Virtualizing Performance-Critical Database Applications in VMware® vSphere™

VMware vSphere 4.0 with ESX™ 4.0

VMware® vSphere™ 4.0 with ESX™ 4.0 makes it easier than ever to virtualize demanding applications such as databases. Database workloads are widely acknowledged to be extremely resource-intensive. The large number of storage commands issued and the network activity to serve remote clients place significant challenges on the platform. The high consumption of CPU and memory resources leaves little room for inefficient virtualization software. The questions we're often asked are:

- Is it possible to run a heavy-duty database application in a virtual machine?
- Can a virtualized environment handle high storage IOPS and network packets/second?

While database applications have been successfully deployed in virtual machines since the earliest versions of VMware ESX, vSphere 4.0 incorporates many performance-related enhancements such as:

- PVSCSI, a new paravirtualized SCSI adapter for virtual machines
- Improved I/O concurrency for enhanced storage performance
- Improved interrupt coalescing and delivery mechanisms for better I/O throughput
- Hardware support for memory management unit virtualization
- Improved CPU scheduler that can better support these larger I/O intensive virtual machines

By quantifying performance gains achieved as a result of these changes for a very high-end Oracle database deployment (with a much larger resource footprint than one would expect to see in most production environments) and highlighting the overall system performance, we show that large database applications deployed in virtual machines have excellent performance.

Highlights

We conducted a set of experiments using the Order-Entry benchmark (described in "Workload Characteristics" on page 3) with the goal of quantifying:

- Performance gains due to enhancements built into ESX 4.0
- Performance differential between ESX 4.0 and ESX 3.5
- Performance differential between ESX 4.0 and native

Results from these experiments show that even the largest database applications can be deployed with excellent performance. For example, a virtual machine with eight virtual CPUs (vCPUs) running on an ESX host with eight physical CPUs (pCPUs), throughput was 85% of native on the same hardware platform. Statistics that give an indication of the load placed on the system in the native and virtual machine configurations are summarized in Table 1.

Table 1. Comparison of Native and Virtual Machine Benchmark Load Profiles

Metric	Native	VM
Throughput in business transactions per minute	293K	250K
Disk IOPS	71K	60K
Disk bandwidth	305MB/s	258MB/s
Network packet rate	12K/s receive 19K/s send	10K/s receive 17K/s send
Network bandwidth	25 Mb/s receive 66 Mb/s send	21 Mb/s receive 56 Mb/s send

A 2007 VMware Capacity Planner study determined that a typical Oracle database application running on a 4-core installation has a workload profile defined by 100 transactions/second and 1,200 IOPS. Table 2 shows the difference in scale between the typical deployment profile and the load placed by the Order-Entry benchmark on an eight-vCPU virtual machine. Note that each Order-Entry business transaction consists of 2.14 individual transactions, hence a throughput of 8.9K individual transactions per second.

Table 2. Typical Load Profile vs. Benchmark Load Profile

Metric	Typical Deployment	Virtual Machine
Throughput (transactions per second)	100	8.9K
Disk IOPS	1,200	60K

The corresponding guest statistics are shown in Table 3. These statistics were collected while running the Order-Entry benchmark and provide another perspective on the resource-intensive nature of the workload.

Table 3. Guest Operating System Statistics

Metric	Value
Interrupts per second	13K
Disk IOPS	60K
Context switches per second	54K

Performance Test Environment

This section describes the workload, hardware and software configurations, and the benchmark methodology.

Workload Characteristics

The workload used in our experiments is based on the TPC-C benchmark. We will refer to this workload as the Order-Entry benchmark. The Order-Entry benchmark is a non-comparable implementation of the TPC-C business model; our results are not TPC-C compliant, and not comparable to official TPC-C results. According to the Transaction Processing Performance Council rules, we need to disclose deviations from the benchmark specification. The deviations from the specification are: batch implementation and an undersized database for the observed throughput.

A compliant run at this throughput level would require approximately 200,000 emulated users with long think times. Typically, a transaction monitor is used to multiplex the load of these users down to a few hundred processes that are connected to the database. Our benchmark was run with a few hundred no-think-time users directly connected to the database.

