Performance Best Practices for VMware vSphere™ 4.1

VMware ESX™ 4.1 and ESXi 4.1

vCenter™ Server 4.1

EN-000005-03

You can find the most up-to-date technical documentation on the VMware Web site at:

http://www.vmware.com/support/

The VMware Web site also provides the latest product updates.

If you have comments about this documentation, submit your feedback to:

docfeedback@vmware.com

Revision: 20101028

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

Contents

Tables

About This Book

This book, *Performance Best Practices for VMware vSphere™ 4.1*, provides performance tips that cover the most performance-critical areas of VMware vSphere 4.1. It is not intended as a comprehensive guide for planning and configuring your deployments.

Chapter 1, "Hardware for Use with VMware vSphere," on page 9, provides guidance on selecting hardware for use with vSphere.

Chapter 2, "ESX and Virtual Machines," on page 15, provides guidance regarding VMware ESX™ software and the virtual machines that run in it.

Chapter 3, "Guest Operating Systems," on page 29, provides guidance regarding the guest operating systems running in vSphere virtual machines.

Chapter 4, "Virtual Infrastructure Management," on page 35, provides guidance regarding resource management best practices.

NOTE For planning purposes we recommend reading this entire book before beginning a deployment. Material in the Virtual Infrastructure Management chapter, for example, might influence your hardware choices.

NOTE Most of the recommendations in this book apply to both ESX and ESXi. The few exceptions are noted in the text.

Intended Audience

This book is intended for system administrators who are planning a VMware vSphere 4.1 deployment and want to maximize its performance. The book assumes the reader is already familiar with VMware vSphere concepts and terminology.

Document Feedback

VMware welcomes your suggestions for improving our documentation. If you have comments, send your feedback to:

docfeedback@vmware.com

VMware vSphere Documentation

The VMware vSphere documentation consists of the combined VMware vCenter™ and VMware ESX™ documentation set.

You can access the most current versions of this documentation by going to:

http://www.vmware.com/support/pubs

Conventions

Table 1 illustrates the typographic conventions used in this manual.

Table 1. Conventions Used in This Manual

Style	Elements
Blue (online only)	Links, cross-references, and email addresses
Black boldface	User interface elements such as button names and menu items
`Monospace`	Commands, filenames, directories, and paths
`Monospace bold`	User input
Italic	Document titles, glossary terms, and occasional emphasis
<Name>	Variable and parameter names

Hardware for Use with VMware vSphere 1

This chapter provides guidance on selecting and configuring hardware for use with VMware vSphere.

Validate Your Hardware

Before deploying a system we recommend the following:

- Verify that all hardware in the system is on the hardware compatibility list for the specific version of VMware software you will be running.
- Make sure that your hardware meets the minimum configuration supported by the VMware software you will be running.
- Test system memory for 72 hours, checking for hardware errors.

Hardware CPU Considerations

- When purchasing hardware, it is a good idea to consider CPU compatibility for VMware vMotion™ and VMware Fault Tolerance. See "VMware vCenter" on page 36.

Hardware-Assisted Virtualization

Most recent processors from both Intel and AMD include hardware features to assist virtualization. These features were released in two generations: the first generation introduced CPU virtualization; the second generation included CPU virtualization and added memory management unit (MMU) virtualization. For the best performance, make sure your system uses processors with at least second-generation hardware-assist features.

Hardware-Assisted CPU Virtualization (Intel VT-x and AMD AMD-V)

The first generation of hardware virtualization assistance, VT-x from Intel and AMD-V from AMD, became available in 2006. These technologies automatically trap sensitive calls, eliminating the overhead required to do so in software. This allows the use of a hardware virtualization (HV) virtual machine monitor (VMM) as opposed to a binary translation (BT) VMM.

For information about configuring the way ESX uses hardware-assisted CPU virtualization, see "Configuring ESX for Hardware-Assisted Virtualization" on page 20.

Hardware-Assisted MMU Virtualization (Intel EPT and AMD RVI)

More recent processors also include a feature that addresses the overheads due to memory management unit (MMU) virtualization by providing hardware support to virtualize the MMU. ESX versions 4.1 supports this feature both in AMD processors, where it is called rapid virtualization indexing (RVI) or nested page tables (NPT), and in Intel processors, where it is called extended page tables (EPT).

Without hardware-assisted MMU virtualization, the guest operating system maintains guest virtual memory to guest physical memory address mappings in guest page tables, while ESX maintains "shadow page tables" that directly map guest virtual memory to host physical memory addresses. These shadow page tables are maintained for use by the processor and are kept consistent with the guest page tables. This allows ordinary memory references to execute without additional overhead, since the hardware translation lookaside buffer (TLB) will cache direct guest virtual memory to host physical memory address translations read from the shadow page tables. However, extra work is required to maintain the shadow page tables.

Hardware-assisted MMU virtualization allows an additional level of page tables that map guest physical memory to host physical memory addresses, eliminating the need for ESX to maintain shadow page tables. This reduces memory consumption and speeds up workloads that cause guest operating systems to frequently modify page tables.

For information about configuring the way ESX uses hardware-assisted MMU virtualization, see "Configuring ESX for Hardware-Assisted Virtualization" on page 20.

Hardware-Assisted I/O MMU Virtualization (Intel VT-d and AMD AMD-Vi)

An even newer processor feature is an I/O memory management unit that remaps I/O DMA transfers and device interrupts. This can allow virtual machines to have direct access to hardware I/O devices, such as network cards. In AMD processors this feature is called AMD I/O Virtualization (AMD-Vi or IOMMU) and in Intel processors the feature is called Intel Virtualization Technology for Directed I/O (VT-d).

For information about using hardware-assisted I/O MMU virtualization, see "VMDirectPath I/O" on page 28.

Hardware Storage Considerations

Back-end storage configuration can greatly affect performance. Refer to the following sources for more information:

- For Fibre Channel SAN best practices: the *Fibre Channel SAN Configuration Guide* for 4.1.
- For iSCSI best practices: the *iSCSI SAN Configuration Guide* for 4.1.

Less than expected storage performance is most often the result of configuration issues with underlying storage devices rather than anything specific to ESX.

Storage performance is a vast topic that depends on workload, hardware, vendor, RAID level, cache size, stripe size, and so on. Consult the appropriate documentation from VMware as well as the storage vendor.

Many workloads are very sensitive to the latency of I/O operations. It is therefore important to have storage devices configured correctly. The remainder of this section lists practices and configurations recommended by VMware for optimal storage performance.

- In order to use VMware Storage vMotion your storage infrastructure must provide sufficient available storage bandwidth. For the best Storage vMotion performance you should make sure that the available bandwidth will be well above the minimum required. We therefore recommend you consider the information in “VMware vMotion and Storage vMotion” on page 39 when planning a deployment.
- Consider choosing storage arrays that support VMware vStorage APIs for Array Integration (VAAI). VAAI, new in vSphere 4.1, significantly improves storage scalability by offloading the following operations to the storage device instead of performing them in ESX:
  - Hardware-accelerated cloning (sometimes called “full copy” or “copy offload”) frees resources on the host and significantly speeds up workloads that rely on cloning, such as Storage vMotion.
  - Block zeroing speeds up creation of eager-zeroed thick disks and improves first-time write performance on lazy-zeroed thick disks and on thin disks.
  - Scalable lock management reduces locking-related overheads, speeding up thin-disk expansion as well as many other administrative and file system-intensive tasks. This helps improve the scalability of very large deployments by speeding up provisioning operations like boot storm, expansion of thin disks, snapshots, and other tasks.

  These features can reduce storage latency for several types of storage operations, can reduce the ESX host CPU utilization for storage operations, and can reduce storage network traffic.

  For information about configuring the way ESX uses VAAI, see “ESX Storage Considerations” on page 25.
- Performance design for a storage network must take into account the physical constraints of the network, not logical allocations. Using VLANs or VPNs does not provide a suitable solution to the problem of link oversubscription in shared configurations. VLANs and other virtual partitioning of a network provide a way of logically configuring a network, but don't change the physical capabilities of links and trunks between switches.
- If you have heavy disk I/O loads, you may need to assign separate storage processors (SPs) to separate systems to handle the amount of traffic bound for storage.
- To optimize storage array performance, spread I/O loads over the available paths to the storage (across multiple host bus adapters (HBAs) and storage processors).
- Configure maximum queue depth for Fibre Channel HBA cards. For additional information see VMware KB article 1267.
- Applications or systems that write large amounts of data to storage, such as data acquisition or transaction logging systems, should not share Ethernet links to a storage device with other applications or systems. These types of applications perform best with dedicated connections to storage devices.

- For iSCSI and NFS, make sure that your network topology does not contain Ethernet bottlenecks, where multiple links are routed through fewer links, potentially resulting in oversubscription and dropped network packets. Any time a number of links transmitting near capacity are switched to a smaller number of links, such oversubscription is a possibility.

  Recovering from these dropped network packets results in large performance degradation. In addition to time spent determining that data was dropped, the retransmission uses network bandwidth that could otherwise be used for new transactions.

- For NFS and iSCSI, if the network switch deployed for the data path supports VLAN, it is beneficial to create a VLAN just for the ESX host's vmknic and the NFS/iSCSI server. This minimizes network interference from other packet sources.

- Be aware that with software-initiated iSCSI and NFS the network protocol processing takes place on the host system, and thus these might require more CPU resources than other storage options.

Hardware Networking Considerations

- Before undertaking any network optimization effort, you should understand the physical aspects of the network. The following are just a few aspects of the physical layout that merit close consideration:
  - Consider using server-class network interface cards (NICs) for the best performance.
  - Make sure the NICs are configured to use autonegotiation to set speed and duplex settings and are configured for full-duplex mode.
  - Make sure the network infrastructure between the source and destination NICs doesn't introduce bottlenecks. For example, if both NICs are 1 Gigabit, make sure all cables and switches are capable of the same speed and that the switches are not configured to a lower speed.
- For the best networking performance, we recommend the use of network adapters that support the following hardware features:
  - Checksum offload
  - TCP segmentation offload (TSO)
  - Ability to handle high-memory DMA (that is, 64-bit DMA addresses)
  - Ability to handle multiple Scatter Gather elements per Tx frame
  - Jumbo frames (JF)
  - Large receive offload (LRO)
- On some 10 Gigabit Ethernet hardware network adapters ESX supports NetQueue, a technology that significantly improves performance of 10 Gigabit Ethernet network adapters in virtualized environments.
- In addition to the PCI and PCI-X bus architectures, we now have the PCI Express (PCIe) architecture. Ideally single-port 10 Gigabit Ethernet network adapters should use PCIe x8 (or higher) or PCI-X 266 and dual-port 10 Gigabit Ethernet network adapters should use PCIe x16 (or higher). There should preferably be no "bridge chip" (e.g., PCI-X to PCIe or PCIe to PCI-X) in the path to the actual Ethernet device (including any embedded bridge chip on the device itself), as these chips can reduce performance.
- Multiple physical network adapters between a single virtual switch (vSwitch) and the physical network constitute a NIC team. NIC teams can provide passive failover in the event of hardware failure or network outage and, in some configurations, can increase performance by distributing the traffic across those physical network adapters.

Hardware BIOS Settings

The default hardware BIOS settings on servers might not always be the best choice for optimal performance. This section lists some of the BIOS settings you might want to check.

NOTE ESX supports Enhanced Intel SpeedStep® and Enhanced AMD PowerNow!™ CPU power management technologies that can save power when a host is not fully utilized. However because these and other power-saving technologies can reduce multi-threaded performance and increase I/O latency in some situations, you should consider disabling them when these particular performance considerations outweigh power considerations.

NOTE Because of the large number of different server models and configurations, the BIOS options discussed below may not be comprehensive for your server.

- Make sure you are running the latest version of the BIOS available for your system.

  NOTE After updating the BIOS, you may need to make sure your BIOS settings are still as you wish.

- Make sure the BIOS is set to enable all populated sockets and to enable all cores in each socket.
- Enable "Turbo Mode" in the BIOS if your processors support it.
- Make sure hyper-threading is enabled in the BIOS for processors that support it.
- Some NUMA-capable systems provide an option in the BIOS to disable NUMA by enabling node interleaving. In most cases you will get the best performance by disabling node interleaving.
- Make sure any hardware-assisted virtualization features (VT-x, AMD-V, EPT, RVI, and so on) are enabled in the BIOS.

  NOTE After these changes are made some systems may need a complete power down before the changes take effect. See http://communities.vmware.com/docs/DOC-8978 for details.

- Disable C1E halt state in the BIOS if multi-threaded performance and I/O latency are important considerations. Consider enabling C1E if "Turbo Mode" is present and single-threaded performance of certain workloads is more relevant. (See note above regarding performance considerations versus power considerations.)
- For the highest performance, potentially at the expense of higher power consumption, set any BIOS power-saving options to high-performance mode. Depending on the manufacturer, this might involve disabling an option (for example, "Demand-Based Power Management") or enabling an option (for example, "Static High Performance Mode").
- Disable from within the BIOS any unneeded devices, such as serial and USB ports.

ESX and Virtual Machines 2

This chapter provides guidance regarding ESX software itself and the virtual machines that run in it.

