vmware

VMware ESX Server

Performance Tuning Best Practices for ESX Server 3

The paper provides a list of performance tips that cover the most performance-critical areas of Virtual Infrastructure 3 (VI3). This is not intended as a comprehensive guide for planning and configuring for your deployments. The intended audience is system administrators who have already deployed VI3 and are looking to maximize their performance. The paper assumes the reader is already familiar with Virtual Infrastructure concepts and terminology.

The following sections are included:

- "Setup and General Issues" on page 1
- "CPU Performance Best Practices" on page 3
- "Memory Performance Best Practices" on page 5
- "Networking Performance Best Practices" on page 6
- "Storage Performance Best Practices" on page 8
- "Resource Management Best Practices" on page 12
- "DRS Best Practices" on page 15
- "Benchmarking Best Practices" on page 17
- "Related Publications" on page 22

Setup and General Issues

VMware recommends the following practices and configurations for optimal performance:

- Validate your hardware:
  - Test memory for 72 hours, checking for hardware errors.
  - Check the performance of hardware components using a Windows or Linux operating system to make sure they are running at their maximum capacities. Run benchmarks if necessary.
  - Check installed items against the supported hardware list.

  - Use minimum configuration or greater.
- Plan the deployment. Allocate enough resources especially for the Service Console.

  The following table lists symptoms of insufficient resources for the service console:

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

- Use supported guests. See the *Guest Operating System Installation Guide*.

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

- Install the latest version of VMware Tools in the guest operating system and ensure it stays updated after each ESX Server upgrade.

  Installing VMware Tools updates the BusLogic driver within the guest operating system to the VMware supplied driver. The VMware driver has certain optimizations that guest-supplied drivers do not.

  The balloon driver used for memory reclamation on ESX is part of VMware Tools. Ballooning will not work if Tools is not installed.
- Always use VMware Tools, the VI Client, or esxtop to measure resource utilization. CPU and memory usage reported in the guest can be different from what ESX reports. See "Related Publications" on page 22, KB article 2032.
- Disable screen savers and Window animations in the virtual machine. On Linux, if using an X server is not required, disable it.

  Screen savers, animations, and X servers all consume extra CPU which can affect performance of other virtual machines and consolidation ratios. The non-trivial consumption of CPU by idling virtual machines can also have an adverse impact on Dynamic Resource Scheduling (DRS) decisions.
- Disconnect unused, unnecessary devices on both the guest and on the host:
  - COM ports
  - LPT ports
  - Floppy drives
  - CD-ROM drives
  - USB adapters

  Disabling USB and other devices on the host frees IRQ resources and eliminates IRQ sharing conflicts that can cause performance problems. Some devices, such as

USB, also consume extra CPU because they operate on a polling scheme. See "Related Publications" on page 22, KB article 1290.

Windows guests poll CD devices quite frequently. When multiple guests try to access the same physical CD drive, performance suffers. Disabling CD devices in virtual machines when they are not needed alleviates this.

- Schedule backups and anti-virus programs in the virtual machines to run at off-peak hours. In general, it is a good idea to distribute CPU consumption evenly, not just across CPUs but also across time. For workloads such as backups and antivirus where the load is predictable, this is easily achieved by scheduling the jobs appropriately.

CPU Performance Best Practices

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

For applications that are CPU-bound (that is, most of the application's time is spent executing instructions rather than waiting for external events such as user interaction, device input or data retrieval), any CPU virtualization overhead translates into a reduction in overall performance. Applications that are not CPU-bound can still deliver comparable performance because there are CPU cycles available to absorb the virtualization overhead.

VMware recommends the following practices and configurations for optimal CPU performance:

- When configuring virtual machines, remember that ESX Server itself has some overhead. Allow for the CPU overhead required by virtualization, and take care not to excessively overcommit processor resources (in terms of CPU utilizations and the total number of VCPUs).
- Use as few virtual CPUs (VCPUs) as possible. Do not use virtual SMP if your application is single threaded and does not benefit from the additional VCPUs, for example.

  Having virtual machines configured with virtual CPUs that are not used still imposes resource requirements on the ESX Server. In some guest operating systems, the unused virtual CPU still consumes timer interrupts and executes the idle loop of the guest operating system which translates to real CPU consumption from the point of view of the ESX Server. See "Related Publications" on page 22, KB articles 1077 and 1730.

  In ESX we try to co-schedule the multiple VCPUs of an SMP virtual machine. That is, we try to run them together in parallel as much as possible. Having unused VCPUs imposes scheduling constraints on the VCPU that is actually being used and can degrade its performance.
- When running multi-threaded or multi-process applications in an SMP guest, it can help to pin guest threads or processes to VCPUs. When processes in the guest migrate from one VCPU to another, they incur small CPU overhead. If the

migration is very frequent, it may help to pin the process in the guest operating system to a particular VCPU.

- Monitor idle loop spin parameter: Certain guest operating systems do not actually halt the virtual CPU when they idle. Since the virtual CPU is not halted, ESX will continue to run and schedule this virtual CPU as if it were still doing useful work. This execution of the idle loop results in unnecessary CPU consumption.

  To prevent such unneeded CPU consumption, ESX detects that a guest VCPU is attempting to idle and deschedules it. The detection heuristic is controlled by the idle loop spin parameter and can sometimes affect performance. For more details on optimally configuring this parameter, please refer to "Related Publications" on page 22, KB article 1730.

