Performance Best Practices for VMware vSphere 6.5

VMware ESXi 6.5

vCenter Server 6.5

vmware®

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: 20210128

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 6.5*, provides performance tips that cover the most performance-critical areas of VMware vSphere® 6.5. It is not intended as a comprehensive guide for planning and configuring your deployments.

Chapter 1, "Hardware for Use with VMware vSphere," on page 11, provides guidance on selecting hardware for use with vSphere.

Chapter 2, "ESXi and Virtual Machines," on page 19, provides guidance regarding VMware ESXi™ software and the virtual machines that run in it.

Chapter 3, "Guest Operating Systems," on page 47, provides guidance regarding the guest operating systems running in vSphere virtual machines.

Chapter 4, "Virtual Infrastructure Management," on page 57, provides guidance regarding infrastructure 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.

Intended Audience

This book is intended for system administrators who are planning a VMware vSphere 6.5 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 ESXi documentation set.

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

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

You can access performance and other technical papers on the VMware Technical Papers page:

http://www.vmware.com/vmtn/resources

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

This section provides guidance regarding CPUs for use with vSphere 6.5.

General CPU Considerations

- When selecting hardware, it is a good idea to consider CPU compatibility for VMware vSphere® vMotion™ (which in turn affects DRS, DPM, and other features) and VMware Fault Tolerance. See "VMware vMotion and Storage vMotion" on page 67, "VMware Distributed Resource Scheduler (DRS)" on page 70, and "VMware Fault Tolerance" on page 79.

Hardware-Assisted Virtualization

Most processors from both Intel® and AMD include hardware features to assist virtualization and improve performance. These features—hardware-assisted CPU virtualization, MMU virtualization, and I/O MMU virtualization—are described below.

NOTE For more information about virtualization techniques, see http://www.vmware.com/files/pdf/software_hardware_tech_x86_virt.pdf.

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

Hardware-assisted CPU virtualization assistance, called VT-x (in Intel processors) or AMD-V (in AMD processors), automatically traps sensitive events and instructions, eliminating the software overhead of monitoring all supervisory level code for sensitive instructions. In this way, VT-x and AMD-V give the virtual machine monitor (VMM) the option of using either hardware-assisted virtualization (HV) or binary translation (BT). While HV outperforms BT for most workloads, there are a few workloads where the reverse is true.

NOTE For a 64-bit guest operating system to run on an Intel processor, the processor must have hardware-assisted CPU virtualization.

For information about configuring the way ESXi uses hardware-assisted CPU virtualization, see "Configuring ESXi for Hardware-Assisted Virtualization" on page 23.

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

Hardware-assisted MMU virtualization, called rapid virtualization indexing (RVI) or nested page tables (NPT) in AMD processors and extended page tables (EPT) in Intel processors, addresses the overheads due to memory management unit (MMU) virtualization by providing hardware support to virtualize the MMU.

Without hardware-assisted MMU virtualization, the guest operating system maintains guest virtual memory to guest physical memory address mappings in guest page tables, while ESXi 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 ESXi to maintain shadow page tables. This reduces memory consumption and speeds up workloads that cause guest operating systems to frequently modify page tables. While hardware-assisted MMU virtualization improves the performance of most workloads, it does increase the time required to service a TLB miss, thus reducing the performance of workloads that stress the TLB. However this increased TLB miss cost can be mitigated by configuring the guest operating system and applications to use large memory pages, as described in "2MB Large Memory Pages for Hypervisor and Guest Operating System" on page 30.

For information about configuring the way ESXi uses hardware-assisted MMU virtualization, see "Configuring ESXi for Hardware-Assisted Virtualization" on page 23.

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

Hardware-assisted I/O MMU virtualization, called Intel Virtualization Technology for Directed I/O (VT-d) in Intel processors and AMD I/O Virtualization (AMD-Vi or IOMMU) in AMD processors, is an I/O memory management feature that remaps I/O DMA transfers and device interrupts. This feature (strictly speaking, a function of the chipset, rather than the CPU) can allow virtual machines to have direct access to hardware I/O devices, such as network cards, storage controllers (HBAs), and GPUs.

For information about using hardware-assisted I/O MMU virtualization, see "DirectPath I/O" on page 40 and "Single Root I/O Virtualization (SR-IOV)" on page 41.

AES-NI Support

vSphere 6.5 includes features that perform significantly better, incur significantly lower CPU load, or both, on hardware that supports AES-NI (Intel's Advanced Encryption Standard New Instruction Set). For the best performance with these features:

- Choose processors that support AES-NI, especially some recent processor versions in which the AES-NI implementation is further optimized.
- Make sure your BIOS version supports AES-NI (in some cases a BIOS upgrade might be required for AES-NI support).
- Make sure AES-NI is enabled in BIOS.

On a host that supports AES-NI, that support is exposed by default to virtual machines running on the host. This can be changed if desired by using Enhanced vMotion Compatibility (EVC) or modifying the CPU mask (see VMware KB article 1993). One situation in which disabling AES-NI passthrough might be desirable is to allow vMotion compatibility from a host that supports AES-NI to one that does not.

For more about the features that use AES-NI, see "vSphere Virtual Machine Encryption Recommendations" on page 36, "VMware vMotion Recommendations" on page 67, and "VMware Virtual SAN (vSAN)" on page 83.

Hardware Storage Considerations

Back-end storage configuration can greatly affect performance. For more information on storage configuration, refer to the *vSphere Storage* document for VMware vSphere 6.5.

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

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.

- VMware Storage vMotion performance is heavily dependent on the available storage infrastructure bandwidth. We therefore recommend you consider the information in "VMware vMotion and Storage vMotion" on page 67 when planning a deployment.
- Consider providing flash devices for the vSphere Flash Infrastructure layer. This layer can be used to store a host swap file (as described in "Memory Overcommit Techniques" on page 27) and for vSphere Flash Read Cache (vFRC) files (as described in "vSphere Flash Read Cache (vFRC)" on page 32).

  The vSphere Flash Infrastructure layer can be composed of PCIe flash cards or SAS- or SATA-connected SSD drives, with the PCIe flash cards typically performing better than the SSD drives.
- Consider choosing PCIe flash cards that use the Non-Volatile Memory Express (NVMe) protocol (see "NVMe Recommendations" on page 36 for details about NVMe support in vSphere 6.5).
- Consider choosing storage hardware that supports vStorage APIs for Storage Awareness (VASA), thus allowing you to use VVols (as described in "VMware Virtual Volumes (VVols)" on page 85).

  Because VVol performance varies significantly between storage hardware vendors, make sure the storage hardware you choose will provide the VVol performance you expect.
- Consider choosing storage hardware that supports VMware vStorage APIs for Array Integration (VAAI), allowing some operations to be offloaded to the storage hardware instead of being performed in ESXi.

  Though the degree of improvement is dependent on the storage hardware, VAAI can improve storage scalability, can reduce storage latency for several types of storage operations, can reduce the ESXi host CPU utilization for storage operations, and can reduce storage network traffic.

  On SANs, VAAI offers the following features:

  - Scalable lock management (sometimes called "hardware-assisted locking," "Atomic Test & Set," or ATS) replaces the use of SCSI reservations on VMFS volumes when performing metadata updates. This can reduce locking-related overheads, speeding up many administrative tasks as well as increasing I/O performance for thin VMDKs. ATS helps improve the scalability of very large deployments by speeding up provisioning operations such as expansion of thin disks, creation of snapshots, and other tasks.
  - Extended Copy (sometimes called "full copy," "copy offload," or XCOPY) allows copy operations to take place completely on the array, rather than having to transfer data to and from the host. This can dramatically speed up operations that rely on cloning, such as Storage vMotion, while also freeing CPU and I/O resources on the host.
  - Block zeroing (sometimes called "Write Same") speeds up creation of eager-zeroed thick disks and can improve first-time write performance on lazy-zeroed thick disks and on thin disks.
  - Dead space reclamation (using the UNMAP command) allows hosts to convey to storage which blocks are no longer in use. On a LUN that is thin-provisioned on the array side this can allow the storage array hardware to reuse no-longer needed blocks.

    NOTE In this context, "thin provisioned" refers to LUNs on the storage array, as distinct from thin provisioned VMDKs, which are described in "Virtual Disk Types" on page 33.

On NAS devices, VAAI offers the following features:

- Hardware-accelerated cloning (sometimes called "Full File Clone," "Full Copy," or "Copy Offload") allows virtual disks to be cloned by the NAS device. This frees resources on the host and can speed up workloads that rely on cloning. (Note that Storage vMotion does not make use of this feature on NAS devices.)
- Native Snapshot Support (sometimes called "Fast File Clone") can create virtual machine linked clones or virtual machine snapshots using native snapshot disks instead of VMware redo logs. This feature, which requires virtual machines running on virtual hardware version 9 or later, offloads tasks to the NAS device, thus reducing I/O traffic and resource usage on the ESXi hosts.

  NOTE Initial creation of virtual machine snapshots with NAS native snapshot disks is slower than creating snapshots with VMware redo logs. To reduce this performance impact, we recommend avoiding heavy write I/O loads in a virtual machine while using NAS native snapshots to create a snapshot of that virtual machine.

  Similarly, creating linked clones using NAS native snapshots can be slightly slower than the same task using redo logs.