ESX General Considerations

This subsection provides guidance regarding a number of general performance considerations in ESX.

- Plan the deployment. Allocate enough resources, especially (in the case of ESX, but not ESXi) for the service console. Table 2-1 lists symptoms of insufficient resources for the service console.

  Table 2-1. Symptoms of Insufficient Resources for the ESX Service Console

Insufficient Resource Type	Symptoms
Memory	Poor response from the vSphere Client
Disk Space	Inability to write diagnostic messages to log; inability to log into the service console

- Allocate only as much virtual hardware as required for each virtual machine. Provisioning a virtual machine with more resources than it requires can, in some cases, reduce the performance of that virtual machine as well as other virtual machines sharing the same host.
- Disconnect or disable unused or unnecessary physical hardware devices, such as:
  - COM ports
  - LPT ports
  - USB controllers
  - Floppy drives
  - Optical drives (that is, CD or DVD drives)
  - Network interfaces
  - Storage controllers

  Disabling hardware devices (typically done in BIOS) can free interrupt resources. Additionally, some devices, such as USB controllers, operate on a polling scheme that consumes extra CPU resources. Lastly, some PCI devices reserve blocks of memory, making that memory unavailable to ESX.
- Unused or unnecessary virtual hardware devices can impact performance and should be disabled.

  For example, Windows guest operating systems poll optical drives (that is, CD or DVD drives) quite frequently. When virtual machines are configured to use a physical drive, and multiple guest operating systems simultaneously try to access that drive, performance could suffer. This can be reduced by configuring the virtual machines to use ISO images instead of physical drives, and can be avoided entirely by disabling optical drives in virtual machines when the devices are not needed.

- ESX 4.0 introduced virtual hardware version 7. By creating virtual machines using this hardware version, or upgrading existing virtual machines to this version, a number of additional capabilities become available. Some of these, such as VMXNET3 and PVSCSI, can improve performance for many workloads. This hardware version is not compatible with versions of ESX prior to 4.0, however, and thus if a cluster of ESX hosts will contain some hosts running pre-4.0 versions of ESX, the virtual machines running on hardware version 7 will be constrained to run only on the ESX 4.x hosts. This could limit vMotion choices for Distributed Resource Scheduling (DRS) or Distributed Power Management (DPM).

ESX CPU Considerations

This subsection provides guidance regarding CPU considerations in VMware ESX.

CPU virtualization adds varying amounts of overhead depending on the percentage of the virtual machine's workload that can be executed on the physical processor as is and the cost of virtualizing the remainder of the workload:

- For many workloads, CPU virtualization adds only a very small amount of overhead, resulting in performance essentially comparable to native.
- Many workloads to which CPU virtualization does add overhead are not CPU-bound — that is, most of their time is spent waiting for external events such as user interaction, device input, or data retrieval, rather than executing instructions. Because otherwise-unused CPU cycles are available to absorb the virtualization overhead, these workloads will typically have throughput similar to native, but potentially with a slight increase in latency.
- For a small percentage of workloads, for which CPU virtualization adds overhead and which are CPU-bound, there might be a noticeable degradation in both throughput and latency.

The rest of this subsection lists practices and configurations recommended by VMware for optimal CPU performance.

- In most environments ESX allows significant levels of CPU overcommitment (that is, running more vCPUs on a host than the total number of physical CPUs in that host) without impacting virtual machine performance.

  If an ESX host becomes CPU saturated (that is, the virtual machines and other loads on the host demand all the CPU resources the host has), latency-sensitive workloads might not perform well. In this case you might want to reduce the CPU load, for example by powering off some virtual machines or migrating them to a different host (or allowing DRS to migrate them automatically).

- ESX reserves for the console a small amount of CPU resources. Very rarely these CPU resources might not be sufficient, resulting in poor console responsiveness. This might happen, for example, when the console has an unusually heavy load and the host CPUs are simultaneously fully committed. (Because it doesn't have a console, this point doesn't apply to ESXi.)

  In this case, you can use CPU reservations to guarantee more CPU availability to the console (which will, however, reduce the CPU resources available for other purposes, potentially slowing down virtual machines running on that host or limiting the host's ability to power-on additional virtual machines) or migrate some virtual machines away from that host.

- It is a good idea to periodically monitor the CPU usage of the host. This can be done through the vSphere Client or by using `esxtop` or `resextop`. Below we describe how to interpret `esxtop` data:

  - If the load average on the first line of the `esxtop` CPU panel is equal to the number of physical processors in the system, this indicates that the system is overloaded.
  - The usage percentage for the physical CPUs on the PCPU line can be another indication of a possibly overloaded condition. In general, 80% usage is a reasonable ceiling and 90% should be a warning that the CPUs are approaching an overloaded condition. However organizations will have varying standards regarding the desired load percentage.

  For information about using `esxtop` or `resextop` see Appendix A of the VMware *Resource Management Guide*.

- Configuring a virtual machine with more virtual CPUs (vCPUs) than its workload can use might cause slightly increased resource usage, potentially impacting performance on very heavily loaded systems. Common examples of this include a single-threaded workload running in a multiple-vCPU virtual machine or a multi-threaded workload in a virtual machine with more vCPUs than the workload can effectively use.

Even if the guest operating system doesn't use some of its vCPUs, configuring virtual machines with those vCPUs still imposes some small resource requirements on ESX that translate to real CPU consumption on the host. For example:

- Unused vCPUs still consume timer interrupts in some guest operating systems. (Though this is not true with "tickless timer" kernels, described in "Guest Operating System CPU Considerations" on page 31.)
- Maintaining a consistent memory view among multiple vCPUs can consume additional resources, both in the guest operating system and in ESX. (Though hardware-assisted MMU virtualization significantly reduces this cost.)
- Most guest operating systems execute an idle loop during periods of inactivity. Within this loop, most of these guest operating systems halt by executing the HLT or MWAIT instructions. Some older guest operating systems, however, use busy-waiting within their idle loops. This results in the consumption of resources that might otherwise be available for other uses (other virtual machines, the VMkernel, the console, etc.). (These older operating systems include Windows 2000 (with certain HALs), Solaris 8 and 9, and MS-DOS.) For additional information see VMware KB articles 1077 and 2231.
- The guest operating system's scheduler might migrate a single-threaded workload amongst multiple vCPUs, thereby losing cache locality.

These resource requirements translate to real CPU consumption on the host.

UP vs. SMP HALs/Kernels

- Although some recent operating systems (including Windows Vista, Windows Server 2008, and Windows 7) use the same HAL (hardware abstraction layer) or kernel for both UP and SMP installations, many operating systems can be configured to use either a UP HAL/kernel or an SMP HAL/kernel. To obtain the best performance on a single-vCPU virtual machine running an operating system that offers both UP and SMP HALs/kernels, configure the operating system with a UP HAL or kernel.

  The UP operating system versions are for single-processor systems. If used on a multiprocessor system, a UP operating system version will recognize and use only one of the processors. The SMP versions, while required in order to fully utilize multiprocessor systems, may also be used on single-processor systems. Due to their extra synchronization code, however, SMP operating system versions used on single-processor systems are slightly slower than UP operating system versions used on the same systems.

  NOTE When changing an existing virtual machine running Windows from multiprocessor to single-processor the HAL usually remains SMP.

Hyper-Threading

- Hyper-threading technology (recent versions of which are called symmetric multithreading, or SMT) allows a single physical processor core to behave like two logical processors, essentially allowing two independent threads to run simultaneously. Unlike having twice as many processor cores—that can roughly double performance—hyper-threading can provide anywhere from a slight to a significant increase in system performance by keeping the processor pipeline busier.

  If the hardware and BIOS support hyper-threading, ESX automatically makes use of it. For the best performance we recommend that you enable hyper-threading, which can be accomplished as follows:

  a. Ensure that your system supports hyper-threading technology. It is not enough that the processors support hyper-threading — the BIOS must support it as well. Consult your system documentation to see if the BIOS includes support for hyper-threading.

  b. Enable hyper-threading in the system BIOS. Some manufacturers label this option **Logical Processor** while others label it **Enable Hyper-threading**.

- An ESX system enabled for hyper-threading will behave almost exactly like a system without it. Logical processors on the same core have adjacent CPU numbers, so that CPUs 0 and 1 are on the first core, CPUs 2 and 3 are on the second core, and so on.

  ESX systems manage processor time intelligently to guarantee that load is spread smoothly across all physical cores in the system. If there is no work for a logical processor it is put into a special halted state that frees its execution resources and allows the virtual machine running on the other logical processor on the same core to use the full execution resources of the core.

- Be careful when using CPU affinity on systems with hyper-threading. Because the two logical processors share most of the processor resources, pinning vCPUs, whether from different virtual machines or from one SMP virtual machine, to both logical processors on one core (CPUs 0 and 1, for example) could cause poor performance.

- ESX provides configuration parameters for controlling the scheduling of virtual machines on hyper-threaded systems. When choosing hyper-threaded core sharing choices, the **Any** option (which is the default) is almost always preferred over **None** or **Internal**.

  The **None** option indicates that when a vCPU from this virtual machine is assigned to a logical processor, no other vCPU, whether from the same virtual machine or from a different virtual machine, should be assigned to the other logical processor that resides on the same core. That is, each vCPU from this virtual machine should always get a whole core to itself and the other logical CPU on that core should be placed in the halted state. This option is like disabling hyper-threading for that one virtual machine.

  The **Internal** option indicates that the vCPUs of the virtual machine can share cores with each other, but not with vCPUs from other virtual machines.

  For nearly all workloads, custom hyper-threading settings are not necessary. In cases of unusual workloads that interact badly with hyper-threading, however, choosing the **None** or **Internal** hyper-threading option might help performance. For example, an application with cache-thrashing problems might slow down applications sharing its physical CPU core. In this case configuring the virtual machine running that problem application with the **None** hyper-threading option might help isolate that virtual machine from other virtual machines.

  The trade-off for configuring **None** or **Internal** should also be considered. With either of these settings, there can be cases where there is no core to which a descheduled virtual machine can be migrated, even though one or more logical cores are idle. As a result, it is possible that virtual machines with hyper-threading set to **None** or **Internal** can experience performance degradation, especially on systems with a limited number of CPU cores.

- The ESX scheduler ensures that each virtual machine receives its fair share of CPU resources while enforcing the resource controls, such as reservations, limits, and shares. On hyper-threaded processors the amount of CPU resources a vCPU receives is affected by the activity on the other logical processor on the same physical core. To guarantee that each vCPU receives its fair share, ESX sometimes schedules just one vCPU on a hyper-threaded core, effectively leaving one of the logical processors idle. As a result, ESX might not take full advantage of the available CPU resources. This happens rarely, and only when the system has high CPU utilization, has close to full CPU commitment (that is, the number of vCPUs is nearly equal to the number of physical cores), and has specific types of bursty CPU usage patterns.

  In these situations, a parameter called HaltingIdleMsecPenalty can be used to adjust the behavior of the fairness algorithm. For additional details, see VMware KB article 1020233.

Non-Uniform Memory Access (NUMA)

VMware vSphere supports AMD (Opteron-based), Intel (Nehalem), and IBM (X-Architecture) non-uniform memory access (NUMA) systems.

NOTE On some systems BIOS settings for node interleaving (also known as interleaved memory) determine whether the system behaves like a NUMA system or like a uniform memory accessing (UMA) system. If node interleaving is disabled, ESX detects the system as NUMA and applies NUMA optimizations. If node interleaving is enabled, ESX does not detect the system as NUMA. For more information, refer to your server's documentation.

The intelligent, adaptive NUMA scheduling and memory placement policies in ESX can manage all virtual machines transparently, so that administrators don't need to deal with the complexity of balancing virtual machines between nodes by hand. Manual override controls are available, however, and advanced administrators may prefer to control the memory placement (through the **Memory Affinity** option) and processor utilization (through the **Only Use Processors** option).

By default, ESX NUMA scheduling and related optimizations are enabled only on systems with a total of at least four CPU cores and with at least two CPU cores per NUMA node.

On such systems, virtual machines can be separated into the following two categories:

- Virtual machines with a number of vCPUs equal to or less than the number of cores in each NUMA node. These virtual machines will be assigned to cores all within a single NUMA node and will be preferentially allocated memory local to that NUMA node. This means that, subject to memory availability, all their memory accesses will be local to that NUMA node, resulting in the lowest memory access latencies.
- Virtual machines with more vCPUs than the number of cores in a NUMA node (called "wide virtual machines," a new feature in vSphere 4.1). These virtual machines will be assigned to two (or more) NUMA nodes and will be preferentially allocated memory local to those NUMA nodes. Because vCPUs in these wide virtual machines might sometimes need to access memory outside their own NUMA node, they might experience higher average memory access latencies than virtual machines that fit entirely within a NUMA node.

Because of this difference, there can be a slight performance advantage in some environments to virtual machines configured with no more vCPUs than the number of cores in each NUMA node.

Conversely, some memory bandwidth bottlenecked workloads benefit from the increased aggregate memory bandwidth available when a virtual machine is split across multiple NUMA nodes. This split can be accomplished by using the `numa.vcpu.maxPerMachineNode` option.

