32.73
  vmhba0:1:0      0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00
  vmhba1:0:0      0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00
  vmhba1:0:1      0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00
  vmhba1:0:2     25.37    19.29     6.00     2.37     2.40     3.35     0.01     3.35     0.00
  vmhba1:0:3      0.34     0.00     0.34     0.00     0.00     4.79     0.01     4.80     0.00
  vmhba1:0:4      3.90     0.00     3.90     0.00     0.02     0.49     0.01     0.50     0.00
  vmhba1:0:5      0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00
  vmhba1:0:6      0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00
  vmhba1:0:7      0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00
  vmhba1:0:8      0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00     0.00
```

LUNs in C:T:L format

C:T:L - Controller: Target: Lun

esxtop disk VM screen (v)

```
10:44:06am up 2 days  3:13, 77 worlds; CPU load average: 0.31, 0.32, 0.32
View VM only
```

ID	GID	NAME	CMDS/s	READS/s	WRITES/s	MBREAD/s	MBWRTN/s
21	21	windows_vm	0.00	0.00	0.00	0.00	0.00
24	24	windows_vm3	2.05	0.00	2.05	0.00	0.01
25	25	windows_vm4	0.00	0.00	0.00	0.00	0.00
27	27	windows_sp2_vm1	0.00	0.00	0.00	0.00	0.00
31	31	windows_sp2_vm	2.87	0.00	2.87	0.00	0.02

running VMs

Disk screen (d)

- **SCSI Reservation stats (new in 4.1)**

ADAPTR	NPTH	CMDS/s	READS/s	WRITES/s	MBREAD/s	MBWRTN/s	RESV/s	CONS/s
vmhba0	1	0.00	0.00	0.00	0.00	0.00	0.00	0.00
vmhba1	2	4.65	2.71	1.55	0.01	0.00	0.19	0.00
vmhba2	10	4.07	0.00	4.07	0.00	0.03	0.00	0.00
vmhba3	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00
vmhba32	1	0.00	0.00	0.00	0.00	0.00	0.00	0.00
vmhba34	0	0.00	0.00	0.00	0.00	0.00	0.00	0.00

RESV/s : SCSI reservations per second
CONS/s: SCSI reservation conflicts per second

LUN screen (u)

VAAI (vStorage API for Array Integration) Stats (new in 4.1)

DEVICE	CLONE_RD	CLONE_WR	CLONE_F	MBC_RD/s	MBC_WR/s	ATS	ATSF	ZERO	ZERO_F	MBZERO/s
mpx.vmhba0:C0:T	0	0	0	0.00	0.00	0	0	0	0	0.00
mpx.vmhba1:C0:T	0	0	0	0.00	0.00	0	0	0	0	0.00
naa.60060160a91	18178	18178	18178	0.00	0.00	3552	0	102406	0	0.00
{NFS}build-tool	0	0	0	0.00	0.00	0	0	0	0	0.00

CLONE_RD, CLONE_WR: Number of Clone read/write requests
CLONE_F: Number of Failed clone operations
MBC_RD/s, MBC_WR/s – Clone read/write MBs/sec
ATS – Number of ATS commands
ATSF – Number of failed ATS commands
ZERO – Number of Zero requests
ZEROF – Number of failed zero requests
MBZERO/s – Megabytes Zeroed per second

VM disk screen

VM disk screen now reports stats using vScsistats (new in 4.1)

ID	GID	VMNAME	VSCSINAME	NDK	CMDS/s	READS/s	WRITES/s	MBREAD/s	MBWRTN/s	LAT/rd	LAT/wr
6242729	6242729	vMA 4.1	-	1	0.78	0.00	0.78	0.00	0.00	0.00	0.57
8314	6342310	Exchange2007	scsi0:0	-	24.68	0.20	24.49	0.01	0.15	6.37	0.59
8315	6342310	Exchange2007	scsi0:1	-	0.59	0.00	0.59	0.00	0.00	0.00	3.45
8316	6342310	Exchange2007	scsi0:2	-	0.59	0.00	0.59	0.00	0.00	0.00	0.36
8317	6342310	Exchange2007	scsi0:3	-	0.39	0.00	0.39	0.00	0.00	0.00	1.57
8318	6342310	Exchange2007	scsi0:4	-	0.39	0.00	0.39	0.00	0.00	0.00	0.28

ESX 3.x and 4.x provides this stats by grouping I/Os based on the world ids

I/O Latency from NFS Volumes

- vSphere 4.1 enables latency information for NFS based storage

DEVICE	CMDS/s	READS/s	WRITES/s	MBREAD/s	MBWRTN/s	DAVG/cmd	KAVG/cmd	GAVG/cmd	QAVG/cmd
mpx.vmhba0:C0:T	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
naa.6000eb39974	1.63	0.00	1.63	0.00	0.01	1.38	0.02	1.40	0.01
naa.600508b1001	1.81	0.00	1.81	0.00	0.01	0.09	0.02	0.10	0.00
{NFS}cloud12Nfs	15113.93	15113.57	0.36	466.32	0.00	-	-	0.77	-
{NFS}nfsMount0	0.00	0.00	0.00	0.00	0.00	-	-	0.00	-
{NFS}nfsMount1	0.00	0.00	0.00	0.00	0.00	-	-	0.00	-
{NFS}nfsMount2	0.00	0.00	0.00	0.00	0.00	-	-	0.00	-

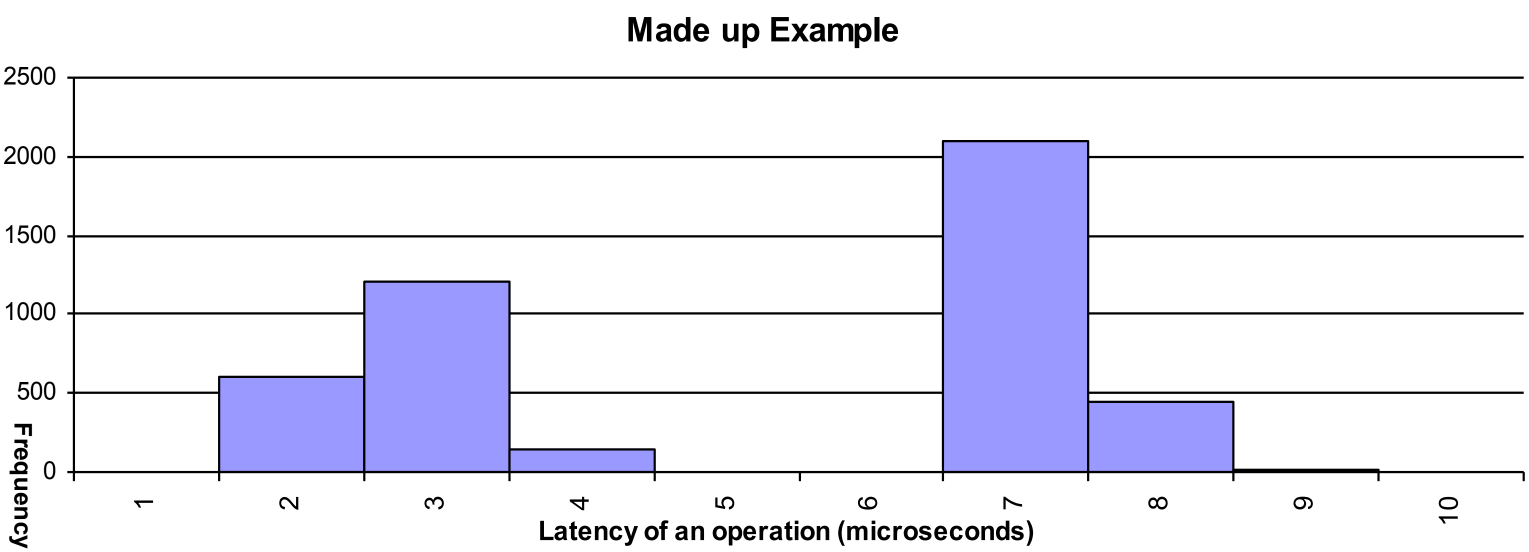
DAVG/cmd	KAVG/cmd	GAVG/cmd
0.00	0.00	0.00
1.38	0.02	1.40
0.09	0.02	0.10
-	-	0.77
-	-	0.00
-	-	0.00
-	-	0.00

vScsiStats

- **Disk I/O characterization of applications is the first step in tuning disk subsystems; key questions:**
  - I/O block size
  - Spatial locality
  - I/O interarrival period
  - Active queue depth
  - Latency
  - Read/Write Ratios
- **Our technique allows transparent and online collection of essential workload characteristics**
  - Applicable to arbitrary, unmodified operating systems running in virtual machines

Workload Characterization Technique

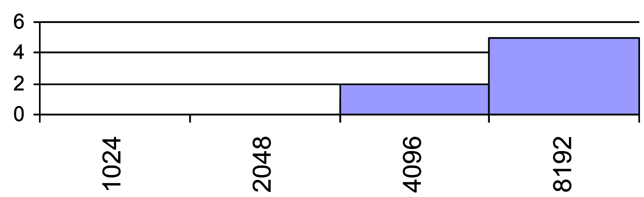
- **Histograms of observed data values can be much more informative than single numbers like mean, median, and standard deviations from the mean**
  - E.g., multimodal behaviors are easily identified by plotting a histogram, but obfuscated by a mean
- **Histograms can actually be calculated efficiently online**
- **Why take one number if you can have a distribution?**

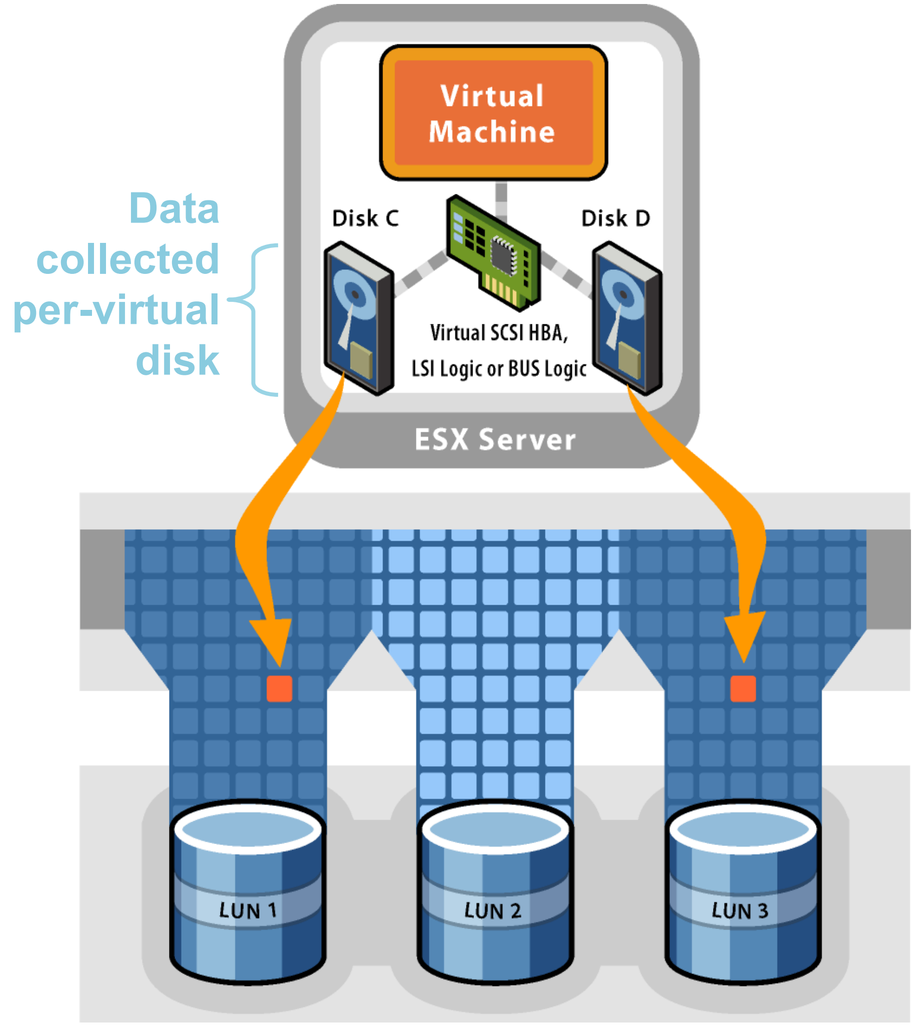


Mean is 5.3!

Workload Characterization Technique

- **The ESX disk I/O workload characterization is on a per-virtual disk basis**
  - Allows us to separate out each different type of workload into its own container and observe trends
- **Histograms only collected if enabled; no overhead otherwise**
- **Technique:**
  - For each virtual machine I/O request in ESX, we insert some values into histograms
    - E.g., size of I/O request → 4KB





Workload Characterization Technique

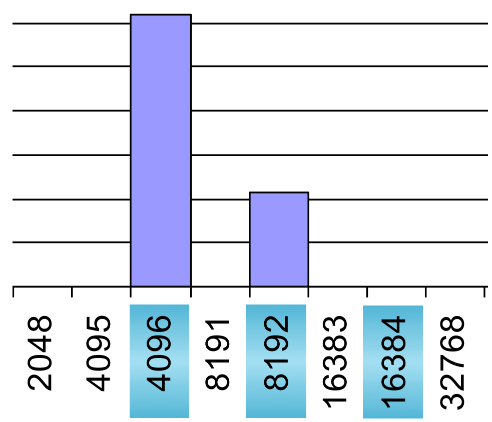
Full List of Histograms

- **Read/Write Distributions are available for our histograms**
  - Overall Read/Write ratio?
  - Are Writes smaller or larger than Reads in this workload?
  - Are Reads more sequential than Writes?
  - Which type of I/O is incurring more latency?
- **In reality, the problem is not knowing which question to ask**
  - Collect data, see what you find

- **I/O Size**
  - All, Reads, Writes
- **Seek Distance**
  - All, Reads, Writes
- **Seek Distance Shortest Among Last 16**
- **Outstanding IOs**
  - All, Reads, Writes
- **I/O Interarrival Times**
  - All, Reads, Writes
- **Latency**
  - All, Reads, Write

Workload Characterization Technique
Histograms Buckets

- **To make the histograms practical, bin sizes are on rather irregular scales**
  - E.g., the I/O length histogram bin ranges like this:
    - …, 2048, 4095, 4096, 8191, 8192, … rather odd: some buckets are big and others are as small as just 1
    - Certain block sizes are really special since the underlying storage subsystems may optimize for them; single those out from the start (else lose that precise information)
    - E.g., important to know if the I/O was 16KB or some other size in the interval (8KB,16KB)

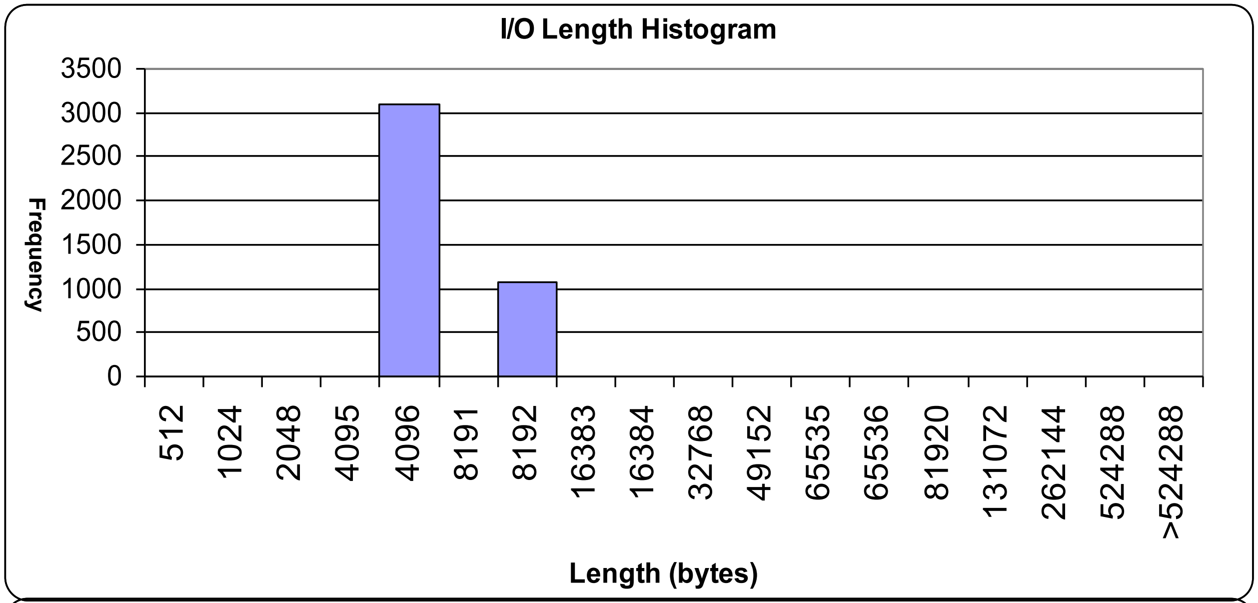


Filebench OLTP (Solaris)

- **Filebench is a model-based workload generator for file systems developed by Sun Microsystems**
  - Input to this program is a model file that specifies processes, threads in a workflow
- **Filebench OLTP “personality” is a model to emulate an Oracle database server generating I/Os under an online transaction processing workload**
  - Other personalities include fileserver, webserver, etc.
- **Used two different filesystems (UFS and ZFS)**
  - To study what effect a filesystem can have on I/O characteristics
  - Ran filebench on Solaris 5.11 (build 55)

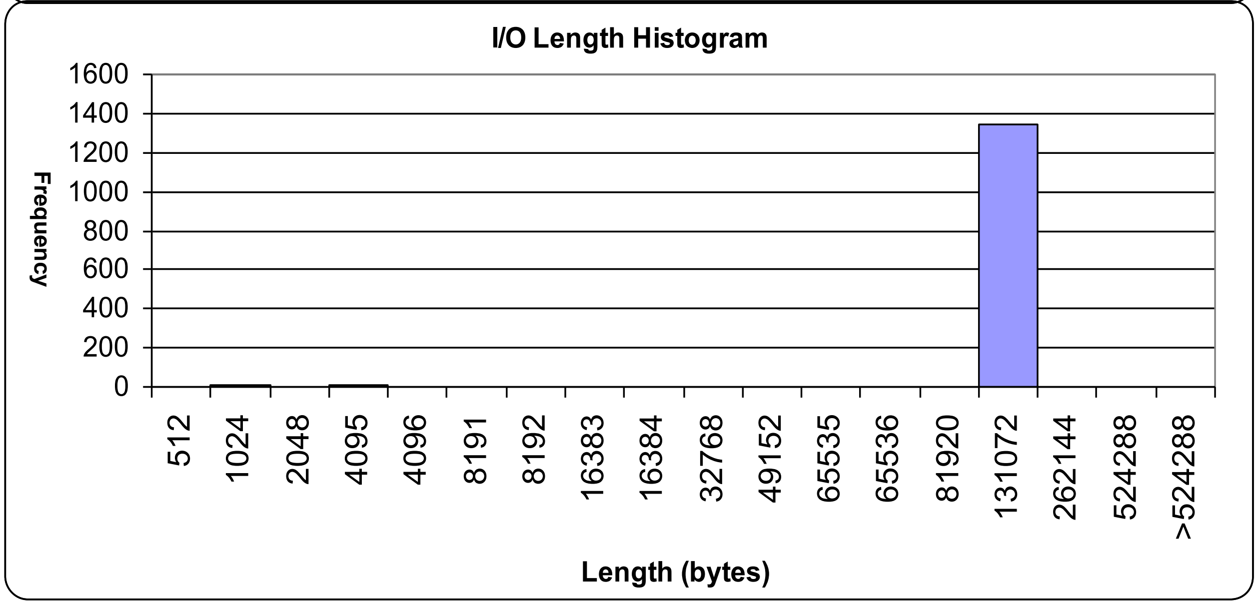
I/O Length
Filebench OLTP

UFS



- 4K and 8K I/O transformed into 128K by ZFS?

ZFS



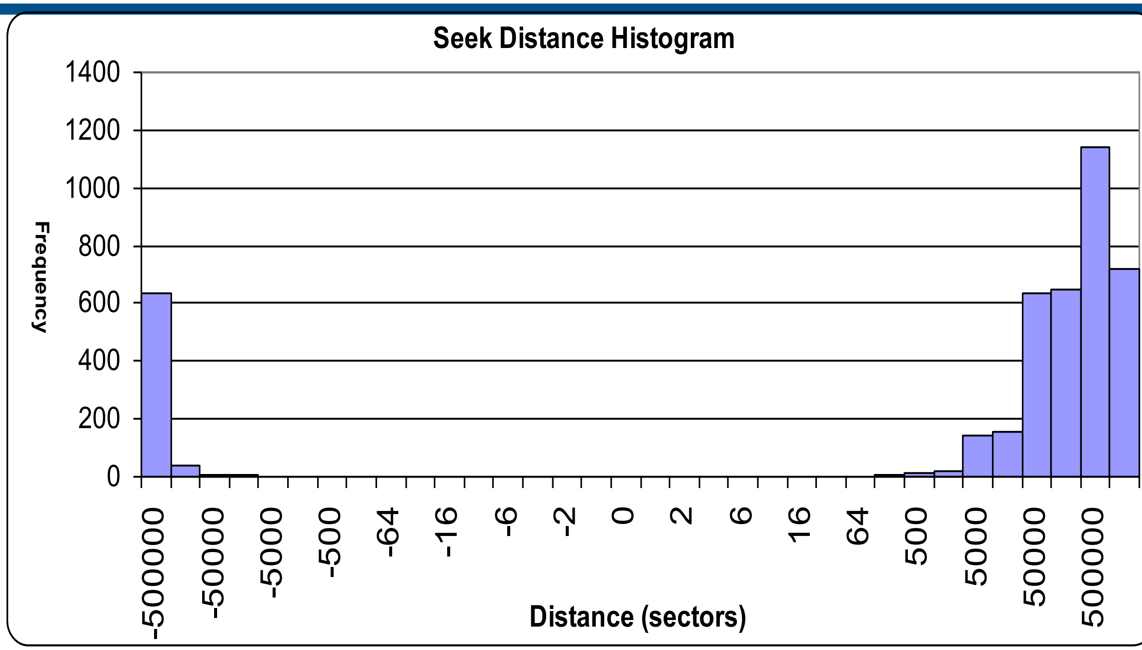
vmware

Seek Distance

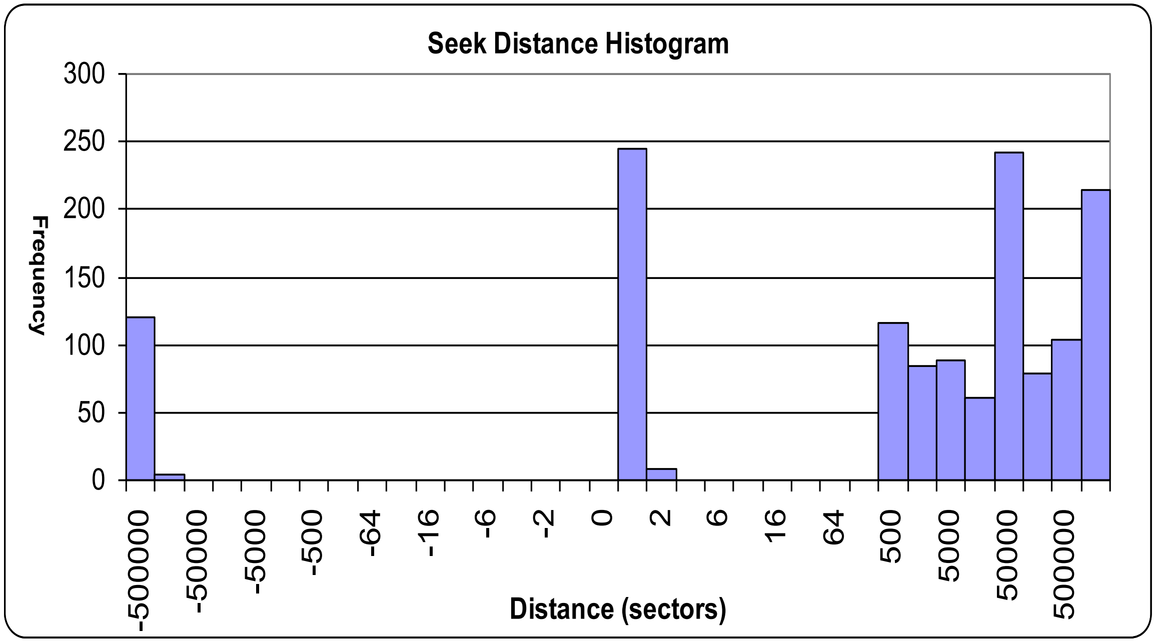
Filebench OLTP

- Seek distance: a measure of ***sequentiality*** versus ***randomness*** in a workload
- Somehow a random workload is transformed into a sequential one by ZFS!
- More details needed ...

UFS



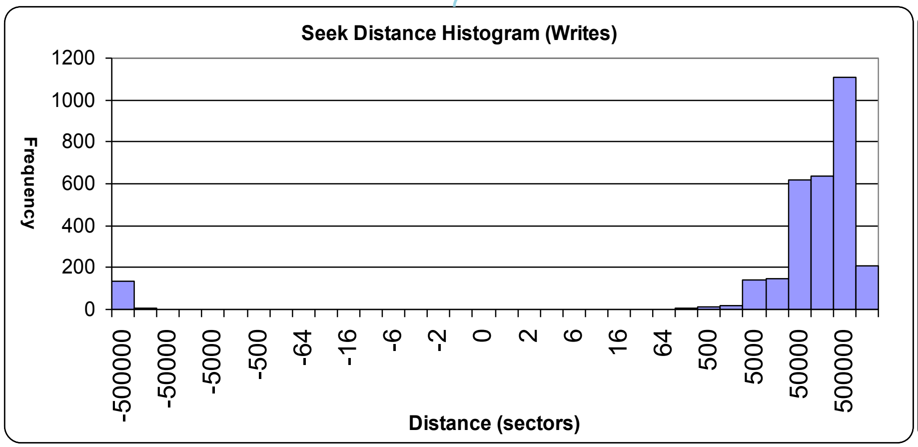
ZFS

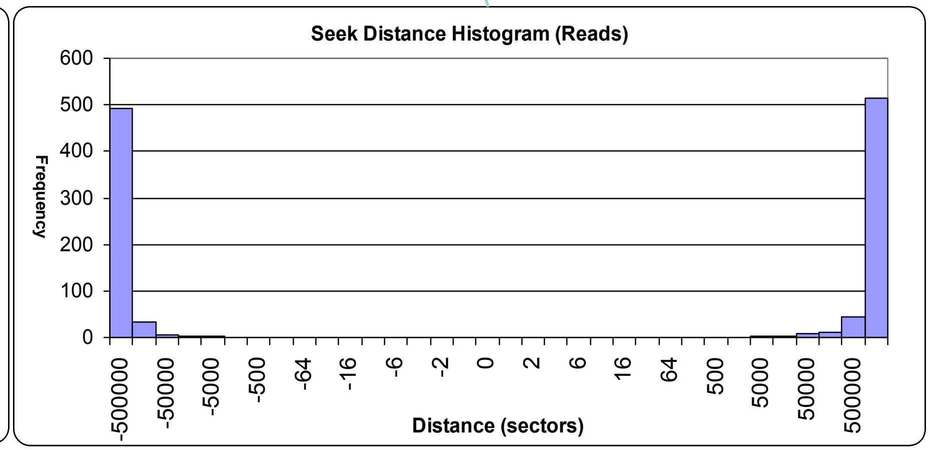


Seek Distance
Filebench OLTP—More Detailed

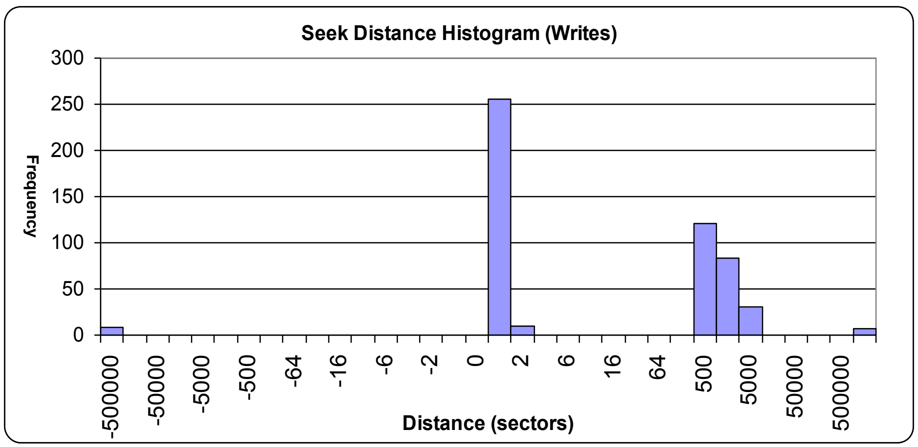
Split out reads & writes

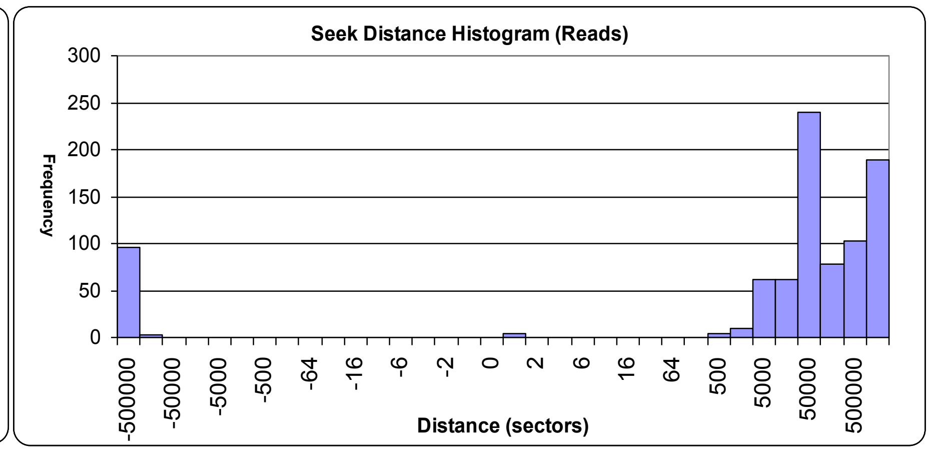
UFS





ZFS





- **Transformation from Random to Sequential: primarily for Writes**
- **Reads: Seek distance is reduced (look at histogram shape & scales)**

Filebench OLTP

Summary

- **So, what have we learnt about Filebench OLTP?**
  - I/O is primarily 4K but 8K isn't uncommon (~30%)
  - Access pattern is mostly random
    - Reads are entirely random
    - Writes do have a forward-leaning pattern
  - ZFS is able to transform random Writes into sequential:
    - Aggressive I/O scheduling
    - Copy-on-write (COW) technique (blocks on disk not modified in place)
    - Changes to blocks from app writes are written to alternate locations
    - Stream otherwise random data writes to a sequential pattern on disk
- **Performed this detailed analysis in just a few minutes**

vscsiStats

vscsiStats -l