- Reserve Space allows ESXi to fully preallocate space for a virtual disk at the time the virtual disk is created. Thus, in addition to the thin provisioning that non-VAAI NAS devices support, VAAI NAS devices also support lazy-zeroed thick provisioning and eager-zeroed thick provisioning.
- Extended Statistics provides visibility into space usage on NAS datastores. This is particularly useful for thin-provisioned datastores, because it allows vSphere to display the actual usage of oversubscribed datastores.

For more information about VAAI, see *VMware vSphere Storage APIs — Array Integration (VAAI)* (though written for vSphere 5.1, most of the content applies to vSphere 6.5). For information about configuring the way ESXi uses VAAI, see "ESXi Storage Considerations" on page 32.

- If you plan to use VMware Virtual SAN (vSAN), consider the advantages of an all-flash vSAN deployment versus a hybrid deployment (see "Hybrid versus All-Flash vSAN" on page 83).
- If you plan to use virtual machine encryption (see "vSphere Virtual Machine Encryption Recommendations" on page 36) consider choosing hardware that supports the AES-NI processor instruction set extensions. For more information, see "AES-NI Support" on page 12.
- 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.

  VLANs and VPNs do, however, allow the use of network Quality of Service (QoS) features that, while not eliminating oversubscription, do provide a way to allocate bandwidth preferentially or proportionally to certain traffic. See also "Network I/O Control (NetIOC)" on page 39 for a different approach to this issue.
- Make sure that end-to-end Fibre Channel speeds are consistent to help avoid performance problems. For more information, see VMware KB article 1006602.
- Configure maximum queue depth if needed 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.

- 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.
- Local storage performance might be improved with write-back cache. If your local storage has write-back cache installed, make sure it's enabled and contains a functional battery module. For more information, see VMware KB article 1006602.
- Make sure storage adapter cards are installed in slots with enough bandwidth to support their expected throughput. Be careful to distinguish between similar-sounding—but potentially incompatible—bus architectures, including PCI, PCI-X, PCI Express (PCIe), and PCIe 3.0 (aka PCIe Gen3), and be sure to note the number of "lanes" for those architectures that can support more than one width.

  For example, in order to supply their full bandwidth potential, single-port 32Gb/s Fibre Channel HBA cards would need to be installed in at least PCIe Gen2 x8 or PCIe Gen3 x4 slots (either of which is capable of a net maximum of 32Gb/s in each direction) and dual-port 32Gb/s Fibre Channel HBA cards would need to be installed in at least PCIe Gen3 x8 slots (which are capable of a net maximum of 64Gb/s in each direction).

  These high-performance cards will typically function just as well in slower PCIe slots, but their maximum throughput could be limited by the slots' available bandwidth. This is most relevant for workloads that make heavy use of large block size I/Os, as this is where these cards tend to develop their highest throughput.

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 network infrastructure between the source and destination NICs doesn't introduce bottlenecks. For example, if both NICs are 10Gb/s, 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)
  - When using a virtualization encapsulation protocol, such as VXLAN or GENEVE, the NICs should support offload of that protocol's encapsulated packets.
  - Receive Side Scaling (RSS)
- Make sure network cards are installed in slots with enough bandwidth to support their maximum throughput. As described in "Hardware Storage Considerations" on page 13, be careful to distinguish between similar-sounding—but potentially incompatible—bus architectures.

  Ideally single-port 10Gb/s Ethernet network adapters should use PCIe x8 (or higher) or PCI-X 266 and dual-port 10Gb/s 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.

  Ideally 40Gb/s Ethernet network adapters should use PCI Gen3 x8/x16 slots (or higher).

  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.

  When using load balancing across multiple physical network adapters connected to one vSwitch, all the NICs should have the same line speed.

  If the physical network switch (or switches) to which your physical NICs are connected support Link Aggregation Control Protocol (LACP), configuring both the physical network switches and the vSwitch to use this feature can increase throughput and availability.

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, particularly when first configuring a new server.

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

General BIOS Settings

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

  NOTE After updating the BIOS you should revisit your BIOS settings in case new BIOS options become available or the settings of old options have changed.

- Make sure the BIOS is set to enable all populated processor sockets and to enable all cores in each socket.
- Enable "Turbo Boost" 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 (in other words, leaving NUMA enabled).
- Make sure any hardware-assisted virtualization features (VT-x, AMD-V, EPT, RVI, and so on) are enabled in the BIOS.

  NOTE After changes are made to these hardware-assisted virtualization features, some systems might need a complete power down before the changes take effect. See http://communities.vmware.com/docs/DOC-8978 for details.

- Disable from within the BIOS any devices you won't be using. This might include, for example, unneeded serial, USB, or network ports. See "ESXi General Considerations" on page 19 for further details.
- If the BIOS allows the memory scrubbing rate to be configured, we recommend leaving it at the manufacturer's default setting.
- If your hardware supports AES-NI (see "AES-NI Support" on page 12), and your deployment will be able to make use of the feature (see "vSphere Virtual Machine Encryption Recommendations" on page 36, "VMware vMotion Recommendations" on page 67, and "VMware Virtual SAN (vSAN)" on page 83):
  - Make sure your BIOS version supports AES-NI (in some cases a BIOS upgrade might be required for AES-NI support)
  - Make sure AES-NI is enabled in BIOS.

Power Management BIOS Settings

VMware ESXi includes a full range of host power management capabilities in the software that can save power when a host is not fully utilized (see "Host Power Management in ESXi" on page 24). We recommend that you configure your BIOS settings to allow ESXi the most flexibility in using (or not using) the power management features offered by your hardware, and that you then make your power-management choices within ESXi.

- In order to allow ESXi to control CPU power-saving features, set power management in the BIOS to "OS Controlled Mode" or equivalent. Even if you don't intend to use these power-saving features, ESXi provides a convenient way to manage them.
- C1E is a hardware-managed state; when ESXi puts the CPU into the C1 state, the CPU hardware can determine, based on its own criteria, to deepen the state to C1E. Availability of the C1E halt state typically provides a reduction in power consumption with little or no impact on performance.

- C-states deeper than C1/C1E (typically C3 and/or C6 on Intel and AMD) are managed by software and enable further power savings. In order to get the best performance per watt, you should enable all C-states in BIOS. This gives you the flexibility to use vSphere host power management to control their use.
- When "Turbo Boost" or "Turbo Core" is enabled, C1E and deep halt states (for example, C3 and C6 on Intel) can sometimes even increase the performance of certain lightly-threaded workloads (workloads that leave some hardware threads idle).

  However, for a very few multithreaded workloads that are highly sensitive to I/O latency, C-states can reduce performance. In these cases you might obtain better performance by disabling them in the BIOS.

  Because C1E and deep C-state implementation can be different for different processor vendors and generations, your results might vary.

ESXi and Virtual Machines 2

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

ESXi General Considerations

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

- Plan your deployment by allocating enough resources for all the virtual machines you will run, as well as those needed by ESXi itself.
- Allocate to each virtual machine only as much virtual hardware as that virtual machine requires. 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 any physical hardware devices that you will not be using. These might include 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 ESXi.
- 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.
- ESXi 6.5 introduces virtual hardware version 13. By creating virtual machines using this hardware version, or upgrading existing virtual machines to this version, a number of additional capabilities become available. This hardware version is not compatible with versions of ESXi prior to 6.5, however, and thus if a cluster of ESXi hosts will contain some hosts running pre-6.5 versions of ESXi, the virtual machines running on hardware version 13 will be constrained to run only on the ESXi 6.5 hosts. This could limit vMotion choices for Distributed Resource Scheduling (DRS) or Distributed Power Management (DPM).

ESXi CPU Considerations

This subsection provides guidance regarding CPU considerations in VMware ESXi.

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 ESXi allows significant levels of CPU overcommitment (that is, running more vCPUs on a host than the total number of physical processor cores in that host) without impacting virtual machine performance.

  If an ESXi 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).

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

  - If the load average on the first line of the `esxtop` CPU panel is equal to or greater than **1**, 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 `resxtop` 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 ESXi 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 49.)
  - Maintaining a consistent memory view among multiple vCPUs can consume additional resources, both in the guest operating system and in ESXi. (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 (including Windows 2000 (with certain HALs), Solaris 8 and 9, and MS-DOS), 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, and so on).

  ESXi automatically detects these loops and de-schedules the idle vCPU. Though this reduces the CPU overhead, it can also reduce the performance of some I/O-heavy workloads. 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.

- Some workloads can easily be split across multiple virtual machines. In some cases, for the same number of vCPUs, more smaller virtual machines (sometimes called "scaling out") will provide better performance than fewer larger virtual machines (sometimes called "scaling up"). In other cases the opposite is true, and fewer larger virtual machines will perform better. The variations can be due to a number of factors, including NUMA node sizes, CPU cache locality, and workload implementation details. The best choice can be determined through experimentation using your specific workload in your environment.

UP vs. SMP HALs/Kernels

There are two types of hardware abstraction layers (HALs) and kernels: UP and SMP. UP historically stood for "uniprocessor," but should now be read as "single-core." SMP historically stood for "symmetric multi-processor," but should now be read as multi-core.