More information about using NUMA systems with ESX can be found in the *vSphere Resource Management Guide* for 4.1.

Configuring ESX for Hardware-Assisted Virtualization

For a description of hardware-assisted virtualization, see "Hardware-Assisted Virtualization" on page 9.

On processors that support hardware-assisted CPU virtualization but not hardware-assisted MMU virtualization, ESX by default chooses between the binary translation (BT) virtual machine monitor (VMM) and a hardware virtualization (HV) VMM based on the processor model and the guest operating system, providing the best performance in the majority of cases. If desired, however, this behavior can be changed, as described below.

On processors that support hardware-assisted MMU virtualization, ESX 4.x by default uses it for virtual machines running certain guest operating systems and uses shadow page tables for others. This default behavior should work well for the majority of guest operating systems and workloads. If desired, however, this behavior can be changed, as described below.

NOTE When hardware-assisted MMU virtualization is enabled for a virtual machine we strongly recommend you also—when possible—configure that virtual machine's guest operating system and applications to make use of large memory pages.

When running on a system with hardware-assisted MMU virtualization enabled ESX will attempt to use large pages to back the guest's memory pages even if the guest operating system and applications don't make use of large memory pages. For more information about large pages, see "Large Memory Pages for Hypervisor and Guest Operating System" on page 24.

The default behavior of ESX regarding hardware-assisted virtualization can be changed using the vSphere Client. To do so:

1. Select the virtual machine to be configured.
2. Click **Edit virtual machine settings**, choose the **Options** tab, and select **CPU/MMU Virtualization**.

3 Select the desired radio button:

- **Automatic** allows ESX to determine the best choice. This is the default; http://communities.vmware.com/docs/DOC-9882 provides a detailed list of which VMM is chosen for each combination of CPU and guest operating system.
- **Use software for instruction set and MMU virtualization** disables both hardware-assisted CPU virtualization (VT-x/AMD-V) and hardware-assisted MMU virtualization (EPT/RVI).
- **Use Intel® VT-x/AMD-V™ for instruction set virtualization and software for MMU virtualization** enables hardware-assisted CPU virtualization (VT-x/AMD-V) but disables hardware-assisted MMU virtualization (EPT/RVI).
- **Use Intel® VT-x/AMD-V™ for instruction set virtualization and Intel® EPT/AMD RVI for MMU virtualization** enables both hardware-assisted CPU virtualization (VT-x/AMD-V) and hardware-assisted MMU virtualization (EPT/RVI).

NOTE Some combinations of CPU, guest operating system, and other variables (i.e, turning on Fault Tolerance or using VMI) limit these options. If the setting you select is not available for your particular combination, the setting will be ignored and **Automatic** will be used.

ESX Memory Considerations

This subsection provides guidance regarding memory considerations in ESX.

Memory Overhead

Virtualization causes an increase in the amount of physical memory required due to the extra memory needed by ESX for its own code and for data structures. This additional memory requirement can be separated into two components:

1. A system-wide memory space overhead for the service console and the VMkernel on ESX or for the VMkernel and various host agents (hostd, vpxa, etc.) on ESXi.
2. An additional memory space overhead for each virtual machine.

   The per-virtual-machine memory space overhead includes space reserved for the virtual machine devices (for example, the SVGA frame buffer and several internal data structures maintained by the VMware software stack). These amounts depend on the number of vCPUs, the configured memory for the guest operating system, whether the guest operating system is 32-bit or 64-bit, and which features are enabled for the virtual machine. For more information about these overheads, see the *vSphere Resource Management Guide* for 4.1.

However ESX also provides optimizations, such as memory sharing (see Memory Overcommit Techniques), to reduce the amount of physical memory used on the underlying server. In some cases these optimizations can save more memory than is taken up by the overhead.

Memory Sizing

- Carefully select the amount of memory you allocate to your virtual machines. You should allocate enough memory to hold the working set of applications you will run in the virtual machine, thus minimizing swapping, but avoid over-allocating memory. Allocating more memory than needed unnecessarily increases the virtual machine memory overhead, thus using up memory that could be used to support more virtual machines.
- When possible, configure 32-bit Linux virtual machines to have no more than 896MB of memory. 32-bit Linux kernels use different techniques to map memory on systems with more than 896MB. These techniques impose additional overhead on the virtual machine monitor and can result in slightly reduced performance.

Memory Overcommit Techniques

- ESX uses four memory management mechanisms—page sharing, ballooning, swapping, and memory compression—to dynamically reduce the amount of machine physical memory required for each virtual machine.
  - **Page Sharing:** ESX uses a proprietary technique to transparently and securely share memory pages between virtual machines, thus eliminating redundant copies of memory pages. Page sharing is used by default regardless of the memory demands on the host system.
  - **Ballooning:** If the virtual machine's memory usage approaches its memory target, ESX uses ballooning to reduce that virtual machine's memory demands. Using a VMware-supplied vmmemctl module installed in the guest operating system as part of VMware Tools suite, ESX can cause the guest operating system to relinquish the memory pages it considers least valuable. Ballooning provides performance closely matching that of a native system under similar memory constraints. To use ballooning, the guest operating system must be configured with sufficient swap space.
  - **Swapping:** If the host free memory gets very low, ESX uses host-level swapping to forcibly reclaim memory from a virtual machine. Because this might swap out active pages, it can cause virtual machine performance to degrade significantly.

- **Memory Compression**: In conjunction with host-level swapping, ESX uses memory compression to reduce the number of memory pages it will need to swap out. Because the decompression latency is much smaller than the swap-in latency, compressing a memory page instead of swapping it out can significantly improve performance when a host has heavy memory overcommitment.

  For further information about memory management, see *Understanding Memory Resource Management in VMware ESX 4.1*.

- While ESX uses page sharing and ballooning to allow significant memory overcommitment, usually with little or no impact on performance, you should avoid overcommitting memory to the point that it requires host-level swapping.

  If you suspect that memory overcommitment is beginning to affect the performance of a virtual machine you can:

  a Look at the value of **Memory Balloon (Average)** in the vSphere Client Performance Chart.
  An absence of ballooning suggests that ESX is not under heavy memory pressure and thus memory overcommitment is not affecting performance. (Note, however, that some ballooning is quite normal and not indicative of a problem.)

  b Check for guest swap activity within that virtual machine.
  This can indicate that ballooning might be starting to impact performance, though swap activity can also be related to other issues entirely within the guest (or can be an indication that the guest memory size is simply too small).

  c Look at the value of **Memory Swap Used (Average)** in the vSphere Client Performance Chart.
  Memory swapping at the host level would indicate more significant memory pressure.

Memory Swapping

As described in "Memory Overcommit Techniques," above, ESX supports a bounded amount of memory overcommitment without swapping. If the overcommitment is large enough that the other memory reclamation techniques are not sufficient, however, ESX uses host-level memory swapping, with a potentially significant effect on virtual machine performance. (Note that this swapping is distinct from the swapping that can occur within the virtual machine under the control of the guest operating system.)

This subsection describes ways to avoid or reduce host-level swapping, and presents techniques to reduce its impact on performance when it is unavoidable.

- Because ESX uses page sharing, ballooning, and memory compression to reduce the need for swapping, don't disable these techniques.

- Host-level memory swapping can be avoided for a specific virtual machine by using the vSphere Client to reserve memory for that virtual machine at least equal in size to the machine's active working set. Be aware, however, that configuring resource reservations will reduce the number of virtual machines that can be run on a system. This is because ESX will keep available enough host memory to fulfill all reservations and won't power-on a virtual machine if doing so would reduce the available memory to less than the reserved amount.

  NOTE The memory reservation is a guaranteed lower bound on the amount of physical memory ESX reserves for the virtual machine. It can be configured through the vSphere Client in the settings window for each virtual machine (select **Edit virtual machine settings**, choose the **Resources** tab, select **Memory**, then set the desired reservation).

- If you choose to overcommit memory with ESX, be sure you have sufficient swap space on your ESX system. At the time a virtual machine is first powered on, ESX creates a swap file for that virtual machine equal in size to the difference between the virtual machine's configured memory size and its memory reservation. The available disk space must therefore be at least this large.

- If swapping cannot be avoided, placing the virtual machine's swap file on a low latency, high bandwidth storage system will result in the smallest performance impact. A fast local disk might be a good choice, as would an SSD disk. In any case, for best performance swap files should not be placed on storage backed by thin-provisioned LUNs.

  The swap file location can be set with the **sched.swap.dir** option in the vSphere Client (select **Edit virtual machine settings**, choose the **Options** tab, select **Advanced**, and click **Configuration Parameters**). If this option is not set, the swap file will be created in the virtual machine's working directory: either the directory specified by `workingDir` in the virtual machine's `.vmx` file, or, if this variable is not set, in the directory where the `.vmx` file is located. The latter is the default behavior.

Large Memory Pages for Hypervisor and Guest Operating System

In addition to the usual 4KB memory pages, ESX also provides 2MB memory pages (commonly referred to as "large pages"). By default ESX assigns these 2MB machine memory pages to guest operating systems that request them, giving the guest operating system the full advantage of using large pages. The use of large pages results in reduced memory management overhead and can therefore increase hypervisor performance.

If an operating system or application can benefit from large pages on a native system, that operating system or application can potentially achieve a similar performance improvement on a virtual machine backed with 2MB machine memory pages. Consult the documentation for your operating system and application to determine how to configure each of them to use large memory pages.

Use of large pages can also change page sharing behavior. ESX ordinarily uses page sharing regardless of memory demands, but does not share large pages. Therefore with large pages, page sharing might not occur until memory overcommitment is high enough to require the large pages to be broken into small pages. For further information see VMware KB articles 1021095 and 1021896.

More information about large page support can be found in the performance study entitled *Large Page Performance* (available at http://www.vmware.com/resources/techresources/1039).

Hardware-Assisted MMU Virtualization

Hardware-assisted MMU virtualization is a technique that virtualizes the CPU's memory management unit (MMU). For a description of hardware-assisted MMU virtualization, see "Hardware-Assisted MMU Virtualization (Intel EPT and AMD RVI)" on page 9; for information about configuring the way ESX uses hardware-assisted MMU virtualization, see "Configuring ESX for Hardware-Assisted Virtualization" on page 20.

ESX Storage Considerations

This subsection provides guidance regarding storage considerations in ESX.

VMware vStorage APIs for Array Integration (VAAI)

- For the best storage performance, consider using VAAI-capable storage arrays. The performance gains from VAAI (described in "Hardware Storage Considerations" on page 11) are especially noticeable in VDI environments (where VAAI improves boot-storm and desktop workload performance), large data centers (where VAAI improves the performance of mass virtual machine provisioning and of thin-provisioned virtual disks), and in other large-scale deployments.

  If your storage array supports VAAI ESX will automatically recognize and use these capabilities. To confirm that your array does support VAAI and that it is being used, follow the instructions in VMware KB article 1021976. If you determine that VAAI is not being used, contact your array vendor to see if a firmware upgrade is required for VAAI support.

LUN Access Methods, Virtual Disk Modes, and Virtual Disk Types

- ESX supports raw device mapping (RDM), which allows management and access of raw SCSI disks or LUNs as VMFS files. An RDM is a special file on a VMFS volume that acts as a proxy for a raw device. The RDM file contains metadata used to manage and redirect disk accesses to the physical device. Ordinary VMFS is recommended for most virtual disk storage, but raw disks may be desirable in some cases.
- ESX supports three virtual disk modes: Independent persistent, Independent nonpersistent, and Snapshot. These modes have the following characteristics:
  - **Independent persistent** – In this mode changes are immediately written to the disk, providing the best performance.
  - **Independent nonpersistent** – In this mode disk writes are appended to a redo log. The redo log is erased when you power off the virtual machine or revert to a snapshot, causing any changes made to the disk to be discarded. When a virtual machine reads from an independent nonpersistent mode disk, ESX first checks the redo log (by looking at a directory of disk blocks contained in the redo log) and, if the relevant blocks are listed, reads that information. Otherwise, the read goes to the base disk for the virtual machine. Because of these redo logs, which track the changes in a virtual machine's file system and allow you to commit changes or revert to a prior point in time, performance might not be as high as independent persistent mode disks.
  - **Snapshot** – In this mode disk writes are appended to a redo log that persists between power cycles. Thus, like the independent nonpersistent mode disks described above, snapshot mode disk performance might not be as high as independent persistent mode disks.
- ESX supports multiple virtual disk types:
  - **Thick** – Thick virtual disks, which have all their space allocated at creation time, are further divided into two types: eager zeroed and lazy zeroed.
    - **Eager-zeroed** – An eager-zeroed thick disk has all space allocated and zeroed out at the time of creation. This extends the time it takes to create the disk, but results in the best performance, even on the first write to each block.

      NOTE The use of VAAI-capable storage (described in "Hardware Storage Considerations" on page 11) speeds up eager-zeroed thick disk creation by offloading zeroing operations to the storage array.

    - **Lazy-zeroed** – A lazy-zeroed thick disk has all space allocated at the time of creation, but each block is only zeroed on first write. This results in a shorter creation time, but reduced performance the first time a block is written to. Subsequent writes, however, have the same performance as on eager-zeroed thick disks.