- Make sure to configure a single-processor virtual machine with a UP HAL/kernel. Multi-processor virtual machines must be configured with an SMP HAL/kernel.

  Most guest operating systems can be configured to use either a UP HAL/kernel or an SMP HAL/kernel. The UP operating system versions are for single-processor systems. If used on a multiprocessor system, a UP operating system 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, SMP operating systems used on single-processor systems are slightly slower than UP operating systems. See "Related Publications" on page 22, KB article 1730.

  NOTE Remember that when changing a virtual machine from multiprocessor to single-processor in Windows, the HAL/kernel usually remains SMP

- Avoid running programs in the service console that consume excessive amounts of CPU or memory. This can adversely affect performance of the virtual machines and ESX Server.

  It is a good idea to periodically monitor service console memory and CPU usage using esxtop. See "Related Publications" on page 22.

- ESX 3.0.1 has full support for 64-bit guest operating systems. 64-bit guests and applications can have better performance than corresponding 32-bit versions.

- The guest operating system timer rate can have an impact on performance. Linux guests keep time by counting timer interrupts. Unpatched 2.4 and earlier kernels program the virtual system timer to request clock interrupts at 100Hz (100 interrupts per second). Some 2.6 kernels request interrupts at 1000 Hz, others at 250 Hz, so you should check your kernel to determine the actual rate. These rates are for the UP kernels only and the SMP Linux kernels request additional timer interrupts. Please refer to "Related Publications" on page 22, Timekeeping in virtual machines. The overhead of delivering so many virtual clock interrupts can negatively impact guest performance and increase host CPU consumption. If you do have a choice, use guests that require lower timer rates.

  Microsoft Windows operating systems timer rates are specific to the version of Microsoft Windows, and which Windows HAL (hardware abstraction layer), is installed. For most uniprocessor Windows installations, the base timer rate is

usually 100Hz. Virtual machines running Microsoft Windows request 1000 interrupts per second if they are running certain applications that make use of the Microsoft Windows multimedia timer service and hence such multimedia applications should be avoided if possible.

Memory Performance Best Practices

There are two kinds of memory related overhead that are incurred by ESX Server virtual machines: the additional time to access memory within a virtual machine and the extra memory needed by ESX Server for its own code and data structures.

In ESX, the VMkernel manages all machine memory except for the memory allocated to the service console. The guest operating system running inside a virtual machine translates guest virtual memory addresses to guest physical memory addresses. The virtualization layer then translates guest physical memory addresses to physical memory addresses on the underlying machine. The virtualization layer does this by intercepting guest operating system instructions to manipulate guest address mappings. It is important to note that the mappings from guest physical memory addresses to machine memory addresses are used directly by the processor's paging hardware so that there is no overhead for each memory access.

The memory overhead is comprised of two components: a fixed system-wide overhead for the service console and the VMkernel, and an additional overhead for each virtual machine. For ESX Server 3.0, the service console typically uses 272MB and the VMkernel uses a smaller amount of memory. Overhead memory includes space reserved for the virtual machine frame buffer and various virtualization data structures. Overhead memory depends on the number of virtual CPUs, the configured memory for the guest operating system, and on whether you are using a 32-bit or 64-bit guest operating system.

The following table gives examples of per virtual machine memory overhead estimations:

Virtual CPUs	Memory (MB)	Overhead for a 32-bit virtual machine (MB)	Overhead for a 64-bit virtual machine (MB)
1	1024	84	180
1	8192	139	236
1	16384	203	300
2	1024	101	300
2	8192	221	413
2	16384	349	541
4	1024	141	523
4	8192	222	605
4	16384	350	734

The VMkernel reclaims memory by ballooning and swapping.

VMware recommends the following memory configurations and practices for optimal performance:

- Avoid frequent memory reclamation. Make sure the host has more physical memory than the total amount of memory that will be used by ESX plus the sum of the working set sizes that will be used by all the virtual machines running at any one time.

  NOTE ESX does, however, allow some memory overcommitment without impacting performance by using the memory management mechanisms described in "Resource Management Best Practices" on page 12.

- Carefully select the amount of virtual memory you allocate to your virtual machines to allow enough memory to hold the working set of applications you will run in the virtual machine.
- If possible, use less than 896MB of guest physical memory on Linux virtual machines. Linux uses different techniques to map memory in the kernel if the amount of physical memory is greater than 896MB. These techniques impose additional overhead on the virtual machine monitor and can result in slightly lowered performance.
- Due to page sharing and other techniques, ESX Server can be overcommitted on memory and still not swap. However, if the over-commitment is large and ESX is swapping, performance in the virtual machines is significantly reduced.
- If you choose to overcommit memory with ESX Server, you need to be sure you have sufficient swap space on your ESX Server. ESX server creates a swap file per virtual machine that is equal in size to the difference between the virtual machine's configured memory size and its reservation. This swap file is created at power on, so there are no performance implications in its creation. This swap space must be greater than or equal to the difference between the virtual machine's configured memory size and its reservation.

  NOTE The reservation is a guaranteed lower bound on the amount of memory that the host reserves for the virtual machine. This can be configured through Virtual Infrastructure client in the Memory resource settings for a virtual machine.