- Although some recent operating systems (including Windows Vista, Windows Server 2008, and Windows 7) use the same HAL 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-core machines. If used on a multi-core machine, a UP operating system version will recognize and use only one of the cores. The SMP versions, while required in order to fully utilize multi-core machines, can also be used on single-core machines. Due to their extra synchronization code, however, SMP operating system versions used on single-core machines are slightly slower than UP operating system versions used on the same machines.

  NOTE When changing an existing virtual machine running Windows from multi-core to single-core the HAL usually remains SMP. For best performance, the HAL should be manually changed back to UP.

Hyper-Threading

- Hyper-threading technology (sometimes also called simultaneous 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, ESXi 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**.

- When ESXi is running on a system with hyper-threading enabled, it assigns adjacent CPU numbers to logical processors on the same core. Thus CPUs 0 and 1 are on the first core, CPUs 2 and 3 are on the second core, and so on.

  ESXi 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 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 a single SMP virtual machine, to both logical processors on one core (CPUs 0 and 1, for example) could cause poor performance.

Non-Uniform Memory Access (NUMA)

This section describes how to obtain the best performance when running ESXi on NUMA hardware.

NOTE A different feature, Virtual NUMA (vNUMA), allowing the creation of NUMA virtual machines, is described in "Virtual NUMA (vNUMA)" on page 50.

VMware vSphere supports AMD (Opteron, Barcelona, etc.), Intel (Nehalem, Westmere, etc.), 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, ESXi detects the system as NUMA and applies NUMA optimizations. If node interleaving is enabled, ESXi does not detect the system as NUMA. For more information, refer to your server's documentation.

More information about using NUMA systems with ESXi can be found in *vSphere Resource Management*.

Manual NUMA Configuration

The intelligent, adaptive NUMA scheduling and memory placement policies in ESXi 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 controls are available to override this default behavior, however, and advanced administrators might prefer to manually set NUMA placement (through the **numa.nodeAffinity** advanced option).

By default, ESXi 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 physical 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 each physical NUMA node (called "wide virtual machines").

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.

NOTE This potential increase in average memory access latencies can be mitigated by appropriately configuring Virtual NUMA (described in "Virtual NUMA (vNUMA)" on page 50), thus allowing the guest operating system to take on part of the memory-locality management task.

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 physical NUMA node.

Conversely, some memory bandwidth bottlenecked workloads can benefit from the increased aggregate memory bandwidth available when a virtual machine that would fit within one NUMA node is nevertheless split across multiple NUMA nodes. This split can be accomplished by limiting the number of vCPUs that can be placed per NUMA node by using the `maxPerMachineNode` option (do also consider the impact on vNUMA, however, by referring to "Virtual NUMA (vNUMA)" on page 50).

On hyper-threaded systems, virtual machines with a number of vCPUs greater than the number of cores in a NUMA node but lower than the number of logical processors in each physical NUMA node might benefit from using logical processors with local memory instead of full cores with remote memory. This behavior can be configured for a specific virtual machine with the `numa.vcpu.preferHT` flag.

Snoop Mode Selection

In order to maintain cache data coherency, memory-related operations require "snooping." Snooping is a mechanism by which a processor probes the cache contents of both local and remote processors to determine if a copy of requested data resides in any of those caches.

There are four snoop modes (though not all modes are available on all processors):

- Early Snoop (ES)
- Home Snoop (HS)
- Home snoop with Directory and Opportunistic Snoop Broadcast (HS with DIS + OSB)
- Cluster on Die (COD)

The newest, the Cluster-on-Die (COD) mode, allows boot-time configuration of NUMA nodes within a processor. This feature can logically partition the processor into either a single NUMA node (containing all the processors resources) or multiple NUMA nodes (each one consisting of a set of CPU cores, a portion of the last-level cache, and an integrated memory controller).

Because the best snoop mode for a particular environment depends on both the specific workloads in that environment and the underlying hardware, we recommend following your server manufacturer's guidance in making this selection.

For additional information about snoop modes see VMware KB article 2142499 and https://software.intel.com/en-us/articles/intel-xeon-processor-e5-2600-v4-product-family-technical-overview.

Configuring ESXi for Hardware-Assisted Virtualization

ESXi supports hardware-assisted CPU and MMU virtualization on Intel and AMD processors (as described in "Hardware-Assisted Virtualization" on page 11). Based on the available processor features and the guest operating system, ESXi chooses from three virtual machine monitor (VMM) modes:

- Software CPU and MMU virtualization
- Hardware CPU virtualization and software MMU virtualization
- Hardware CPU and MMU virtualization

In most cases the default behavior provides the best performance; overriding it will often reduce performance. If desired, however, the default 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 2MB large memory pages.

When running on a system with hardware-assisted MMU virtualization enabled, ESXi will attempt to use large memory pages to back the guest's memory pages even if the guest operating system and applications don't make use of large pages. For more information about large pages, see "2MB Large Memory Pages for Hypervisor and Guest Operating System" on page 30.

The VMM mode used for a specific virtual machine can be changed using the vSphere Web Client. To do so:

1 Select the virtual machine to be configured.

2 Click **Edit virtual machine settings**, choose the **Virtual Hardware** tab, and expand the **CPU** option.

3 Under **CPU/MMU Virtualization**, use the drop-down menu to select one of the following options:

- **Automatic** allows ESXi to determine the best choice. This is the default.
- **Software CPU and MMU** disables both hardware-assisted CPU virtualization (VT-x/AMD-V) and hardware-assisted MMU virtualization (EPT/RVI).
- **Hardware CPU, Software MMU** enables hardware-assisted CPU virtualization (VT-x/AMD-V) but disables hardware-assisted MMU virtualization (EPT/RVI).
- **Hardware CPU and MMU** 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 factors (for example, turning on Fault Tolerance, described in "VMware Fault Tolerance" on page 79) 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.

4 Click the **OK** button.

Host Power Management in ESXi

Host power management in ESXi is designed to reduce the power consumption of ESXi hosts while they are powered-on.

NOTE While Host Power Management applies to powered-on hosts, a very different power-saving technique, Distributed Power Management, attempts to power-off ESXi hosts when they are not needed. This is described in "VMware Distributed Power Management (DPM)" on page 74.

Host Power Management and Distributed Power Management can be used together for the best power conservation.

Power Policy Options in ESXi

ESXi offers the following power policy options:

- **High performance**
  This power policy maximizes performance, using no power management features.
- **Balanced**
  This power policy (the default in ESXi 6.5) is designed to reduce host power consumption while having little or no impact on performance.
- **Low power**
  This power policy is designed to more aggressively reduce host power consumption at the risk of reduced performance.

- **Custom**
  This power policy starts out the same as **Balanced**, but allows for the modification of individual parameters.

For details on selecting a power policy, search for "Select a CPU Power Management Policy" in the *vSphere Resource Management* guide.

For details on modifying individual power management parameters for the **Custom** policy, search for "Using CPU Power Management Policies" in the *vSphere Resource Management* guide.

Be sure, also, that your server's BIOS settings are configured correctly, as described in "Hardware BIOS Settings" on page 17.

Confirming Availability of Power Management Technologies

In some cases, the underlying hardware won't have one or more of the technologies ESXi can use to save power, or won't make those technologies available to ESXi. This will not cause problems, but it might result in the system using more power than necessary.

If desired, you can take the following steps to confirm that the hardware has these technologies and that they're available to ESXi:

1 Using the vSphere Web Client, select the host of interest.

2 Select the **Manage** tab for that host.

3 In the left pane within the **Manage** tab, under **Hardware** (not **System**), select **Power Management**.

4 The **Technology** field will show a list of the technologies currently available to ESXi on that host: ACPI P-states, ACPI C-states, or both. For the best power savings, you'd want both technologies to be available to ESXi.

Choosing a Power Policy

While the default power policy in ESX/ESXi 4.1 was **High performance**, in ESXi 5.0 and later the default is **Balanced**. This power policy will typically not impact the performance of CPU-intensive workloads. Rarely, however, the **Balanced** policy might slightly reduce the performance of latency-sensitive workloads. In these cases, selecting the **High performance** power policy will provide the full hardware performance. For more information on this, see "Running Network Latency Sensitive Applications" on page 43.

ESXi Memory Considerations

This subsection provides guidance regarding memory considerations in ESXi.

Memory Overhead

Virtualization causes an increase in the amount of physical memory required due to the extra memory needed by ESXi 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 VMkernel and various host agents (hostd, vpxa, etc.).

ESXi allows the use of a system swap file to reduce this memory overhead by up to 1GB when the host is under memory pressure. To use this feature, a system swap file must first be manually created. This can be accomplished by issuing the following command from the ESXi console:

```
esxcli sched swap system set –d true –n <datastore name>
```

The swap file is 1GB and is created at the root of the specified datastore.

NOTE This system swap file is different from the per-virtual-machine swap files, which store memory pages from the VMX component of the per-virtual-machine memory space overhead.

2 An additional memory space overhead for each virtual machine.

The per-virtual-machine memory space overhead can be further divided into the following categories:

- Memory reserved for the virtual machine executable (VMX) process.
  This is used for data structures needed to bootstrap and support the guest (i.e., thread stacks, text, and heap).
- Memory reserved for the virtual machine monitor (VMM).
  This is used for data structures required by the virtual hardware (i.e., TLB, memory mappings, and CPU state).
- Memory reserved for various virtual devices (i.e., mouse, keyboard, SVGA, USB, etc.)
- Memory reserved for other subsystems, such as the kernel, management agents, etc.