      NOTE The use of VAAI-capable storage improves lazy-zeroed thick disk first-time-write performance by offloading zeroing operations to the storage array.

- **Thin** – Space required for a thin-provisioned virtual disk is allocated and zeroed upon first write, as opposed to upon creation. There is a higher I/O cost during the first write to an unwritten file block, but thin-provisioned disks have the same performance as eager-zeroed thick disks on subsequent writes.

  NOTE The use of VAAI capable storage improves thin-provisioned disk first-time-write performance by improving file locking capability and offloading zeroing operations to the storage array.

Virtual disks created using the vSphere Client (whether connected directly to the ESX host or through vCenter) can be lazy-zeroed thick disks (thus incurring the first-write performance penalty described above) or thin disks. Eager-zeroed thick disks can be created from the console command line using `vmkfstools`. For more details refer to the `vmkfstools` man page.

NOTE Virtual disks created on NFS volumes are always thin-provisioned.

Partition Alignment

The alignment of file system partitions can impact performance. VMware makes the following recommendations for VMFS partitions:

- Like other disk-based file systems, VMFS suffers a performance penalty when the partition is unaligned. Using the vSphere Client to create VMFS partitions avoids this problem since it automatically aligns the partitions along the 64KB boundary.
- To manually align your VMFS partitions, check your storage vendor's recommendations for the partition starting block. If your storage vendor makes no specific recommendation, use a starting block that is a multiple of 8KB.
- Before performing an alignment, carefully evaluate the performance impact of the unaligned VMFS partition on your particular workload. The degree of improvement from alignment is highly dependent on workloads and array types. You might want to refer to the alignment recommendations from your array vendor for further information.

SAN Multipathing

- By default, ESX uses the **Most Recently Used** (MRU) path policy for devices on Active/Passive storage arrays. Do not use **Fixed** path policy for Active/Passive storage arrays to avoid LUN path thrashing. For more information, see the *VMware SAN Configuration Guide*.
- By default, ESX uses the **Fixed** path policy for devices on Active/Active storage arrays. When using this policy you can maximize the utilization of your bandwidth to the storage array by designating preferred paths to each LUN through different storage controllers. For more information, see the *VMware SAN Configuration Guide*.
- In addition to the Fixed and MRU path policies, ESX can also use the **Round Robin** path policy, which can improve storage performance in some environments. Round Robin policy provides load balancing by cycling I/O requests through all Active paths, sending a fixed (but configurable) number of I/O requests through each one in turn.

- If your storage array supports ALUA (Asymmetric Logical Unit Access), enabling this feature on the array can improve storage performance in some environments. ALUA, which is automatically detected by ESX, allows the array itself to designate paths as "Active Optimized." When ALUA is combined with the Round Robin path policy, ESX cycles I/O requests through these Active Optimized paths.

Storage I/O Resource Allocation

VMware vSphere 4.1 provides mechanisms to dynamically allocate storage I/O resources, allowing critical workloads to maintain their performance even during peak load periods when there is contention for I/O resources. This allocation can be performed at the level of the individual host or for an entire datastore. Both methods are described below.

- The storage I/O resources available to an ESX host can be proportionally allocated to the virtual machines running on that host by using the vSphere Client to set disk shares for the virtual machines (select **Edit virtual machine settings**, choose the **Resources** tab, select **Disk**, then change the **Shares** field).
- An entire datastore's I/O resources can be proportionally allocated to the virtual machines accessing that datastore using Storage I/O Control (SIOC), a new feature in vSphere 4.1. When enabled, SIOC evaluates the disk share values set for all virtual machines accessing a datastore and allocates that datastore's resources accordingly. SIOC can be enabled using the vSphere Client (select a datastore, choose the **Configuration** tab, click **Properties...** (at the far right), then under **Storage I/O Control** add a checkmark to the **Enabled** box).

With SIOC disabled (the default), all hosts accessing a datastore get an equivalent portion of that datastore's resources. Any shares values determine only how each host's portion is divided amongst its virtual machines.

With SIOC enabled, the disk shares are evaluated globally and the portion of the datastore's resources each host receives depends on the sum of the shares of the virtual machines running on that host relative to the sum of the shares of all the virtual machines accessing that datastore.

General ESX Storage Recommendations

- I/O latency statistics can be monitored using `esxtop` (or `resxtop`), which reports device latency, time spent in the kernel, and latency seen by the guest.
- Make sure I/O is not queueing up in the VMkernel by checking the number of queued commands reported by `esxtop` (or `resxtop`).

  Since queued commands are an instantaneous statistic, you will need to monitor the statistic over a period of time to see if you are hitting the queue limit. To determine queued commands, look for the **QUED** counter in the `esxtop` (or `resxtop`) storage resource screen. If queueing occurs, try adjusting queue depths. For further information see VMware KB article 1267.
- The driver queue depth can be set for some VMkernel drivers. For example, while both the Fibre Channel and iSCSI QLogic drivers have a default queue depth of 32, specifying a larger queue depth might yield higher performance. You can also adjust the maximum number of outstanding disk requests per virtual machine in the VMkernel through the vSphere Client. Setting this parameter can help equalize disk bandwidth across virtual machines. For additional information see VMware KB article 1268.
- Don't allow your service console's root file system to become full. (Because it does not include a service console, this point doesn't apply to ESXi.)

ESX Networking Considerations

This subsection provides guidance regarding networking considerations in ESX.

General ESX Networking Considerations

- In a native environment, CPU utilization plays a significant role in network throughput. To process higher levels of throughput, more CPU resources are needed. The effect of CPU resource availability on the network throughput of virtualized applications is even more significant. Because insufficient CPU resources will limit maximum throughput, it is important to monitor the CPU utilization of high-throughput workloads.
- Use separate virtual switches, each connected to its own physical network adapter, to avoid contention between the ESX service console, the VMkernel, and virtual machines, especially virtual machines running heavy networking workloads.
- To establish a network connection between two virtual machines that reside on the same ESX system, connect both virtual machines to the same virtual switch. If the virtual machines are connected to different virtual switches, traffic will go through wire and incur unnecessary CPU and network overhead.

Network I/O Control (NetIOC)

- New in vSphere 4.1, Network I/O Control (NetIOC) allows the allocation of network bandwidth to six network resource groups: vMotion, NFS, iSCSI, Fault Tolerance, virtual machine, and management. Network bandwidth can be allocated using either shares or limits:
  - Using shares, each network traffic type is allowed a proportion of the network bandwidth equivalent to the proportion of its share value to the total shares. With shares, unused bandwidth is available for use by other traffic types.
  - Using limits, a maximum bandwidth utilization (in Mbps) is set for each traffic type. Unused bandwidth is not available for other traffic types.

  This feature can guarantee bandwidth for specific needs and can prevent any one traffic type from impacting the others.

  For further information about NetIOC, see *VMware Network I/O Control: Architecture, Performance and Best Practices*.

VMDirectPath I/O

- VMDirectPath I/O leverages Intel VT-d and AMD-Vi hardware support (described in "Hardware-Assisted I/O MMU Virtualization (Intel VT-d and AMD AMD-Vi)" on page 10) to allow guest operating systems to directly access hardware devices. In the case of networking, VMDirectPath I/O allows the virtual machine to access a physical NIC directly rather than using an emulated device (E1000) or a para-virtualized device (VMXNET, VMXNET3). While VMDirectPath I/O has limited impact on throughput, it reduces CPU cost for networking-intensive workloads. VMDirectPath I/O is incompatible with many core virtualization features, however. These include vMotion, Snapshots, Suspend/Resume, Fault Tolerance, NetIOC, Memory Overcommit, and VMSafe. Typical virtual machines and their workloads don't require the use of VMDirectPath I/O. For workloads that are very networking intensive and don't need the core virtualization features mentioned above, however, VMDirectPath I/O might be useful to reduce CPU usage.

3 Guest Operating Systems

This chapter provides guidance regarding the guest operating systems running in virtual machines.

Guest Operating System General Considerations

- Use guest operating systems that are supported by ESX. See the *Guest Operating System Installation Guide* for a list.

  NOTE VMware Tools might not be available for unsupported guest operating systems.

- Install the latest version of VMware Tools in the guest operating system. Make sure to update VMware Tools after each ESX upgrade.

  Installing VMware Tools in Windows guests updates the BusLogic SCSI driver included with the guest operating system to the VMware-supplied driver. The VMware driver has optimizations that guest-supplied Windows drivers do not.

  VMware Tools also includes the balloon driver used for memory reclamation in ESX. Ballooning (described in "Memory Overcommit Techniques" on page 22) will not work if VMware Tools is not installed.

- Disable screen savers and Window animations in virtual machines. On Linux, if using an X server is not required, disable it. Screen savers, animations, and X servers all consume extra physical CPU resources, potentially affecting consolidation ratios and the performance of other virtual machines.

- Schedule backups and virus scanning programs in virtual machines to run at off-peak hours and avoid scheduling them to run simultaneously in multiple virtual machines on the same ESX host. In general, it is a good idea to evenly distribute CPU usage, not just across CPUs but also across time. For workloads such as backups and virus scanning, where the load is predictable, this is easily achieved by scheduling the jobs appropriately.

- For the most accurate timekeeping, consider configuring your guest operating system to use NTP, Windows Time Service, or another timekeeping utility suitable for your operating system. The time-synchronization option in VMware Tools can be used instead, but it is not designed for the same level of accuracy as these other tools and does not adjust the guest time when it is ahead of the host time. We recommend, however, that within any particular virtual machine you use either the VMware Tools time-synchronization option or another timekeeping utility, but not both.

Running Paravirtualized Operating Systems

ESX includes support for virtual machine interface (VMI), used for communication between the guest operating system and the hypervisor, thus improving performance and efficiency. Enabling this support will improve the performance of virtual machines running operating systems with VMI by reducing their CPU utilization and memory space overhead (the later being especially true for SMP virtual machines). Even when only some of the virtual machines on an ESX system use VMI, the performance of all virtual machines on that server might benefit due to the hardware resources freed up for ESX to allocate elsewhere.

ESX VMI support can be enabled for a virtual machine through the vSphere Client by selecting **Edit virtual machine settings**, choosing the **Options** tab, selecting **Paravirtualization**, then marking the box next to **Support VMI Paravirtualization**. Enabling VMI for a virtual machine reduces the number of available PCI slots in the guest operating system running in that virtual machine by one. There is no performance benefit to enabling VMI for a virtual machine running a non-VMI operating system.

Kernel support for VMI is included in some recent Linux distributions (Ubuntu 7.04 and later and SLES 10 SP2, for example), and can be compiled into other Linux distributions, typically by compiling the kernel with `CONFIG_PARAVIRT` and `CONFIG_VMI`. No Microsoft Windows operating systems support VMI. Check the VMware *Guest Operating System Installation Guide* to see which VMI operating systems are supported in ESX. More information about VMI can be found in
Performance of VMware VMI (http://www.vmware.com/resources/techresources/1038)
and the Paravirtualization API Version 2.5 (http://www.vmware.com/pdf/vmi_specs.pdf).

For best performance, consider the following regarding enabling VMI support:

- If running 32-bit Linux guest operating systems that include kernel support for VMI on hardware that does not support hardware-assisted MMU virtualization (i.e., EPT or RVI), enabling VMI will improve performance.
- VMI-enabled virtual machines always use Binary Translation (BT) and shadow page tables, even on systems that support hardware-assisted MMU virtualization. Because hardware-assisted MMU virtualization almost always provides more of a performance increase than VMI, we recommend disabling VMI for virtual machines running on hardware that supports hardware-assisted MMU virtualization. (No kernel change is required in the guest operating system, as the VMI kernel can run in a non-VMI enabled virtual machine.)

Measuring Performance in Virtual Machines

Be careful when measuring performance from within virtual machines.

- Timing numbers measured from within virtual machines can be inaccurate, especially when the processor is overcommitted.

  NOTE One possible approach to this issue is to use a guest operating system that has good timekeeping behavior when run in a virtual machine, such as a guest that uses the NO_HZ kernel configuration option (sometimes called "tickless timer") or the VMI paravirtual timer. More information about this topic can be found in *Timekeeping in VMware Virtual Machines* (http://www.vmware.com/files/pdf/Timekeeping-In-VirtualMachines.pdf).

- Measuring performance from with virtual machines can fail to take into account resources used by ESX for tasks it has offloaded from the guest operating system, as well as resources consumed by virtualization overhead. For additional information see VMware KB article 2032.

Measuring resource utilization using tools at the ESX level, such as the vSphere Client, VMware Tools, `esxtop`, or `resxtop`, can avoid these problems.