```
Virtual Machine worldGroupID: 17981, Virtual Machine Display Name: MSSQL2005 {
   Virtual SCSI Disk handleID: 8192
   Virtual SCSI Disk handleID: 8193
   Virtual SCSI Disk handleID: 8194
   Virtual SCSI Disk handleID: 8195
}
Virtual Machine worldGroupID: 1181739, Virtual Machine Display Name: sles10-vm1
{
   Virtual SCSI Disk handleID: 8202
   Virtual SCSI Disk handleID: 8203
   Virtual SCSI Disk handleID: 8204
   Virtual SCSI Disk handleID: 8205
   Virtual SCSI Disk handleID: 8206
   Virtual SCSI Disk handleID: 8207
}
```

World group leader id

Virtual Machine Name

Virtual scsi disk handle ids - unique across virtual machines

vscsiStats – latency histogram

 # vscsiStats -p latency -w 118739 -i 8205

```
Histogram: latency of Write IOs in Microseconds (us) for virtual
 machine worldGroupID : 1181739, virtual disk handleID : 8205 {
 min : 472
 max : 322869
 mean : 15552
 count : 161493
    {
       0                    (<=              1)
       0                    (<=             10)
       0                    (<=            100)
       9                    (<=            500)
       10894                (<=           1000)
       40520                (<=           5000)
       26513                (<=          15000)
       62477                (<=          30000)
       20285                (<=          50000)
       790                  (<=         100000)
       5                    (>          100000)
    }
}
```

I/O distribution count

Latency in microseconds

vmware®

vscsiStats – iolength histogram

vscsiStats -p iolength -w 118739 -i 8205