The amounts of memory reserved for these purposes depend on a variety of factors, including the number of vCPUs, the configured memory for the guest operating system, whether the guest operating system is 32-bit or 64-bit, the VMM execution mode, and which features are enabled for the virtual machine. For more information about these overheads, see *vSphere Resource Management*.

While the VMM and virtual device memory needs are fully reserved at the time the virtual machine is powered on, a feature called VMX swap can significantly reduce the per virtual machine VMX memory reservation, allowing more memory to be swapped out when host memory is overcommitted. This represents a significant reduction in the overhead memory reserved for each virtual machine.

The creation of a VMX swap file for each virtual machine (and thus the reduction in host memory reservation for that virtual machine) is automatic. By default, this file is 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) but a different location can be set with `sched.swap.vmxSwapDir`.

The amount of disk space required varies, but even for a large virtual machine is typically less than 100MB (typically less than 300MB if the virtual machine is configured for 3D graphics). VMX swap file creation can be disabled by setting `sched.swap.vmxSwapEnabled` to `FALSE`.

NOTE The VMX swap file is entirely unrelated to swap to host cache or regular host-level swapping, both of which are described in "Memory Overcommit Techniques" on page 27.

In addition, ESXi also provides optimizations, such as page sharing (see "Memory Overcommit Techniques" on page 27), 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 be sure to allocate enough memory to hold the working set of applications you will run in the virtual machine, thus minimizing thrashing. Because thrashing can dramatically impact performance, it is very important not to under-allocate memory.
- On the other hand, though the performance impact of over-allocating memory is far less than under-allocating it, you should nevertheless avoid substantially over-allocating as well.

  Memory allocated to a virtual machine beyond the amount needed to hold the working set will typically be used by the guest operating system for file system caches. If memory resources at the host level become low, ESXi can generally use memory ballooning (described in "Memory Overcommit Techniques" on page 27) to reclaim the portion of memory used for these caches.

  But over-allocating memory also unnecessarily increases the virtual machine memory overhead. While ESXi can typically reclaim the over-allocated memory, it can't reclaim the *overhead* associated with this over-allocated memory, thus consuming memory that could otherwise be used to support more virtual machines.

Memory Overcommit Techniques

- ESXi uses five memory management mechanisms—page sharing, ballooning, memory compression, swap to host cache, and regular swapping—to dynamically reduce the amount of machine physical memory required for each virtual machine.
  - **Page Sharing:** ESXi can use a proprietary technique to transparently share memory pages between virtual machines, thus eliminating redundant copies of memory pages. While pages are shared by default *within* virtual machines, as of vSphere 6.0 pages are no longer shared by default *between* virtual machines.

    In most environments, this change should have little effect. For details on the environments in which it might impact memory usage, and instructions on how to enable the previous default behavior if desired, see "Memory Page Sharing" on page 28.
  - **Ballooning:** If the host memory begins to get low and the virtual machine's memory usage approaches its memory target, ESXi will use ballooning to reduce that virtual machine's memory demands. Using a VMware-supplied vmmemctl module installed in the guest operating system as part of the VMware Tools suite, ESXi 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.
  - **Memory Compression**: If the virtual machine's memory usage approaches the level at which host-level swapping will be required, ESXi will use 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 memory pages has significantly less impact on performance than swapping out those pages.
  - **Swap to Host Cache:** If memory compression doesn't keep the virtual machine's memory usage low enough, ESXi will next forcibly reclaim memory using host-level swapping to a host cache (if one has been configured). Swap to host cache is a feature that allows users to configure a special swap cache on SSD storage. In most cases this host cache (being on SSD) will be much faster than the regular swap files (typically on hard disk storage), significantly reducing access latency. Thus, although some of the pages ESXi swaps out might be active, swap to host cache has a far lower performance impact than regular host-level swapping.

- **Regular Host-Level Swapping:** If the host cache becomes full, or if a host cache has not been configured, ESXi will next reclaim memory from the virtual machine by swapping out pages to a regular swap file. Like swap to host cache, some of the pages ESXi swaps out might be active. Unlike swap to host cache, however, this mechanism can cause virtual machine performance to degrade significantly due to its high access latency. (Note that this swapping is distinct from the swapping that can occur within the virtual machine under the control of the guest operating system.)

For further information about memory management, see *Understanding Memory Resource Management in VMware vSphere 5.0* (though this paper specifically addresses vSphere 5.0, most of the concepts are applicable to vSphere 6.5) and *Memory Overcommitment in the ESX Server*.

- While ESXi uses page sharing, ballooning, memory compression, and swap to host cache to allow significant memory overcommitment, usually with little or no impact on performance, you should avoid overcommitting memory to the point that regular host-level swapping is used to swap out active memory pages.

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

  NOTE The point at which memory overcommitment begins to affect a workload's performance depends heavily on the workload. The areas we describe in this section will be relevant for most workloads. For some of those workloads, however, performance will be noticeably impacted earlier than for others.

  a. In the vSphere Web Client, select the virtual machine in question, select **Monitor** > **Performance** > **Advanced**, in the drop down at the upper right select **Memory**, then look at the value of **Ballooned memory (Average)**.
  An absence of ballooning suggests that ESXi is not under heavy memory pressure and thus memory overcommitment is not affecting the performance of that virtual machine.

  NOTE This indicator is only meaningful if the balloon driver is installed in the virtual machine and is not prevented from working.

  NOTE Some ballooning is quite normal and not indicative of a problem.

  b. In the vSphere Web Client, select the virtual machine in question, select **Monitor** > **Performance** > **Advanced**, in the drop down at the upper right select **Memory**, then compare the values of **Consumed** memory and **Active** memory. If consumed is higher than active, this suggests that the guest is currently getting all the memory it requires for best performance.

  c. In the vSphere Web Client, select the virtual machine in question, select **Monitor** > **Utilization**, expand the **Guest Memory** pane, then look at the values of **Swapped** and **Compressed**. Swapping and compressing at the host level indicate more significant memory pressure.

  d. Check for guest operating system 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 operating system (or can be an indication that the guest memory size is simply too small).

Memory Page Sharing

Beginning with vSphere 6.0 (and included in patches or updates to some earlier releases), page sharing is enabled by default *within* virtual machines ("intra-VM sharing"), but is enabled *between* virtual machines ("inter-VM sharing") only when those virtual machines have the same "salt" value (described below). This change was made to ensure the highest security between virtual machines.

Because of the prevalence of 2MB large memory pages (see "2MB Large Memory Pages for Hypervisor and Guest Operating System" on page 30), which are not shared even when page sharing is enabled, this change will likely have only a small effect, if any, on memory usage in most deployments.

In environments that make extensive use of small memory pages, however (such as many VDI deployments), sharing pages between virtual machines might provide a considerable reduction in memory usage.

By default, the salt value for a specific virtual machine is that virtual machine's `vc.uuid`, which is always unique among virtual machines on a single ESXi host, thus preventing inter-VM page sharing.

You can enable inter-VM page sharing using one of these methods:

- To enable inter-VM page sharing between all virtual machines on an ESXi host, set the host configuration option `Mem.ShareForceSalting` to 0. This duplicates the default behavior of previous versions of vSphere.
- Manually specify a salt value in virtual machines' `.vmx` files with the configuration option `sched.mem.pshare.salt`. Using this option, you can create pools of virtual machines that have the same salt value, and can thus share memory pages with other virtual machines in that pool.

  If desired, you can configure the same salt value for every virtual machine in your environment, effectively creating a single page-sharing pool (thus duplicating the default behavior of previous versions of vSphere).

For more information on page sharing and the relevant security issues, see VMware KB articles 2080735 and 2097593.

Memory Swapping Optimizations

As described in "Memory Overcommit Techniques," above, ESXi supports a bounded amount of memory overcommitment without host-level swapping. If the overcommitment is so large that the other memory reclamation techniques are not sufficient, however, ESXi uses host-level memory swapping, with a potentially significant impact on virtual machine performance.

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 ESXi uses page sharing, ballooning, and memory compression to reduce the need for host-level memory swapping, don't disable these techniques.
- If you choose to overcommit memory with ESXi, be sure you have sufficient swap space on your ESXi system. At the time a virtual machine is first powered on, ESXi 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 (plus the space required for VMX swap, as described in "Memory Overhead" on page 26).
- You can optionally configure a special host cache on an SSD (if one is installed) to be used for the swap to host cache feature. This swap cache will be shared by all the virtual machines running on the host and host-level swapping of their most active pages will benefit from the low latency of SSD. This allows a relatively small amount of SSD storage to potentially significantly increase performance

  NOTE Using swap to host cache and putting the regular swap file in SSD (as described below) are two different approaches for improving host swapping performance. Swap to host cache makes the best use of potentially limited SSD space while also being optimized for the large block sizes at which some SSDs work best.

- Even if an overcommitted host uses swap to host cache, it still needs to create regular swap files. Swap to host cache, however, makes much less important the speed of the storage on which the regular swap files are placed.

  If a host does *not* use swap to host cache, and memory is overcommitted to the point of thrashing, placing the virtual machine swap files on low latency, high bandwidth storage systems will result in the smallest performance impact from swapping.

  - The best choice will usually be local SSD.