Guest Operating System CPU Considerations

- In SMP virtual machines the guest operating system can migrate processes from one vCPU to another. This migration can incur a small CPU overhead. If the migration is very frequent it might be helpful to pin guest threads or processes to specific vCPUs. (Note that this is another reason not to configure virtual machines with more vCPUs than they need.)
- Many operating systems keep time by counting timer interrupts. The timer interrupt rates vary between different operating systems and versions. For example:
  - Unpatched 2.4 and earlier Linux kernels typically request timer interrupts at 100 Hz (that is, 100 interrupts per second), though this can vary with version and distribution.
  - 2.6 Linux kernels have used a variety of timer interrupt rates, including 100 Hz, 250 Hz, and 1000 Hz, again varying with version and distribution.
  - The most recent 2.6 Linux kernels introduce the NO_HZ kernel configuration option (sometimes called "tickless timer") that uses a variable timer interrupt rate.
  - Microsoft Windows operating system timer interrupt rates are specific to the version of Microsoft Windows and the Windows HAL that is installed. Windows systems typically use a base timer interrupt rate of 64 Hz or 100 Hz.
  - Running applications that make use of the Microsoft Windows multimedia timer functionality can increase the timer interrupt rate. For example, some multimedia applications or Java applications increase the timer interrupt rate to approximately 1000 Hz.

  In addition to the timer interrupt rate, the total number of timer interrupts delivered to a virtual machine also depends on a number of other factors:

  - Virtual machines running SMP HALs/kernels (even if they are running on a UP virtual machine) require more timer interrupts than those running UP HALs/kernels.
  - The more vCPUs a virtual machine has, the more interrupts it requires.

  Delivering many virtual timer interrupts negatively impacts virtual machine performance and increases host CPU consumption. If you have a choice, use guest operating systems that require fewer timer interrupts. For example:

  - If you have a UP virtual machine use a UP HAL/kernel.
  - In some Linux versions, such as RHEL 5.1 and later, the "divider=10" kernel boot parameter reduces the timer interrupt rate to one tenth its default rate. See VMware KB article 1006427 for further information.

    NOTE A bug in the RHEL 5.1 x86_64 kernel causes problems with the divider option. For RHEL 5.1 use the patch that fixes the issue at https://bugzilla.redhat.com/show_bug.cgi?id=305011. This bug is also fixed in RHEL 5.2. For more information see http://rhn.redhat.com/errata/RHSA-2007-0993.html.

  - Kernels with tickless-timer support (NO_HZ kernels) do not schedule periodic timers to maintain system time. As a result, these kernels reduce the overall average rate of virtual timer interrupts, thus improving system performance and scalability on hosts running large numbers of virtual machines.

  For background information on the topic of timer interrupts, refer to *Timekeeping in Virtual Machines*.

Guest Operating System Storage Considerations

- The default storage adapter in ESX 4.1 is either BusLogic Parallel, LSI Logic Parallel, or LSI Logic SAS, depending on the guest operating system and the virtual hardware version. However, ESX also includes a virtual storage adapter, paravirtualized SCSI (PVSCSI, also called VMware Paravirtual, introduced in ESX 4.0). The PVSCSI adapter offers a significant reduction in CPU utilization as well as potentially increased throughput compared to the default virtual storage adapters, and are thus the best choice for environments with very I/O-intensive guest applications.

  NOTE In order to use PVSCSI, your virtual machine must be using virtual hardware version 7, as described under "ESX General Considerations" on page 15.

  NOTE For boot drives PVSCSI adapters are supported only in some operating systems.

- If you choose to use the BusLogic Parallel virtual SCSI adapter, and are using a Windows guest operating system, you should use the custom BusLogic driver included in the VMware Tools package.
- The depth of the queue of outstanding commands in the guest operating system SCSI driver can significantly impact disk performance. A queue depth that is too small, for example, limits the disk bandwidth that can be pushed through the virtual machine. See the driver-specific documentation for more information on how to adjust these settings.
- In some cases large I/O requests issued by applications in a virtual machine can be split by the guest storage driver. Changing the guest operating system's registry settings to issue larger block sizes can eliminate this splitting, thus enhancing performance. For additional information see VMware KB article 9645697.
- Make sure the disk partitions within the guest are aligned. For further information you might want to refer to the literature from the operating system vendor regarding appropriate tools to use as well as recommendations from the array vendor.

Guest Operating System Networking Considerations

The default virtual network adapter emulated in a virtual machine is either an AMD PCnet32 device (vlance) or an Intel E1000 device (E1000). VMware also offers the VMXNET family of paravirtualized network adapters, however, that provide better performance than these default adapters and should be used for optimal performance within any guest operating system for which they are available.

The paravirtualized network adapters in the VMXNET family implement an idealized network interface that passes network traffic between the virtual machine and the physical network interface cards with minimal overhead. Drivers for VMXNET-family adapters are available for most guest operating systems supported by ESX.

The VMXNET family contains VMXNET, Enhanced VMXNET (available since ESX 3.5), and VMXNET Generation 3 (VMXNET3; available since ESX 4.0).

NOTE The network speeds reported by the guest network driver in the virtual machine do not necessarily reflect the actual speed of the underlying physical network interface card. For example, the vlance guest driver in a virtual machine reports a speed of 10Mbps, even if the physical card on the server is 100Mbps or 1Gbps, because the AMD PCnet cards that ESX emulates are 10Mbps. However, ESX is not limited to 10Mbps and transfers network packets as fast as the resources on the physical server machine allow

- For the best performance, use the VMXNET3 paravirtualized network adapter for operating systems in which it is supported. This requires that the virtual machine use virtual hardware version 7, and that VMware Tools be installed in the guest operating system.

  NOTE A virtual machine with a VMXNET3 device cannot vMotion to a host running ESX 3.5.x or earlier.

- For guest operating systems in which VMXNET3 is not supported, or if you don't wish to use virtual hardware version 7 (to maintain vMotion compatibility with older versions of ESX, for example), the best performance can be obtained with the use of Enhanced VMXNET for operating systems in which it is supported. This requires that VMware Tools be installed in the guest operating system.

  NOTE A virtual machine with an Enhanced VMXNET device cannot vMotion to a host running ESX 3.0.x or earlier.

- For the operating systems in which Enhanced VMXNET is not supported, use the flexible device type. In ESX, the "flexible NIC" automatically converts each vlance network device to a VMXNET device (a process also called "NIC Morphing") if the VMware Tools suite is installed in the guest operating system and the operating system supports VMXNET.
- The VMXNET3 and Enhanced VMXNET devices support jumbo frames for better performance. (Note that the E1000 and vlance devices do not support jumbo frames.) To enable jumbo frames, set the MTU size to 9000 in both the guest network driver and the virtual switch configuration. The physical NICs at both ends and all the intermediate hops/routers/switches must also support jumbo frames.
- In ESX 4.x, TCP Segmentation Offload (TSO) is enabled by default in the VMkernel, but is supported in virtual machines only when they are using the VMXNET3 device, the Enhanced VMXNET device, or the E1000 device. TSO can improve performance even if the underlying hardware does not support TSO.
- In some cases, low receive throughput in a virtual machine can be caused by insufficient receive buffers in the receiver network device. If the receive ring in the guest operating system's network driver overflows, packets will be dropped in the VMkernel, degrading network throughput. A possible workaround is to increase the number of receive buffers, though this may increase the host physical CPU workload.

  For VMXNET, the default number of receive and transmit buffers is 100 each, with the maximum possible being 128. For Enhanced VMXNET, the default number of receive and transmit buffers are 150 and 256, respectively, with the maximum possible receive buffers being 512. You can alter these settings by changing the buffer size defaults in the `.vmx` (configuration) files for the affected virtual machines.

For VMXNET3 and E1000, the default number of receive and transmit buffers are controlled by the guest driver, with the maximum possible for both being 4096. For Linux, these values can be changed from within the guest by using `ethtool`. In Windows, the values can be changed from within the guest in the device properties window. For additional information see VMware KB article 1010071.

- Receive-side scaling (RSS) allows network packet receive processing to be scheduled in parallel on multiple processors. Without RSS, receive interrupts can be handled on only one processor at a time. With RSS, received packets from a single NIC can be processed on multiple CPUs concurrently. This will help receive throughput in cases where a single vCPU would otherwise be saturated with receive processing and become a bottleneck. To prevent out-of-order packet delivery, RSS always schedules all of a flow's packets to the same CPU.

  The VMXNET3 device supports RSS for all Windows guest operating systems that support RSS natively (such as Windows Server 2008 and Windows 7) and also supports multiple transmit queues. By default, on an *n*-vCPU virtual machine RSS configures *n* receive queues (up to 4). (RSS doesn't affect the transmit queues, which default to 1 with or without RSS.) In order to obtain the maximum performance with your specific workloads and resource availability you can try out different values for both the number of transmit queues (up to a maximum of 8) and the number of receive queues (up to a maximum of 8). These values are set from the advanced driver configuration tab within the guest operating system.

Virtual Infrastructure Management 4

This chapter provides guidance regarding resource management best practices. Most of the suggestions included in this section can be implemented using the vSphere Client connected to a VMware vCenter™ Server. Some can also be implemented using the vSphere Client connected to an individual ESX host.

Further detail about many of these topics, as well as background information, can be found in the *VMware vCenter Server Performance and Best Practices* white paper.

General Resource Management

ESX provides several mechanisms to configure and adjust the allocation of CPU and memory resources for virtual machines running within it. Resource management configurations can have a significant impact on virtual machine performance.

This section lists resource management practices and configurations recommended by VMware for optimal performance.

- Use resource settings (that is, **Reservations**, **Shares**, and **Limits**) only if needed in your environment.
- If you expect frequent changes to the total available resources, use **Shares**, not **Reservation**, to allocate resources fairly across virtual machines. If you use **Shares** and you subsequently upgrade the hardware, each virtual machine stays at the same relative priority (keeps the same number of shares) even though each share represents a larger amount of memory or CPU.
- Use **Reservation** to specify the *minimum* acceptable amount of CPU or memory, not the amount you would like to have available. After all resource reservations have been met, ESX allocates the remaining resources based on the number of shares and the resource limits configured for your virtual machine.

  As indicated above, reservations can be used to specify the minimum CPU and memory reserved for each virtual machine. In contrast to shares, the amount of concrete resources represented by a reservation does not change when you change the environment, for example by adding or removing virtual machines. Don't set **Reservation** too high. A reservation that's too high can limit the number of virtual machines you can power on in a resource pool, cluster, or host.

  When specifying the reservations for virtual machines, always leave some headroom for memory virtualization overhead and migration overhead. In a DRS-enabled cluster, reservations that fully commit the capacity of the cluster or of individual hosts in the cluster can prevent DRS from migrating virtual machines between hosts. As you approach fully reserving all capacity in the system, it also becomes increasingly difficult to make changes to reservations and to the resource pool hierarchy without violating admission control.
- Use resource pools for delegated resource management. To fully isolate a resource pool, make the resource pool type **Fixed** and use **Reservation** and **Limit**.
- Group virtual machines for a multi-tier service into a resource pool. This allows resources to be assigned for the service as a whole.

VMware vCenter

This section lists VMware vCenter practices and configurations recommended for optimal performance. It also includes a few features that are controlled or accessed through vCenter.

- Large numbers of managed hosts, managed virtual machines, and connected VMware vSphere Clients can affect the performance of a vCenter Server. Exceeding the supported maximums, though it might work, is even more likely to impact vCenter performance. For more information, see *VMware vSphere 4.1 Release Notes* and *Configuration Maximums for VMware vSphere 4.1*.
- Make sure you are running vCenter Server and the vCenter Server database on hardware with sufficient CPU, memory, and storage resources for your deployment size. For additional information see *ESX and vCenter Server Installation Guide* for 4.1.
- To minimize the latency of vCenter operations, keep to a minimum the number of network hops between the vCenter Server system and the ESX hosts.
- For the best performance, avoid overly-aggressive vCenter alarm settings. Each time an alarm condition is met the vCenter Server must take appropriate action. If this happens very often, the added load could affect system performance.
- Although VMware vCenter Update Manager can be run on the same system and use the same database as vCenter Server, for maximum performance, especially on heavily-loaded vCenter systems, you should consider running Update Manager on its own system and providing it with a dedicated database. For additional information see "VMware vCenter Update Manager" on page 47.
- Similarly, VMware vCenter Converter can be run on the same system as vCenter Server, but doing so might impact performance, especially on heavily-loaded vCenter systems.
- Beginning with vSphere 4.0, and continuing in vSphere 4.1, a Java Servlet implementation of Apache Tomcat is used for various services. Appropriately setting the Java virtual machine (JVM) max heap memory size during installation can assure good performance while avoiding unnecessary memory commitment. Table 4-1 shows some recommended sizes.