```
Histogram: IO lengths of Write commands for virtual machine
dGroupID : 1181739, virtual disk handleID : 8205 {
 min : 2048
 max : 4096
 mean : 2048
 count : 161493
   {
     0                   (<=           512)
     0                   (<=          1024)
     161486              (<=          2048)
     0                   (<=          4095)
     7                   (<=          4096)
     0                   (<=          8191)
     0                   (<=          8192)
     0                   (<=         16383)
     0                   (<=         16384)
     0                   (<=         32768)
     0                   (<=         49152)
     0                   (<=         65535)
     0                   (<=         65536)
     0                   (<=         81920)
     0                   (<=        131072)
     0                   (<=        262144)
     0                   (<=        524288)
     0                   (>         524288)
   }
}
```

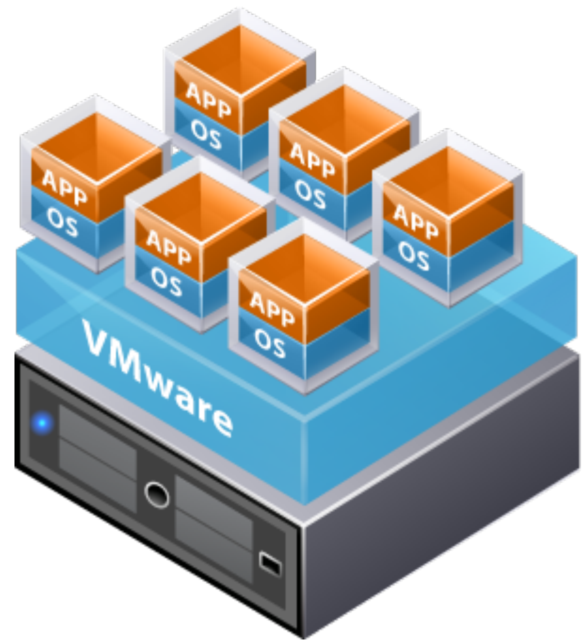
Distribution Count

I/O block size

vmware®

Storage Recommendations

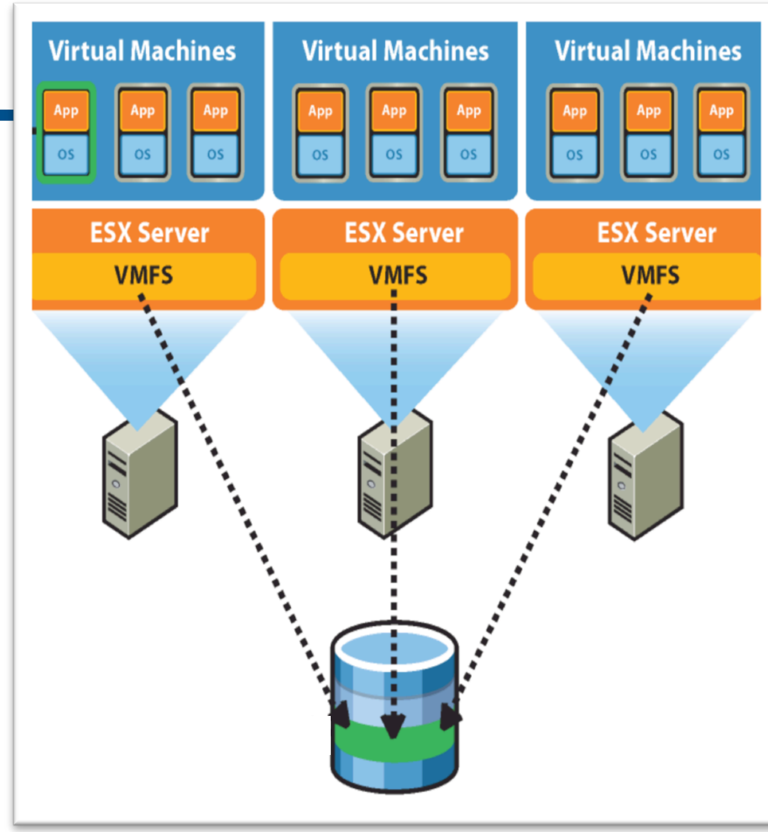
- **The fundamental relationship between consumption and supply has not changed**
  - Spindle count and RAID configuration still rule
  - But host demand is an aggregate of VMs
- **What is the impact of virtual disk consolidation**
  - Full isolation
  - Shared VMFS

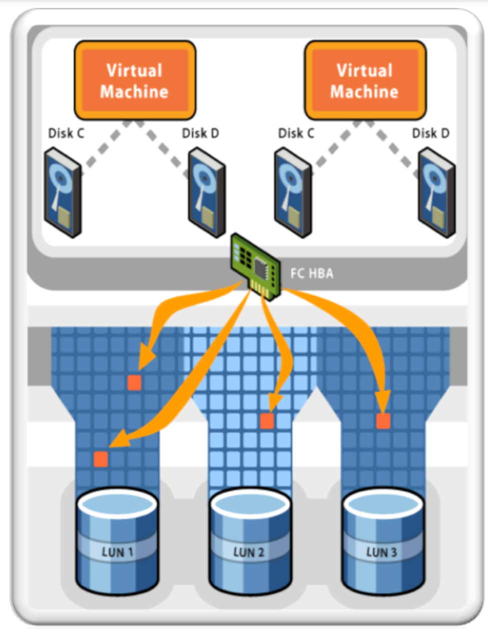


K

Differences in VMs

- **VMware deployments**
  - Large set of physical machines consolidated
  - Diverse set of applications
- **Workload characteristics**
  - Different IO patterns to the same volume, or
  - IO from one app split to different volumes
  - Provisioning operations along with applications (Create VM, Power On VM)
- **Hypervisor and the storage subsystem**
  - Clustered file system locking
  - CPU and virtual device emulation can impact storage performance
- **System setup can affect performance**
  - Partition alignment affects performance.
  - Raw Device Mapping or File system
  - New Hardware Assist technology
  - CPU and memory affinity settings

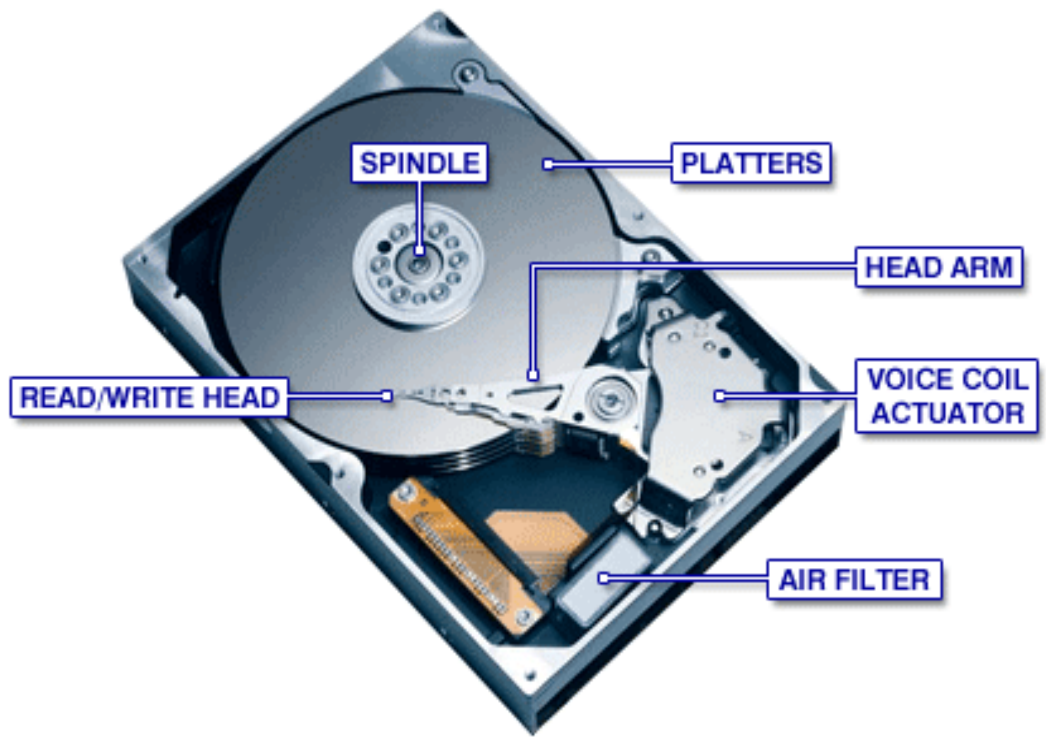




vmware

Disk Fundamentals

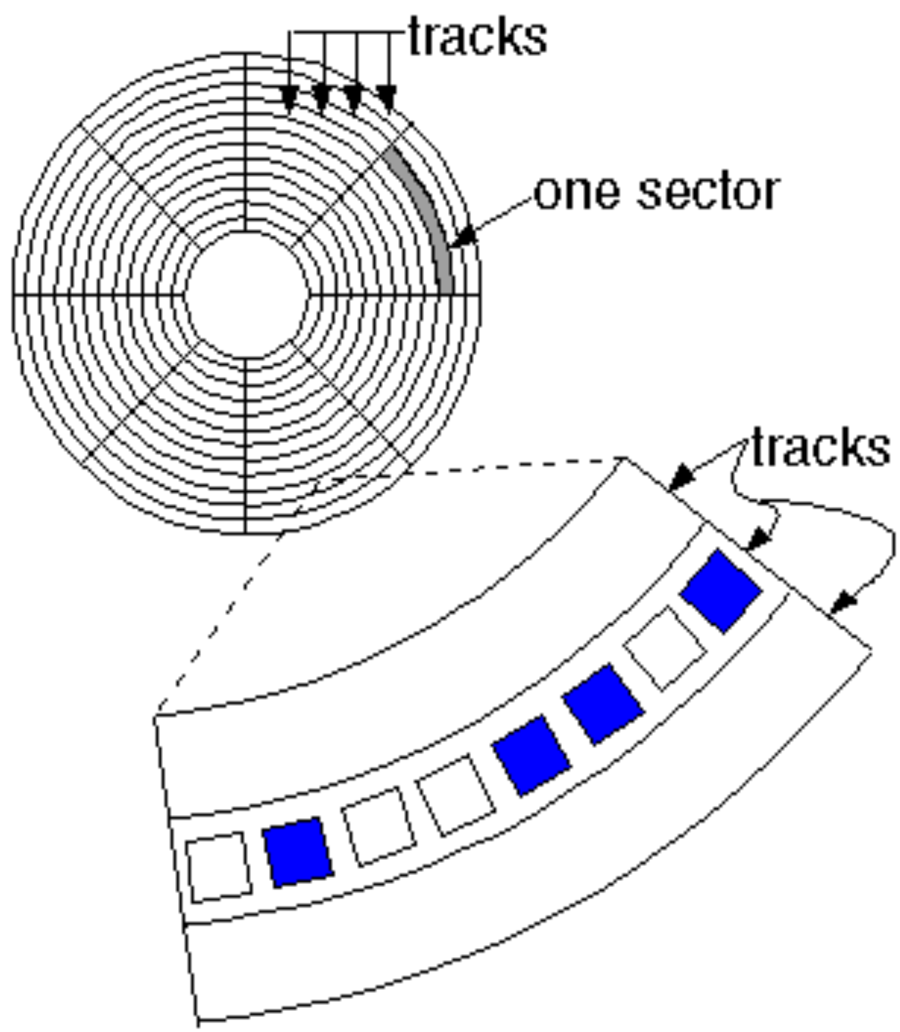
- **Databases are mostly random I/O access patterns**
- **Accesses to disk are dominated by seek/rotate**
  - 10k RPM Disks: 150 IOPS max, ~80 IOPS Nominal
  - 15k RPM Disks: 250 IOPS max, ~120 IOPS Nominal
- **Database Storage Performance is controlled by two primary factors**
  - Size and configuration of cache(s)
  - Number of physical disks at the back-end



Disk Performance

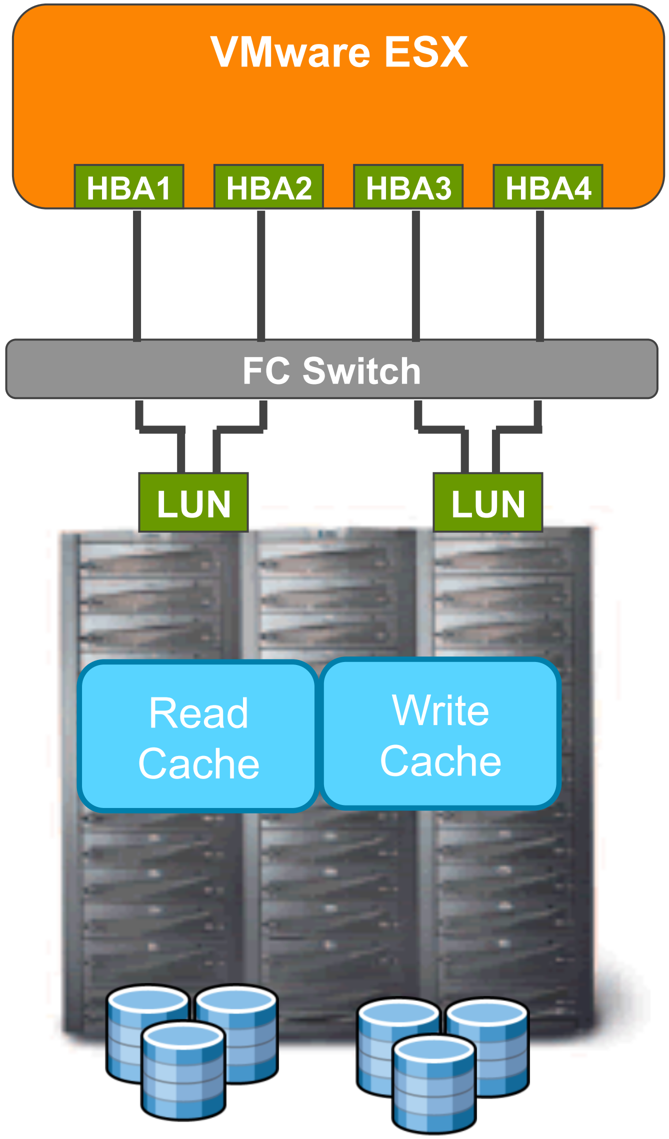
- **Higher sequential performance (bandwidth) on the outer tracks**

From Computer Desktop Encyclopedia
© 1998 The Computer Language Co. Inc.



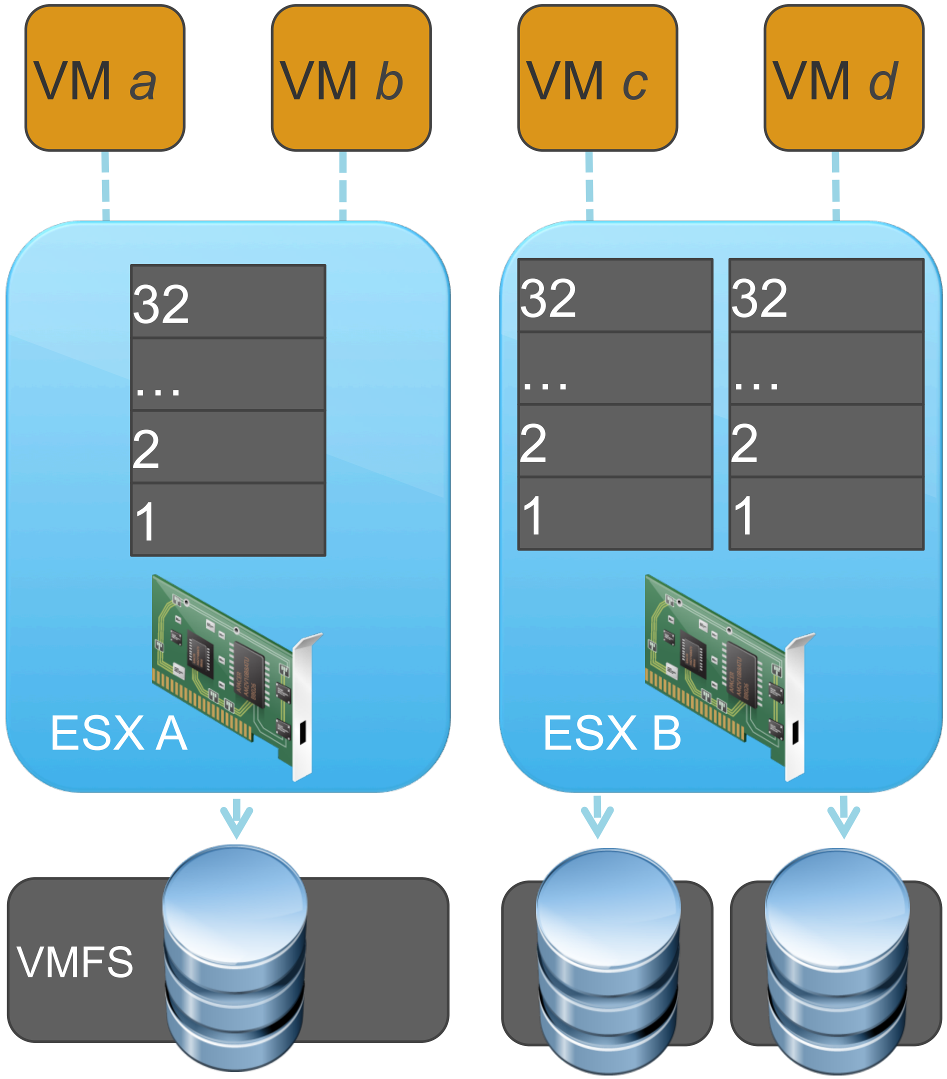
Disk Arrays

- **Lowest level resource is disk**
  - 150 IOPS, 70-150MByte/sec
- **Disks are aggregated into LUNS**
  - Increase performance and availability
- **LUNS can be (should be) cached**
  - Read caches or write caches
  - Write caches hide *wait-for-write*
- **Disk arrays share FC Connections**
  - Typically 200 or 400MBytes/sec

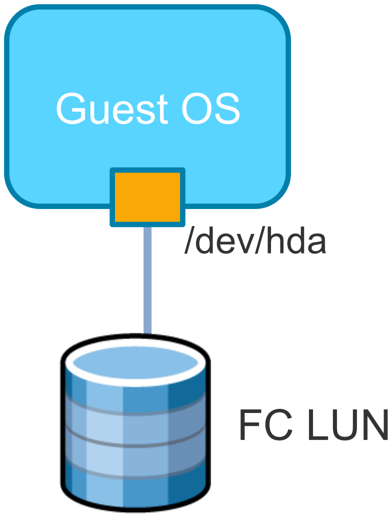


LUN Sizing and Its Impact On Load

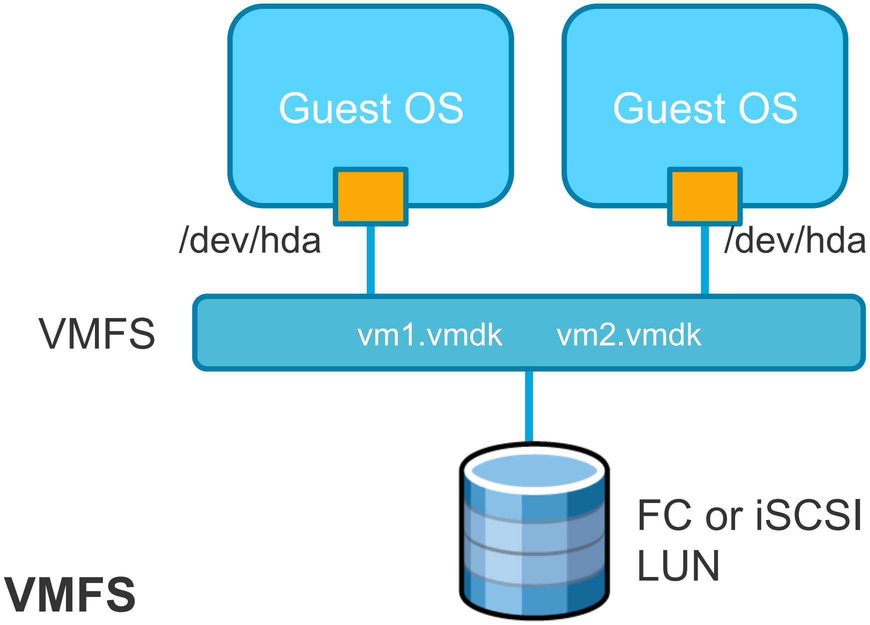
- **In example on the right, ESX B can generate twice as much IO as ESX A**
- **Improved aggregate throughput of multiple LUNs is the primary reason for thinking RDM is faster**
- **Implications for the array**
  - Greater number of smaller LUNs increases burst intensity
    - Many HBA/LUN pairs could be used simultaneously
  - Smaller number of LUNs stabilizes demand
    - Fewer HBA/LUN pairs will be used concurrently



Storage – VMFS or RDM

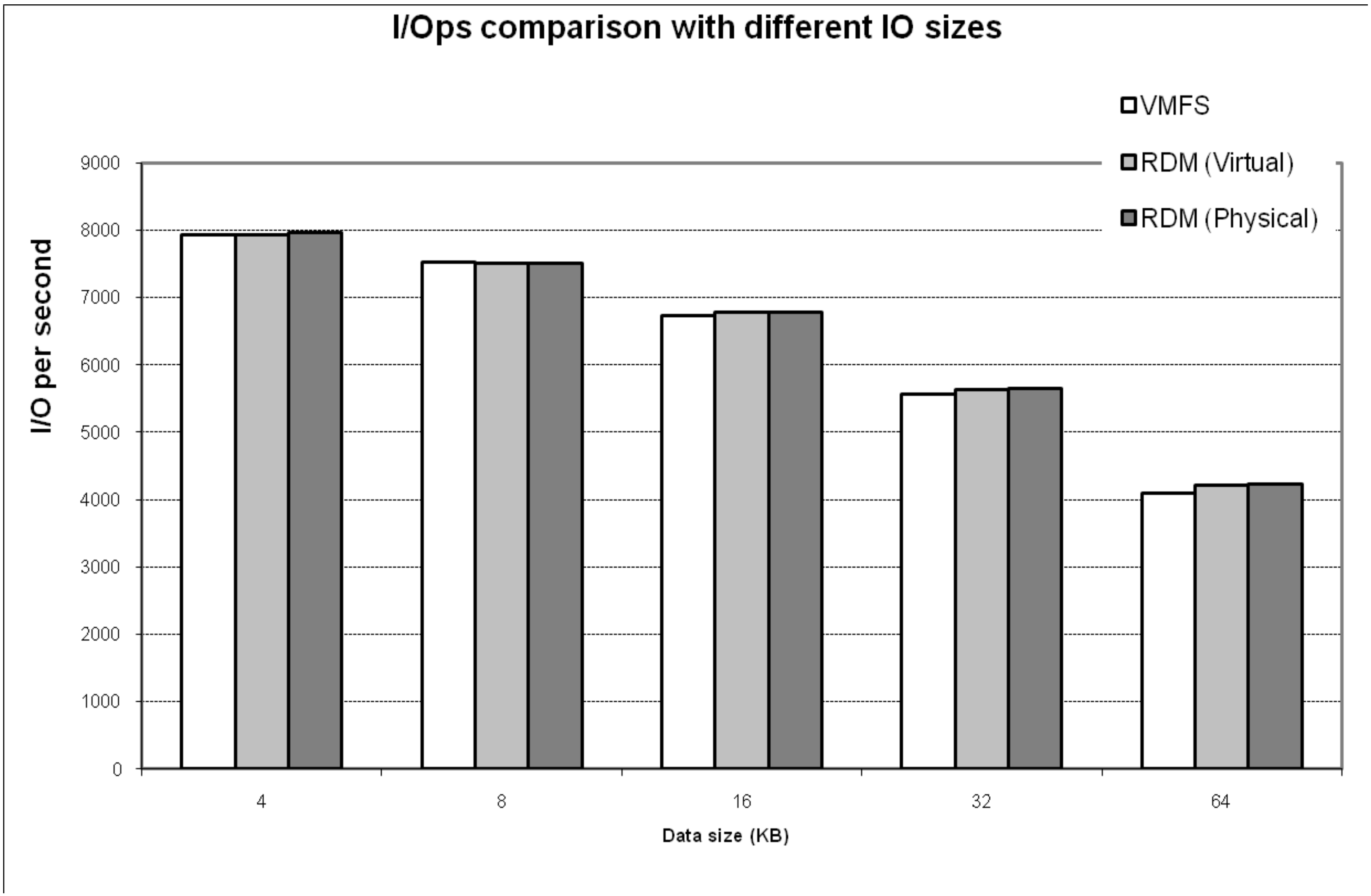


- **RAW**
- **RAW provides direct access to a LUN from within the VM**
- **Allows portability between physical and virtual**
- **RAW means more LUNs**
  - More provisioning time
- **Advanced features still work**

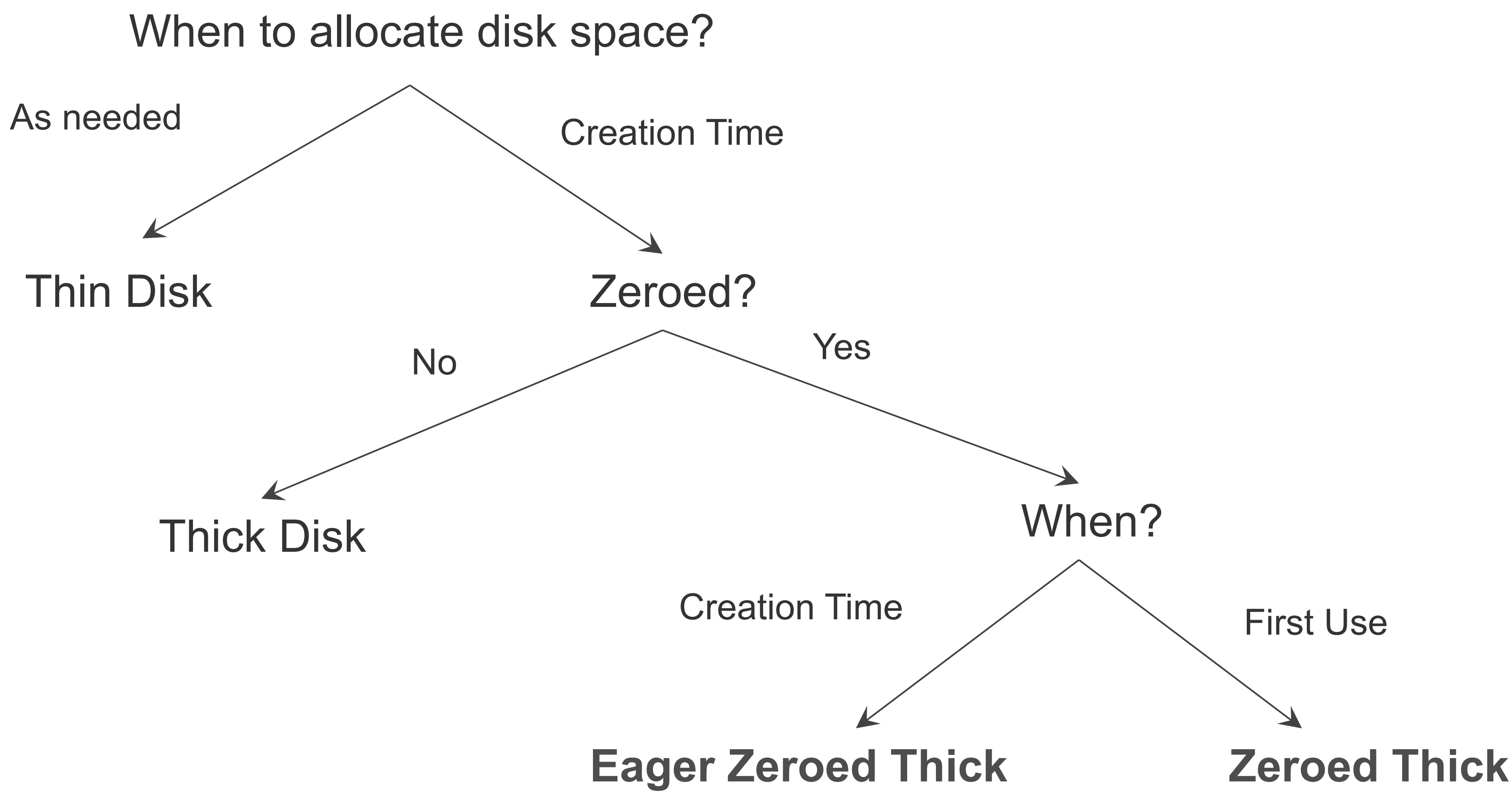


- **VMFS**
- **Easier provisioning**
- **Snapshots, clones possible**
- **Leverage templates and quick provisioning**
- **Scales better with Consolidated Backup**
- **Preferred Method**

VMFS vs. RDM Performance



Creating VM: Disk Type?

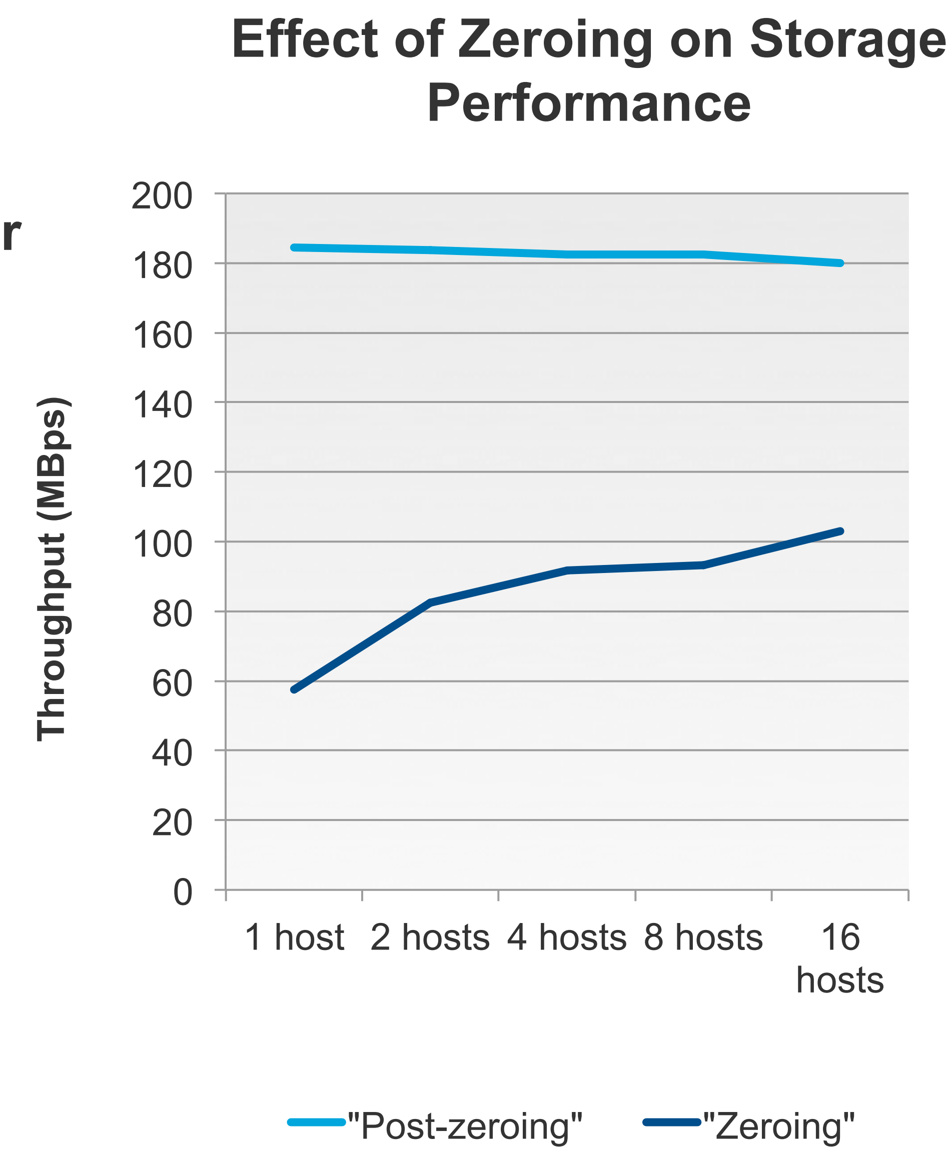


Creating VM: Disk Type?

- **Speed Vs Space**
  - Thin disk is space efficient but higher per IO overhead
  - Thick disk has lower per IO overhead but consumes space
  - Zeroed thick disk pays extra write cost at the first write
  - Eager zeroes thick disk or thick disk gives best performance
  - Use vmkfstool to create or convert
- **RDM Vs VMFS**
  - Physical RDM disables VMotion
  - VMFS performance is close to the RDM

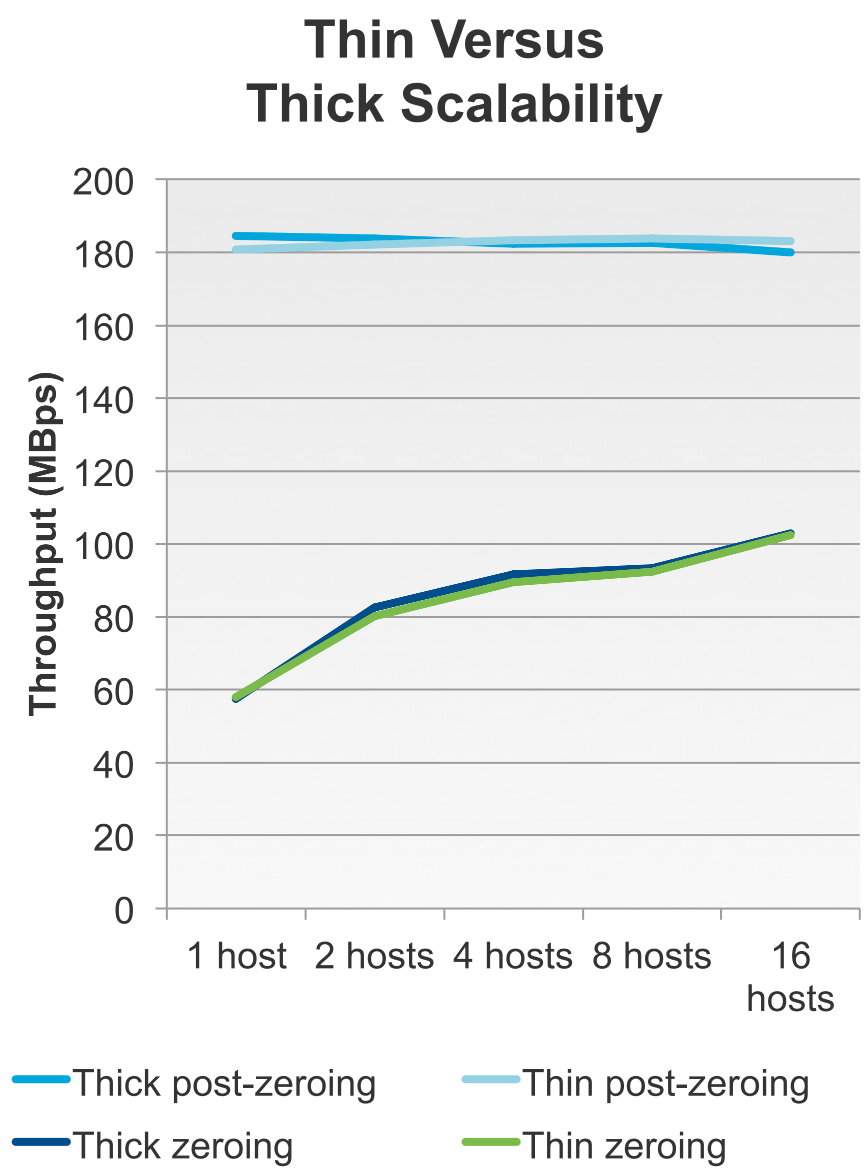
VMDK Lazy Zeroing

- **Default VMDK allocation policy "lazy zeroes" 1M VMFS blocks on first write**
- **Writes on an untouched VMDK incur a penalty**
- **Difference usually not seen in production**