    NOTE Placing the regular swap file in SSD and using swap to host cache in SSD (as described above) are two different approaches to improving host swapping performance. Because it is unusual to have enough SSD space for a host's entire swap file needs, we recommend using local SSD for swap to host cache.

- If the host doesn't have local SSD, the second choice would be remote SSD. This would still provide the low-latencies of SSD, though with the added latency of remote access.
- If you can't use SSD storage, place the regular swap file on the fastest available storage. This might be a Fibre Channel SAN array or a fast local disk.
- Placing swap files on local storage (whether SSD or hard drive) could potentially reduce vMotion performance. This is because if a virtual machine has memory pages in a local swap file, they must be swapped in to memory before a vMotion operation on that virtual machine can proceed.
- Regardless of the storage type or location used for the regular swap file, for the best performance, and to avoid the possibility of running out of space, swap files should not be placed on thin-provisioned storage.

The regular swap file location for a specific virtual machine can be set in the vSphere Web Client (right-click the virtual machine, select **Edit Settings**, choose the **VM Options** tab, expand **Advanced**, and select a **Swap file location**). 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.

- Host-level memory swapping can, in many cases, be reduced for a specific virtual machine by reserving memory for that virtual machine at least equal in size to the machine's active working set. Host-level memory swapping can be *eliminated* for a specific virtual machine by reserving memory equal in size to the virtual machine's entire memory.

NOTE When applied to a running virtual machine, the effect of these memory reservations might appear only gradually. To immediately see the full effect you would need to power-cycle the virtual machine.

Be aware, however, that configuring resource reservations will reduce the number of virtual machines that can be run on a system. This is because ESXi will keep available enough host memory to fulfill all reservations (both for virtual machines and for overhead) 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 ESXi reserves for the virtual machine. It can be configured through the vSphere Web Client in the settings window for each virtual machine (right-click the virtual machine, select **Edit Settings**, choose the **Virtual Hardware** tab, expand **Memory**, then set the desired reservation).

2MB Large Memory Pages for Hypervisor and Guest Operating System

In addition to the usual 4KB memory pages, ESXi also provides 2MB memory pages (commonly referred to as "large pages"). (Note that although some CPUs support 1GB memory pages, in this book the term "large pages" is used only to refer to 2MB pages.)

ESXi assigns these 2MB machine memory pages to guest operating systems whenever possible; on systems with hardware-assisted MMU virtualization, ESXi does this even if the guest operating system doesn't request them (though the full benefit of large pages comes only when the guest operating system and applications use them as well). The use of large pages can significantly reduce TLB misses, improving the performance of most workloads, especially those with large active memory working sets. In addition, large pages can slightly reduce the per-virtual-machine memory space overhead.

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. While ESXi ordinarily uses page sharing regardless of memory demands (though see "Memory Page Sharing" on page 28 to read how this behavior changed in vSphere 6.0), ESXi 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.

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 12; for information about configuring the way ESXi uses hardware-assisted MMU virtualization, see "Configuring ESXi for Hardware-Assisted Virtualization" on page 23.

ESXi Storage Considerations

This subsection provides guidance regarding storage considerations in ESXi.

vSphere Flash Read Cache (vFRC)

vSphere Flash Read Cache (vFRC) allows flash storage resources on the ESXi host (the vSphere Flash Infrastructure layer) to be used for read caching of virtual machine I/O requests. This can dramatically increase the performance of workloads that exhibit locality of access. This section describes how to obtain the best performance using vFRC.

- Select appropriate flash hardware for the vSphere Flash Infrastructure layer (for details see "Hardware Storage Considerations" on page 13).
- Create a vSphere Flash Infrastructure layer on the ESXi host. (This is not created by default.)
- Enable vFRC by selecting that option when you create a virtual disk, or enabling at a later time. (vFRC is not enabled by default.)
- Configure the cache size appropriately. Ideally the cache would be large enough to hold the active working set, but not much larger. Configuring a larger-than-necessary cache can reduce performance by making that flash space unavailable for other vFRC-enabled virtual machines or for host swap cache. It can also slow vMotion of the virtual machine.

  NOTE The cache file can also impact vMotion performance; see "VMware vMotion Recommendations" on page 67 for more information.

- Configure the cache block size appropriately. This should usually match the I/O size of the workload.

For further details about vFRC, including guidance on how to determine the best cache and block sizes, see *Performance of vSphere Flash Read Cache in VMware vSphere 5.5* (though specifically addressing vFRC in vSphere 5.5, this paper is also relevant to vFRC in vSphere 6.5).

VMware vStorage APIs for Array Integration (VAAI)

- For the best storage performance, consider using VAAI-capable storage hardware. The performance gains from VAAI (described in "Hardware Storage Considerations" on page 13) can be especially noticeable in VDI environments (where VAAI can improve boot-storm and desktop workload performance), large data centers (where VAAI can improve the performance of mass virtual machine provisioning and of thin-provisioned virtual disks), and in other large-scale deployments.

  The behavior of ESXi with VAAI-capable storage is slightly different for SAN versus NAS:

  - If your SAN storage hardware supports VAAI, ESXi will automatically recognize and use these capabilities.
  - If your NAS storage hardware supports VAAI, ESXi will automatically recognize and use these capabilities only if the vendor-specific plugin is present.

  To confirm that your hardware 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 storage hardware vendor to see if a firmware upgrade is required for VAAI support.

LUN Access Methods

- ESXi 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 might be desirable in some cases.

  You can use RDMs in virtual compatibility mode or physical compatibility mode:

- Virtual mode specifies full virtualization of the mapped device, allowing the guest operating system to treat the RDM like any other virtual disk file in a VMFS volume and allowing the use of VMware redo logs to take snapshots of the RDM.
- Physical mode specifies minimal SCSI virtualization of the mapped device, allowing the greatest flexibility for SAN management software or other SCSI target-based software running in the virtual machine.

For more information about RDM, see *vSphere Storage*.

Virtual Disk Modes

- ESXi supports three virtual disk modes: Independent persistent, Independent nonpersistent, and Dependent.

NOTE An independent disk does not participate in virtual machine snapshots. That is, the disk state will be independent of the snapshot state; creating, consolidating, or reverting to snapshots will have no effect on the disk.

These modes have the following characteristics:

- **Independent persistent** – In this mode changes are persistently 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, ESXi 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.
- **Dependent** – In this mode, if a snapshot has been taken, disk writes are appended to a redo log that persists between power cycles. Thus, like the independent nonpersistent mode disks described above, dependent mode disk performance might not be as high as independent persistent mode disks. If a snapshot has *not* been taken, however, dependent disks are just as fast as independent disks.

Virtual Disk Types

ESXi supports multiple virtual disk types:

- **Thick provisioned** – Thick virtual disks, which have all their space allocated at creation time, are further divided into two types: eager-zeroed and lazy-zeroed.

NOTE With VMware Virtual Volumes (VVols; described in "VMware Virtual Volumes (VVols)" on page 85) the array itself automatically does zeroing as necessary; there's thus no need (and no way) to specify "eager" versus "lazy" zeroing for VVols.

- **Eager zeroed** – An eager-zeroed thick disk has all space allocated and zeroed out at the time of creation. This increases 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 SAN storage (described in "Hardware Storage Considerations" on page 13) can speed up eager-zeroed thick disk creation by offloading zeroing operations to the storage array.

NOTE The performance advantages of eager-zeroed thick disks are true only for independent persistent virtual disks or for dependent virtual disks that have no snapshots. Independent nonpersistent virtual disks, or dependent virtual disks that have snapshots, will perform like thin provisioned disks.

- **Lazy zeroed** – A lazy-zeroed thick disk has all space allocated at the time of creation, but each block is zeroed only 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 SAN or NAS storage can improve lazy-zeroed thick disk first-time-write performance by offloading zeroing operations to the storage array.

- **Thin provisioned** – 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 (similar to that of lazy-zeroed thick disks) during the first write to an unwritten file block, but on subsequent writes thin-provisioned disks have the same performance as eager-zeroed thick disks.

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

All three types of virtual disks can be created using the vSphere Web Client (right-click the virtual machine, select **Edit Settings**, choose the **Virtual Hardware** tab, in **New device** (at the bottom) select **New Hard Disk**, click **Add**, expand the **New Hard disk** line, then select the disk type).

Virtual disks can also be created from the vSphere Command-Line Interface (vSphere CLI) using `vmkfstools`. For details refer to *vSphere Command-Line Interface Reference* and the `vmkfstools` man page.

NOTE Some of these virtual disk types might not be available when creating virtual disks on NFS volumes, depending on the hardware vendor and whether or not the hardware supports VAAI.

Partition Alignment

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

- Like other disk-based filesystems, VMFS filesystems suffer a performance penalty when the partition is unaligned. Using the vSphere Web Client to create VMFS partitions avoids this problem because, beginning with ESXi 5.0, it automatically aligns VMFS3, VMFS5, or VMFS6 partitions along the 1MB boundary.

  NOTE If a VMFS3 partition was created using an earlier version of ESX/ESXi that aligned along the 64KB boundary, and that filesystem is then upgraded to VMFS5, it will retain its 64KB alignment. 1MB alignment can be obtained by deleting the partition and recreating it using the vSphere Web Client with an ESXi 5.0 or later host.