- If swapping cannot be avoided, for better performance ensure that the virtual machine's swap file is placed on a high speed/bandwidth storage system. By default, a virtual machine's swap file is created in the same location where the virtual machine is located. This can be changed by setting the **sched.swap.dir** option (in the VI client, **Edit Settings** > **Options** > **Advanced** > **Configuration Parameters**) to the appropriate location path.

Networking Performance Best Practices

VMware recommends the following networking configurations for optimal performance:

- Before undertaking any network optimization effort, you must understand the physical aspects of the network. The following are just a few aspects of the physical layout that merit close consideration:

  - Are both the client and the server members of a simple local area network (LAN) where the systems are connected to the same switch? If not, how many network hops do the packets need to traverse between the two systems?
  - What are the types of network cards in interacting machines? Server-class NICs are often able to offer better performance.
  - Are both the client and the server configured to use autonegotiation to set speed and duplex settings? Are they configured for half-duplex or full-duplex mode?
  - What size packets are transmitted through the network? Do the packets need to be fragmented or re-assembled along the transmission path?
- Multiple network adapters from a single vSwitch to the physical network form a NIC team. Such a NIC team can increase performance by distributing the traffic across those physical network adapters and provide passive failover in the event of hardware failure or network outage.
- The default virtual network adapter emulated inside 32-bit guests is the AMD PCnet32 device configured with VMware's vlance driver (e1000 for 64-bit guests). However, vmxnet provides much better performance than vlance and should be used for optimal performance.

  The vmxnet driver implements an idealized network interface that passes through network traffic from the virtual machine to the physical cards with minimal overhead. To use the vmxnet network adapter, install the vmxnet driver (available in VMware Tools) on your virtual machines.

  Note that the network speeds reported by the guest networking driver on the virtual machine do not necessarily reflect the actual capacity of the underlying physical network interface card. For example, the vlance guest driver on 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 Server emulates are 10Mbps, by definition. However, ESX Server is not limited to 10Mbps and transfers network packets as fast as the resources on the physical server machines allow.
- Use separate vSwitches (and consequently separate physical network adapters) to avoid contention between service console, VMkernel, and virtual machines, and between virtual machines running heavy networking workloads.
- In a physical environment, CPU utilization plays a significant role in reaching acceptable network throughput. To process higher levels of throughput, more CPU resources are needed. The effect of CPU resource availability on network throughput of virtualized applications is even more significant. Insufficient CPU resources will reduce maximum throughput, and it is important to monitor CPU utilization for such high throughput workloads.
- The VMkernel network device drivers start with a default speed and duplex setting of auto-negotiate. This setting will work correctly with network switches set to auto-negotiate. If your switch is configured for a specific speed and duplex setting, you must force the network driver to use the same speed and duplex setting (for gigabit links, network settings should be set to auto-negotiate and not forced).

If you encounter problems — in particular, very low bandwidth — it is likely that the NIC did not auto-negotiate properly, and you should configure the speed and duplex settings manually. To resolve the problem, either change the settings on your switch or change the settings for the VMkernel network device using the VI Client.

- To establish a network between two virtual machines that reside on the same ESX host, 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.
- In some cases, low throughput between virtual machines on the same ESX Server may be caused by buffer overflows in the guest driver. Buffer overflows require the sender to retransmit data, thereby limiting bandwidth.

  Possible workarounds are to increase the number of receive buffers, reduce the number of transmit buffers, or both. These workarounds may increase workload on the physical CPUs. The default number of receive and transmit buffers for vmxnet is 100 each. The maximum possible for vmxnet is 128. You can alter the default settings by changing the buffer size defaults in .vmx (configuration) files for the affected virtual machines. See "Related Publications" on page 22, KB article 1428.
- In ESX 3.0, the NIC morphing feature automatically replaces a virtual machine's vlance device with the vmxnet device for improved performance. This requires VMware Tools be installed within the guest.
- For the best networking performance, we recommend using a network adapter that supports the following hardware features:
  - Checksum offload
  - TCP segmentation offload (TSO)
  - Capability to handle high memory DMA (i.e. handle 64-bit DMA addresses)
  - Capability to handle multiple Scatter Gather elements per Tx frame
- In addition to the PCI and PCI-X architectures, we now have the PCIe architecture. 10Gig Ethernet network adapters should ideally use PCIe-8x or PCI-X 266. Preferably, there is 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) that can lower performance.

Storage Performance Best Practices

Back-end storage configuration greatly affects performance:

- See "Using ESX Server with SAN: Concepts" in the *SAN Configuration Guide* for SAN best practices.
- See "Advanced Networking" in the *Server Configuration Guide* for NFS best practices.
- See "Configuring Storage" in the *Server Configuration Guide* for iSCSI best practices.

Storage performance issues you encounter are most often the result of configuration issues with underlying storage devices and are not specific to ESX Server. If you suspect a storage performance problem, you may want to try running applications natively, if possible, to determine if the problem is due to a storage configuration issue. If storage is centrally managed or configured in your environment, contact your storage administrator.

This is a vast area that depends on workload, hardware, vendor, RAID levels, cache sizes, stripe sizes, 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. Therefore, it is very important to have the storage device configured correctly. VMware recommends the following storage configurations and practices for optimal performance:

- Make sure I/O is not queueing up in the VMkernel by checking the number of queued commands reported by esxtop.