Table 4-1. Java Virtual Machine Max Heap Size Recommendations

Deployed Inventory Size	JVM Max Memory
Small (fewer than 100 hosts)	1024MB
Medium (between 100 and 400 hosts)	2048MB
Large (more than 400 hosts)	4096MB

VMware vCenter Database Considerations

vCenter relies heavily on a database to store configuration information and statistics data. Due to the importance of this database to the reliability and performance of your vCenter Server, VMware recommends the following database practices:

- To minimize the latency of operations between vCenter Server and the database, keep to a minimum the number of network hops between the vCenter Server system and the database system.
- Because the database data files and the database transaction logs generate very different storage loads, the best performance can be obtained when they are placed on separate physical disks. Placing tempDB on yet another set of disks is also a good idea.
- Maximize bandwidth to the database by using high-performance RAID storage for both the database data files and log files.
- Configure the database log setting to **Normal** unless you have a specific reason to set it to **High**.
- Configure the vCenter statistics level to a setting appropriate for your uses. This setting can range from 1 to 4, but a setting of 1 is recommended for most situations. Higher settings can slow the vCenter Server system. You can also selectively disable statistics rollups for particular collection levels.
- Periodically reindex the database to maximize its performance.
- vCenter Server starts up with a database connection pool of 50 threads. This pool is then dynamically sized, growing adaptively as needed based on the vCenter Server workload, and does not require modification. However, if a heavy workload is expected on the vCenter Server, the size of this pool at startup can be increased, with the maximum being 128 threads. Note that this might result in increased memory consumption by vCenter Server and slower vCenter Server startup.
- If you are using a SQL Server database, setting the transaction logs to **Simple recovery** mode significantly reduces the disk space used by the database logs.
- If you are using an Oracle database, the following points can improve vCenter Server performance:
  - Allocate sufficient memory for the Oracle process. When using Automatic Memory Management in Oracle, pay attention to the **MEMORY_TARGET** and **MEMORY_MAX_TARGET** values. **MEMORY_TARGET** specifies the amount of memory that Oracle can utilize and **MEMORY_MAX_TARGET** is the maximum value that can be set for **MEMORY_TARGET**.
  - In vCenter Server 4.1, add a Unique Index in the **VPXV_DEVICE_COUNTER** table for better performance.

    ```
    create unique index VPXI_DEVICE_COUNT_IDX on VPXV_DEVICE_COUNTER(entity_id, device_name,
                  stat_id);
    ```
  - Set appropriate **PROCESSES** and **SESSIONS** parameters. Oracle creates a new server process for every new connection that is made to it. The number of connections an application can make to the Oracle instance thus depends on how many processes Oracle can create. **PROCESSES** and **SESSIONS** together determine how many simultaneous processes Oracle can create. In large vSphere environments we recommend a value of 800 for each of these parameters.

For more information on vCenter Server database performance, see *VMware vCenter 4.0 Database Performance for MS SQL Server 2008*.

VMware vSphere Client

VMware vSphere environments are typically managed with either the vSphere Client or the vSphere Web Services SDK.

- For the best performance, disconnect VMware vSphere Clients from the vCenter Server when they are no longer needed. Because the vCenter Server must keep all client sessions current with inventory changes, the number of vSphere Client sessions attached to the vCenter Server can affect the server's CPU usage and user interface speed.
- If running many vSphere Clients on a single system, monitor the resource usage of that system.
- When selecting an entity (virtual machine, host, datastore, network, etc.) within the vSphere Client, using the inventory search feature is less resource intensive on the vCenter Server than navigating through the vCenter Client inventory panel.
- In many cases, using the vSphere Web Services SDK can be an efficient way to manage the vSphere environment. To learn more about the VMware vSphere API and supported SDK libraries, refer to the *VMware vSphere Web Services SDK Documentation*. For examples of good programming practices see code samples from the VMware Communities sample code page (http://communities.vmware.com/community/developer/codecentral).

For more information on working with the vSphere Client, see *Customizing the vSphere Client*.

VMware vMotion and Storage vMotion

This section provides performance best practices for vMotion™ and Storage vMotion.

VMware vMotion

- ESX 4.0 introduced virtual hardware version 7. Because virtual machines running on hardware version 7 can't run on prior versions of ESX, such virtual machines can be moved using VMware vMotion only to other ESX 4.x hosts. ESX 4.x is compatible with virtual machines running on virtual hardware version 4, however, and these machines can be moved using VMware vMotion to ESX 3.x or ESX 4.x hosts.
- vMotion performance will increase as additional network bandwidth is made available to the vMotion network. Consider provisioning a 10Gb vMotion network interface for maximum vMotion performance.
- ESX opportunistically reserves CPU resources on both the source and destination hosts of a vMotion operation. In vSphere 4.1, this is 30% of a processor core for 1Gb network interfaces and 100% of a processor core for 10Gb network interfaces. Therefore leaving some unreserved CPU capacity in a cluster can help ensure that vMotion tasks get the resources required in order to fully utilize available network bandwidth.

VMware Storage vMotion

- Before using VMware Storage vMotion make sure you have sufficient storage bandwidth between the ESX host where the virtual machine is running and both the source and destination data stores. This is necessary because the virtual machine will continue to read from and write to the source data store, while at the same time the virtual disk to be moved is being read from the source data store and written to the destination data store. While this is happening, the storage array or arrays containing these data stores might have other traffic (from other virtual machines on the same ESX host, from other ESX hosts, etc.) that can further reduce the available bandwidth.

  With barely sufficient bandwidth Storage vMotion will progress very slowly. With insufficient storage bandwidth Storage vMotion will eventually time out.
- Storage vMotion will have the highest performance during times of low storage activity (when available storage bandwidth is highest) and when the workload in the virtual machine being moved is least active.
- Storage vMotion will often have significantly better performance on VAAI-capable storage arrays (described in "Hardware Storage Considerations" on page 11).

VMware Distributed Resource Scheduler (DRS)

Clustering configurations can have a significant impact on performance. This section lists Distributed Resource Scheduler (DRS) practices and configurations recommended by VMware for optimal performance.

- When deciding which hosts to group into DRS clusters, try to choose hosts that are as homogeneous as possible in terms of CPU and memory. This ensures higher performance predictability and stability. VMware vMotion is not supported across hosts with incompatible CPU's. Hence with 'incompatible CPU' heterogeneous systems, the opportunities DRS has to improve the load balance across the cluster are limited.

  To ensure CPU compatibility make sure systems are configured with the same CPU vendor, with similar CPU families, and with matching SSE instruction-set capability. For more information on this topic see VMware KB articles 1991, 1992, and 1993.

  You can also use Enhanced vMotion Compatibility (EVC) to facilitate vMotion between different CPU generations. For more information on this topic see *VMware vMotion and CPU Compatibility* and VMware KB article 1003212.

  When heterogeneous systems have compatible CPUs, but have different CPU frequencies and/or amounts of memory, DRS generally prefers to locate virtual machines on the systems with more memory and higher CPU frequencies (all other things being equal) since those systems have more capacity to accommodate peak loads.

- The more vMotion compatible ESX hosts DRS has available, the more choices it has to better balance the DRS cluster. Besides CPU incompatibility, there are other misconfigurations that can block vMotion between two or more hosts. For example, if the hosts' vMotion network adapters are not connected by a Gigabit (or faster) Ethernet link then the vMotion might not occur between the hosts. Other configuration settings to check for are virtual hardware version compatibility, misconfiguration of the vMotion gateway, incompatible security policies between the source and destination host vMotion network adapter, and virtual machine network availability on the destination host. Refer to the *Resource Management Guide* for 4.1 and the *vSphere Datacenter Administration Guide* for 4.1 for further details.

- Don't configure more hosts or virtual machines in a DRS cluster than the maximum allowed in the *Configuration Maximums for VMware vSphere 4.1*.

- Virtual machines with smaller memory sizes and/or fewer vCPUs provide more opportunities for DRS to migrate them in order to improve balance across the cluster. Virtual machines with larger memory sizes and/or more vCPUs add more constraints in migrating the virtual machines. This is one more reason to configure virtual machines with only as many vCPUs and only as much virtual memory as they need.

- Have virtual machines in DRS automatic mode when possible, as they are considered for cluster load balance migrations across the ESX hosts before the virtual machines that are not in automatic mode.

- All powered-on virtual machines consume CPU resources. Thus even idle virtual machines, though their CPU utilization is usually small, can affect DRS decisions. For this and other reasons, a marginal performance increase might be obtained by shutting down or suspending virtual machines that are not being used.

- The frequency with which the DRS algorithm is invoked for balancing can be controlled through the vpxd configuration file, `vpxd.cfg`, with the following option:

  ```
  <config>
     <drm>
        <pollPeriodSec>
           300
        </pollPeriodSec>
     </drm>
  </config>
  ```

  The default frequency is 300 seconds, but it can be set to anything between 60 seconds and 3600 seconds. We recommend against changing the default value, however, except in specific cases where the user would like to invoke the algorithm less frequently at the cost of potential loss in application performance.

- The migration threshold should be set to more aggressive levels when the following conditions are satisfied:
  - If the hosts in the cluster are relatively homogeneous.
  - If the virtual machines' resource utilization does not vary too much over time and you have relatively few constraints on where a virtual machine can be placed.

  The migration threshold should be set to more conservative levels in the converse situations.

- In addition to the migration threshold, DRS offers a number of advanced options that allow further control over the algorithm performance. Table 4-2 lists some of these options and explains their use. We recommend that these DRS options be left at their default values unless there is a strong reason to change them; for example, a cluster with severe resource contention on some hosts and idle capacity on others.

Table 4-2. DRS Advanced Options for Performance Tuning

Advanced option name	Description	Default Value	Most Aggressive Value
CostBenefit	Whether to take migration cost into account	1	0 (No cost-benefit analysis)
MinImbalance	Used to compute target imbalance	50	0
MinGoodness	Minimum improvement in cluster imbalance required for each move	Adaptive	0 (All moves are considered)
MaxMovesPerHost	Maximum number of moves per host recommended per invocation	Adaptive	0 (No limit)

- DRS affinity rules can keep virtual machines on the same ESX host (“VM/VM affinity”) or make sure they are always on different hosts (“VM/VM anti-affinity”). New in vSphere 4.1, DRS affinity rules can also be used to make sure a group of virtual machines runs only on (or has a preference for) a specific group of ESX hosts (“VM/Host affinity”) or never runs on (or has a preference against) a specific group of hosts (“VM/Host anti-affinity”).

  In most cases leaving the affinity settings unchanged will provide the best results. In rare cases, however, specifying affinity rules can help improve performance. To change affinity settings select a cluster from within the vSphere Client, choose the **Summary** tab, click **Edit Settings**, choose **Rules**, click **Add**, enter a name for the new rule, choose a rule type, and proceed through the GUI as appropriate for the rule type you selected.

  Besides the default setting, the three affinity setting types are:
  - **Keep Virtual Machines Together** can improve performance due to lower latencies of communication between machines.
  - **Separate Virtual Machines** maintains maximal availability of the virtual machines. For instance, if they are both web server front ends to the same application, you might want to make sure that they don't both go down at the same time. Also co-location of I/O intensive virtual machines could end up saturating the host I/O capacity, leading to performance degradation. DRS currently does not make virtual machine placement decisions based on their I/O resources usage (though see “Storage I/O Resource Allocation” on page 27 for one way to allocate storage resources).
  - **Virtual Machines to Hosts** (including **Must run on...**, **Should run on...**, **Must not run on...**, and **Should not run on...**) can be useful for clusters with software licensing restrictions or specific availability zone requirements.

- To allow DRS the maximum flexibility:
  - Place virtual machines on shared datastores accessible from all hosts in the cluster.
  - Make sure virtual machines are not connected to host devices that would prevent them from moving off of that host.

- Carefully select the resource settings (that is, reservations, shares, and limits) for your virtual machines.

    - Setting reservations too high can leave few unreserved resources in the cluster, thus limiting the options DRS has to balance load.
    - Setting limits too low could keep virtual machines from using extra resources available in the cluster to improve their performance.

  Use reservations to guarantee the minimum requirement a virtual machine needs, rather than what you might like it to get. Note that shares take effect only when there is resource contention. Note also that additional resources reserved by virtual machine memory overhead need to be accounted for when sizing resources in the cluster.

- Resource pools help improve manageability and troubleshooting of performance problems. We recommend, however, that resource pools and virtual machines not be made siblings in a hierarchy. Instead, each level should contain only resource pools or only virtual machines. This is because by default resource pools are assigned share values that might not compare appropriately with those assigned to virtual machines, potentially resulting in unexpected performance.

VMware Distributed Power Management (DPM)

VMware Distributed Power Management (DPM) conserves power by migrating virtual machines to fewer hosts when utilizations are low. DPM is most appropriate for clusters in which composite virtual machine demand varies greatly over time; for example, clusters in which overall demand is higher during the day and significantly lower at night. If demand is consistently high relative to overall cluster capacity DPM will have little opportunity to put hosts into standby mode to save power.

Because DPM uses DRS, most DRS best practices (described in "VMware Distributed Resource Scheduler (DRS)" on page 40) are relevant to DPM as well.