  Existing VMFS3 or VMFS5 volumes can't be directly converted to VMFS6. Instead, a VMFS6 volume should be created from scratch and virtual machines then migrated to the new volume using Storage vMotion.

- To manually align your VMFS partitions, check your storage vendor's recommendations for the partition starting block address. If your storage vendor makes no specific recommendation, use a starting block address 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

SAN path policies can have a significant effect on storage performance. In general, the path policy ESXi uses by default for a particular array will be the best choice. If selecting a non-default path policy, we recommend choosing from among those policies tested and supported on that array by its vendor.

This section provides a brief overview of the subject of SAN path policies.

- For most Active/Passive storage arrays ESXi uses the **Most Recently Used** (MRU) path policy by default. We don't recommend using **Fixed** path policy for Active/Passive storage arrays as this can result in frequent and rapid path switching (often called "path-thrashing"), which can cause slow LUN access. For optimal performance with the arrays for which ESXi defaults to MRU you might also consider the Round Robin path policy (described below).

  NOTE With some Active/Passive storage arrays that support ALUA (described below) ESXi can use Fixed path policy. We recommend, however, that you only use it on arrays where it is specifically supported by the SAN vendor. In most cases Round Robin is a better and safer choice for Active/Passive arrays.

- For most Active/Active storage arrays ESXi uses the **Fixed** path policy by default. 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 optimal performance with these arrays you might also consider the Round Robin path policy (described below).
- ESXi can 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.

  NOTE Not all storage arrays support the Round Robin path policy. Switching to an unsupported or undesirable path policy can result in connectivity issues to the LUNs or even a storage outage. Check your array documentation or with your storage vendor to see if Round Robin is supported and/or recommended for your array and configuration.

- ESXi also supports third-party path selection plugins (PSPs). In some cases, these might provide the best performance for a specific array by exploiting specific features of the array or knowledge of its design.
- 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 ESXi, allows the array itself to designate paths as "Active Optimized." When ALUA is combined with the Round Robin path policy, ESXi by default cycles I/O requests through these Active Optimized paths.

For more information, see the *VMware SAN Configuration Guide* and VMware KB article 1011340.

Storage I/O Resource Allocation

VMware vSphere 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 on the storage resources available to an individual host (as described in this section) or on an entire datastore's resources (as described in "VMware vSphere Storage I/O Control" on page 76).

The storage I/O resources available to an ESXi host can be allocated to that host's virtual disks by setting disk Shares or Limits.

- The proportion of a datastore's I/O resources available to each virtual disk can be controlled with **Shares**. Each virtual disk will receive a proportion of the datastore bandwidth equal to that virtual disk's share divided by the sum of the shares of all the virtual disks on that host.
- The maximum storage I/O resources available to each virtual disk can be set using **Limits**. These limits, set in I/O operations per second (IOPS), can be used to provide strict isolation and control of certain workloads. By default, these are set to **unlimited**.

To set Shares or Limits, from the vSphere Web Client right-click a virtual machine, select **Edit Settings**, choose the **Virtual Hardware** tab, expand **Hard disk 1** (or whichever hard disk you want to configure), then set the **Shares** or **Limit - IOPs** to the desired value.

iSCSI and NFS Recommendations

- For iSCSI and NFS it's sometimes beneficial to create a VLAN, if the network infrastructure supports it, just for the ESXi host's vmknic and the iSCSI/NFS server. This minimizes network interference from other packet sources.

NVMe Recommendations

Flash storage on PCIe cards initially used SCSI or SATA interfaces and protocols, often limiting their performance to much less than what the underlying SSD devices were capable of. Some newer PCIe flash devices use the Non-Volatile Memory Express (NVMe) protocol, potentially allowing much higher performance.

NOTE vSphere also supports virtual NVMe (vNVMe), which is described in "Guest Operating System Storage Considerations" on page 52.

NVMe performance considerations:

- NVMe devices are highly parallelized; their maximum throughput is generally achieved when they are backing multiple virtual disks and/or multiple virtual machines.
- The high throughput of NVMe creates a correspondingly high CPU load. NVMe devices are thus a better fit for hardware configurations with many cores per socket (preferably at least eight) and multiple sockets (preferably at least two). High CPU frequencies are also desirable.
- The virtual storage adapters used by the guest virtual machines should be appropriately chosen. These will typically be either PVSCSI or the new vNVMe virtual adapter. (See "Guest Operating System Storage Considerations" on page 52 for details.)

vSphere Virtual Machine Encryption Recommendations

Virtual machine encryption, new in vSphere 6.5, can encrypt disk files (`.vmdk`), snapshot files (`.vmsn`), NVRAM files (`.nvram`), swap files (`.vswp`), and portions of configuration files (`.vmx`). Encryption and decryption is performed by the CPU and therefore incurs a CPU cost. When possible, vSphere uses the AES-NI processor instruction set extensions (supported by many recent processors) to reduce that CPU cost.

- The resource overhead of virtual machine encryption is primarily in CPU utilization. Thus, with sufficient CPU resources, typical deployments (that is, those that don't use ultra-fast storage) will see no significant performance impact.
- To significantly minimize additional CPU utilization due to VM encryption, choose processors that support AES-NI (see "AES-NI Support" on page 12), especially some recent processor versions in which the AES-NI implementation is further optimized.
- Make sure AES-NI is enabled in BIOS. In some cases a BIOS upgrade might be required for AES-NI support.
- Because the encryption takes place on the host, storage devices see only encrypted data. Special features of some storage devices, such as deduplication or compression, might therefore be ineffective or not provide their full benefit. (For an encryption option that takes place at the storage level, thus allowing deduplication and compression, see "VMware Virtual SAN (vSAN)" on page 83.)
- When virtual machine encryption is used with storage having very low latencies (below about 200 microseconds), the time taken by the CPU to process encryption requests can begin to impact performance.

Further detail on this topics can be found in the *VMware vSphere Virtual Machine Encryption Performance* white paper.

General ESXi Storage Recommendations

- The number of LUNs in a storage array, and the way virtual machines are distributed across those LUNs, can affect performance:
  - Provisioning more LUNs, with fewer virtual machines on each one, can enable the ESXi servers to simultaneously present more I/O requests to the array. This has the potential to improve performance by ensuring full utilization of all array resources and giving the array more opportunities to optimize the I/Os.
  - On the other hand provisioning too many LUNs, especially when many ESXi servers are connected to a single array, can allow the ESXi hosts to simultaneously send so many I/O requests that they fill the array queue and the array returns QFULL/BUSY errors. This can reduce performance due to the need to retry the rejected I/O requests.
- 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 operating system.
- Make sure that the average latency for storage devices is not too high. This latency can be seen in `esxtop` (or `resxtop`) by looking at the **GAVG/cmd** metric. A reasonable upper value for this metric depends on your storage subsystem. If you use Storage I/O Control ("VMware vSphere Storage I/O Control" on page 76), you can use your Storage I/O Control setting as a guide — your **GAVG/cmd** value should be well below your Storage I/O Control setting. The default Storage I/O Control setting is 30 ms, but if you have very fast storage (SSDs, for example) you might have reduced that value. For further information on average latency see VMware KB article 1008205.
- You can adjust the maximum number of outstanding disk requests per VMFS volume, which can help equalize the bandwidth across virtual machines using that volume. For further information see VMware KB article 1268.
- If you often observe QFULL/BUSY errors, enabling and configuring queue depth throttling might improve storage performance. This feature can significantly reduce the number of commands returned from the array with a QFULL/BUSY error. If any system accessing a particular LUN or storage array port has queue depth throttling enabled, all systems (both ESXi hosts and other systems) accessing that LUN or storage array port should use an adaptive queue depth algorithm. For more information about both QFULL/BUSY errors and this feature see VMware KB article 1008113.

Running Storage Latency Sensitive Applications

By default the ESXi storage stack is configured to drive high storage throughput at low CPU cost. While this default configuration provides better scalability and higher consolidation ratios, it comes at the cost of potentially higher storage latency. Applications that are highly sensitive to storage latency might therefore benefit from the following:

- Adjust the host power management settings:

  Some of the power management features in newer server hardware can increase storage latency. Disable them as follows:
  - Set the ESXi host power policy to **Maximum performance** (as described in "Host Power Management in ESXi" on page 24; this is the preferred method) or disable power management in the BIOS (as described in "Power Management BIOS Settings" on page 17).
  - Disable C1E and other C-states in BIOS (as described in "Power Management BIOS Settings" on page 17).
  - Enable Turbo Boost in BIOS (as described in "General BIOS Settings" on page 17).
- If you are using an LSILogic vHBA or a PVSCSI vHBA (with hardware version 11) in the virtual machine, you can adjust the reqCallThreshold value.

  The lower the reqCallThreshold value, the less time the I/O requests are likely to stay in the vHBA's queue. For instance, if reqCallThreshold is set to 1, it means when there is even one I/O request in the vHBA's queue, the I/Os will be dispatched to the lower layer. (The default reqCallThreshold value is 8.)

There are two ways to adjust the reqCallThreshold value:

- Change the value for *all* vHBAs in the system by running the command:
  `esxcfg-advcfg -s x /Disk/ReqCallThreshold`
  (where *x* is the desired reqCallThreshold value).
- Override the system-wide configuration for a *specific* vHBA by adding:
  `scsiY.reqCallThreshold = X`
  to the `.vmx` file (where *Y* is the target SCSI number and *X* is the desired reqCallThreshold value).