  - But common with benchmarks
- **Zero offload capability in VAAI improves zeroing in supported arrays**



Thin Provisioning Performance

- vSphere introduced thin provisioned VMDKs
- In *theory*, LUN locking during VMDK growth might hurt performance
- In *reality*, zeroing more impactful than locking
- ATS and zero-offloading in VAAI enabled arrays will speed up “first-writes”



Device Paravirtualization (4.0)

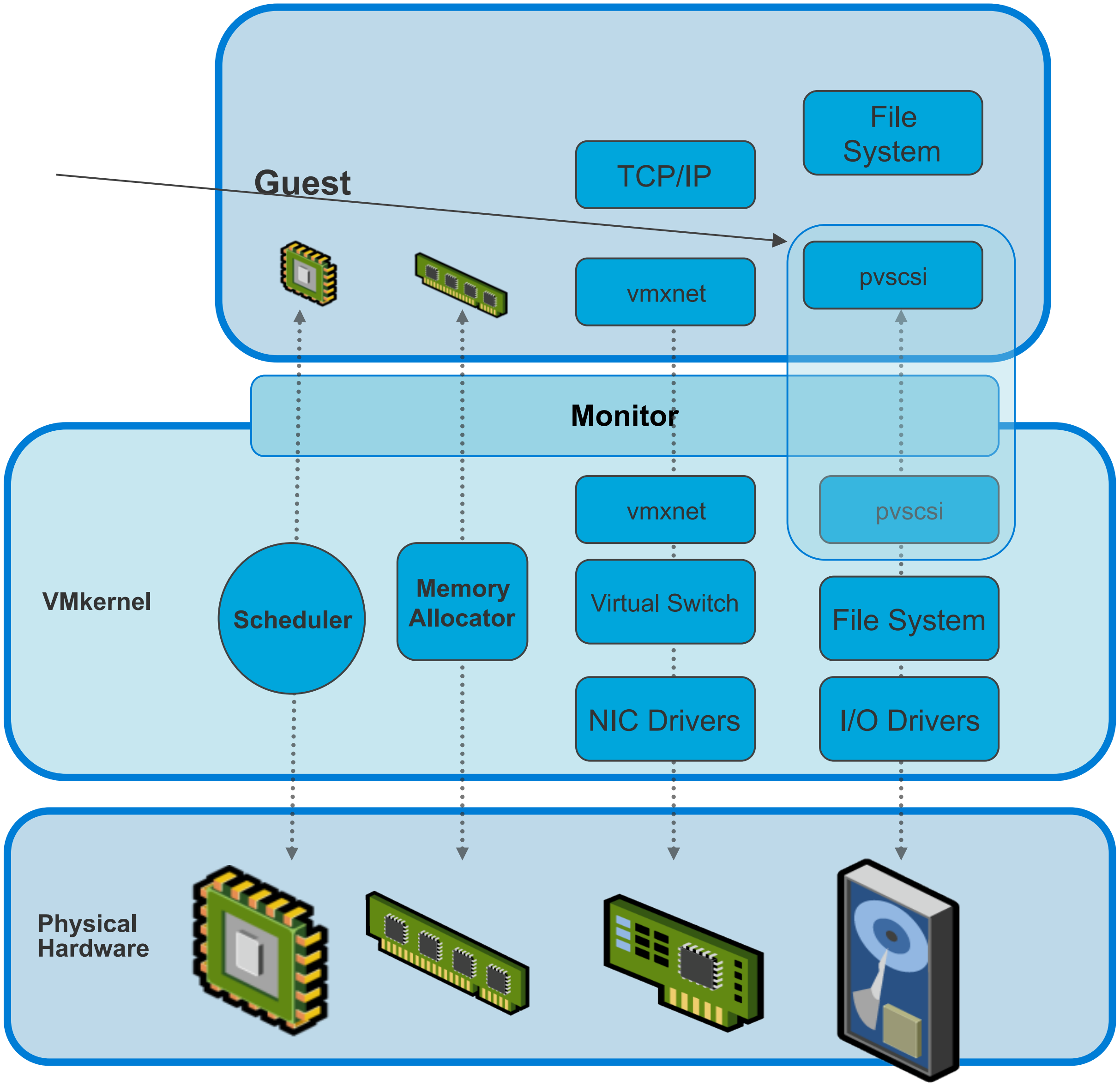
Device Paravirtualization places A high performance virtualization-Aware device driver into the guest

Paravirtualized drivers are more CPU efficient (less CPU over-head for virtualization)

Paravirtualized drivers can also take advantage of HW features, like partial offload (checksum, large-segment)

VMware ESX uses para-virtualized network drivers

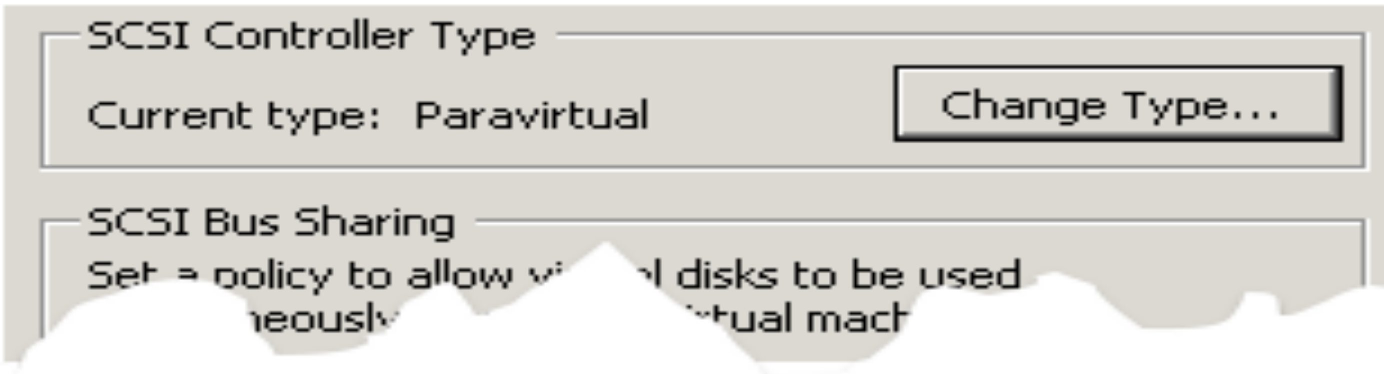
vSphere 4 now provides *pvscsi*



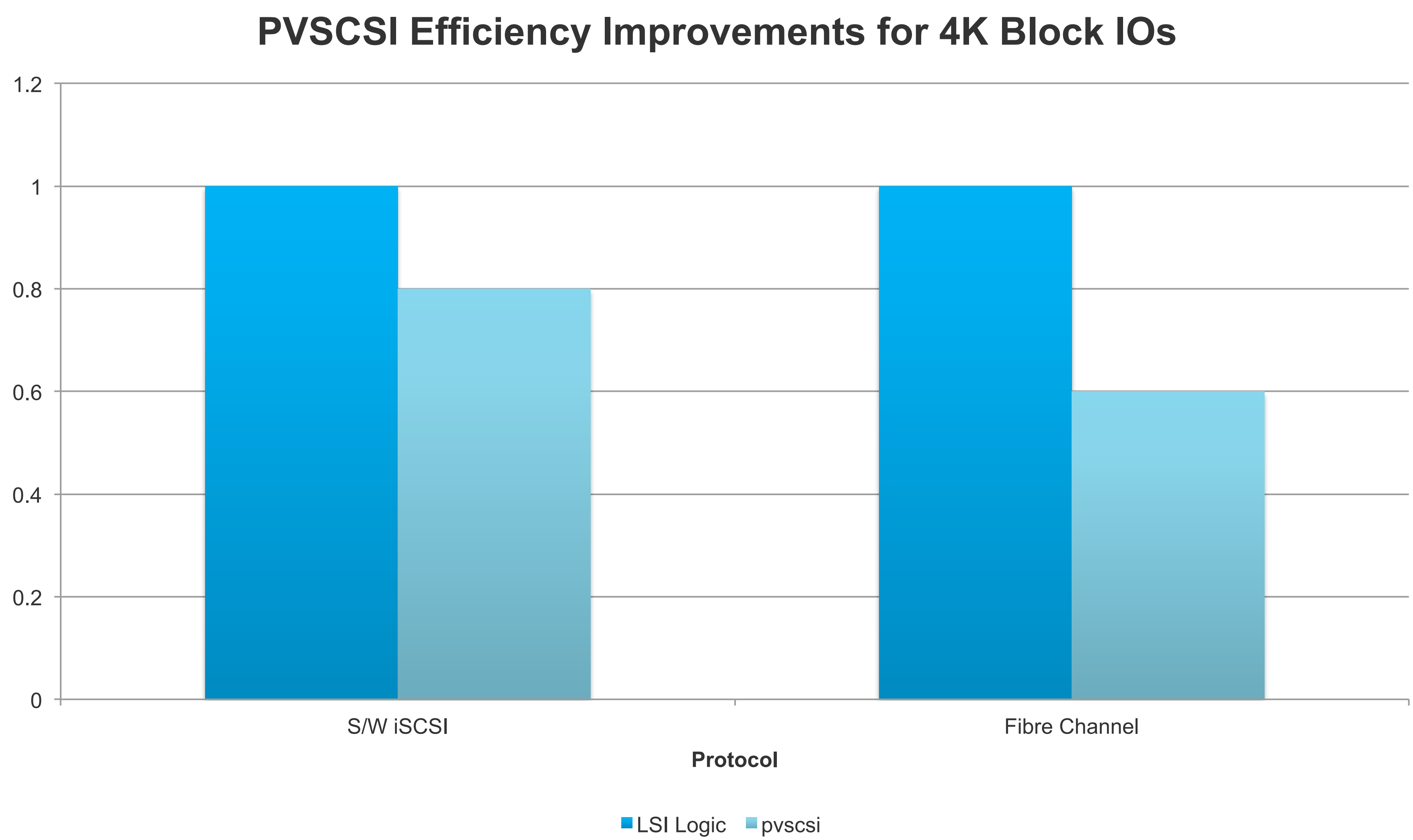
PVSCSI Architecture

- PVSCSI looks like a PCI-E device to the guest OS
- Uses MSI or MSI-X interrupt delivery (instead of legacy INTx) to reduce the cost of interrupt virtualization
- Boot capable
- New Windows/Linux SCSI HBA drivers
- Windows driver uses the Storport driver model
- Exports itself as a Serial Attached SCSI adapter

Enabling the PVSCSI Driver



PVSCSI Efficiency



vmware

Benchmarks for I/O

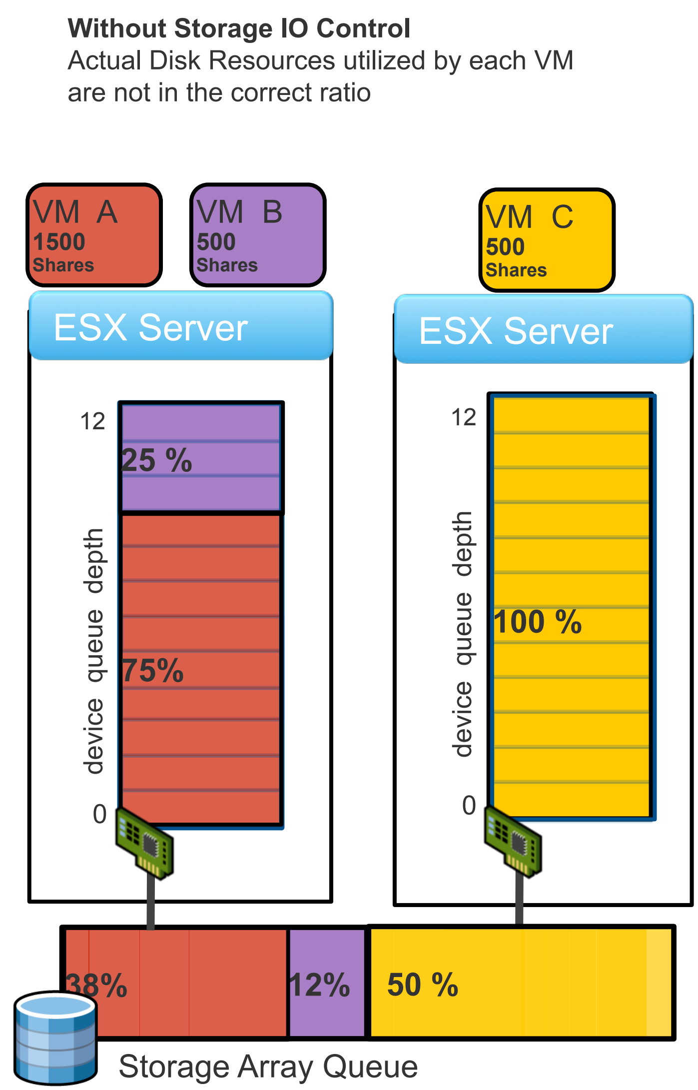
- **Microbenchmarks**
  - Iometer
  - Aiostress
  - Filebench
  - Orion
  - Sqliosim
  - Jetstress

Macrobenchmarks

> TPC-C/E
> MS Exchange
> Oracle
> SQLserver
> Etc…

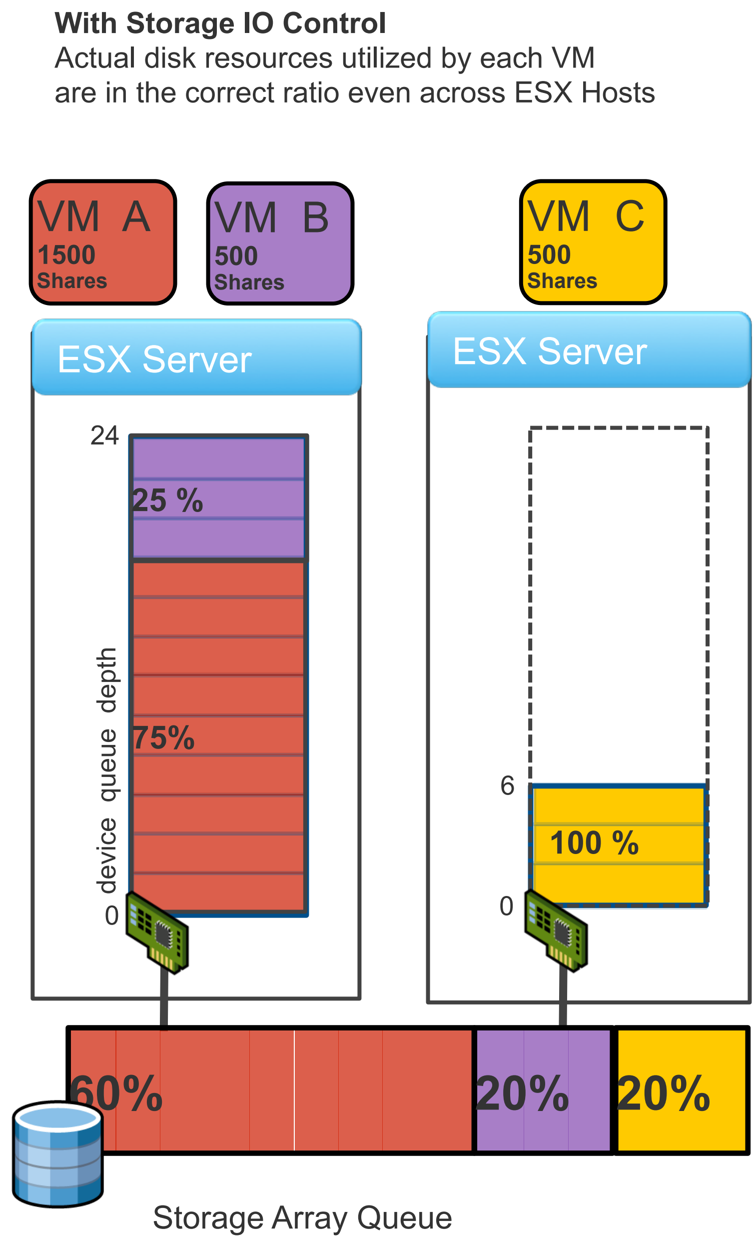
Storage Contention Problems

- **In vSphere 4, an isolated VM can dominate a shared LUN**
  - IO shares determine access to LUN relative to other VMs on the same host
  - A VM can get uncontested access to the device queue negatively affecting VMs that share the LUN but are running on other ESX hosts
  - Regardless of shares, VMs on the same host contend for one queue
- **Existing storage resource management controls only affects VMs on a single host**



Storage Contention Solution: Storage IO Control

- **SIOC calculates data store latency to identify storage contention**
  - Latency is normalized, averaged across virtual machines
  - IO size and IOPS included
- **SIOC enforces fairness when data store latency crosses threshold**
  - Default of 30 ms
  - Sustained for four seconds
  - Fairness enforced by limiting VMs access to queue slots
- **Can have small detrimental effect on throughput at LUN**

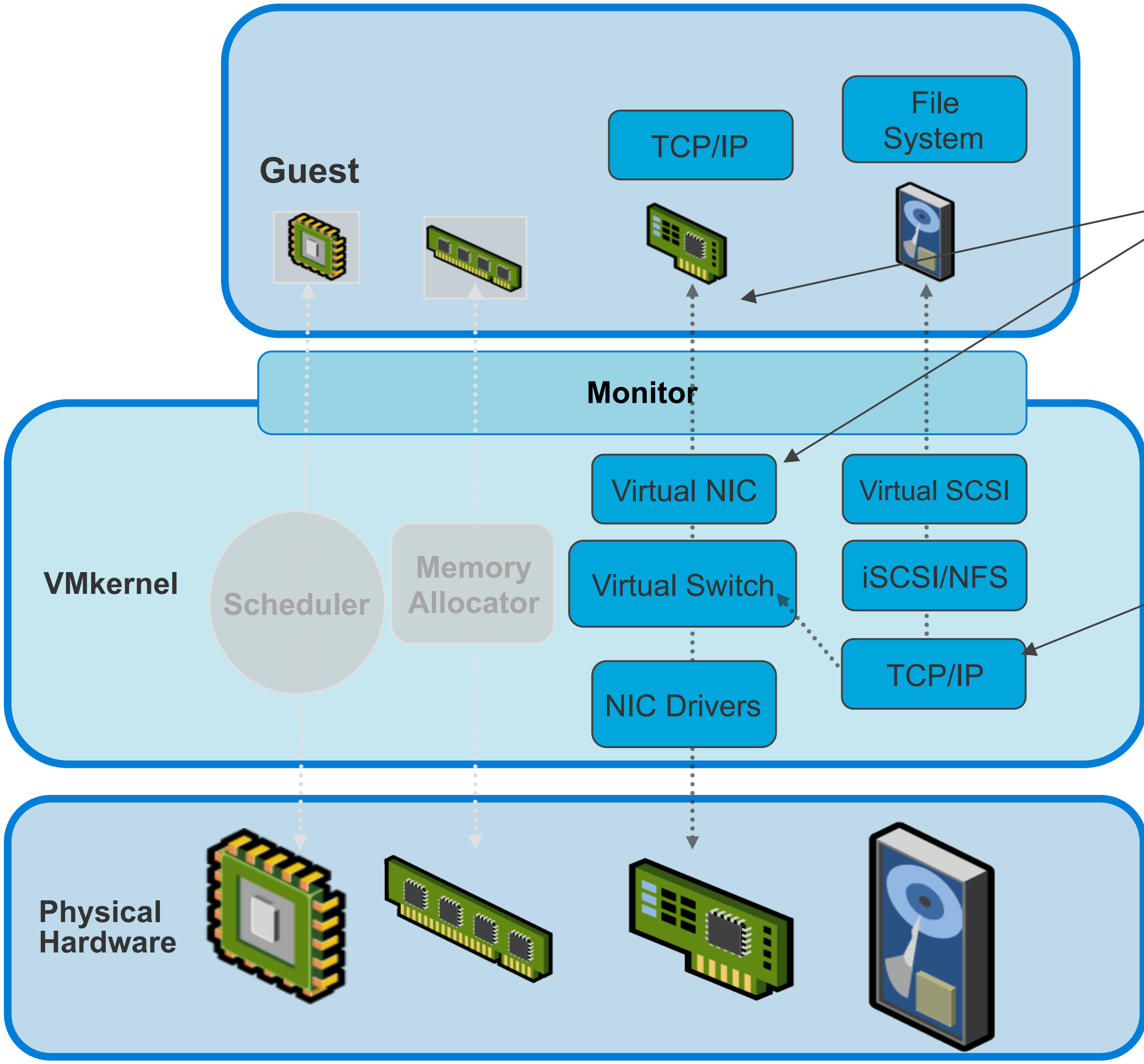


Notes and Caveats on SIOC

- **SIOC is not a storage panacea**
  - Important VMs can be protected
  - Poorly performing storage remains poorly performing, and the infrastructure suffers!
- **SIOC trades throughput for latency**
  - The feature is enabled when latency crosses a certain threshold, implying a storage bottleneck
  - Throughput is throttled for less performance critical VMs to provide fast access to high priority VMs
- **SIOC may make some of your happy application owners unhappy**
  - Your current configuration may allow storage hogs to lock their neighbors out of the array
  - When you enable SIOC, these “bad neighbors” will be throttled

NETWORKING

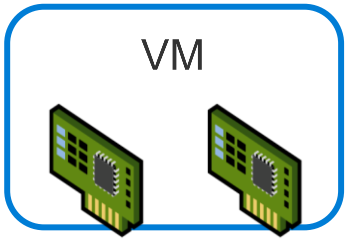
VMware ESX Networking Architecture



Virtual NIC Device
- **Full Virt: e1000g**
- **Paravirt: vmxnet2,**
- **vSphere adds vmxnet3**

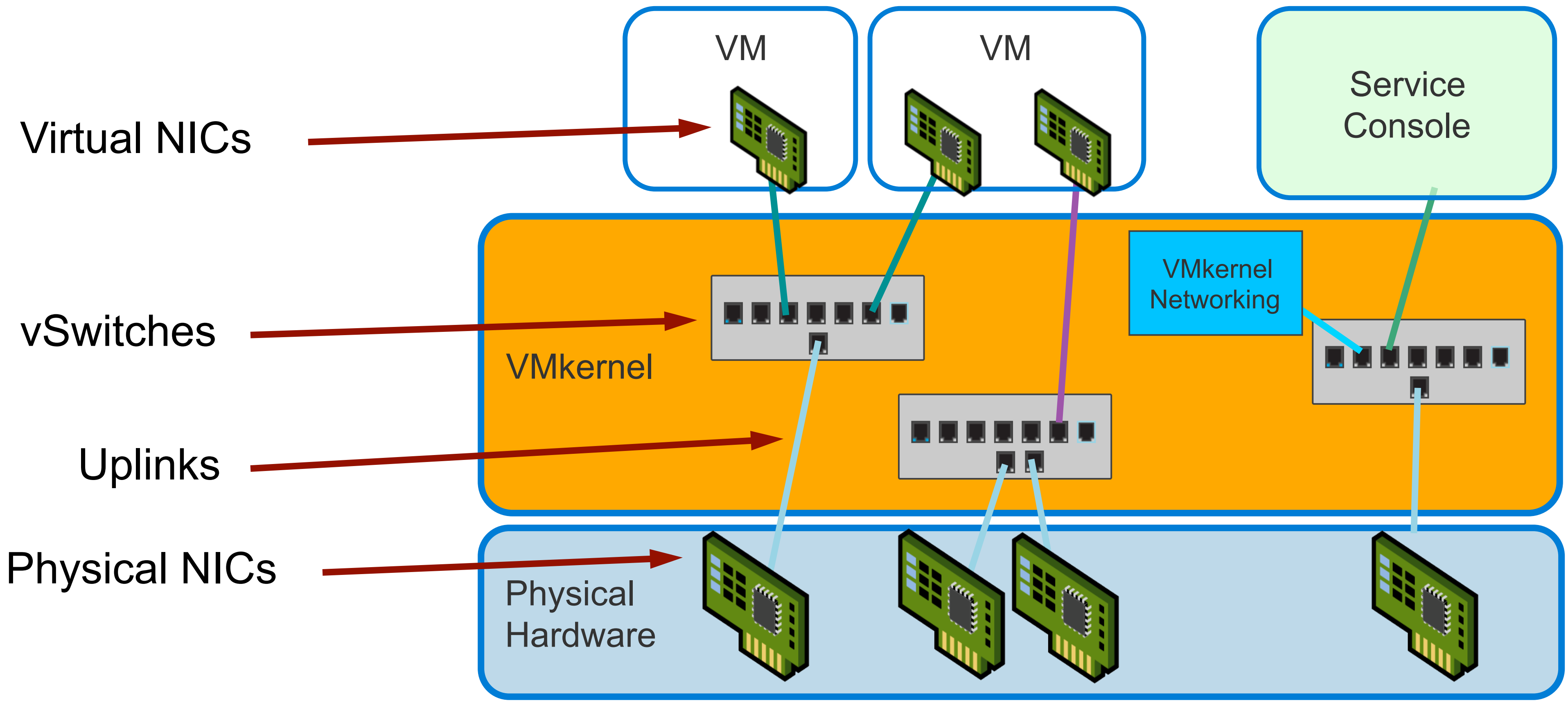
TCP/IP Stack
- **For vMotion,iSCSI and NFS**
- **New v2 Stack for vSphere**

VM Network I/O Virtualization



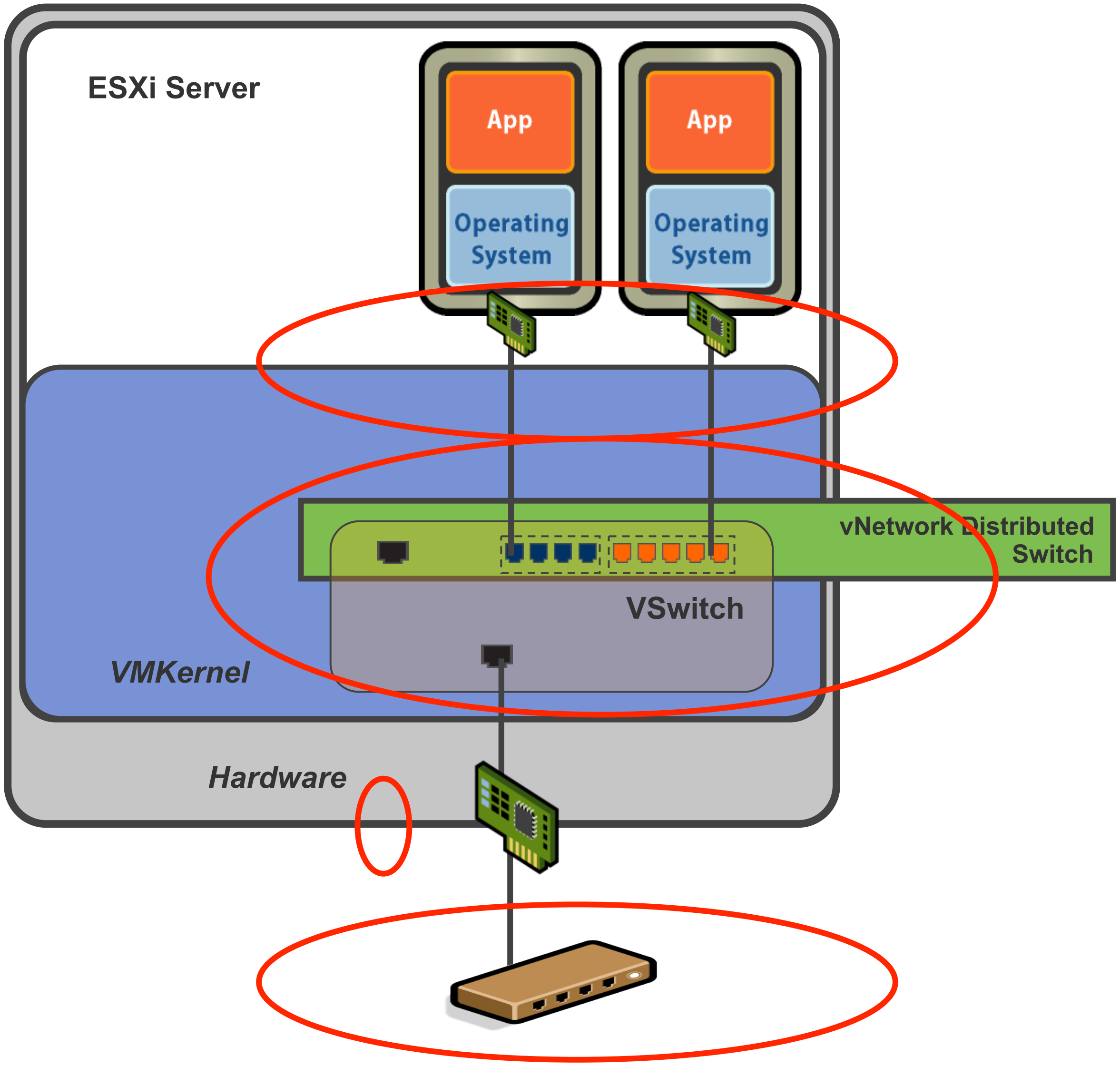
- **Guest OS sees a virtual NIC**
  - AMD Lance, Intel e1000, or VMware vmxnet
    - Virtual devices acting just like physical one (except vmxnet)
  - Each virtual NIC has a unique MAC address
  - Up to 4 virtual NICs per VM
- **Virtual NIC enhancements**
  - No physical crystal limiting transmit/receive
  - Disallow promiscuous mode
  - Disallow MAC address changes by the OS
  - Disallow forged source MAC transmits

ESX Server Networking I/O



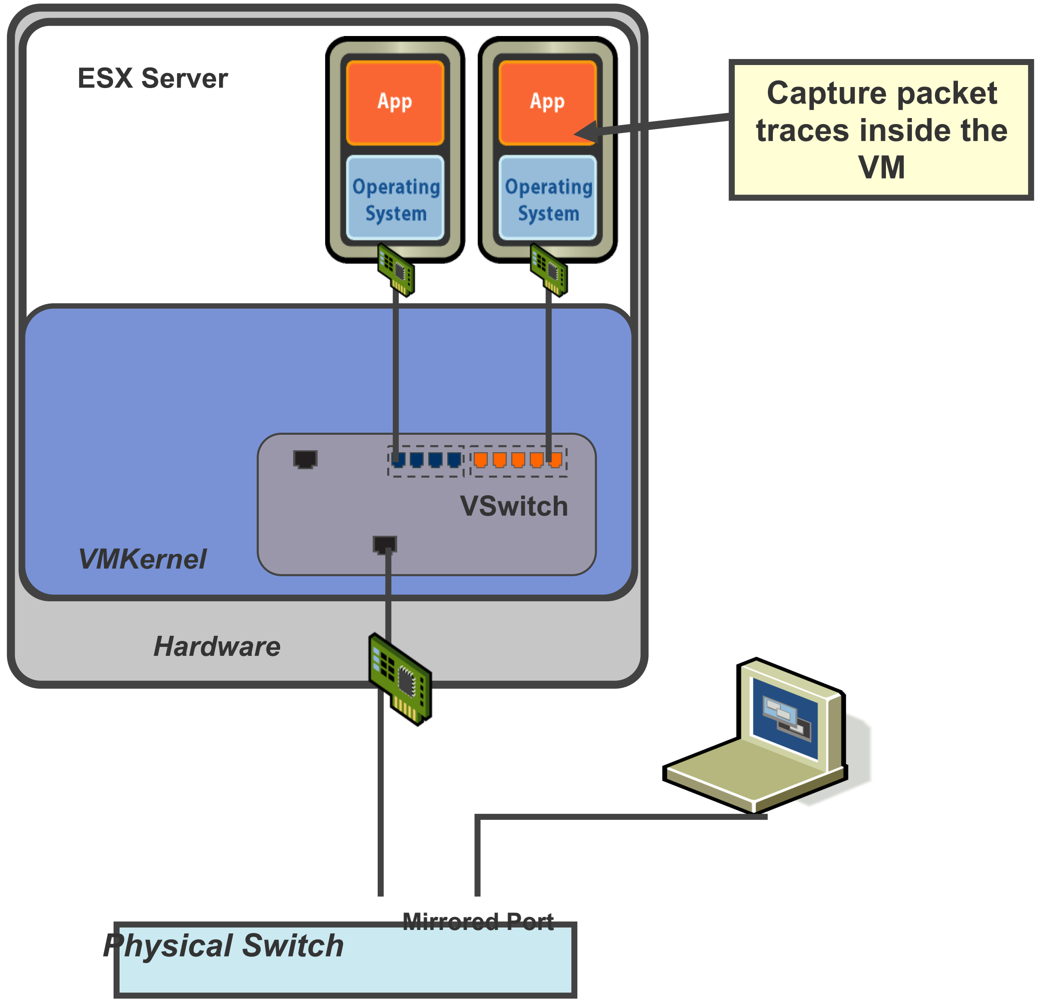
Troubleshooting Networking

- **Troubleshoot one component at a time**
  - Physical NICs
  - vNetwork Distributed Switch
  - Virtual NICs
  - Physical Network
- **Tools for troubleshooting**
  - vSphere Client (aka VI)
  - Command Line Utilities
    - vSphere CLI
  - Third party tools
    - Ping and traceroute
    - Traffic sniffers and Protocol Analyzers
      - Wireshark
  - Logs



Sniffing For Trouble

- **Sniff for packets at different layers for isolation**
  - Physical Switch Port Level (SPAN)
  - VM Level (Promiscuous mode)
- **Look for**
  - Lost Packets
  - Large number of packet retransmissions
  - Anomalies reported by protocol analyzers like Wireshark etc.
- **Look for patterns**
  - Are packets of a certain type causing problems?
  - Are packets of a certain size causing problems?



Getting Information about the vnic i/o

Output of esxtop/resxtop