- The more hosts in the cluster, the more opportunity there is for power savings through DPM. However don't exceed the maximum allowed in the *Configuration Maximums for VMware vSphere 4.1*.
  - DPM can only be enabled on hosts running ESX 3.5 or later. (Note that DPM *is* supported between hosts running EX 3.5 and ESX 4.x, subject to the vMotion limitations detailed in "VMware vMotion and Storage vMotion" on page 39.)
  - Having vMotion-compatible hosts in the cluster is important, since DPM needs to be able to migrate running virtual machines onto fewer hosts to be able to put some hosts into standby mode, thus saving power.
  - Similarly, having wake-on-LAN (WOL), IPMI, or iLO capable hosts in the cluster is beneficial. The ability of DPM to place a host in standby and power it on again later depends on ESX host software and on the host having hardware support for one of these protocols. The more of the hosts in the cluster support DPM standby/power-on operations the more choices DPM has for saving power.
- DPM is complementary to host power management (HPM) policies that use both processor P-states and C-states to reduce host power consumption when the workloads running on a host don't require full CPU capacity or when some CPUs are idle. Using DPM and HPM together can offer greater power savings than when either solution is used alone.
- A new feature in vSphere 4.1 allows DPM to be enabled or disabled on a predetermined schedule. When DPM is disabled, all hosts in a cluster will be powered on. This might be useful, for example, to reduce the delay in responding to load spikes expected at certain times of the day.
- Enabling DPM with all hosts in the cluster being in automatic mode gives DPM the most flexibility in choosing hosts for power down/up. For hosts in manual DPM mode, DPM must query the user about power down/up of the host. For this reason, DPM prefers choosing hosts in automatic mode over those in manual mode. If desired, DPM can also be disabled for specific hosts.
- DPM considers historical demand in determining how much capacity to keep powered on and keeps some excess capacity available for changes in demand. DPM will also power on additional hosts when needed for unexpected increases in the demand of existing virtual machines or to allow virtual machine admission.
- In a cluster with VMware High Availability (HA) enabled, DRS/DPM maintains excess powered-on capacity to meet the High Availability settings. The cluster might therefore not allow additional virtual machines to be powered on and/or some hosts might not be powered down even when the cluster appears to be sufficiently idle. These factors should be considered when configuring HA.

VMware High Availability

VMware High Availability (HA) minimizes virtual machine downtime by restarting failed virtual machines on alternate hosts.

- HA uses heartbeats to monitor the status of hosts. By default, these heartbeats are sent by the host once per second.

  The heartbeat packets are very small and thus create minimal network traffic. While we recommend leaving this heartbeat interval at the default, the advanced option `das.failuredetectioninterval` can be used to change it. (Note that in order to avoid false failover, if the heartbeat interval is increased, the failure detection time should also be increased, as described below.)

  If high network latency, intermittent network problems, or the presence of heavy network traffic over the management port groups cause lost heartbeats, resulting in false failover, you can increase the failure detection time, which is the amount of time the HA agent will wait without receiving heartbeats from a host before attempting to ping the host to see if it is still alive. The advanced option `das.failuredetectiontime` can be increased from its default value of 15000 (15 seconds). A reasonable value to try would be 30000 (30 seconds). We don't recommend values greater than 120000 (120 seconds) because at this point TCP timeouts will kick in.

- HA designates five hosts in the cluster as "primary hosts," responsible for various HA tasks. If you experience performance issues from less-powerful hosts being designated as primaries, you can put these less-powerful hosts into maintenance mode before enabling HA. In this case, five of the remaining hosts will become primaries. After HA is enabled you can take the less-powerful hosts out of maintenance mode, at which point they will become secondaries. (Note that if a primary host is moved out of the cluster, put into maintenance mode, or disconnected, these secondaries might still be promoted to primaries.)

- Enabling HA on a host reserves some host resources for HA agents, slightly reducing the available host capacity for powering on virtual machines.

- When HA is enabled, the vCenter Server reserves sufficient unused resources in the cluster to support the failover capacity specified by the chosen admission control policy. This can reduce the number of virtual machines the cluster can support.

VMware Fault Tolerance

VMware Fault Tolerance (FT) provides continuous virtual machine availability in the event of a server failure.

Because FT uses HA, most HA best practices (described in "VMware High Availability" on page 44) are relevant to FT as well.

- For each virtual machine there are two FT-related actions that can be taken: turning on or off FT and enabling or disabling FT.

  "Turning on FT" prepares the virtual machine for FT by prompting for the removal of unsupported devices, disabling unsupported features, and setting the virtual machine's memory reservation to be equal to its memory size (thus avoiding ballooning or swapping).

  "Enabling FT" performs the actual creation of the secondary virtual machine by live-migrating the primary.

  NOTE Turning on FT for a powered-on virtual machine will also automatically "Enable FT" for that virtual machine.

  Each of these operations has performance implications.

  - Don't turn on FT for a virtual machine unless you will be using (i.e., Enabling) FT for that machine. Turning on FT automatically disables some features for the specific virtual machine that can help performance, such as hardware virtual MMU (if the processor supports it).
  - Enabling FT for a virtual machine uses additional resources (for example, the secondary virtual machine uses as much CPU and memory as the primary virtual machine). Therefore make sure you are prepared to devote the resources required before enabling FT.

- The live migration that takes place when FT is enabled can briefly saturate the vMotion network link and can also cause spikes in CPU utilization.
  - If the vMotion network link is also being used for other operations, such as FT logging (that is transmission of all the primary virtual machine's inputs (incoming network traffic, disk reads) to the secondary host), the performance of those other operations can be impacted. For this reason it is best to have separate and dedicated NICs (or use Network I/O Control, described in "Network I/O Control (NetIOC)" on page 28) for FT logging traffic and vMotion, especially when multiple FT virtual machines reside on the same host.
  - Because this potentially resource-intensive live migration takes place each time FT is enabled, we recommend that FT not be frequently enabled and disabled.
  - FT-enabled virtual machines must use eager-zeroed thick-provisioned virtual disks. Thus when FT is enabled for a virtual machine with thin-provisioned virtual disks or lazy-zeroed thick-provisioned virtual disks these disks need to be converted. This conversion process uses fewer resources when the virtual machine is on storage hardware that supports VAAI (described in "Hardware Storage Considerations" on page 11).
- Because FT logging traffic is asymmetric (the majority of the traffic flows from primary to secondary), congestion on the logging NIC can be reduced by distributing primaries onto multiple hosts. For example on a cluster with two ESX hosts and two virtual machines with FT enabled, placing one of the primary virtual machines on each of the hosts allows the network bandwidth to be utilized bidirectionally.
- FT virtual machines that receive large amounts of network traffic or perform lots of disk reads can create significant bandwidth on the NIC specified for the logging traffic. This is true of machines that routinely do these things as well as machines doing them only intermittently, such as during a backup operation. To avoid saturating the network link used for logging traffic limit the number of FT virtual machines on each host or limit disk read bandwidth and network receive bandwidth of those virtual machines.
- Make sure the FT logging traffic is carried by at least a Gigabit-rated NIC (which should in turn be connected to at least Gigabit-rated network infrastructure).

- Avoid placing more than four FT-enabled virtual machines on a single host. In addition to reducing the possibility of saturating the network link used for logging traffic, this also limits the number of simultaneous live-migrations needed to create new secondary virtual machines in the event of a host failure.
- If the secondary virtual machine lags too far behind the primary (which usually happens when the primary virtual machine is CPU bound and the secondary virtual machine is not getting enough CPU cycles), the hypervisor may slow the primary to allow the secondary to catch up. The following recommendations help avoid this situation:
  - Make sure the hosts on which the primary and secondary virtual machines run are relatively closely matched, with similar CPU make, model, and frequency.
  - Make sure that power management scheme settings (both in the BIOS and in ESX) that cause CPU frequency scaling are consistent between the hosts on which the primary and secondary virtual machines run.
  - Enable CPU reservations for the primary virtual machine (which will be duplicated for the secondary virtual machine) to ensure that the secondary gets CPU cycles when it requires them.
- Though timer interrupt rates do not significantly affect FT performance, high timer interrupt rates create additional network traffic on the FT logging NIC. Therefore, if possible, reduce timer interrupt rates as described in "Guest Operating System CPU Considerations" on page 31.

VMware vCenter Update Manager

VMware vCenter Update Manager provides a patch management framework for VMware vSphere. It can be used to apply patches, updates, and upgrades to VMware ESX and ESXi hosts, Windows and Linux virtual machines, VMware Tools, and so on.

Further detail about many of these topics can be found in *VMware vCenter Update Manager Performance and Best Practices*.

Update Manager Setup and Configuration

- When there are 300+ virtual machines or 30+ hosts, separate the Update Manager database from the vCenter database.
- When there are 1000+ virtual machines or 100+ hosts, separate the Update Manager server from the vCenter Server and the Update Manager database from the vCenter Server database.
- Allocate separate physical disks for the Update Manager patch store and the Update Manager database.
- To reduce network latency and packet drops, keep to a minimum the number of network hops between the Update Manager server system and the ESX hosts.
- In order to cache patch files in memory, make sure the Update Manager server host has at least 2GB of RAM.
- Upgrade to Update Manager 4.1 or later (or at least Update Manager 4.0 U2) if you are working on a slow network.

Guest Operating System Considerations

- Configure each virtual machine with at least 1GB of RAM so large patch files can fit in the system cache.
- Because the Update Manager guest agent is single threaded, multiple vCPUs in virtual machines don't make Update Manager operations faster.
- If on-access virus scanning software is running on the Update Manager server host, excluding from those virus scans the mounted disks of the virtual machines (`/Device/vstor*`) can dramatically improve performance on high-latency networks.

Update Manager Operations

- Because the Update Manager guest agent will be installed in each Windows virtual machine the first time a powered-on scan is run on that machine, the first powered-on scan command can take longer than subsequent scans. It may therefore be desirable to run the first scan command when this additional time will not be an issue.
- For compliance view for all attached baselines, latency is increased linearly with the number of attached baselines. We therefore recommend the removal of unused baselines, especially when the inventory size is large.
- On a high-latency network, powered-on virtual machine scans are preferred because they are not sensitive to network latency.
- For a large setup, powered-on virtual machine scans are preferred if Update Manager server resources are constrained or more concurrency is needed for scans.
- Upgrading VMware Tools is faster if the virtual machine is already powered on. Otherwise, Update Manager will power on the virtual machine before the VMware Tools upgrade. This could increase the overall latency.
- Upgrading virtual machine hardware is faster if the virtual machine is already powered off. Otherwise, Update Manager will power off the virtual machine before upgrading the virtual hardware. This could increase the overall latency.

Glossary

A

ALUA (Asymmetric Logical Unit Access)
A feature included in some storage arrays that allows the array itself to designate paths as "Active Optimized."

AMD Virtualization (AMD-V)
AMD's version of hardware-assisted CPU virtualization, included in some 64-bit AMD processors. See also Virtualization Assist.

AMD-Vi
AMD's version of hardware-assisted I/O MMU, included in some 64-bit Intel processors, also called AMD I/O Virtualization or IOMMU. See also I/O MMU.

B

Ballooning
A technique used in VMware ESX to reclaim the guest memory pages that are considered the least valuable by the guest operating system. This is accomplished using the `vmmemctl` driver, which is installed as part of the VMware Tools suite.

C

Checksum Offload
An option enabling a network adapter to calculate the TCP checksum, thus reducing CPU load.

Clone
A copy of a virtual machine. See also Full Clone and Linked Clone.

Console
See VMware Virtual Machine Console.

Core
A processing unit. Often used to refer to multiple processing units in one package (a so-called "multi-core CPU"). Also used by Intel to refer to a particular family of processors (with the "Core microarchitecture"). Note that the Intel "Core" brand did not include the Core microarchitecture. Instead, this microarchitecture began shipping with the "Core 2" brand.

D

Distributed Power Management (DPM)
A feature that uses DRS to unload servers, allowing them to be placed into standby, and thereby saving power. When the load increases, the servers can be automatically brought back online.

Distributed Resource Management (DRS)
A feature that monitors utilization across resource pools and uses vMotion to move running virtual machines to other servers.

E

E1000
One of the virtual network adapters available in a virtual machine running in ESX. The E1000 adapter emulates an Intel E1000 device. See also vlance and VMXNET.

Enhanced VMXNET
One of the virtual network adapters available in a virtual machine running in ESX. The Enhanced VMXNET adapter is a high-performance paravirtualized device with drivers (available in VMware Tools) for many guest operating systems. See also VMXNET, VMXNET3, E1000, vlance, and NIC Morphing.

EPT (Extended Page Tables)
Intel's implementation of hardware virtual MMU.

F

Fault Tolerance (FT)
A feature in vSphere 4.x that runs a secondary copy of a virtual machine on a secondary host and seamlessly switches to that secondary copy in the event of failure of the primary host.

Fibre Channel
A networking technology used for storage. See also iSCSI, NAS, NFS, and SAN.

Full Clone
A copy of the original virtual machine that has no further dependence on the parent virtual machine. See also Linked Clone.

G

Growable Disk
A type of virtual disk in which only as much host disk space as is needed is initially set aside, and the disk grows as the virtual machine uses the space. Also called thin disk. See also Preallocated Disk.

Guest
A virtual machine running within VMware Workstation. See also Virtual Machine.

Guest Operating System
An operating system that runs inside a virtual machine. See also Host Operating System.

H

Hardware Abstraction Layer (HAL)
A layer between the physical hardware of a computer and the software that runs on that computer designed to hide differences in the underlying hardware, thus allowing software to run on a range of different architectures without being modified for each one. Windows uses different HALs depending, among other factors, on whether the underlying system has one CPU (Uniprocessor (UP) HAL) or multiple CPUs (Symmetric Multiprocessor (SMP) HAL). See also Kernel.

Hardware Virtual MMU
A feature of some recent CPUs that performs virtualization of the memory management unit (MMU) in hardware, rather than in the virtualization layer. Also called RVI or NPT by AMD and EPT by Intel.

Hardware Virtualization Assist
See Virtualization Assist.