For further information on running storage latency sensitive applications, the paper *Best Practices for Performance Tuning of Latency-Sensitive Workloads in vSphere VMs*, while addressing primarily network latency-sensitive applications, might prove helpful.

ESXi Networking Considerations

This subsection provides guidance regarding networking considerations in ESXi.

General ESXi 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.
- If a physical NIC is shared by multiple consumers (that is, virtual machines and/or the vmkernel), each such consumer could impact the performance of others. Thus, for the best network performance, use separate physical NICs for consumers with heavy networking I/O (because these could reduce the network performance of other consumers) and for consumers with latency-sensitive workloads (because these could be more significantly affected by other consumers).

  Because they have multi-queue support, 10Gb/s and faster NICs can separate these consumers onto different queues. Thus as long as these faster NICs have sufficient queues available, this recommendation typically doesn't apply to them.

  Additionally, NetIOC (described below) can be used to reserve bandwidth for specific classes of traffic, providing resource isolation between different flows. While this can be used with NICs of any speed, it's especially useful for 10Gb/s and faster NICs, as these are more often shared by multiple consumers.
- To establish a network connection between two virtual machines that reside on the same ESXi 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)

Network I/O Control (NetIOC) allows the allocation of network bandwidth to network resource pools. You can either select from among nine predefined resource pools (management traffic, Fault Tolerance traffic, iSCSI traffic, NFS traffic, Virtual SAN traffic, vMotion traffic, vSphere Replication (VR) traffic, vSphere Data Protection Backup traffic, and virtual machine traffic) or you can create user-defined resource pools. Each resource pool is associated with a portgroup.

NOTE vSphere 6.0 introduced NetIOC version 3 (NetIOCv3), which allocates bandwidth at the level of the entire distributed switch, rather than at the physical adapter level, as in NetIOC version 2 (NetIOCv2). This book describes only NetIOCv3.

When creating a new vSphere Distributed Switch in vSphere 6.5, that switch will use NetIOCv3 by default. However switches upgraded from a previous ESXi version will, in some cases, not be automatically upgraded to NetIOCv3. For more information about this, see *vSphere Networking* for version 6.5.

NetIOC can guarantee bandwidth for specific needs and can prevent any one resource pool from impacting the others.

As of vSphere 6.0, when DRS ("VMware Distributed Resource Scheduler (DRS)" on page 70) load-balances virtual machines across hosts, it will include NetIOC bandwidth reservations in its calculations.

For further information about NetIOC, see *Performance Evaluation of Network I/O Control in VMware vSphere 6.0* and *vSphere Networking* for version 6.5.

Network I/O Control Configuration

With NetIOC, network bandwidth can be allocated to resource pools as well as to individual virtual machines using shares, reservations, or limits:

- Shares can be used to allocate to a resource pool or virtual machine a proportion of a network link's bandwidth equivalent to the ratio of its shares to the total shares. If a resource pool or virtual machine doesn't use its full allocation, the unused bandwidth is available to other bandwidth consumers using the same physical NIC.
- Bandwidth reservation can be used to guarantee a minimum bandwidth (in Mb/s) to a resource pool or virtual machine. The total bandwidth reservations are limited to 75% of the capacity of the underlying physical adapter. If a resource pool or virtual machine doesn't use its full reservation, the unused bandwidth is available to other bandwidth consumers using the same physical NIC.
- Limits can be used to set a resource pool or virtual machine's maximum bandwidth utilization (in Mb/s or Gb/s) per physical NIC through a specific virtual distributed switch (vDS). These limits are enforced even if a vDS is not saturated, potentially limiting a consumer's bandwidth while simultaneously leaving some bandwidth unused. On the other hand, if a consumer's bandwidth utilization is less than its limit, the unused bandwidth *is* available to other bandwidth consumers using the same physical NIC.

Network I/O Control Advanced Performance Options

- A new option in vSphere 6.5, HClock Multiqueue, can improve performance in some environments with small packets and high packet rate. This option, which is disabled by default, allows multiple vNICs or multiple vmknics to distribute traffic across multiple hardware transmit queues on a single physical NIC.

  You can enable and configure this option for an ESXi host as follows:

  a Enable HClock Multiqueue by running the following command:
  `esxcli system settings advanced set –i x –o /Net/NetSchedHClkMQ`
  (where a value for *x* of `1` enables the feature and `0` disables it).

  b Configure the maximum number of queues HClock will use by running the following command:
  `esxcli system settings advanced set –i n –o /Net/NetSchedHClkMaxHwQueue`
  (where *n* is the maximum number of queues; the default is 2).

  c Take down then bring up each physical NIC on the ESXi host:
  `esxcli network nic down –n vmnicX`
  `esxcli network nic up –n vmnicX`
  (where *X* identifies which vmnic the command is applied to).

DirectPath I/O

vSphere DirectPath I/O leverages Intel VT-d and AMD-Vi hardware support (described in "Hardware-Assisted I/O MMU Virtualization (VT-d and AMD-Vi)" on page 12) to allow guest operating systems to directly access hardware devices. In the case of networking, DirectPath I/O allows the virtual machine to access a physical NIC directly rather than using an emulated device (such as the E1000) or a paravirtualized device (such as VMXNET or VMXNET3). While DirectPath I/O provides limited increases in throughput, it reduces CPU cost for networking-intensive workloads.

DirectPath I/O is not compatible with certain core virtualization features, however. This list varies with the hardware on which ESXi is running:

- When ESXi is running on certain configurations of the Cisco Unified Computing System (UCS) platform, DirectPath I/O for networking is compatible with vMotion, physical NIC sharing, snapshots, and suspend/resume. It is not compatible with Fault Tolerance, NetIOC, memory overcommit, VMCI, VMSafe, or NSX network virtualization.
- For server hardware other than the Cisco UCS platform, DirectPath I/O is not compatible with vMotion, physical NIC sharing, snapshots, suspend/resume, Fault Tolerance, NetIOC, memory overcommit, VMSafe, or NSX network virtualization.

Typical virtual machines and their workloads don't require the use of DirectPath I/O. For workloads that are very networking intensive and don't need the core virtualization features mentioned above, however, DirectPath I/O might be useful to reduce CPU usage.

Single Root I/O Virtualization (SR-IOV)

vSphere supports Single Root I/O virtualization (SR-IOV), a new mode of direct guest access to hardware devices. In addition to Intel VT-d or AMD-Vi hardware support (described in "Hardware-Assisted I/O MMU Virtualization (VT-d and AMD-Vi)" on page 12), SR-IOV also requires BIOS, physical NIC, and network driver support. For more information on supported configurations, see the 6.5 version of *vSphere Networking*.

SR-IOV offers performance benefits and trade-offs similar to those of DirectPath I/O. SR-IOV is beneficial in workloads with very high packet rates or very low latency requirements. Like DirectPath I/O, SR-IOV is not compatible with certain core virtualization features, such as vMotion (for details, see the compatibility list for hardware other than the Cisco UCS platform in "DirectPath I/O" on page 40). SR-IOV does, however, allow for a single physical device to be shared amongst multiple guests.

SplitRx Mode

SplitRx mode uses multiple physical CPUs to process network packets received in a single network queue. This feature can significantly improve network performance for certain workloads. These workloads include:

- Multiple virtual machines on one ESXi host all receiving multicast traffic from the same source. (SplitRx mode will typically improve throughput and CPU efficiency for these workloads.)
- Traffic via the vNetwork Appliance (DVFilter) API between two virtual machines on the same ESXi host. (SplitRx mode will typically improve throughput and maximum packet rates for these workloads.)

In vSphere 5.1 and later this feature is automatically enabled for a VMXNET3 virtual network adapter (the only adapter type on which it is supported) when ESXi detects that a single network queue on a physical NIC is both (a) heavily utilized and (b) getting more than 10,000 broadcast/multicast packets per second.

NOTE SplitRx mode will be automatically enabled as described above *only* if the network traffic is coming in over a physical NIC, not when the traffic is entirely internal to the ESXi host. In cases where the traffic is entirely internal, however, SplitRx mode can be manually enabled if desired, as described in "Enabling or Disabling SplitRx Mode for an Individual Virtual NIC" on page 41.

Disabling SplitRx Mode for an Entire ESXi Host

This automatic behavior can be disabled for an entire ESXi host by using the `NetSplitRxMode` variable.

The possible values for this variable are:

- `NetSplitRxMode = "0"`
  This value disables splitRx mode for the ESXi host.
- `NetSplitRxMode = "1"`
  This value (the default) enables splitRx mode for the ESXi host.

To change this variable through the vSphere Web Client:

1. Select the ESXi host you wish to change.
2. Under the **Configure** tab, expand **System**, then click on **Advanced System Settings**.
3. Click **Edit** (in the upper right corner).
4. Find **NetSplitRxMode** (it's under **Net.NetSplitRxMode**).
5. Click on the value to be changed and configure it as you wish.
6. Click **OK** to close the **Edit Advanced System Settings** window.

The change will take effect immediately and does not require the ESXi host to be restarted.