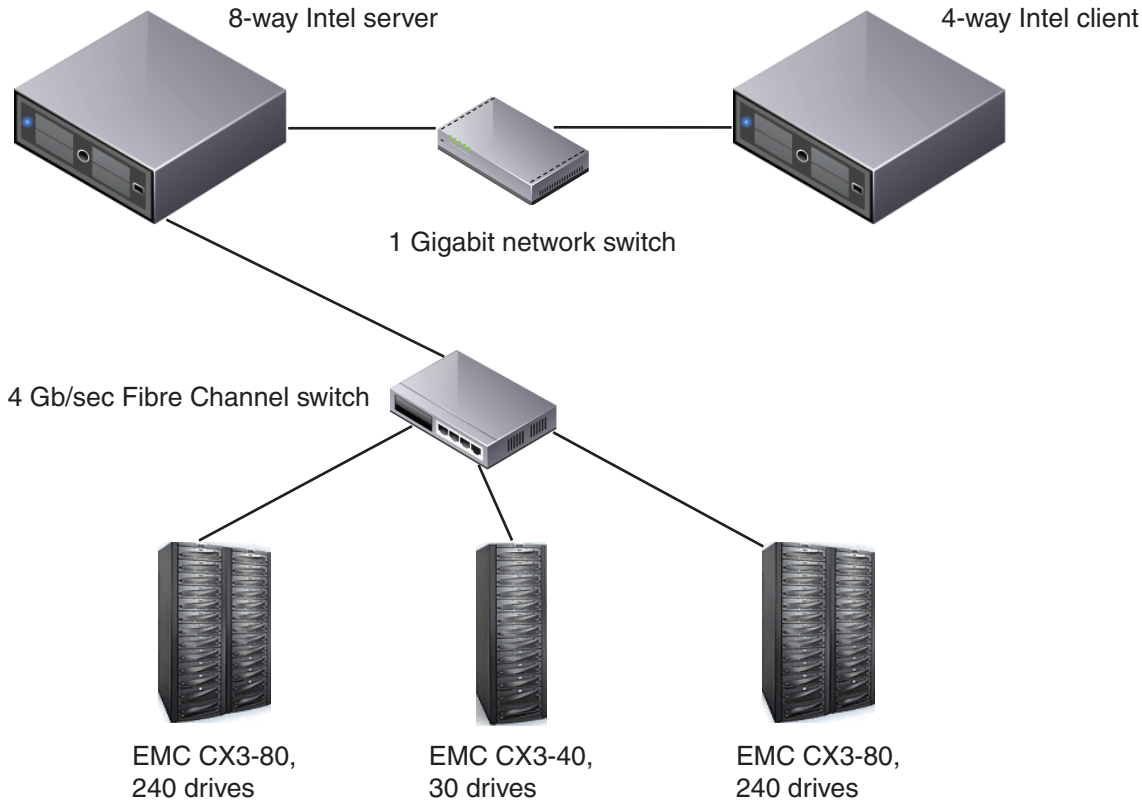
Database size is an important parameter in this benchmark. Vendors sometimes size their database based on a small warehouse count, which results in a "cached" database and a low I/O rate with misleading results. The performance levels we measured would normally correspond to a database size of 20-23K warehouses. Since we were limited by the memory available to us, the benchmark was configured with 7,500 warehouses. A property of this benchmark is that it produces very similar results, with similar disk I/O rates, as long as the Oracle SGA size and the database size are scaled together. Therefore, 7,500 warehouses with 36GB of memory gives results comparable to those from a benchmark with 23,500 warehouses on a system with 108GB of memory, which is the about the right memory size for this class of machine. So, all in all, this is a reasonable database size for the performance levels we are seeing, producing the right disk I/O rate for this level of performance.