```
11:45:05pm up 5 min, 63 worlds; CPU load average: 0.00, 0.04, 0.00

   PORT-ID              USED-BY  TEAM-PNIC        DNAME   PKTTX/s  MbTX/s   PKTRX/s  MbRX/s %DRPTX %DRPRX
  16777217               vmnic0          -     vSwitch0      1.98    0.00     15.05    0.01   0.00   0.00
  16777218                0:NCP          -     vSwitch0      0.00    0.00      0.00    0.00   0.00   0.00
  16777219                0:CDP          -     vSwitch0      0.00    0.00      0.00    0.00   0.00   0.00
  16777220             0:vswif0     vmnic0     vSwitch0      1.98    0.00     14.26    0.01   0.00   0.00
  33554433               vmnic3          -     vSwitch1      0.00    0.00     13.27    0.01   0.00   0.00
  33554434                0:NCP          -     vSwitch1      0.00    0.00      0.00    0.00   0.00   0.00
  33554435               vmnic4          -     vSwitch1   1597.13    1.19   1610.20    1.20   0.00   0.00
  33554436                0:CDP          -     vSwitch1      0.00    0.00      0.00    0.00   0.00   0.00
  33554437            0:vswif10     vmnic3     vSwitch1      0.00    0.00     12.28    0.01   0.00   0.00
  33554438 0:vmk-tcpip-10.17.41     vmnic4     vSwitch1   1597.13    1.19   1609.41    1.20   0.00   0.00
```

Output of esxcfg-info

```
\==+Port :
   |----Port Id...............................33554438
   |----World Leader..........................0
   |----Client Name...........................vmk-tcpip-10.17.41.72
   |----MAC Addr..............................00:50:56:7c:31:61
   |----Blocked...............................false
   |----Type..................................Tcp/Ip
   |----Portgroup Name........................vmkPG
   \==+Stats :
      |----Packets Tx Ok......................84489
      |----Bytes Tx Ok........................8281172
      |----Dropped Tx.........................0
      |----Packets TSO Tx Ok..................0
      |----Bytes TSO Tx Ok....................0
      |----Dropped TSO Tx.....................0
      |----Packets SW TSO Tx..................0
      |----Dropped SW TSO Tx..................0
      |----Packets Zero Copy Tx Ok............0
      |----Packets Rx Ok......................113655
      |----Bytes Rx Ok........................10153132
```

Look for Rx/Tx information for the vNIC you are interested in

Real time traffic information

Search for the port ID of the vNIC in the esxcfg-info output

Cumulative Traffic Information

Check the physical NIC

- **Check that the right uplinks are connected**
  - Use vSphere client or esxcfg-vswitch –l
- **Check the Rx/Tx counters of the physical nic using esxcfg-info or resxtop**
- **Check connected physical port**
  - Use Network Hint or CDP