  Since queued commands are an instantaneous statistic, you will need to monitor it over a period of time to see if you are hitting the queue limit. To determine queued commands, 'QUED' is the counter to look for in the esxtop, storage resource screen. If queueing, then try to adjust queue depths. See "Related Publications" on page 22, KB article 1267.

- To optimize storage array performance, spread I/O loads over the available paths to the storage (across multiple HBAs and SPs).

- For Active/Active arrays with fixed failover policy, designate preferred paths to each logical unit of storage. Doing so allows for the best possible utilization of your bandwidth to the disk array.

- Avoid operations that would excessively open/close files on the VMFS, a distributed file system, partition as they tend to be expensive. If possible access a file, do all that needs to be done with it and close it, instead of opening, doing something and closing in a tight loop. As an example, the `watch tail file_on_vmfs_partition` command in the service console can impact performance adversely.

- Virtual Disk Modes – ESX Server supports the following disk modes: Independent persistent, Independent nonpersistent and snapshots. These modes have the following characteristics:

  - Independent Persistent – Changes are immediately and permanently written to the disk, so they have high performance.

  - Independent Nonpersistent – Changes to the disk are discarded when you power off or revert to the snapshot. In this mode disk writes are appended to a redo log. When a virtual machine reads from disk, it first checks the redo log (by looking at a directory of disk blocks contained in the redo log) and, if the redo log is present, reads that information. Otherwise, the read goes to the base disk for the virtual machine. 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, can incur a performance penalty.

- Snapshot – A snapshot captures the entire state of the virtual machine at the time you take the snapshot. This includes the memory and disk states as well as the virtual machine settings. When you revert to a snapshot, you return all these items to the state they were in at the time you took that snapshot

- The alignment of your file system partitions can impact performance. We make the following recommendations for VMware VMFS partitions:
  - Like other known disk based file systems, VMFS suffers a penalty when the partition is unaligned. Use the Virtual Infrastructure client to create VMFS partitions since the VI client automatically aligns the partitions along the 64 KB boundary.
  - To manually align your VMware VMFS partitions, first check your storage vendor's recommendations for the partition starting block.
  - Carefully evaluate the I/O workload against your unaligned VMware VMFS partitions.

    For example, workloads consisting of mostly sequential reads gain the most from partition alignment. If the workload is light, the system already meets service level agreements, or the workload contains many random writes, consider delaying the partition alignment procedure until a scheduled outage or not migrating at all.
  - Please ensure that file system partitions within the guest are also aligned.

  See "Related Publications" on page 22, Recommendations for Aligning VMFS Partitions.

- Fibre Channel SANs typically yield higher performance than NAS and iSCSI. Fibre Channel SANs do not suffer from the congestion and oversubscription problems as readily as NAS and iSCSI because of the Fibre Channel protocol.

  Fibre Channel includes a mechanism for port-to-port throttling, not just end-to-end throttling. This means a Fibre Channel switch can limit the amount of data a connected system sends and avoids situations in which information must be dropped. Therefore, Fibre Channel does not degrade into a retransmission recovery scheme that saps the storage network's performance, and is more capable of operating near wire speed. Fibre Channel protocol transmission bandwidth is 2Gbps versus the 1Gbps bandwidth of high speed Ethernet used by NAS and iSCSI, so the FC protocol has the edge in terms of raw bandwidth compared to either NAS or iSCSI. With the more widespread adoption of 10 Gigabit Ethernet (support for it exists in ESX 3.0), the performance scales may shift towards NAS and iSCSI with these 10G deployments.

- Ensure that heavily-used virtual machines do not all access the same VMFS volume concurrently and that they are spread across multiple VMFS volumes (typically a VMFS volume will span a single LUN, but this is not always true). Heavy SAN I/O when a large number of virtual machines access the same VMFS volume concurrently will cause poor disk performance.

- Avoid operations that require excessive file locks or meta data locks, such as dynamically growing vmdk files and file permissions manipulation.

- Configure maximum queue depth for the HBA card. See "Related Publications" on page 22, KB article 1267.
- Increase virtual machines' maximum outstanding disk requests if needed. See "Related Publications" on page 22, KB article 1268.
- For iSCSI/NFS, make sure several input Ethernet links are not funneled into fewer output links, resulting in an oversubscribed link. Any time a number of links transmitting near capacity are switched to a smaller number of links, oversubscription is a possibility.

  Recovering from 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 current transactions.
- Applications or systems that write a lot of data to storage, such as data acquisition or transaction logging systems, should not share Ethernet links to a storage device. These types of applications perform best with multiple connections to storage devices.
- 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 designing a network, but don't change the physical capabilities of links and trunks between switches.
- If you are working with systems with heavy disk I/O loads, you may need to assign separate storage processors to separate systems to handle the amount of traffic bound for the storage.
- Ensure adequate CPU, particularly for software-initiated iSCSI and NAS.
- Protect your service console's root file system from becoming full.
- Guest storage drivers typically set the I/O size at 64K as default. If applications issue I/Os that are larger than 64K then they are split into 64K chunks. Changing the registry settings to issue larger block sizes can enhance performance. See "Related Publications" on page 22, KB article 9645697.
- ESX Server emulates either a BusLogic or LSI Logic SCSI adapter, which is likely to be different from the physical adapter installed on the server. The specifics of the implementation of the SCSI driver loaded into the guest operating system can affect disk I/O throughput. For example, the depth of the queue of outstanding commands in a driver can significantly impact disk performance. A queue depth that is too small limits the disk bandwidth that can be pushed through the virtual machine.