The Order-Entry benchmark is an OLTP benchmark with many small transactions. Of the five transaction types, three update the database and the other two, which occur with relatively low frequency, are read-only. The I/O load is quite heavy and consists of small access sizes (2K-16K). The disk I/O accesses consist of random reads and writes with a 2:1 ratio in favor of reads. In terms of the impact on the system, this benchmark spends considerable execution time in the operating system kernel context, which is harder to virtualize than user-mode code. Specifically, how well we virtualize the hardware interrupt processing, I/O handling, context switching, and scheduler portions of the guest operating system code is critical to the performance of this benchmark. The workload is also very sensitive to the processor cache and translation lookaside buffer (TLB) hit ratios.

Hardware Configuration

The experiment testbed consisted of a server machine, a client machine used to drive the benchmark, and backend storage capacity such that disk latencies are at acceptable levels. Figure 1 shows the connectivity between the various components and the subsequent subsections provide details of the hardware.

Figure 1. Experimental Testbed Configuration



Server Hardware

The server hardware is detailed in Table 4.

Table 4. Server Hardware

Whitebox using a prototype of Intel® Xeon® processors
Two 2.93GHz quad-core Intel Xeon X5570 (“Nehalem”) processors
36GB memory (18 2GB DIMMs)
SMT and turbo mode disabled in BIOS

Storage Hardware

The storage hardware is detailed in Table 5.

Table 5. Storage Hardware

Two EMC CLARiiON CX3-80 arrays and one CX3-40 array
510 15K RPM disk drives[1]
16 LUNs on each CX3-80 SP controller

1. The large number of disks is needed to maintain reasonable response time. Even with this configuration, the read response time was about 8ms.

Client Hardware

The client hardware is detailed in Table 6.

Table 6. Client Hardware

Single-socket, quad-core server
2.50GHz Intel E5420 (“Harpertown”) processor
4GB memory

Software

The software components of the test bed are listed in Table 7.

Table 7. Software Versions

Software	Version
VMware ESX 4.0	VMware ESX Build # 136362
VMware ESX 3.5	VMware ESX 3.5, Update 3
Operating system (guest and native)	RHEL 5.1 64-bit
DBMS	Trial version of Oracle 11g R1

The virtual machine and native configurations were identical in that the same operating system and DBMS software were used. The same configuration and benchmarking scripts were used to set up and run the benchmark. In both cases, large memory pages were used to ensure optimum performance. Virtual machine and native tests were run against the same database.

Benchmark Methodology

All experiments were conducted with the number of pCPUs used by ESX equal to the number of vCPUs configured in the virtual machine. By fully committing CPU resources in this way we ensure that performance comparisons between ESX and native are fair.

In an under-committed test environment, a virtual machine running on an ESX host can offload certain tasks, such as I/O processing, to the additional processors (beyond the number of its virtual CPUs). For example, when a 4-vCPU virtual machine is run on an 8-pCPU ESX host, the throughput is approximately 8% higher than when that virtual machine is run on a 4-pCPU ESX host.

Using fewer than the eight cores in the test machine required additional consideration. When configured to use fewer than the available number of physical cores ESX round-robins between sockets while selecting cores. Native Linux selects cores from the same socket. This would have made comparisons with native unfair in the scaling performance tests. Therefore in the two- and four-CPU configurations the same set of cores was made available to both ESX and native Linux by configuring the appropriate number of cores in BIOS.