```
\==+Port :
   |----Port Id..................................33554435
   |----World Leader.............................0
   |----Client Name..............................vmnic4
   |----MAC Addr.................................00:00:00:00:00:00
   |----Blocked..................................false
   |----Type.....................................Pnic
   \==+Stats :
      |----Packets Tx Ok.........................84488
      |----Bytes Tx Ok...........................8281112
      |----Dropped Tx............................0
      |----Packets TSO Tx Ok.....................0
      |----Bytes TSO Tx Ok.......................0
      |----Dropped TSO Tx........................0
      |----Packets SW TSO Tx.....................0
      |----Dropped SW TSO Tx.....................0
      |----Packets Zero Copy Tx Ok...............0
      |----Packets Rx Ok.........................118277
      |----Bytes Rx Ok...........................10538216
      |----Dropped Rx............................0
      |----Dropped TSO Rx........................0
      |----Packets SW TSO Rx.....................0
      |----Dropped SW TSO Rx.....................0
      |----Actions...............................0
      |----Uplink Rx Packets.....................0
      |----Pks Billed............................0
      |----Dropped Tx Due to Page Absent.........0
      |----Dropped Rx Due to Page Absent.........0
```

Information about Uplink Port (vmnic4)

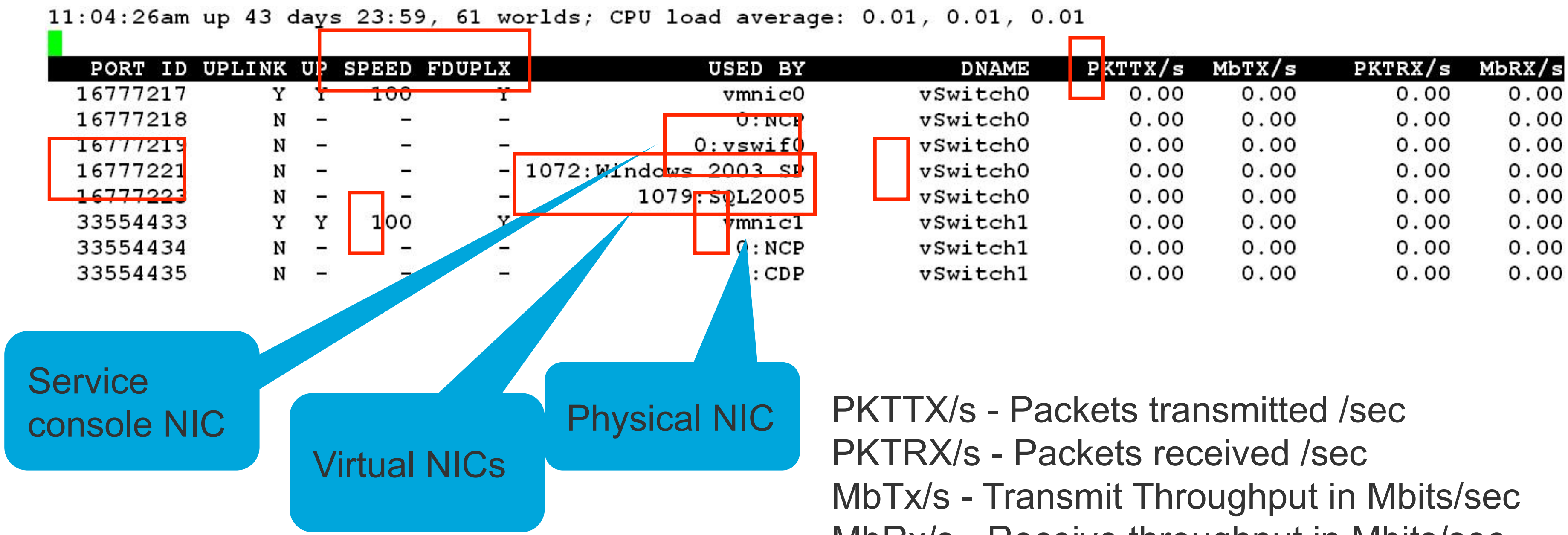
VI Client Networking Statistics

- **Mostly high-level statistics**
  - Bandwidth
    - KBps transmitted, received
    - Network usage (KBps): sum of TX, RX over all NICs
  - Operations/s
    - Network packets received during sampling interval (real-time: 20s)
    - Network packets transmitted during sampling interval
- **Per-adapter and aggregated statistics**

Esxtop Networking Statistics

- **Bandwidth**
  - Receive (MbRX/s), Transmit (MbRX/s)
- **Operations/s**
  - Receive (PKTRX/s), Transmit (PKTTX/s)
- **Configuration info**
  - Duplex (FDUPLX), speed (SPEED)
- **Errors**
  - Packets dropped during transmit (%DRPTX), receive (%DRPRX)

esxtop network screen (n)



PKTTX/s - Packets transmitted /sec
PKTRX/s - Packets received /sec
MbTx/s - Transmit Throughput in Mbits/sec
MbRx/s - Receive throughput in Mbits/sec

Port ID: every entity is attached to a port on the virtual switch
DNAME - switch where the port belongs to

Multicast/Broadcast stats

Multicast/Broadcast stats are new for 4.1

PORT-ID	USED-BY	TEAM-PNIC	DNAME	PKTTXMUL/s	PKTRXMUL/s	PKTTXBRD/s	PKTRXBRD/s
16777217	Management	n/a	vSwitch0	0.00	0.00	0.00	0.00
16777218	vmnic0	-	vSwitch0	0.00	0.39	0.00	0.00
16777219	vmk0	vmnic0	vSwitch0	0.00	0.00	0.00	0.00
33554433	Management	n/a	vswitch_global	0.00	0.00	0.00	0.00
33554434	vmnic2	-	vswitch_global	0.00	0.58	0.00	0.00

PKTTXMUL/s – Multicast packets transmitted per second
PKTRXMUL/s – Multicast packets received per second

PKTTXBRD/s – Broadcast packets transmitted per second
PKTRXBRD/s – Broadcast packets received per second

Platform Optimization: Network

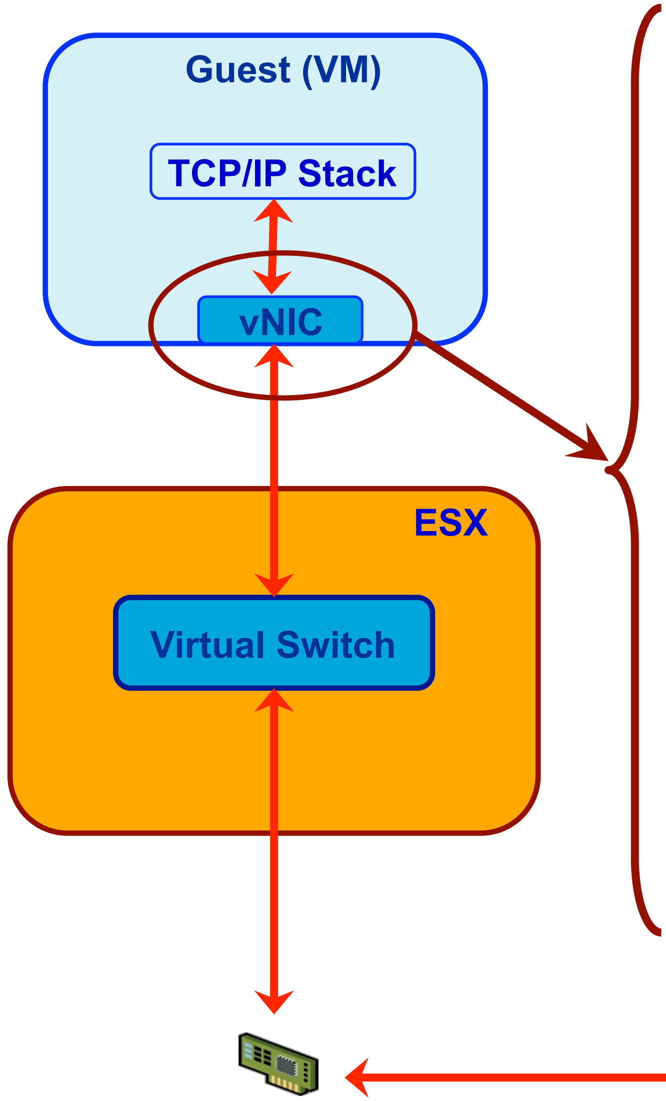
- **Use a network adapter that supports:**
  - Checksum offload, TCP segmentation offload (TSO), Jumbo frames (JF)
  - Enable JF when hardware is available (default is off!)
  - Capability to handle high memory DMA (64-bit DMA addresses)
  - Capability to handle multiple scatter/gather elements per Tx frame
- **Check configuration**
  - Ensure host NICs are running with highest supported speed and full-duplex
  - NIC teaming distributes networking load across multiple NICs
    - Better throughput and allows passive failover
- **Use separate NICs to avoid traffic contention**
  - For Console OS (host management traffic), VMKernel (vmotion, iSCSI, NFS traffic), and VMs

Jumbo Frames

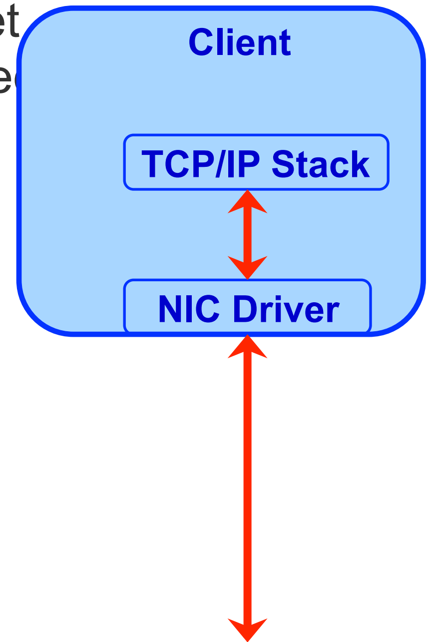
- **Before transmitting, IP layer fragments data into MTU (Maximum Transmission Unit) sized packets**
  - Ethernet MTU is 1500 bytes
  - Receive side reassembles the data

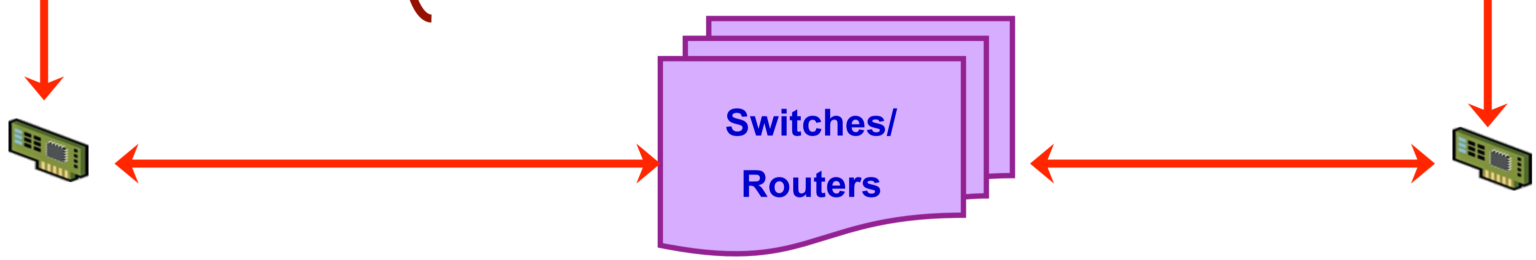
- **Jumbo Frames**
  - Ethernet frame with bigger MTU
  - Typical MTU is 9000 bytes
  - Reduces number of packets transmitted
  - Reduces the CPU utilization on transmit and receive side

Jumbo Frames

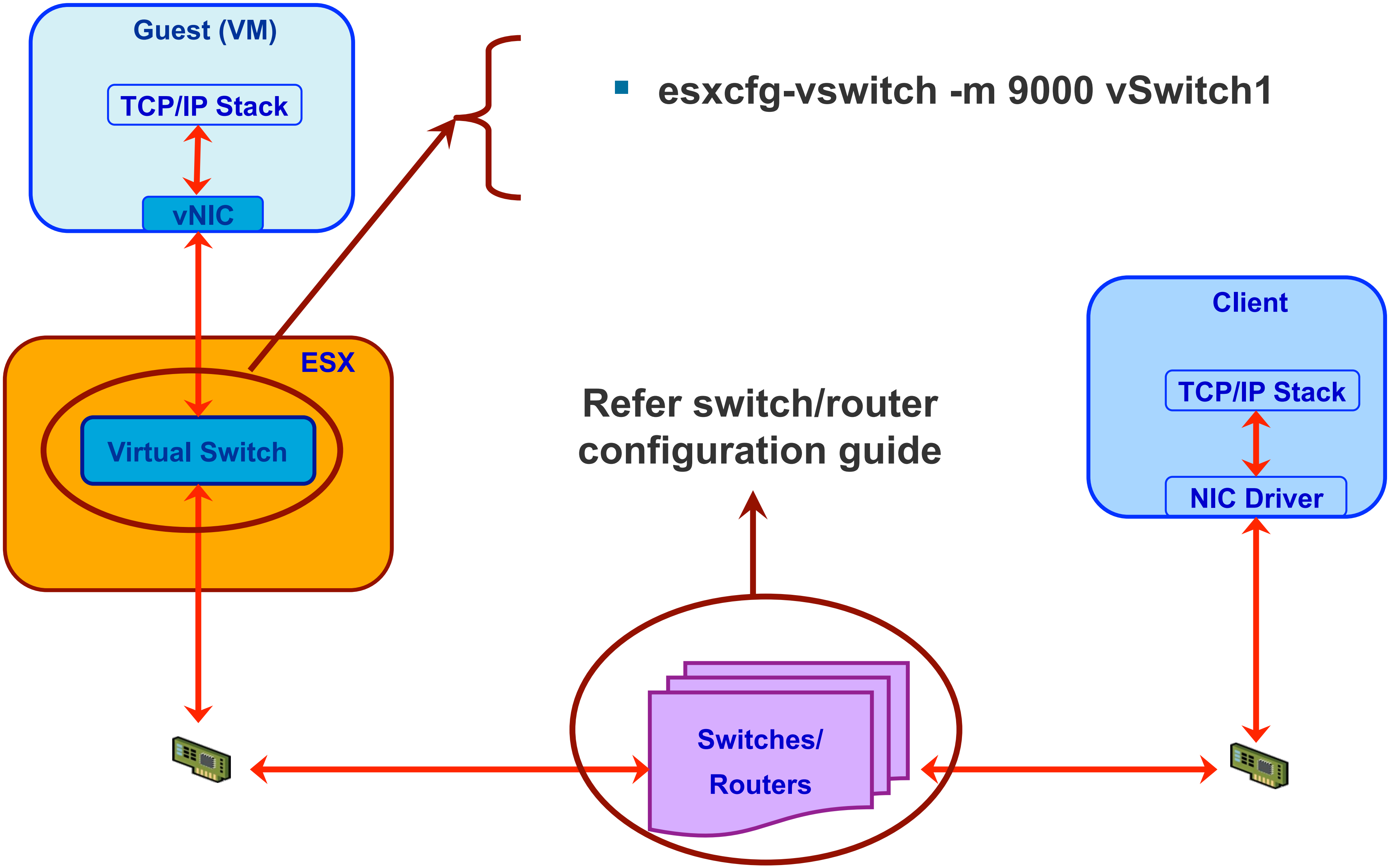


- **Linux**
  - ifconfig eth0 mtu 9000
- **Windows**
  - Device Manager -> Network adapters -> VMware PCI Ethernet Adapter -> Properties -> Advanced -> MTU to 9000





Jumbo Frames



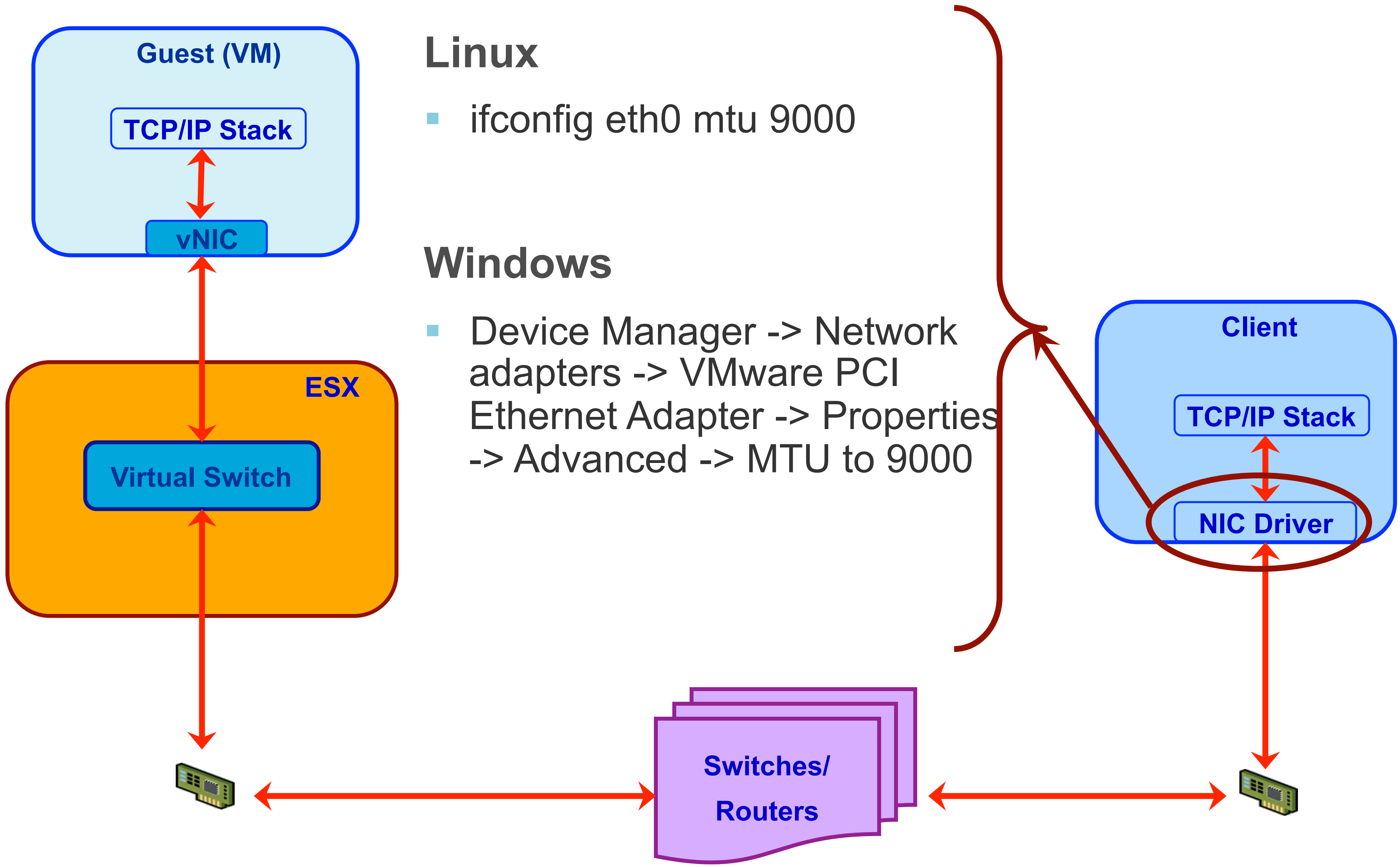