  For the BusLogic adapters, VMware provides a custom BusLogic driver for Windows guest operating systems that is recommended for applications requiring high performance. The BusLogic driver is part of VMware Tools and can be installed when you install VMware Tools. The guest driver queues may also be tuned. See the driver-specific documentation for more information on how to do this.

The driver queue depth can also be set for some VMkernel drivers. For example, the default queue depth of the QLogic driver is 16. However, specifying larger queue depths may yield higher performance. You can also adjust the number of outstanding disk requests per virtual machine in the VMkernel through the ESX Server Management User Interface. Setting this parameter can help equalize disk bandwidth across virtual machines. See "Related Publications" on page 22, KB article 1268.

- 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 meta data used to manage and redirect disk accesses to the physical device. Virtual disks are recommended for most virtual disk storage. Raw disks may be needed in some cases.
- ESX supports multiple disk formats. Their description and performance implications are as following:
  - Thick – A thick disk has all space allocated at creation time. It has optimal I/O latencies during usage but may be a security issue as it may contain stale data as it exists on the physical media on creation.
  - Eager zeroed – An eager zeroed thick disk has all space allocated and zeroed out at creation time. Longer creation time, but optimal performance and better security.
  - Lazy zeroed – Zeroing out the block on first write, but same performance as thick and eager zeroed on subsequent writes.
  - Thin – Space required for thin-provisioned virtual disk is allocated and zeroed on demand as opposed to upon creation. There is a higher I/O penalty during the first write to an unwritten file block, but same performance as thick and eager zeroed on subsequent writes.

  Virtual machine disks configured through the Virtual Center interface are of type Lazy Zeroed. The other disk formats can be created from the console command line using vmkfstools. For more details refer to the vmkfstools man page.

Resource Management Best Practices

ESX 3 provides several mechanisms to configure and adjust the allocation of resources, both CPU and Memory for virtual machines running on the ESX host. Resource management configurations can have a significant impact on virtual machine performance.

VMware recommends the following resource management configurations and practices for optimal performance:

- 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 upgrade the host, 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. The host assigns additional resources as available based on the number of shares and the limit for your virtual machine.

  As indicated before, reservations can be used to specify the minimum CPU and memory reservation 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.

  When specifying the reservations for virtual machines, always leave some headroom. Do not commit all resources. As you move closer to fully reserving all capacity in the system, it becomes increasingly difficult to make changes to reservations and to the resource pool hierarchy without violating admission control. 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.

- 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 in a resource pool. This allows resources to be assigned for the service as a whole.

- Hyperthreading technology allows a single physical processor to behave like two logical processors. The processor can run two independent applications at the same time. While hyperthreading does not double the performance of a system, it can slightly increase performance by keeping the processor pipeline busier.

  If the hardware and BIOS support hyperthreading, ESX automatically makes use of it. To enable hyperthreading:

  a Ensure that your system supports hyperthreading technology. All Intel Xeon MP processors and all Intel Xeon DP processors with 512 L2 cache support hyperthreading. However, not every Intel Xeon system ships with a BIOS that supports hyperthreading. Consult your system documentation to see if the BIOS includes support for hyperthreading.

  b Enable hyperthreading in the system BIOS. Some manufacturers label this option Logical Processor while others call it Enable Hyperthreading.

- An ESX Server system enabled for hyperthreading should behave almost exactly like a standard system. Logical processors on the same core have adjacent CPU numbers, so that CPUs 0 and 1 are on the first core together, CPUs 2 and 3 are on the second core, and so on.

  VMware ESX Server 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, which 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.

- ESX 3 provides configuration parameters for controlling the scheduling of virtual machines on hyperthreaded systems. When choosing hyperthreading sharing choices, sometimes **None** is preferable to **Any** (which is the default) or **Internal**. None indicates that the virtual CPUs of the virtual machine should not share cores with each other or with virtual CPUs from other virtual machines. That is, each virtual CPU from this virtual machine should always get a whole core to itself, with the other logical CPU on that core being placed into the halted state.

  For typical workloads, custom hyperthreading settings should not be necessary, but it can help in case of unusual workloads that interact badly with hyperthreading. For example, an application with cache thrashing problems might slow down an application sharing its physical core. You could place the virtual machine running the application in the none or internal hyperthreading status to isolate it from other virtual machines.

  The trade-off for configuring **None** should also be considered. On system with a limited number of cores (per virtual machine) there might be no core to which the virtual machine that is descheduled can be migrated. As a result, it is possible that virtual machines with hyperthreading set to none or internal can experience performance degradation, especially on systems with a limited number of cores.

- Both IBM and AMD NUMA systems are supported in ESX 3.0. AMD Opteron-based systems, such as the HP ProLiant DL585 Server, can be configured as NUMA architecture. The BIOS setting for node interleaving determines whether the system behaves more like a NUMA system, or more like a Uniform Memory Architecture (UMA) system. For more information, see the HP ProLiant DL585 Server Technology manual.

  If node interleaving is disabled ESX Server detects the system as NUMA and applies NUMA optimizations. If you enable node interleaving (also known as interleaved memory), ESX Server does not detect the system as NUMA.