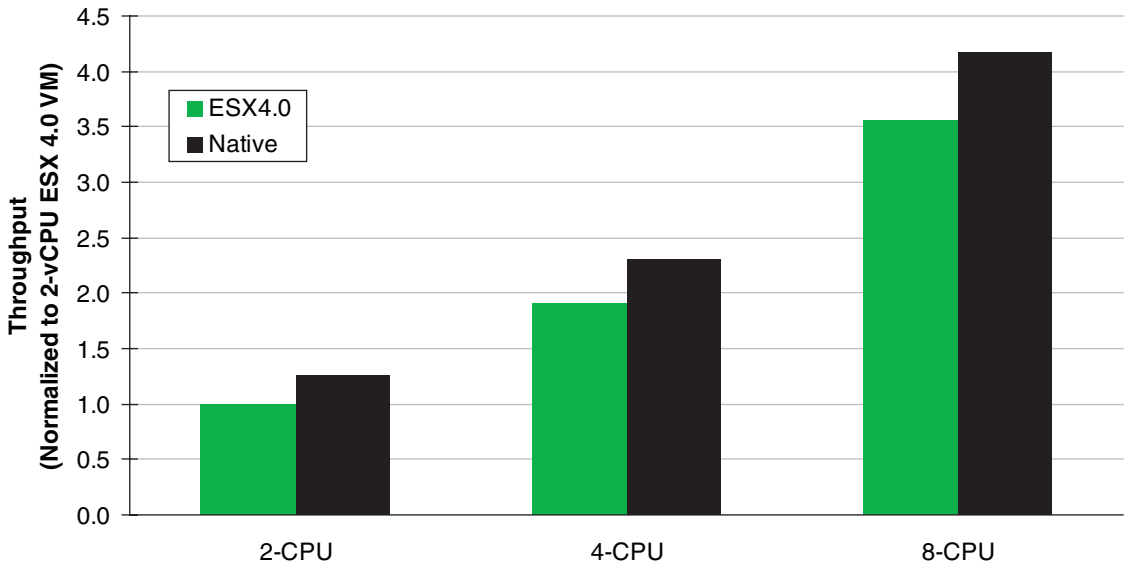
Performance Results

Results from experiments executing the Order-Entry benchmark both natively and in a virtual machine are detailed in the sections below.

ESX 4.0 Performance Relative to Native

We show how ESX 4.0 scales by normalizing both virtualized and native benchmark scores to the smallest configuration, the 2-vCPU virtual machine. Figure 2 illustrates ESX 4.0 performance scalability and virtual machine throughput relative to native.

Figure 2. Throughout: ESX 4.0 vs. Native



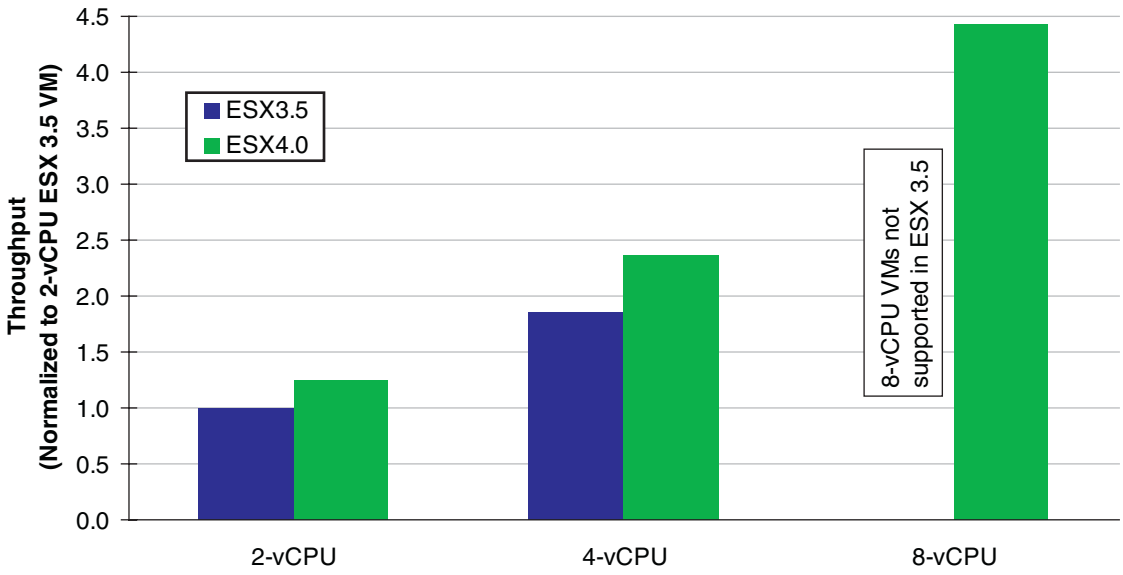
As the ratios in Figure 2 show, ESX 4.0 scales extremely well; for each doubling of processors, throughput increases by about 90%.

Comparison with ESX 3.5

Experimental data comparing ESX 4.0 with ESX 3.5 show that with ESX 4.0 throughput is about 24% higher for the two-processor configuration and about 28% higher for the four-processor configuration. These ratios are an indication of performance gains achieved by upgrading from ESX 3.5 to ESX 4.0.