High Availability (HA)
VMware High Availability is a product that continuously monitors all physical servers in a resource pool and restarts virtual machines affected by server failure.

Host Bus Adapter (HBA)
A device that connects one or more peripheral units to a computer and manages data storage and I/O processing (often for Fibre Channel, IDE, or SCSI interfaces).An HBA can be physical (attached to a host) or virtual (part of a virtual machine).

Hyper-Threading
A processor architecture feature that allows a single processor to execute multiple independent threads simultaneously. Hyper-threading was added to Intel's Xeon and Pentium® 4 processors. Intel uses the term "package" to refer to the entire chip, and "logical processor" to refer to each hardware thread. Also called symmetric multithreading (SMT).

I

Independent Virtual Disk
Independent virtual disks are not included in snapshots. Independent virtual disks can in turn be either Persistent or Nonpersistent.

Intel VT-x
Intel's version of hardware-assisted CPU virtualization, included in some 64-bit Intel processors. See also Virtualization Assist.

Intel VT-d
Intel's version of hardware-assisted I/O MMU, included in some 64-bit Intel processors, also called Intel Virtualization Technology for Directed I/O. See also I/O MMU.

I/O MMU
A processor feature that remaps I/O DMA transfers and device interrupts. This can allow virtual machines to have direct access to hardware I/O devices, such as network cards. See also AMD Vi and Intel VT-d.

iSCSI
A protocol allowing SCSI commands to be transmitted over TCP/IP (typically using ordinary Ethernet cabling). An iSCSI client is called an initiator (can be software or hardware); an iSCSI server is called a target.

J

Jumbo frames
Ethernet frames with a payload of more than 1,500 bytes. Because there is a CPU cost per network packet, larger packets can reduce the CPU cost of network traffic. Not all Gigabit Ethernet cards or switches support jumbo frames. Jumbo frames are supported by the Enhanced VMXNET and the VMXNET3 virtual network adapters (but not by the normal VMXNET adapter).

K

Kernel
The heart of an operating system. The kernel usually includes the functionality of a Hardware Abstraction Layer (HAL). Though applying to any operating system, the term is more often used in reference to Linux than to Windows.

L

Large Pages
A feature offered by most modern processors allowing the TLB (translation lookaside buffer) to index 2MB or 4MB pages in addition to the standard 4KB pages.

Linked Clone
A copy of the original virtual machine that must have access to the parent virtual machine's virtual disk(s). The linked clone stores changes to the virtual disk(s) in a set of files separate from the parent's virtual disk files. See also Full Clone.

LRO (Large Receive Offload)
A method of increasing network receive throughput included in some network adapters.

LUN (Logical Unit Number)
A number identifying a single logical unit, can represent a single disk or a partition on an array. Used in many storage technologies, including SCSI, iSCSI, and Fibre Channel.

M

Management User Interface (MUI)
A web-based interface to previous versions of ESX Server. For ESX Server 3.0 and later the functionality of the MUI has largely been replaced by the Virtual Infrastructure Client (VI Client) and the vSphere Client.

Memory Compression
One of a number of techniques used by ESX to allow memory overcommitment.

MMU (Memory Management Unit)
Part of a computer's hardware that acts as an interface between the core of the CPU and main memory. Typically part of the CPU package in modern processors.

N

NAS
See Network Attached Storage.

Native Execution
Execution of an application directly on a physical server, as contrasted with running the application in a virtual machine.

Native System
A computer running a single operating system, and in which the applications run directly in that operating system.

NetQueue
A technology that significantly improves performance of 10 Gigabit Ethernet network adapters in virtualized environments.

Network-Attached Storage (NAS)
A storage system connected to a computer network. NAS systems are file-based, and often use TCP/IP over Ethernet (although there are numerous other variations). See also Storage Area Network.

Network File System (NFS)
A specific network file system protocol supported by many storage devices and operating systems. Traditionally implemented over a standard LAN (as opposed to a dedicated storage network).

Network I/O Control (NetIOC)
A vSphere feature that allows the allocation of network bandwidth to six network resource groups: vMotion, NFS, iSCSI, Fault Tolerance, virtual machine, and management.

NIC
Historically meant "network interface card." With the recent availability of multi-port network cards, as well as the inclusion of network ports directly on system boards, the term NIC is now sometimes used to mean "network interface controller" (of which there may be more than one on a physical network card or system board).

NIC Morphing
The automatic conversion on some guest operating systems from the vlance virtual network adapter to the higher-performance VMXNET virtual network adapter.

NIC Team
The association of multiple NICs with a single virtual switch to form a team. Such teams can provide passive failover and share traffic loads between members of physical and virtual networks.

Non-Uniform Memory Access (NUMA)
A computer architecture in which memory located closer to a particular processor is accessed with less delay than memory located farther from that processor.

Nonpersistent Disk
All disk writes issued by software running inside a virtual machine with a nonpersistent virtual disk appear to be written to disk, but are in fact discarded after the session is powered down. As a result, a disk in nonpersistent mode is not modified by activity in the virtual machine. See also Persistent Disk.

NPT (Nested Page Tables)
AMD's implementation of hardware virtual MMU. Also called RVI.

P

Pacifica
A code name for AMD's version of virtualization assist, included in some 64-bit AMD processors. See AMD Virtualization.

Paravirtualization
Paravirtualization is a technique in which a modified guest operating system kernel communicates to the hypervisor its intent to perform privileged CPU and memory operations. See also VMI.

PCI (Peripheral Component Interconnect)
A computer bus specification. Now largely being superseded by PCIe.

PCI-X ()
A computer bus specification. Similar to PCI, but twice as wide and with a faster clock. Shares some compatibility with PCI devices (that is, PCI-X cards can sometimes be used in PCI slots and PCI cards can sometimes be used in PCI-X slots).

PCIe (PCI Express)
A computer bus specification. PCIe is available in a variety of different capacities (number of "lanes"): x1, x2, x4, x8, x16, and x32. Smaller cards will fit into larger slots, but not the reverse. PCIe is not slot-compatible with either PCI or PCI-X.

Persistent Disk
All disk writes issued by software running inside a virtual machine are immediately and permanently written to a persistent virtual disk. As a result, a disk in persistent mode behaves like a conventional disk drive on a physical computer. See also Nonpersistent Disk.

Physical CPU
A processor within a physical machine. See also Virtual CPU.

Preallocated Disk
A type of virtual disk in which all the host disk space for the virtual machine is allocated at the time the virtual disk is created. See also Growable Disk.

PVSCSI
A virtual storage adapter that offers a significant reduction in CPU utilization as well as potentially increased throughput compared to the default virtual storage adapters.

R

RAID (Redundant Array of Inexpensive Disks)
A technology using multiple hard disks to improve performance, capacity, or reliability.

Raw Device Mapping (RDM)
The use of a mapping file in a VMFS volume to point to a raw physical device.

Receive-side scaling (RSS)
Allows network packet receive processing to be scheduled in parallel on multiple processors.

RVI (Rapid Virtualization Indexing)
AMD's implementation of hardware virtual MMU. Also called NPT.

S

SAN
See Storage Area Network.

Secure Virtual Machine (SVM)
Another name for AMD's version of virtualization assist, included in some 64-bit AMD processors. See AMD Virtualization.

Service Console
The service console boots the systems and runs support, management, and administration applications.

Shadow Page Tables
A set of page tables maintained by ESX that map the guest operating system's virtual memory pages to the underlying pages on the physical machine.

Snapshot
A snapshot preserves the virtual machine just as it was when you took that snapshot — including the state of the data on all the virtual machine's disks and whether the virtual machine was powered on, powered off, or suspended. VMware Workstation lets you take a snapshot of a virtual machine at any time and revert to that snapshot at any time.

Socket
A connector that accepts a CPU package. With multi-core CPU packages, this term is no longer synonymous with the number of cores.

Storage Area Network (SAN)
A storage system connected to a dedicated network designed for storage attachment. SAN systems are usually block-based, and typically use the SCSI command set over a Fibre Channel network (though other command sets and network types exist as well). See also Network-Attached Storage.

Storage I/O Control (SIOC)
A vSphere feature that allows an entire datastore's I/O resources to be proportionally allocated to the virtual machines accessing that datastore.

Storage vMotion
A feature allowing running virtual machines to be migrated from one datastore to another with no downtime.

Symmetric Multiprocessor (SMP)
A multiprocessor architecture in which two or more processors are connected to a single pool of shared memory. See also Uniprocessor (UP).

Symmetric Multithreading (SMT)
Another name for hyper-threading. See also Hyper-Threading.

T

Template
A virtual machine that cannot be deleted or added to a team. Setting a virtual machine as a template protects any linked clones or snapshots that depend on the template from being disabled inadvertently.

Thick Disk
A virtual disk in which all the space is allocated at the time of creation.

Thin Disk
A virtual disk in which space is allocated as it is used.

Thrashing
A situation that occurs when virtual or physical memory is not large enough to hold the full working set of a workload. This mismatch can cause frequent reading from and writing to a paging file, typically located on a hard drive, which can in turn severely impact performance.

TLB (Translation Lookaside Buffer)
A CPU cache used to hold page table entries.

TSO (TCP Segmentation Offload)
A feature of some NICs that offloads the packetization of data from the CPU to the NIC. TSO is supported by the E1000, Enhanced VMXNET, and VMXNET3 virtual network adapters (but not by the normal VMXNET adapter).

U

Uniprocessor (UP)
A single-processor architecture. See also Symmetric Multiprocessor (SMP).

V

Vanderpool
A code name for Intel's version of virtualization assist, included in some 64-bit Intel processors. See Virtualization Technology.

Virtual CPU (vCPU)
A processor within a virtual machine. ESX 4.x supports up to eight virtual CPUs per virtual machine.

Virtual Disk
A virtual disk is a file or set of files that appears as a physical disk drive to a guest operating system. These files can be on the host machine or on a remote file system. When you configure a virtual machine with a virtual disk, you can install a new operating system into the disk file without the need to repartition a physical disk or reboot the host.

Virtual Machine
A virtualized x86 PC environment in which a guest operating system and associated application software can run. Multiple virtual machines can operate on the same host system concurrently.

Virtual SMP
A VMware proprietary technology that supports multiple virtual CPUs in a single virtual machine.

Virtual Switch (vSwitch)
A software equivalent to a traditional network switch.

Virtualization Assist
A general term for technology included in some 64-bit processors from AMD and Intel that can allow 64-bit operating systems to be run in virtual machines (where supported by VMware Workstation). More information is available in VMware knowledge base article 1901. See also AMD Virtualization and Virtualization Technology.

Virtualization Overhead
The cost difference between running an application within a virtual machine and running the same application natively. Since running in a virtual machine requires an extra layer of software, there is by necessity an associated cost. This cost may be additional resource utilization or decreased performance.

Virtualization Technology (VT)
Intel's version of virtualization assist, included in some 64-bit Intel processors. See also Virtualization Assist.

vlance
One of the virtual network adapters available in a virtual machine running in ESX. The vlance adapter emulates an AMD PCnet32 device. Note that in some cases NIC morphing can automatically convert a vlance device into a VMXNET device. See also NIC Morphing, E1000, and VMXNET.

VMDirectPath I/O
A vSphere feature that leverages Intel VT-d and AMD-Vi hardware support to allow guest operating systems to directly access hardware devices.

VMFS (Virtual Machine File System)
A high performance cluster file system.

VMI (Virtual Machine Interface)
A paravirtualization interface supported by ESX. See also Paravirtualization.

vMotion
A feature allowing running virtual machines to be migrated from one physical server to another with no downtime.

VMware Infrastructure Client (VI Client)
A graphical user interface used to manage ESX hosts or VirtualCenter servers. Renamed vSphere Client in vSphere 4.0.

VMware vCenter Update Manager
Provides a patch management framework for VMware vSphere. It can be used to apply patches, updates, and upgrades to VMware ESX and ESXi hosts, Windows and Linux virtual machines, VMware Tools, and so on.

VMware vStorage APIs for Array Integration (VAAI)
A set of APIs that can significantly improve storage scalability by offloading a number of operations to the storage device instead of performing them in ESX.

VMware Tools
A suite of utilities and drivers that enhances the performance and functionality of your guest operating system. Key features of VMware Tools include some or all of the following, depending on your guest operating system: an SVGA driver, a mouse driver, the VMware Tools control panel, and support for such features as shared folders, shrinking virtual disks, time synchronization with the host, VMware Tools scripts, and connecting and disconnecting devices while the virtual machine is running.

VMXNET
One of the virtual network adapters available in a virtual machine running in ESX. The VMXNET adapter is a high-performance paravirtualized device with drivers (available in VMware Tools) for many guest operating systems. See also Enhanced VMXNET, VMXNET3, E1000, vlance, and NIC Morphing.

VMXNET3 (VMXNET Generation 3)
The latest in the VMXNET family of paravirtualized network drivers. Requires virtual hardware version 7.

vSphere Client.
A graphical user interface used to manage ESX hosts or VirtualCenter servers. Previously called the VMware Infrastructure Client (VI Client).

W

Wake-on-LAN
A feature allowing a computer system to be powered on or brought out of suspend by sending a command over Ethernet.

Index