  The intelligent, adaptive NUMA scheduling and memory placement policies in VMware ESX Server 3 can manage all virtual machines transparently, so that administrators do not need to deal with the complexity of balancing virtual machines between nodes by hand. However, manual override controls are also available 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 hand.

  This may be useful, for example, if a virtual machine runs a memory-intensive workload, such as an in-memory database or a scientific computing application with a large data set. Such an application may see performance improvements if 100% of its memory is allocated locally, while virtual machines managed by the automatic NUMA optimizations often have a small percentage (5-15%) of their memory located remotely. An administrator may also wish to optimize NUMA placements manually if the system workload is known to be simple and unchanging; for example, an eight processor system running eight virtual machines with similar workloads would be easy to optimize by hand. Keep in mind if you manually set the processor or memory affinity for a virtual machine, the NUMA scheduler may not be able to automatically manage this virtual machine.

- Please be aware that ESX NUMA scheduling and related optimizations are enabled only on dual-core systems with a total of at least four cores. On such ESX NUMA scheduling enabled systems, for ideal performance ensure that for each virtual machine # of VCPUs + 1 <= # of cores per node. Virtual machines that are not managed automatically by the NUMA scheduler (single core and/or less then total of four cores) still run fine; they simply don't benefit from ESX Server's NUMA optimizations.
- ESX Server uses three memory management mechanisms — page sharing, ballooning, and swapping — to expand or contract the amount of memory allocated dynamically for each virtual machine. An ESX Server host uses a proprietary transparent page sharing technique to securely eliminate redundant copies of memory pages and is the most preferred memory reclamation technique among the three.

  Between Swapping and Ballooning, Ballooning is the preferred method and provides predictable performance, which closely matches the behavior of a native system under similar memory constraints. With ballooning, ESX Server works with a VMware-supplied vmmemctl module loaded into the guest operating system to reclaim pages that are considered least valuable by the guest operating system. The vmmemctl driver is installed as part of VMware Tools. To use ballooning, the guest operating system must be configured with sufficient swap space.

  Swapping is used to forcibly reclaim memory from a virtual machine when both page sharing and ballooning fail to reclaim sufficient memory from an overcommitted system. If the working set (active memory) of the virtual machine resides in physical memory, using the swapping mechanism and having inactive pages swapped out does not affect performance. However, if the working set is so large that active pages are continuously being swapped in and out (that is, the swap I/O rate is high), then performance may degrade significantly. To avoid swapping in specific virtual machines please configure memory reservations for them (through the VI Client) at least equal in size to their active working sets. But be aware that configuring resource reservations can limit the number of virtual machines one can consolidate on a system.

  The vmmemctl approach (ballooning) is used whenever possible for optimum performance. Swapping is a reliable mechanism of last resort that the system uses to reclaim memory only when necessary and leads to significant performance hit. So place the per virtual machine swap files on the fastest available storage.

DRS Best Practices

Clustering configurations can have significant impact on performance.

VMware recommends the following DRS configurations and practices for optimal performance:

- Generally speaking, don't specify affinity rules unless needed. In some cases, however, specifying affinity rules can help improve performance.
  - **Keep Virtual Machines Together** can improve performance due to lower latencies of communication.

  - **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 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.

- 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.

- When deciding which hosts to group into a DRS clusters, try to choose hosts that are as homogeneous as possible, in terms of CPU and memory. This would ensure higher performance predictability and stability. VMotion is not supported across hosts with incompatible CPU's. Hence with 'incompatible CPU' heterogeneous systems, DRS is limited in the number of opportunities for improving the load balance across the cluster.

  To ensure CPU compatibility ensure systems are configured with the same CPU vendor, with similar CPU family and SSE3 status. In the case where heterogeneous systems do have compatible CPUs, but have different CPU frequency and/or amounts of memory, the systems with more memory and higher CPU frequencies are generally preferred (all other things being equal) to locate virtual machines since they have more room to accommodate peak load.

- Virtual machines with a smaller memory sizes or fewer virtual CPUs provide more opportunities for DRS to migrate them in order to improve balance across the cluster. Virtual machines with larger memory size and/or more virtual CPUs add more constraints in migrating the virtual machines. Hence this is one more reason to configure only as many virtual CPUs and as much virtual memory for a virtual machine as needed.

- The more ESX hosts that are VMotion compatible, the more the choices to better balance the DRS cluster. Besides the incompatibility of CPUs, there may be some misconfiguration that may make two or more hosts incompatible. For instance, if the hosts' VMotion network adapters are not connected by GigE link then the VMotion may not occur between the hosts. Other configuration settings to check for are the VMotion gateway being misconfigured, VMotion network adapters security policies being incompatible between the source and destination hosts and virtual machine network availability on the destination host. Refer to the *Resource Management Guide* and the *Server Configuration Guide* for further details.

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

Benchmarking Best Practices

This section gives guidelines on how to benchmark virtual machines running with ESX 3. It describes how to generate fair comparisons and avoid common benchmarking pitfalls in virtual environments.