Another way of looking at this is to examine the throughput ratios using throughput for a 2-vCPU virtual machine running on ESX 3.5 as the reference. This gives an indication of the combined performance gain observed both with an ESX upgrade as well as an increase in the number of configured vCPUs. From this perspective, as shown in Figure 3, we see that while throughput from an ESX 3.5 4-vCPU virtual machine is 1.85 times that of the ESX 3.5 2-vCPU virtual machine, an ESX 4.0 4-vCPU virtual machine gives 2.37 times the throughput of the ESX 3.5 2-vCPU virtual machine.

Figure 3. Throughput: ESX 4.0 vs. ESX 3.5



From Figure 3 we can also see that throughput from an 8-vCPU virtual machine is 4.43 times that of the reference virtual machine and 2.4 times higher than the throughput from an ESX 3.5 4-vCPU virtual machine. In other words, with support for 8-vCPU virtual machines, maximum throughput achievable from a single virtual machine is much higher in ESX 4.0 than in ESX 3.5.

Performance Impact of New Features and Enhancements

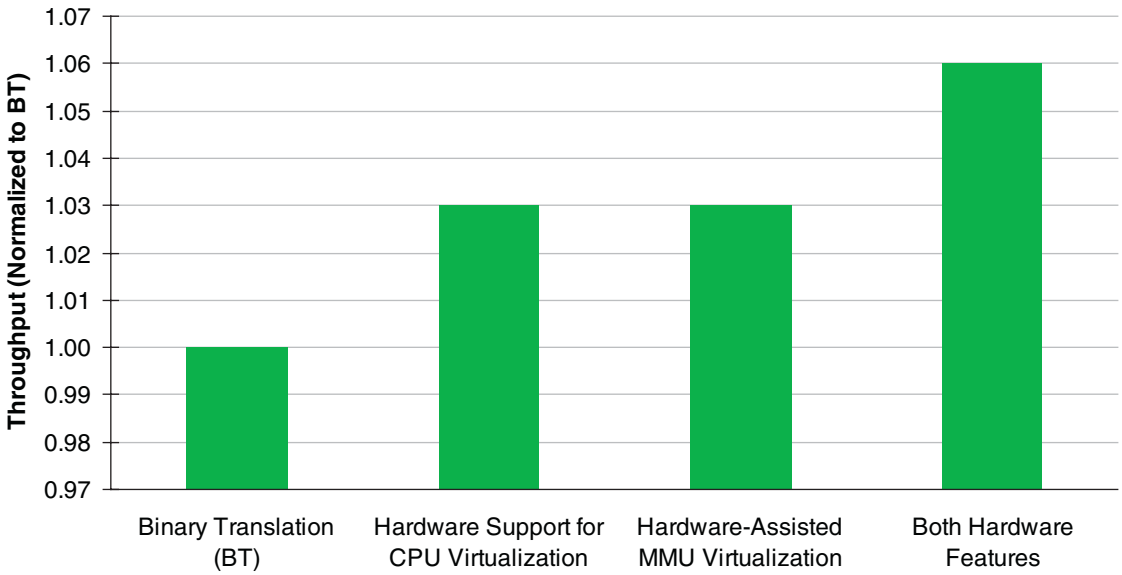
Numerous performance enhancements have made ESX 4.0 an even better platform for performance-critical applications than the ESX 3.5 release. Performance gains from added hardware support for memory management unit virtualization, a more efficient and feature-rich storage stack, and significantly better CPU resource management are analyzed in this section. The Appendix includes brief descriptions of the key features that produced gains in this workload.

Virtual Machine Monitor

Virtualization acceleration features in current processors improve performance for most workloads. Hardware support for CPU virtualization has been available in earlier versions of ESX. Both AMD and Intel have added hardware support for memory management unit (MMU) virtualization. Support for AMD-V with RVI has been available since ESX 3.5; ESX 4.0 adds support for Intel VT-x with EPT. The Appendix includes a brief description of the ESX virtual monitor types.

Figure 4 shows the performance gains obtained in our experiments from this hardware support.