Jumbo Frames

Linux

- ifconfig eth0 mtu 9000

Windows

- Device Manager -> Network adapters -> VMware PCI Ethernet Adapter -> Properties -> Advanced -> MTU to 9000

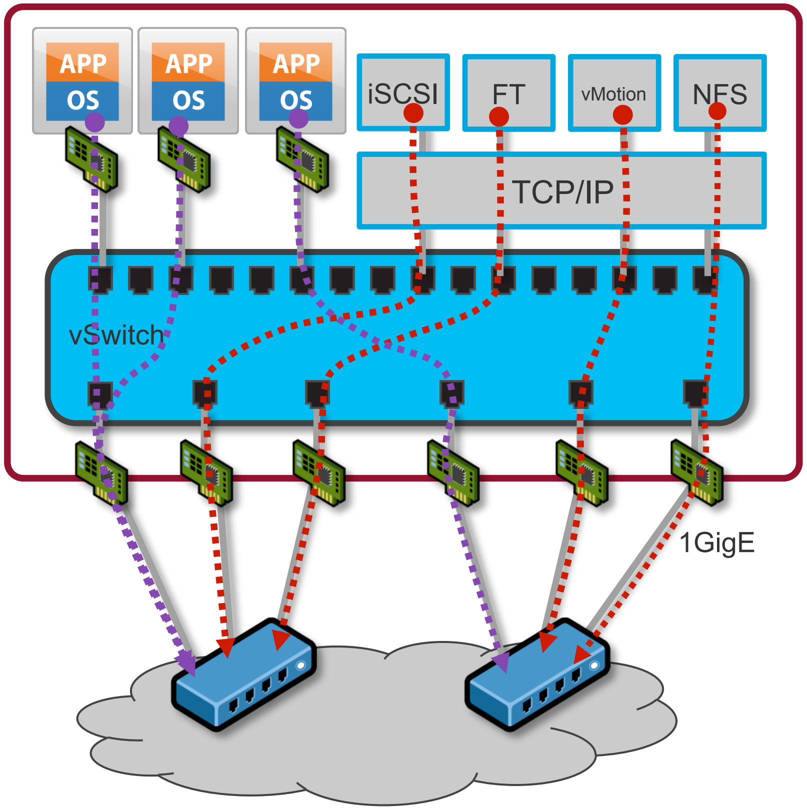


MTU Size

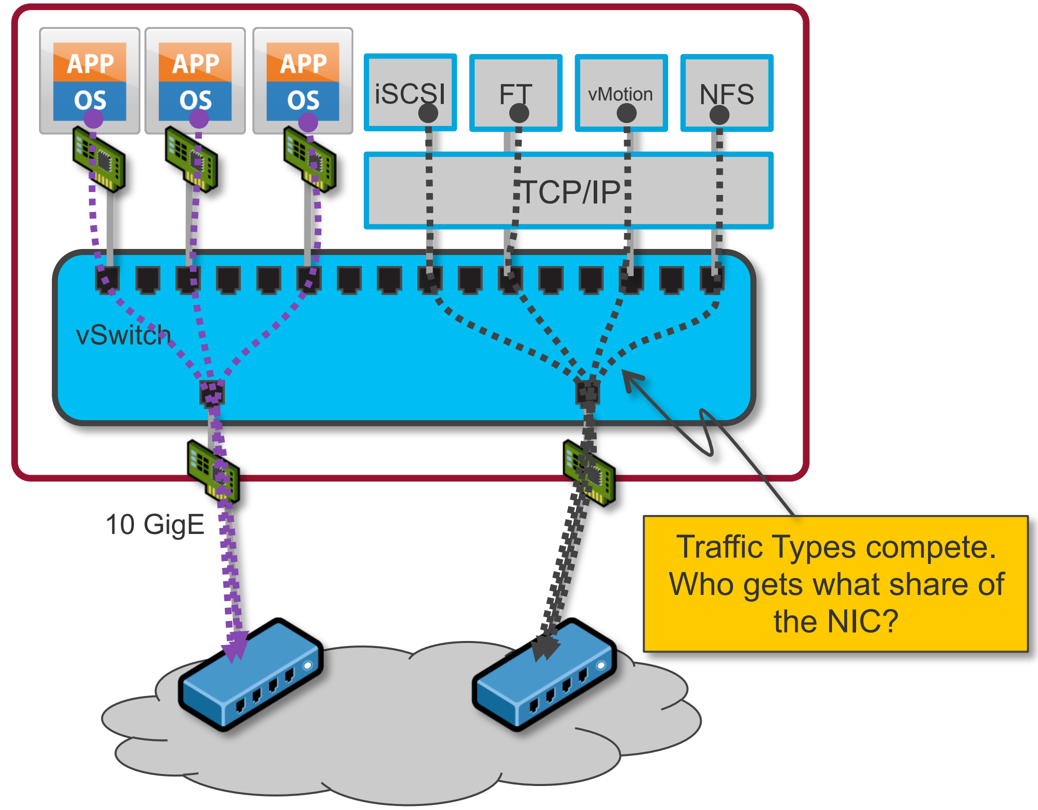
- **Verify it is not a jumbo frame related issue**
  - Verify that the vnic MTU is the same as the vswitch MTU
  - Run ping –s <packet size> from the guest

Network Traffic Management – Emergence of 10 GigE

1GigE NICs



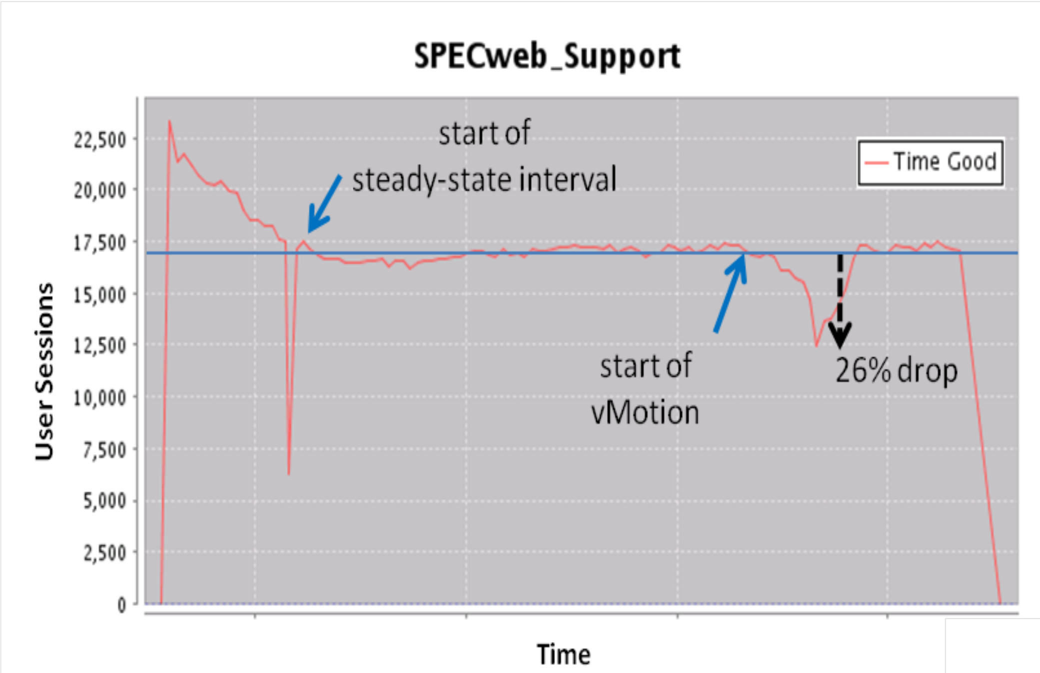
10 GigE NICs



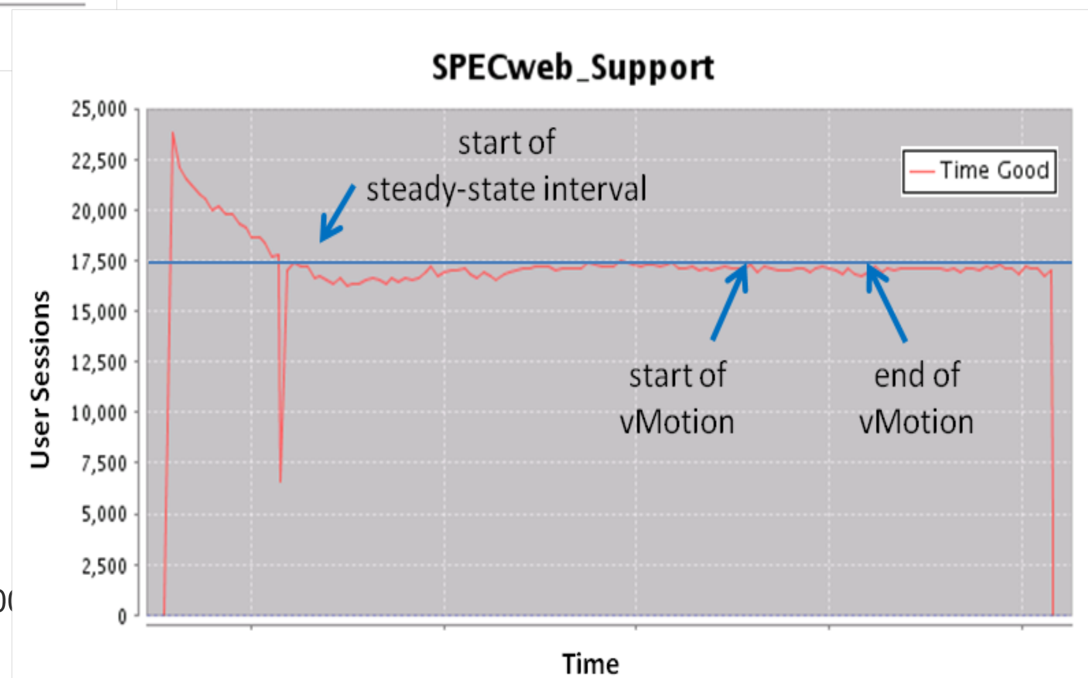
- Dedicated NICs for different traffic types e.g. vMotion, IP storage
- Bandwidth assured by dedicated NICs

- Traffic typically converged to two 10 GigE NICs
- Some traffic flows could dominate others through oversubscription

Network IO control – Protect your important bandwidth



- **Without Network IO Control**
  - VM traffic can be impacted by less performance-critical traffic such as vMotion

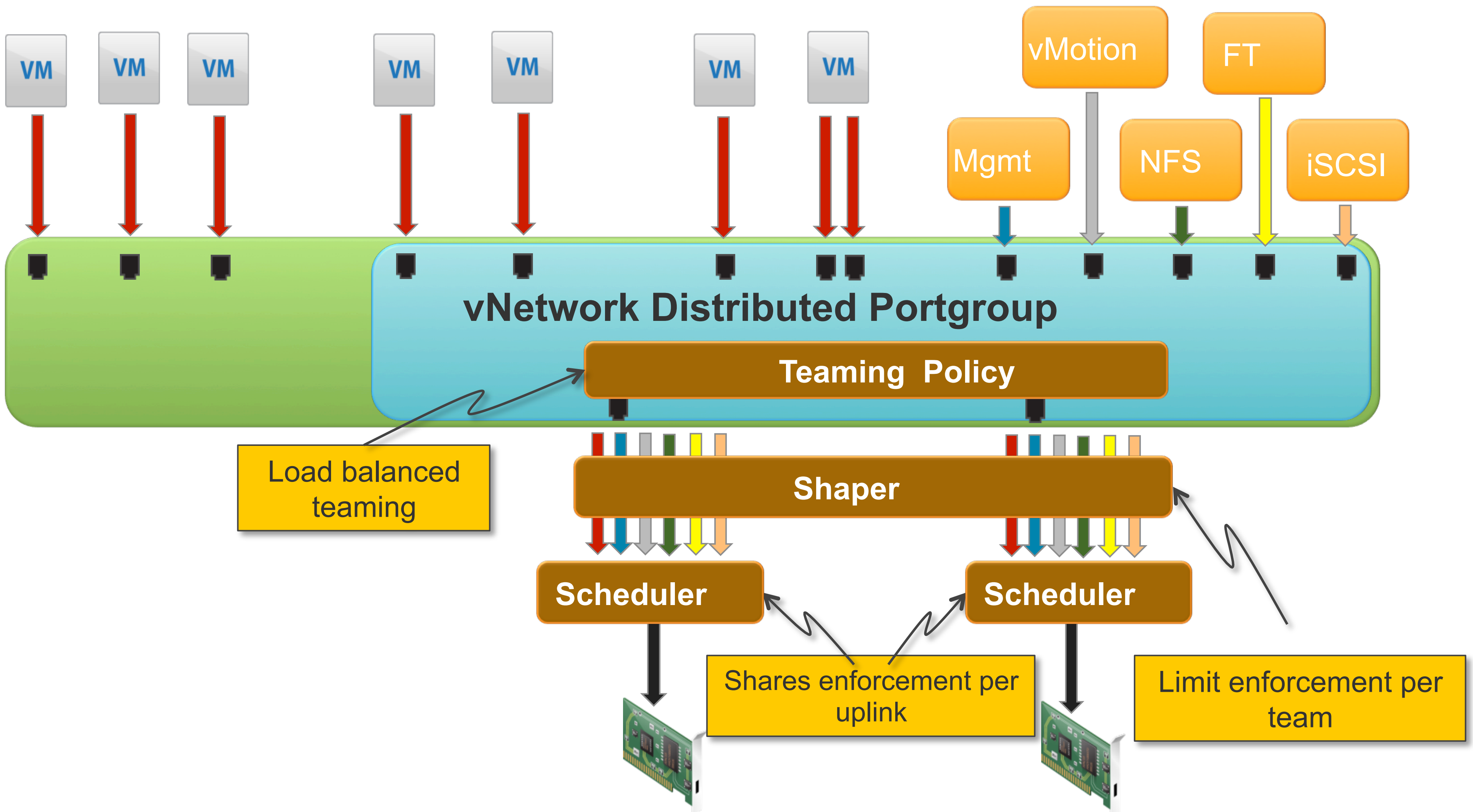


- **With Network IO Control**
  - VM traffic is protected and can maintain application SLAs
  - vMotion is designated lower priority and can take longer

* Y-Axis shows number of User Sessions that meet SPECweb20(latency requirements

vmware K

Network I/O Control Architecture



- Note: NetIOC is only supported with vNetwork Distributed Switch (vDS)
- Team: Group of NICs used for load balancing and fault tolerance

CONFIGURING WORKLOADS

Enterprise Workload Demands vs. Capabilities

	Workload Requires	vSphere 4
Oracle 11g	8vcpus for 95% of DBs 64GB for 95% of DBs 60k IOPS max for OLTP @ 8vcpus 77Mbits/sec for OLTP @ 8vcpus	8vcpus per VM 256GB per VM 120k IOPS per VM 9900Mbits/sec per VM
SQLserver	8vcpus for 95% of DBs 64GB @ 8vcpus 25kIOPS max for OLTP @ 8vcpus 115Mbits/sec for OLTP @ 8vcpus	8vcpus per VM 256GB per VM 120k IOPS per VM 9900Mbits/sec per VM
SAP SD	8vcpus for 90% of SAP Installs 24GB @ 8vcpus 1k IOPS @ 8vcpus 115Mbits/sec for OLTP @ 8vcpus	8vcpus per VM 256GB per VM 120k IOPS per VM 9900Mbits/sec per VM
Exchange	4cpus per VM, Multiple VMs 16GB @ 4vcpus 1000 IOPS for 2000 users 8Mbits/sec for 2000 users	8vcpus per VM 256GB per VM 120k IOPS per VM 9900Mbits/sec per VM

Databases: Top Ten Tuning Recommendations

1. Optimize Storage Layout, # of Disk Spindles
2. Use 64-bit Database
3. Add enough memory to cache DB, reduce I/O
4. Optimize Storage Layout, # of Disk Spindles
5. Use Direct-IO high performance un-cached path in the Guest Operating System
6. Use Asynchronous I/O to reduce system calls
7. Optimize Storage Layout, # of Disk Spindles
8. Use Large MMU Pages
9. Use the latest H/W – with AMD RVI or Intel EPT
10. Optimize Storage Layout, # of Disk Spindles

Databases: Workload Considerations

- **OLTP**

- **Short Transactions**
- **Limited number of standardized queries**
- **Small amounts of data accessed**
- **Uses data from only one source**
- **I/O Profile**
  - Small Synchronous reads/writes (2k->8k)
  - Heavy latency-sensitive log I/O
- **Memory and I/O intensive**

- **DSS**

Long Transactions

Complex queries

Large amounts of data accessed

Combines data from different sources

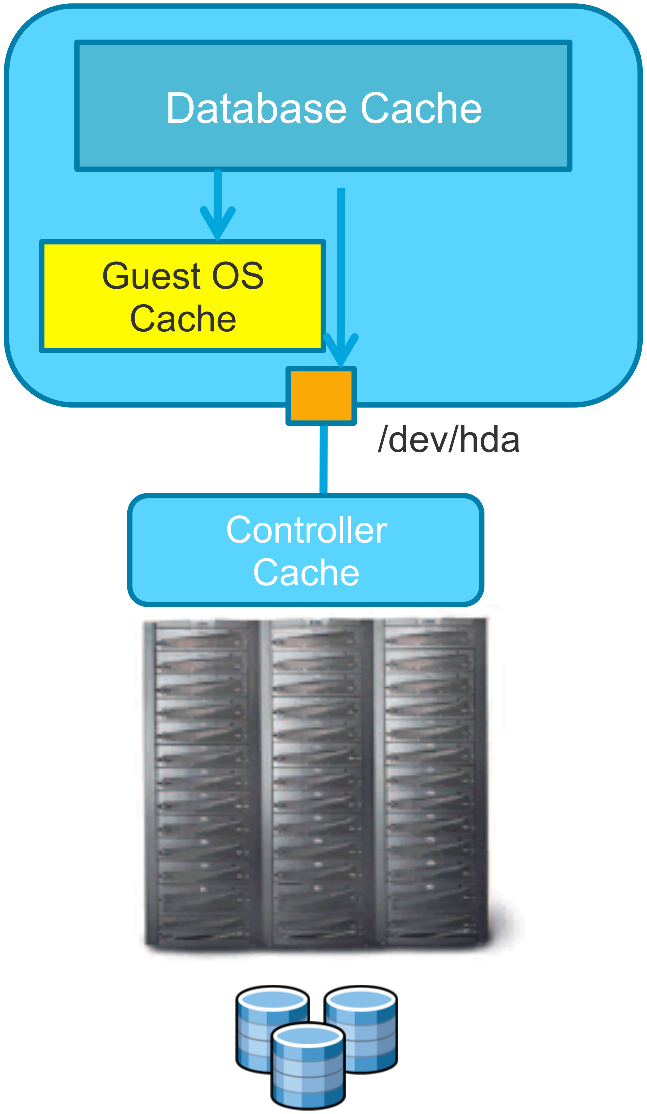
- **I/O Profile**
  - Large, Sequential I/Os (up to 1MB)
  - Extreme Bandwidth Required
  - Heavy ready traffic against data volumes
  - Little log traffic
- **CPU, Memory and I/O intensive**
- **Indexing enables higher performance**

Databases: Storage Configuration

- **Storage considerations**
  - VMFS or RDM
  - Fibre Channel, NFS or iSCSI
  - Partition Alignment
  - Multiple storage paths
- **OS/App, Data, Transaction Log and TempDB on separate physical spindles**
- **RAID 10 or RAID5 for Data, RAID 1 for logs**
- **Queue depth and Controller Cache Settings**
- **TempDB optimization**

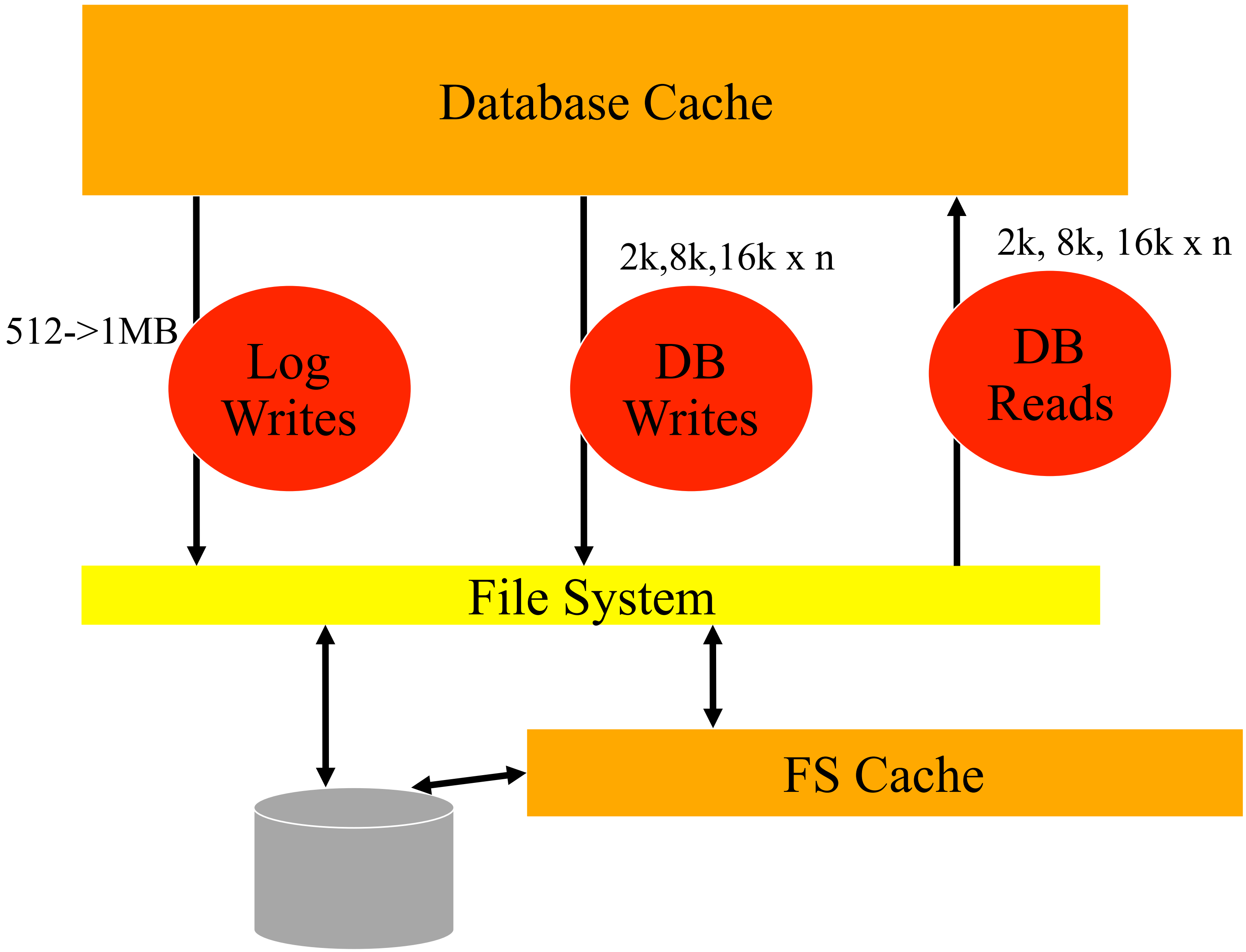
vmware®

Databases: Storage Hierarchy

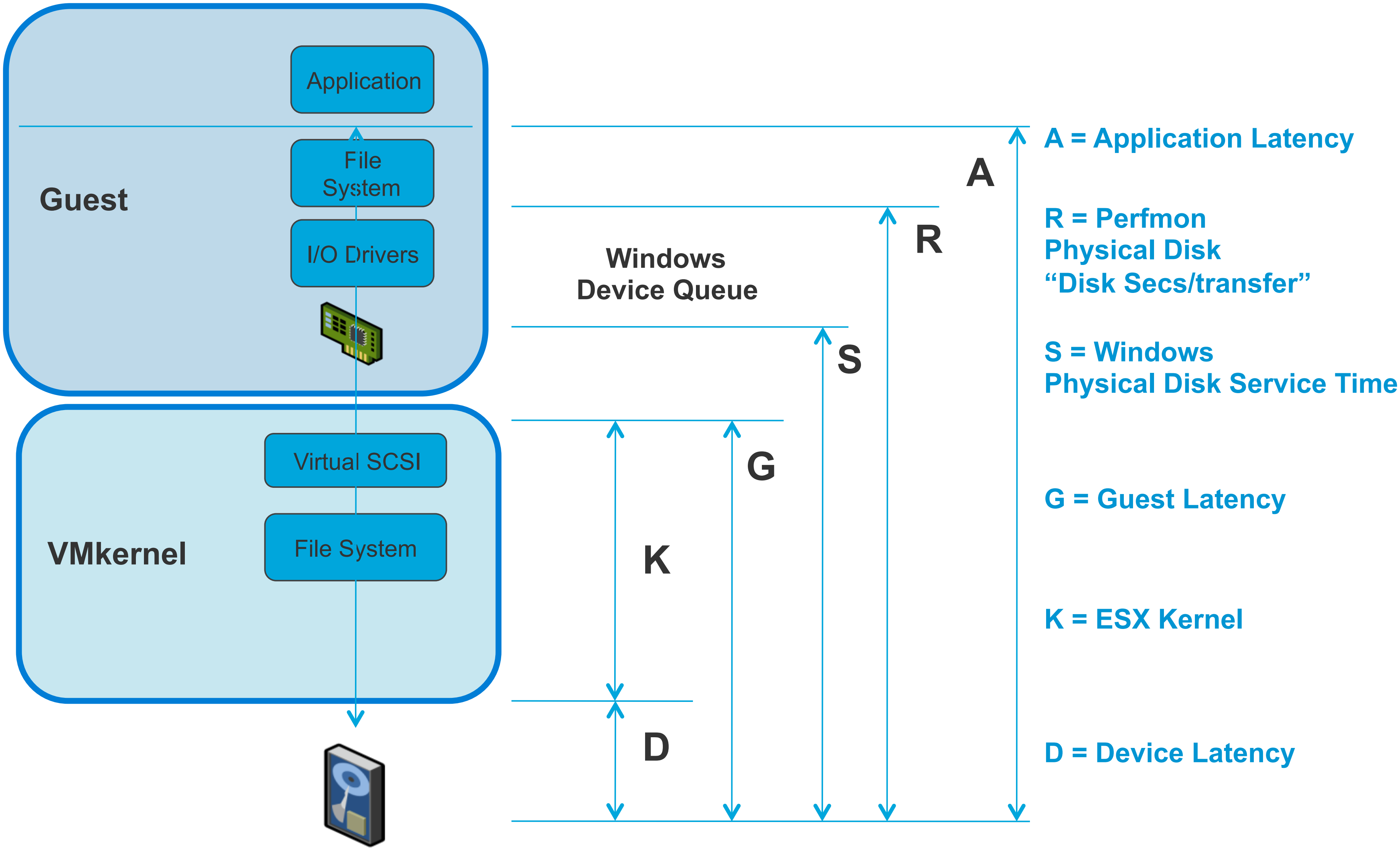


- In a recent study, we scaled up to 320,000 IOPS to an EMC array from a single ESX server.
  - 8K Read/Write Mix
- Cache as much as possible in caches
- Q: What's the impact on the number of disks if we improve cache hit rates from 90% to 95%?
  - 10 in 100 => 5 in 100…
  - #of disks reduced by 2x!

Databases: Typical I/O Architecture



Know your I/O: Use a top-down Latency analysis technique

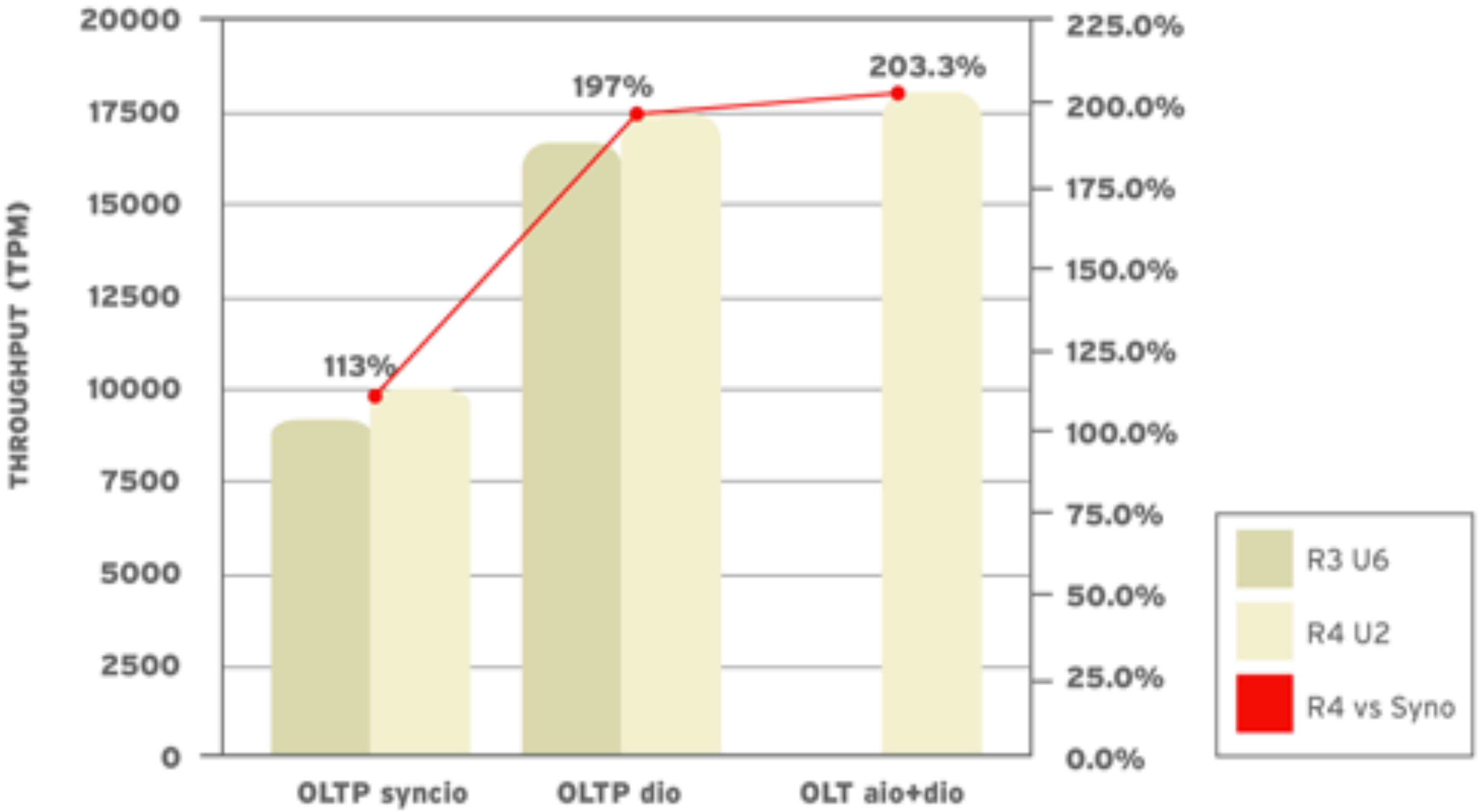


Checking for Disk Bottlenecks

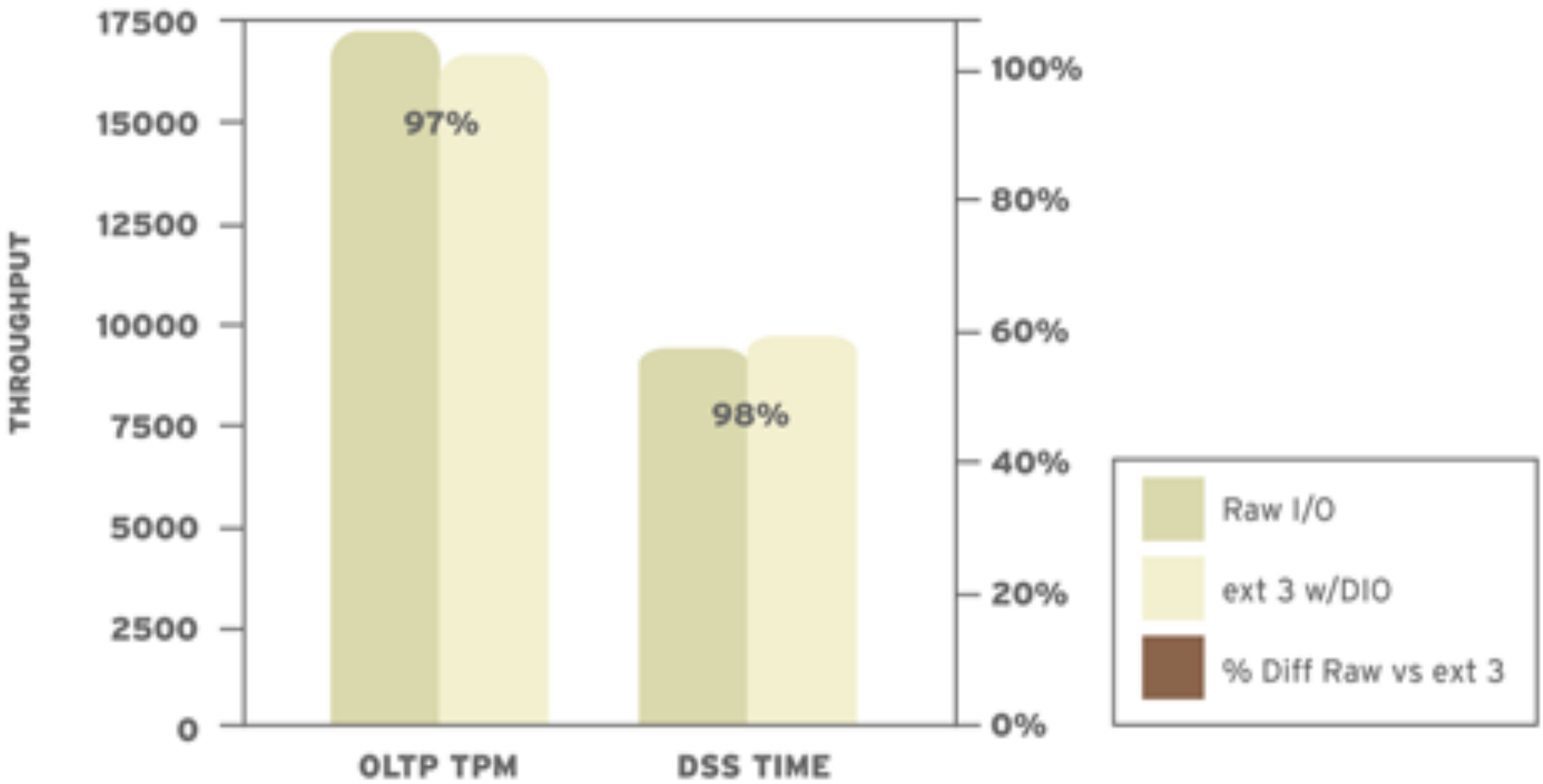
- **Disk latency issues are visible from Oracle stats**
  - Enable statspack
  - Review top latency events