The following sections describe best practices for benchmarking particular performance areas:

- "General Guidelines" on page 17
- "CPU" on page 19
- "Memory" on page 20
- "Networking" on page 20
- "Storage" on page 21

General Guidelines

VMware recommends the following general guidelines for benchmarking tests:

- Before planning the testing, clearly define the parameters being measured and a metric with which to measure them (such as operations per second, jobs per hour, or average response time).
- If you are running a publicly-available benchmark, make sure you follow all the guidelines associated with configuring, running, and reporting for that benchmark, both on the native system and within the guest. If you are running a custom benchmark, make sure that it incorporates the well understood principles of benchmarking: specific test purpose, a meaningful metric (such as time, operations per second, or bytes transferred), reproducibility, and so forth.
- Any report of the benchmark results should include enough details about the experimental setup for the reader to reproduce the results. In your report please include details of:

  Host hardware configuration:

  - Firmware version
  - CPUs
  - Memory
  - Disks: Array configuration, number and speed (RPM) of disks backing the LUN, RAID configuration
  - Network adapters.
  - ESX and VMFS versions

  Virtual machine configuration:

  - Virtual CPU
  - Virtual Memory
  - Virtual Disk: size, disk format, vmdk spec

- Virtual network adapters
- Shares, reservations, limits and affinities of CPU, Memory and Disk
- Guest operating system and version (including service pack) installed
- Benchmark configuration

- When trying to saturate and benchmark any specific system component (such as CPU, memory, or network), ensure that no other resource on the system is constrained in either the native or the virtual machine. Be aware that virtualization has overhead in terms of CPU, memory, etc., and provision for this overhead for components not being tested. For example, before making a native to virtual machine network-bandwidth comparison, make sure the processor load is not limiting the bandwidth achieved in either case.
- For performance comparisons, it is best to run all tests on the same system. When running on the same system is not possible, at least use identical systems. Make sure to run all tests on hardware supported by the VMware software you are using.
- Timing numbers reported within the virtual machine can be inaccurate, especially when the processor is overcommitted. If you report numbers from within the guest, this fact should be noted in your results.

  One method of timing workloads is to ping an external machine and capture timestamps when that machine receives the pings. To use ping to time workloads, run tcpdump (in Linux) or WinDUMP (in Windows) on an external system. (The examples below show tcpdump in a C shell. WinDUMP works similarly.)

  On the external system, run `tcpdump ip proto \\icmp and host`.

  In this example, ip is the IP address of the system under test. From within the virtual machine, before starting the benchmark and after the benchmark finishes execution, run: `ping -c 1 time_host`.

  In this case, time_host is the IP address of the external system running tcpdump or WinDUMP. You can then determine the duration of the experiment by simply subtracting one time stamp from the other. If the benchmark takes long enough, you can also use a stopwatch to measure time.

  NOTE For more information about tcpdump and WinDUMP, see: http://www.tcpdump.org/ and http://www.winpcap.org/windump/

- In some cases your workload can't use the above ping method, perhaps because the timing granularity is so small that the network latency of the pings is significant. It is also possible for the guest operating system to read the real TSC's value as a pseudo performance counter using a VMware feature. rdpmc can be used to measure time from within the virtual machine by:

  a Adding `monitor_control.pseudo_perfctr=1` to the virtual machine config file.

  b Issuing `rdpmc 0x10000` in the virtual machine.

  See "Related Publications" on page 22, KB article 1420.

- Make sure to run the tests enough number of times to ensure that the percentage difference is minimal and the numbers are reproducible.
- During the execution of the workload, ensure that there is no other activity other than the benchmarks under consideration (virus scanner, NFS, cron jobs etc.).

CPU

While benchmarking please follow the following CPU related benchmarking best practices:

- For a fair performance comparison between native and virtual machines, configure the same number of physical CPUs in the native system as you configured virtual CPUs in the virtual machine.

  This may require reducing the number of processors in your system while doing the native tests. To do this in Windows, use the `/numproc` switch in the `boot.ini` file (Vista no longer contains the `boot.ini` file, instead use the `Msconfig.exe` or `bcdedit.exe` binaries to configure the number of processors). To do this in Linux, use the `maxcpus` variable in either the `grub.conf` or `lilo.conf` file depending on your Linux version.

- If comparing a native system to a virtual machine, make sure the HAL/kernel types in both are kept the same: either both UP or both SMP. When running a single-threaded benchmark, use a single-processor virtual machine. When running a multi-threaded benchmark with an SMP HAL/kernel, make sure there is enough parallelism to keep both CPUs busy.
- When conducting SMP scaling experiments, consider configuring both the single-processor and the multi-processor virtual machines with SMP HALs/kernels. If you keep the HAL/kernel the same in both systems you change only one variable. If your benchmarks don't involve processor scaling, it may be best to stick with single processor configurations.
- Create fresh virtual machines specifically for benchmarking tests rather than to reuse machines created previously.

  Most operating systems automatically select an appropriate HAL/kernel when they are first installed in a virtual machine. When a virtual machine with a UP HAL/kernel is reconfigured to have two processors, the typical behavior of the guest operating system is to automatically switch to an SMP HAL/kernel. If that virtual machine is later reconfigured to have a single processor, however, it typically does not automatically switch to a UP HAL/kernel.

  NOTE Some newer versions of Windows, with appropriate BIOS and hardware support, may be able to seamlessly switch between UP and SMP HALs.

  See "Related Publications" on page 22, KB article 1077.