Figure 4. Performance Benefits of Hardware-Assisted Virtualization



Throughput from the binary translation (BT) monitor is used as the reference value in the above graph. It can be seen that hardware support for CPU virtualization increases throughput by 3%, as does hardware-assisted MMU virtualization. By enabling the use of both technologies we observed 6% higher throughput as compared to the case where virtualization was implemented entirely in software.

Another enhancement in ESX 4.0 is that virtual machines can now be configured with up to eight vCPUs (compared to a limit of four vCPUs in ESX 3.5). As shown in Table 8, we found that each doubling of vCPUs resulted in about a 90% increase in throughput on the Order-Entry benchmark.

Table 8. Scale-Up Performance in ESX 4.0

Comparison	Performance Gain in ESX 4.0
4-vCPU VM vs. 2-vCPU VM	1.90
8-vCPU VM vs. 4-vCPU VM	1.87

Storage Subsystem

The storage stack has undergone significant changes. Some of these enhancements are in the form of more efficient internal data structures and code while others include new features such as PVSCSI, the new SCSI adapter emulation. A summary of some of the performance gains made in the area are shown in the Table 9.

Table 9. Performance Impact of Storage Subsystem Enhancements in ESX 4.0

Feature	Performance Gain
PVSCSI	3%
Improved I/O Concurrency	5%
Virtual Interrupt Coalescing	3%
Virtual Interrupt Delivery Optimization	1%

The Appendix includes brief descriptions of each of these features and enhancements.

Network Stack

Coalescing of receive and transmit traffic between the client and server has been found to improve throughput of this benchmark. There are several configuration options which define the rate at which network packets are batched. New values for these parameters can be set in the Advanced Settings section in the vSphere Client. This section describes the performance impact of these options.

The values of the receive and transmit coalescing parameters Net.CoalesceLowRxRate and Net.CoalesceLowTxRate can make a noticeable difference in throughput. Default values for both parameters are 4; reducing them to a value of 1 improved benchmark throughput by approximately 1%.

Further optimization can be done by altering the network parameter Net.vmxnetThroughputWeight from the default value 0 to 128, thus favoring transmit throughput over response time. Note that the virtual machine must be rebooted for this change to take effect. With this change, throughput increases by approximately 3% over the default setting. Even after making this change, transaction response times for the virtual machine were within 10ms of native response times. It does not appear that this change in networking coalescing behavior had a negative impact on transaction response times.

Conclusion

Typical Oracle database applications generate much less I/O and support far fewer transactions per second than vSphere 4.0 supports. Previously-reported Capacity Planner data showed the average Oracle database application running on a 4-core installation to have the following load profile:

- 100 transactions/second
- 1,200 IOPS

In comparison, our experiments show that an 8-vCPU virtual machine can handle 8.9K DBMS transactions/second with the accompanying 60K IOPS. This demonstrates capabilities over 50 times that needed by most Oracle database applications, proof-positive that the vast majority of the most demanding applications can be run, with excellent performance, in a virtualized environment with vSphere 4.0. With a near-linear scale-up, and a 24% performance boost over ESX 3.5, vSphere 4.0 is the best platform for virtualizing Oracle databases.

Disclaimers

All data is based on in-lab results with a developmental version of ESX.

Our benchmark was a fair-use implementation of the TPC-C business model; our results are not TPC-C compliant and are not comparable to official TPC-C results. TPC Benchmark and TPC-C are trademarks of the Transaction Processing Performance Council.

Prototypes of Intel Xeon X5570 (“Nehalem”) processors were used. Our performance is not a guarantee of the true performance of what will be generally available.

Our throughput is not meant to indicate the absolute performance of Oracle, or to compare its performance to another DBMS. Oracle was simply used to place a DBMS workload on ESX, and observe and optimize the performance of ESX.

Our goal was to show the relative-to-native performance of ESX, and its ability to handle a heavy database workload, not to measure the absolute performance of the hardware and software components used in the study.

Appendix

This section describes the features and enhancements in vSphere 4.0 with ESX 4.0 that had the most significant positive effect on the performance of the Order-Entry benchmark.