```
Top 5 Timed Events
                                                            % Total

Event                          Waits         Time (s)      Ela Time

--------------------------- ------------ ----------- -----------
db file sequential read        2,598         7,146           48.54
db file scattered read        25,519         3,246           22.04
library cache load lock          673         1,363            9.26
CPU time                       2,154           934            7.83
log file parallel write       19,157           837            5.68
```

Oracle File System Sync vs DIO



Oracle DIO vs. RAW



Direct I/O

- **Guest-OS Level Option for Bypassing the guest cache**
  - Uncached access avoids multiple copies of data in memory
  - Avoid read/modify/write module file system block size
  - Bypasses many file-system level locks

- **Enabling Direct I/O for Oracle and MySQL on Linux**

```
# vi init.ora
filesystemio_options="setall"

Check:

# iostat 3
(Check for I/O size matching the
DB block size…)
```

```
# vi my.cnf
 innodb_flush_method to O_DIRECT

Check:

# iostat 3
(Check for I/O size matching the
DB block size…)
```

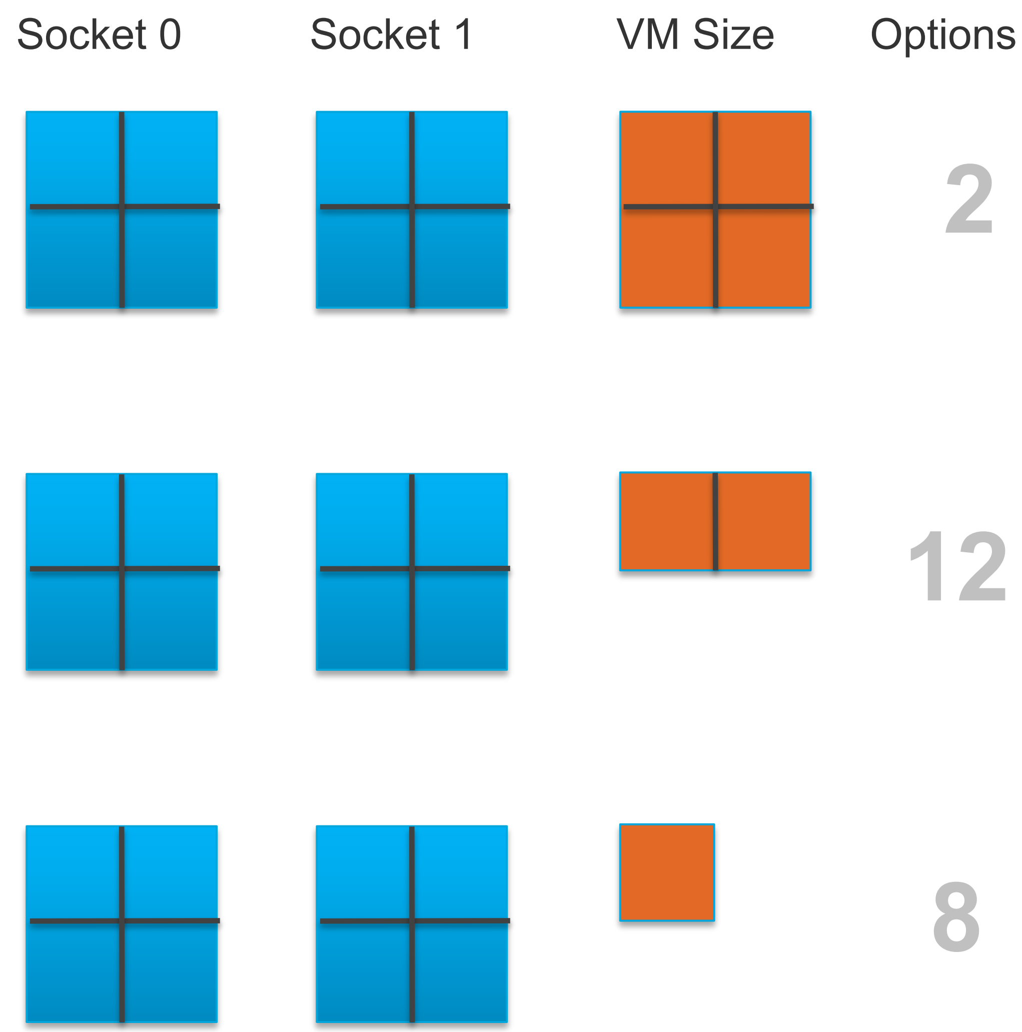
Asynchronous I/O

- **An API for single-threaded process to launch multiple outstanding I/Os**
  - Multi-threaded programs could just just multiple threads
  - Oracle databases uses this extensively
  - See aio_read(), aio_write() etc...
- **Enabling AIO on Linux**

```
# rpm -Uvh aio.rpm
# vi init.ora
filesystemio_options="setall"

Check:

# ps -aef |grep dbwr
# strace -p <pid>
io_submit()…              <- Check for io_submit in syscall trace
```

Picking the size of each VM

- **vCPUs from one VM stay on one socket***
- **With two quad-core sockets, there are only two positions for a 4-way VM**
- **1- and 2-way VMs can be arranged many ways on quad core socket**
- **Newer ESX schedulers more efficiency use fewer options**
  - Relaxed co-scheduling



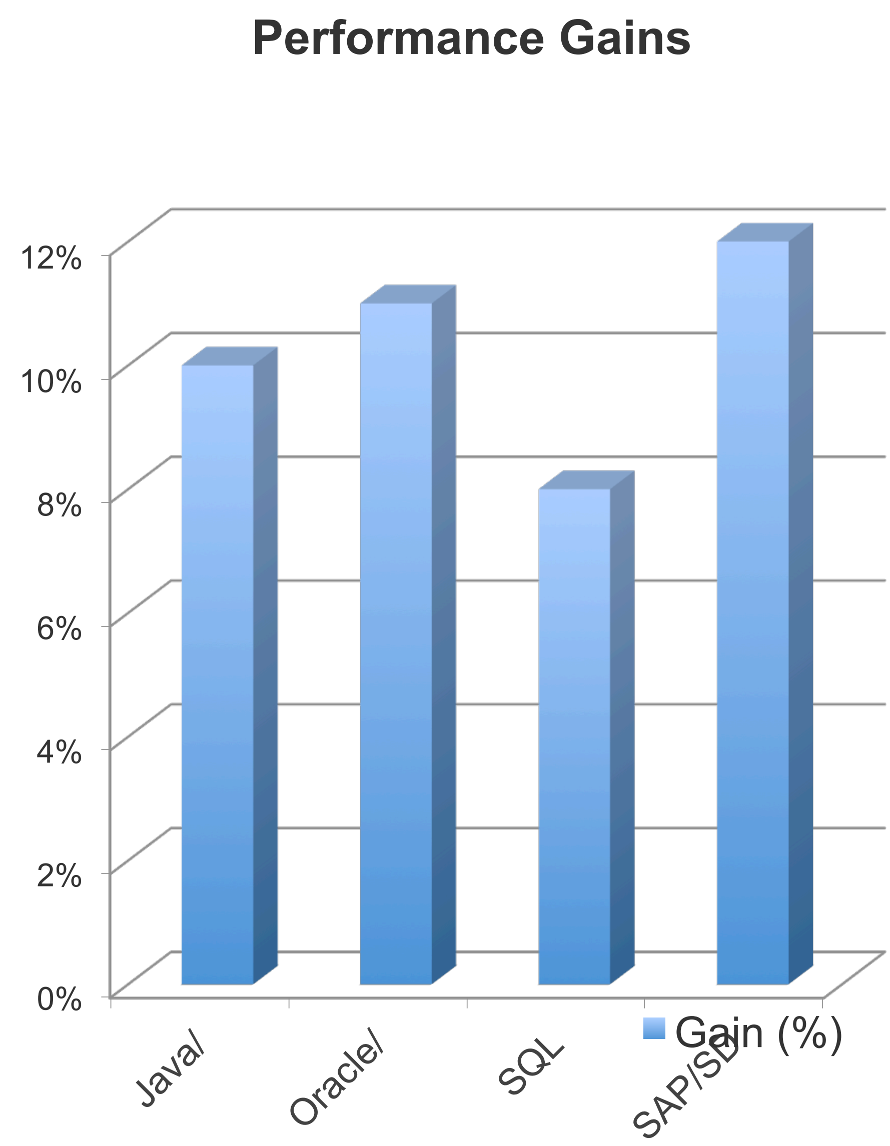
Use Large Pages

- **Guest-OS Level Option to use Large MMU Pages**
  - Maps the large SGA region with fewer TLB entries
  - Reduces MMU overheads
  - ESX 3.5 Uniquely Supports Large Pages!
- **Enabling Large Pages on Linux**

```
# vi /etc/sysctl.conf
(add the following lines:)

vm/nr_hugepages=2048
vm/hugetlb_shm_group=55

# cat /proc/vminfo |grep Huge
HugePages_Total:  1024
HugePages_Free:    940
Hugepagesize:     2048 kB
```

Large Pages

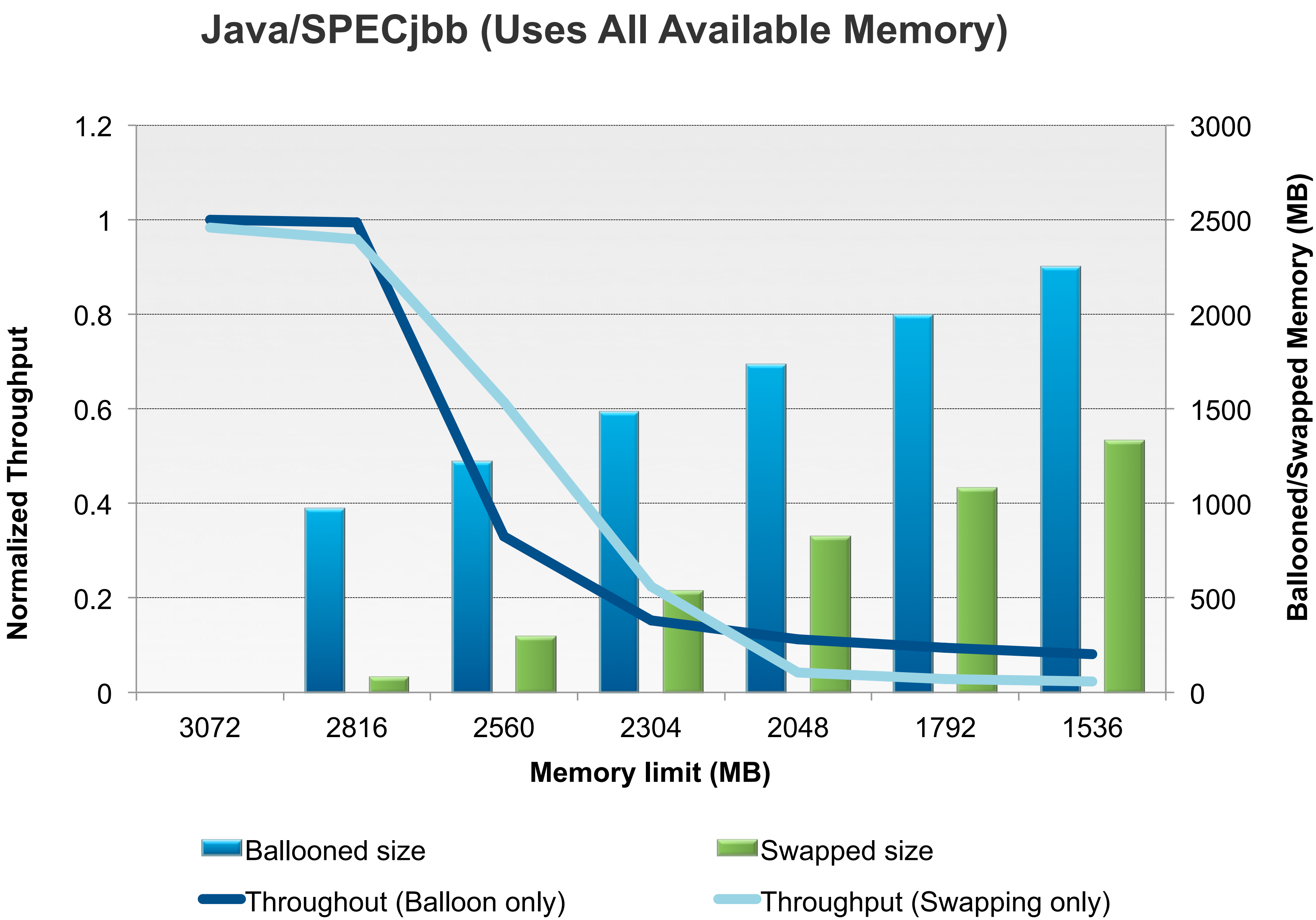
- **Increases TLB memory coverage**
  - Removes TLB misses, improves efficiency
- **Improves performance of applications that are sensitive to TLB miss costs**
- **Configure OS and application to leverage large pages**
  - LP will not be enabled by default



Linux Versions

- **Some older Linux versions have a 1Khz timer to optimize desktop-style applications**
  - There is no reason to use such a high timer rate on server-class applications
  - The timer rate on 4vcpu Linux guests is over 70,000 per second!
- **Use RHEL >5.1 or latest tickless timer kernels**
  - Install 2.6.18-53.1.4 kernel or later
  - Put divider=10 on the end of the kernel line in grub.conf and reboot, or default on tickless kernel
  - All the RHEL clones (CentOS, Oracle EL, etc.) work the same way

Java Requires Careful Memory Management

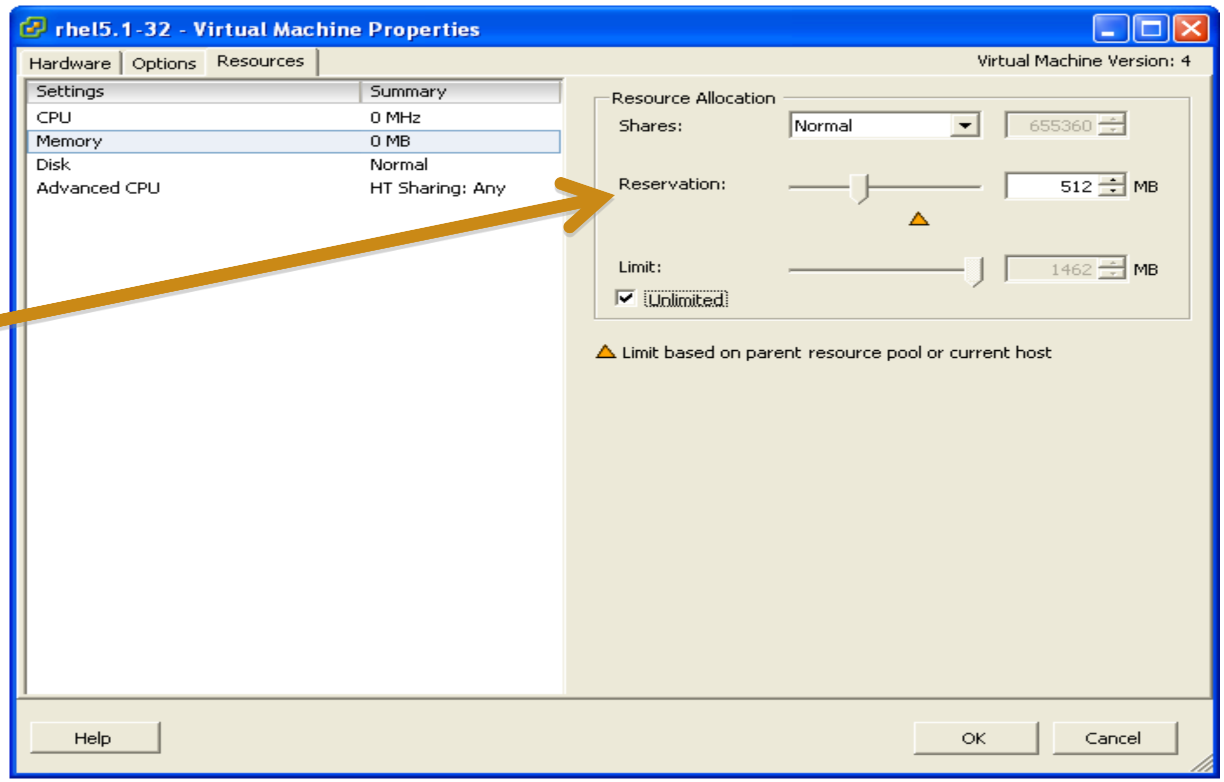


Managing Memory in Java Environments

- Calculate OS memory
- Estimate JVM needs
- Specify heap exactly

- Reservations = OS + JVM + heap

- Also applies to other applications with static memory needs
  - Oracle SGA



vmware

For More Information

- **VMware's Performance Technology Pages**
  - http://vmware.com/technical-resources/performance
- **VMware's Performance Blog**
  - http://blogs.vmware.com/performance
- **Performance Community**
  - http://communities.vmware.com/community/vmtn/general/performance
- **VMware Performance Class**
  - Check with VMware Education or VMware Authorized Training Center
- **VMware Performance Service Offering**
  - Ask VMware account team

VMware Performance for Gurus

Richard McDougall

CTO of Application Infrastructure, VMware

rmc@vmware.com   *twitter @richardmcdougll*