- Don't overcommit CPU resources while running benchmarks. For example:
  - Avoid running a four-processor virtual machine on a dual-processor host system, even if the dual-processor host has hyper-threading (that is, four logical CPUs).

- Avoid running two or more SMP virtual machines on a dual-processor host system, even if the dual-processor host has hyper-threading.
- Since virtualization has its own CPU overhead, make sure that CPU resources are not overcommitted on the host system.
- If benchmarking server performance, use a remote client running on physical machine and make sure that client is not a bottleneck.

See "CPU Performance Best Practices" on page 3.

- 64-bit versions of both operating systems and application software are becoming increasingly common. Make sure you use similar types of operating system and application software (that is, 32-bit or 64-bit), both natively and within the guest (for example, compare a 64-bit operating system running a 32-bit application natively to a 64-bit operating system running a 32-bit application within the guest, not to a 64-bit operating system running a 64-bit application within the guest).

Memory

While benchmarking please follow the following Memory related benchmarking best practices:

- For a fair performance comparison between native and virtual machines, configure the same amount of physical memory in the native system as you configured virtual memory in the virtual machine.

  This may require reducing the amount of memory in your system when doing the native tests. To do this in Windows, use the `/maxmem` switch in the `boot.ini` file (Vista no longer contains the `boot.ini` file, instead use the `Msconfig.exe` or `bcdedit.exe` binaries to configure the amount of system memory). To do this in Linux, use the `mem` variable in either the `grub.conf` or `lilo.conf` file depending on your Linux version.

Networking

While benchmarking please follow the following Networking related benchmarking best practices:

- Avoid cross-traffic noise over the network while conducting the tests. Either use direct cables between the systems or use a private network switch. Use dedicated network interface cards on both machines for the connection.
- Use network switches instead of hubs.
- Make sure that all the networking infrastructure is appropriately rated. For example, when connecting systems containing Gigabit network interface cards, make sure to use Gigabit switches and Gigabit-rated cables.
- If possible, use similar network interface cards on systems under test so that they function well with each other. Using similar cards also helps to ensure that send and receive have similar performance. Ideally, you should use similar client and server machines as well, with similar PCI architecture and configuration. The difference between PCI and PCI-X, for example, can affect networking performance.

- Don't have more physical network interface cards than absolutely necessary. This avoids the unnecessary overhead associated with processing broadcast packets, protocol control packets, and so forth.

  If your system has multiple physical network interface cards, make sure you are using the intended network interface cards for the test. If you have more than one network interface card, it is easy to enable and use the wrong one. To avoid this confusion, disable the network interface cards you do not plan to use.

- Do not use the same NIC for the service console and for your virtual machine(s).
- Most modern network interface cards can operate in multiple modes (such as 10, 100, or 1000Mbps; half duplex or full duplex). Make sure the network interface cards are in full duplex mode and are configured at their maximum possible bandwidth.
- Don't change any of the default network interface card driver settings unless there is a valid reason to do so. Use the OEM listed recommendations.
- To establish a network between two virtual machines that reside on the same ESX host, 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.
- Remove or disable any virtual networking devices that are not required for your tests.

Storage

While benchmarking please follow the following Storage related benchmarking best practices:

- For storage tests involving a single LUN, please make sure you use the same LUN for all tests. If you are testing various versions of VMFS, please recreate VMFS on the same LUN. There is no need to re-create any LUN between tests.
- For storage tests involving multiple LUNs, assign a set of LUNs required for the experiment and make sure the same LUNs are used for all tests.
- Zone the LUNs used for the benchmarking experiments strictly to the benchmarking host.
- Please make sure that the VMFS partitions used in the experiments are aligned (by creating them using Virtual Infrastructure 3 client or Virtual Center 2.0 and later).
- Please make sure that the same version of VMFS is used for all storage tests. In the event that there is testing across versions, one of the things to keep in mind is using the same block size while re-creating the file system. Use the default creation options if you are not sure.
- If it is not possible to use the same LUN/FS for the test, choose another LUN that satisfies the following conditions:
  - The new LUN should be on the same array.
  - The new LUN should have the same number and speed(RPM) of disks as the LUN it is being compared to.

- The RAID type should be the same as the LUN it is being compared to.

Related Publications

For technical papers providing more detail on topics discussed in this paper refer www.vmware.com/vmtn/resources.

Issue	Publication
Mismatched HALs	KB article 1077
IRQ sharing	KB article 1290
Idle loop	KB article 1730
CPU utilization	KB article 2032
Network throughput between virtual machines	KB article 1428
Queue depths on QLogic	KB article 1267
Guest storage drivers	KB article 9645697
Disk outstanding commands parameter	KB article 1268
Linux guest timing	KB article 1420
Timing in virtual machines	Timekeeping in VMware Virtual Machines
Virtual SMP Best Practices	Best Practices Using Virtual SMP
VMFS partitions	Recommendations for Aligning VMFS Partitions
esxtop	Using esxtop to Troubleshoot Performance Problems
SAN best practices	*SAN Configuration Guide*
NFS best practices	*Server Configuration Guide*
iSCSI best practices	*Server Configuration Guide*
Memory allocation	*Resource Management Guide*
Swap space	*Resource Management Guide*

VMware, Inc. 3145 Porter Drive Palo Alto, CA 94304 www.vmware.com
 Revision 20070125 Version: 1.0 Item: ESX ENG Q107 351