Hardware-Assisted Memory Management Unit (MMU) Virtualization

Database applications and database management systems typically use large amounts of memory. Maintaining application working sets in memory helps reduce storage I/O. In addition, data access patterns can cause large changes in the contents of the working set. These two factors make efficient memory management a determinant factor in database performance.

Memory management in virtual machines differs from physical machines in one key aspect: virtual memory address translation. Guest virtual memory addresses are translated to guest physical addresses using the guest operating system page tables. ESX then translates the guest physical addresses into machine physical addresses on the host. ESX maintains mappings from the guest virtual addresses to the machine physical addresses in shadow page tables. The shadow page tables allow ESX to avoid doing two levels of translation for every memory access and are cached in the hardware's TLB. However, creating and maintaining the shadow page tables causes some overhead.

Hardware support for memory management unit virtualization is available in current processors. Offerings from Intel and AMD are called EPT and RVI, respectively. This support consists of an additional level of page tables implemented in hardware. These page tables contain guest physical to machine physical memory address translations.

While this hardware support obviates the need for maintaining shadow page tables (and the associated performance overhead) it introduces some costs of its own: TLB miss costs, in the form of increased latency, are higher because of the extra level of page tables. The number of TLB misses can be reduced by using large memory pages, a feature which has been available since ESX 3.5. Because TLB miss latency is higher with this hardware virtualization assist, use of large pages by the application in the virtual machine and availability of large pages at the ESX level are a prerequisite for good database performance.

The following papers contain detailed descriptions of the hardware assist mechanisms and performance results from several benchmarks:

Performance Evaluation of AMD RVI Hardware Assist
http://www.vmware.com/pdf/RVI_performance.pdf

Performance Evaluation of Intel EPT Hardware Assist
http://www.vmware.com/pdf/Perf_ESX_Intel-EPT-eval.pdf

PVSCSI Adapter

Previous versions of ESX emulated BusLogic and LSI Logic storage adapters within the virtual machine. The virtual machine would load a BusLogic or LSI Logic driver within the guest to access the virtual storage adapter. The advantage of this full virtualization is that most operating systems ship drivers for these devices. ESX 4.0 includes a new paravirtualized storage adapter called PVSCSI. This extends to the storage stack the kind of performance gains associated with other paravirtualized devices, such as the VMXNET network adapter available in earlier versions of ESX.

As with other paravirtualized devices, PVSCSI emulation improves efficiency by:

- Reducing the cost of virtual interrupts
- Batching the processing of I/O requests
- Batching I/O completion interrupts

A further optimization, which is specific to virtual environments, reduces the number of context switches between the guest and Virtual Machine Monitor.

Improved I/O Concurrency

I/O requests issued by the guest are routed to the VMkernel via the virtual machine monitor (VMM).

Once the requests reach the VMkernel, they execute asynchronously. This has always been the case in ESX. The execution model in ESX 4.0 further improves I/O concurrency and allows vCPUs in the guest to execute other tasks immediately after initiating an I/O request.

This improved I/O concurrency model is designed around two ring structures per adapter which are shared by the virtual machine monitor (VMM) and the VMkernel. I/O requests from the guest are sent to the VMM where they are placed in the request ring. The VMkernel periodically initiates all new entries it finds there. The processing of the entries in the request ring is done asynchronously from the perspective of the guest.

I/O completions to the VMM are posted in the shared completion rings and an interrupt is generated for the guest operating system. By using this mechanism instead of requiring the guest to poll, these interrupts can be coalesced for delivery, which further optimizes I/O processing.

Virtual Interrupt Coalescing

The performance of high-IOPS workloads can be improved, in terms of higher throughput and lower CPU utilization, by turning on interrupt coalescing for Fibre Channel HBAs. This optimization has been in place since ESX 3.5.

Analogous to this, in order to further reduce the CPU cost of I/O in high-IOPS workloads, ESX 4.0 implements virtual interrupt coalescing to allow for some batching of I/O completions to happen at the guest level. In order to avoid the undesirable side-effect of increasing latency, this coalescing only occurs when I/O levels exceed a threshold. This threshold is set high enough that performance does not degrade under trickle I/O or latency-bound conditions.

Another optimization, related to inter-processor interrupt (IPI) coalescing, addresses a problem specific to virtual environments. Hardware interrupts that signal I/O completions are routed by the ESX storage stack to the virtual machine that issued the I/O. If the interrupt is not received on the physical CPU where the relevant VMM is executing, an IPI must be sent to the pCPU executing that VMM. As processor core density increases, it becomes more likely that hardware interrupts will be received on processors not running the target virtual machine, thus increasing the number of times the IPIs need to be issued. In ESX 4.0, we provide a mechanism to reduce the number of IPIs issued using the time-stamp of the last interrupt that was sent to the guest operating system. Before sending an IPI, we check the current time against this timestamp. If the time difference is greater than a threshold, 100µs by default, the IPI is posted. Otherwise, the IPI is delayed and the VMM has the opportunity to batch it with other I/O completions at a later time. This reduces the overall number of IPIs while bounding the latency of notifying the guest operating system about an I/O completion. As with other interrupts, lowering the rate of IPIs reduces the CPU cost for this benchmark.

For a detailed discussion of interrupt coalescing improvements, see the following publication:

Irfan Ahmad, Ajay Gulati, and Ali Mashtizadeh, “Improving Performance with Interrupt Coalescing for Virtual Machine Disk I/O in VMware ESX Server,” Second International Workshop on Virtualization Performance: Analysis, Characterization, and Tools (VPACT'09), held with IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS), 2009.

Virtual Interrupt Delivery Optimization

Optimized virtual interrupt delivery in ESX 4.0 improves the performance of some Linux guests. On an I/O completion, previous versions of ESX scheduled the vCPU that originated the I/O request. If this vCPU was found to not be the target vCPU chosen by the guest for further processing the I/O completion interrupt request, the VMM forwarded the I/O completion to the correct target vCPU with an additional inter-processor interrupt. ESX 4.0 avoids the need for these additional interrupts by predicting which vCPU is the target of a particular IRQ and delivering the I/O completion to it instead of to the originating vCPU. This reduces the CPU overhead substantially.

Resource Management

This benchmark is characterized by the heavy stress it places on the I/O and CPU subsystems and an even distribution of the workload across all CPUs. It also pushes the processor caches and TLB extremely hard. Due to the heavy I/O load, interrupt delivery and the associated processing costs are a large component of the overhead of this benchmark. All of these make efficient resource management critical to getting the highest possible throughput.

Compared to ESX 3.5, the ESX 4.0 scheduler includes several new features and enhancements that help improve the throughput of this benchmark. Relaxed co-scheduling of vCPUs is allowed in earlier releases of ESX but has been further fine-tuned in ESX 4.0. This is especially beneficial to efficient resource usage as the number of vCPUs in a virtual machine increases. In ESX 3.5, the scheduler would acquire a lock on a group of pCPUs within which vCPUs of a virtual machine were to be scheduled. In ESX 4.0 this has been replaced with finer-grained locking. It reduces scheduling overheads in cases where frequent scheduling decisions are needed.

Another significant improvement has been in the area of cache-aware scheduling of worlds. Worlds, in VMware terminology, are schedulable entities analogous to processes in conventional operating systems. A world is said to have migrated when it is scheduled on a core other than the one on which it last executed. When this happens, the hardware cache miss rates increase. Cache miss rates can be minimized by always scheduling a world on the same core. However this can result in delays in scheduling worlds as well as high CPU idle times. With intelligent world migrations, the scheduler in ESX 4.0 strikes a good balance between low CPU idle time and low cache miss rates. The scheduler also takes into account the processor cache architecture. This is especially important in light of the differences between the various processor architectures on the market. Migration algorithms have also been enhanced to take into account the load on the vCPUs and physical CPUs. These changes have helped to significantly improve throughput for this benchmark.

If you have comments about this documentation, submit your feedback to: docfeedback@vmware.com

VMware, Inc. 3401 Hillview Ave., Palo Alto, CA 94304 www.vmware.com

Revision: 20090420; Item: EN-000215-00