Enabling or Disabling SplitRx Mode for an Individual Virtual NIC

The SplitRx mode feature can also be individually configured for each virtual NIC (overriding the global NetSplitRxMode setting) using the `ethernet`*X*`.emuRxMode` variable in each virtual machine's `.vmx` file (where *X* is replaced with the network adapter's ID).

The possible values for this variable are:

- `ethernet`*X*`.emuRxMode = "0"`
  This value disables splitRx mode for `ethernet`*X*.
- `ethernet`*X*`.emuRxMode = "1"`
  This value enables splitRx mode for `ethernet`*X*.

To change this variable through the vSphere Web Client:

1. Select the virtual machine you wish to change.
2. Under the **Configure** tab, expand **Settings**, then select **VM Options**
3. Click **Edit** (in the upper right corner).
4. Expand **Advanced**, then in **Configuration Parameters** click **Edit Configuration**.
5. Look for **ethernet*X*.emuRxMode** (where ***X*** is the number of the desired NIC) and configure it as you wish. If the variable isn't present, enter it as a new variable and click **Add**.
6. Click **OK** to close the **Configuration Parameters** window.
7. Click **OK** to close the **Edit Settings** window.

The change will not take effect until the virtual machine has been restarted or the relevant vNIC is disconnected from the virtual machine and connected again.

Receive Side Scaling (RSS)

Receive Side Scaling (RSS) is a feature that allows network packets from a single NIC to be scheduled in parallel on multiple CPUs by creating multiple hardware queues. While this might increase network throughput for a NIC that receives packets at a high rate, it can also increase CPU overhead.

When using certain 10Gb/s or 40Gb/s Ethernet physical NICs, ESXi allows the RSS capabilities of the physical NICs to be used by the virtual NICs. This can be especially useful in environments where a single MAC address gets large amounts of network traffic (for example VXLAN or network-intensive virtual appliances). Because of the potential for increased CPU overhead, this feature is disabled by default.

You can enable this feature as follows:

- For Intel 82599EB SFI/SFP+ 10Gb/s NICs:

  Load the driver by running `vmkload_mod ixgbe RSS="4"`

  To enable the feature on multiple Intel 82599EB SFI/SFP+ 10Gb/s NICs, include another comma-separated `4` for each additional NIC (for example, to enable the feature on three such NICs, you'd run `vmkload_mod ixgbe RSS="4,4,4"`).

- For Mellanox Technologies MT27500 Family ConnectX-3 10Gb/s or 40Gb/s NICs:

  Load the driver by running `vmkload_mod nmlx4_en num_rings_per_rss_queue=4`

  NOTE After loading the driver with `vmkload_mod`, you should make `vmkdevmgr` rediscover the NICs with the following command:
  `'kill –HUP "<pid of vmkdevmgr>"'`
  (where <*pid of vmkdevmgr*> is the process ID of the `vmkdevmgr` process).

  To enable the feature on multiple MT27500 Family ConnectX-3 10Gb/s or 40Gb/s NICs, include an additional comma-separated `4` for each additional NIC (for example, to enable the feature on three such NICs, you'd run `vmkload_mod nmlx4_en num_rings_per_rss_queue=4,4,4`.

- In the `.vmx` file for each virtual machine that will use this feature add `ethernet`*X*`.pNicFeatures = "4"` (where *X* is the number of the virtual network card to which the feature should be added).

Virtual Network Interrupt Coalescing

Virtual network interrupt coalescing can reduce the number of interrupts, thus potentially decreasing CPU utilization. Though this could increase network latency, many workloads are not impacted by additional network latency of anywhere from a few hundred microseconds to a few milliseconds, and the reduction in virtual networking overhead can potentially allow more virtual machines to be run on a single ESXi host.

By default, this feature is enabled for all virtual NICs in ESXi. For VMXNET3 virtual NICs, however, this feature can be set to one of three schemes or disabled by changing the `ethernetX.coalescingScheme` variable (where *X* is the number of the virtual NIC to configure).

- The feature will be enabled (the default) if the `ethernetX.coalescingScheme` variable is not present in the `.vmx` file or if the variable is present and set to **`rbc`**.

  In this case, the `ethernetX.coalescingParams` variable can be used to set the virtual network interrupt rate in interrupts per second. The value can range from 100 to 100,000, with a default of 4,000.

- The feature can be set to queue a predefined number of packets before interrupting the virtual machine or transmitting the packets by setting `ethernetX.coalescingScheme` to **`static`**.

  In this case, the `ethernetX.coalescingParams` variable can be used to set the number of packets ESXi will queue. The value can range from 1 to 64, with a default of 64. A larger queue size can reduce the number of context switches between the virtual machine and the VMkernel, potentially reducing CPU utilization both in the virtual machine and in the VMkernel.

  Regardless of the number of packets queued, however, ESXi waits no longer than 4 milliseconds before sending an interrupt or transmitting the packets. Other events, such as the virtual machine being idle, can also trigger virtual machine interrupts or packet transmission, so packets are rarely delayed the full 4 milliseconds.

- The feature can be set to be adaptive by setting `ethernetX.coalescingScheme` to **`adapt`**. This setting, new in vSphere 6.5, bases the interrupt rate on both the virtual machine load and the overall system load; it uses a lower interrupt rate when the load is high and a higher interrupt rate when the load is low.

- The feature can be disabled by setting `ethernetX.coalescingScheme` to **`disabled`**. Disabling this feature will typically result in more interrupts (and thus higher CPU utilization), but will typically also lead to lower network latency.

To configure VMXNET3 virtual interrupt coalescing through the vSphere Web Client:

1. Right-click the virtual machine you wish to change, then select **Edit Settings**.
2. Under the **VM Options** tab, expand **Advanced**, then click **Edit Configuration**.
3. Look for and set the variable you wish to change:
   - If selecting a virtual interrupt coalescing scheme, look for **ethernetX.coalescingScheme** (where **X** is the number of the virtual NIC to configure) and set it to your desired value.

     If the variable isn't present, enter it as a new variable, enter a value, and click **Add**.
   - If configuring the virtual network interrupt rate or queue size, look for **ethernetX.coalescingParams** (where **X** is the number of the virtual NIC to configure) and set it to your desired value.

     If the variable isn't present, enter it as a new variable, enter a value, and click **Add**.

The change will not take effect until the virtual machine has been restarted.

Running Network Latency Sensitive Applications

By default the ESXi network stack is configured to drive high network throughout at low CPU cost. While this default configuration provides better scalability and higher consolidation ratios, it comes at the cost of potentially higher network latency. vSphere provides a variety of configuration options to improve performance of applications that are highly sensitive to network latency (latency on the order of a few tens of microseconds). This section describes those options. For further information on this topic, see *Best Practices for Performance Tuning of Latency-Sensitive Workloads in vSphere VMs*.

- Designate specific virtual machines as highly sensitive to network latency.

  vSphere allows you to set the network latency sensitivity of a virtual machine to **High** (from the default of **Normal**). This causes the scheduler to attempt to exclusively assign one physical CPU core for each vCPU in the virtual machine. If the assignment succeeds, network-related latency and jitter is greatly reduced. This also means, though, that the assigned physical cores are then no longer available to other virtual machines or vmkernel worlds.

  When setting network latency sensitivity to high, consider these additional important points:

  - Before network latency sensitivity can be set to high for a virtual machine, memory for that virtual machine must be reserved 100%.
  - You can ensure the virtual machine is granted exclusive assignment of the CPU cores it uses by setting a 100% CPU reservation in advance. While this is not required in order to set latency sensitivity to high, failing to do so might result in the virtual machine not getting exclusive access to the physical CPUs due to pre-existing CPU over-commitment.
  - The reservations for both CPU cores and memory, as well as the latency-sensitivity=high setting, will persist when the virtual machine is moved to other hosts using vMotion.

- Fully reserve memory.

  In addition to being required in order to use latency-sensitivity=high (as described above), fully reserving memory can help reduce latency even when that option is not set.

- Fully reserve CPUs.

  In addition to allowing latency-sensitivity=high to work reliably (as described above), making a full CPU reservation, even without setting latency-sensitivity=high, can help to reduce network latency.

- To further reduce latency and jitter for a virtual machine that has been set to latency-sensitivity=high and has a full CPU reservation, you can also reserve CPU cores exclusively for related system contexts using the `sched.cpu.latencySensitivity.sysContexts` option (**Edit Settings** > **VM Options** > **Advanced** > **Edit Configuration**; if the option is not present you can add it). This option specifies the number of extra cores to exclusively reserve for system contexts detected by the CPU scheduler as related to this VM.

  Exclusive affinity for the system contexts is obtained on a best-effort basis and can be automatically unset.

- Use SR-IOV or DirectPath I/O for latency sensitive traffic.

  Virtual network adapters such as VMXNET3 and E1000 incur virtualization overhead on the order of a few microseconds per packet. When this overhead is not desirable, and certain core virtualization features are not needed, you might obtain lower network latency by providing the virtual machine direct access to the network device using DirectPath I/O ("DirectPath I/O" on page 40) or SR-IOV ("Single Root I/O Virtualization (SR-IOV)" on page 41). In either case, we also recommend tuning the interrupt rate for the device in the guest.

  The VMXNET3 virtual network adapter is the best choice if direct device access is unavailable or undesirable (see "Guest Operating System Networking Considerations" on page 53).

- Adjust the host power management settings.

  Some of the power management features in newer server hardware can increase network latency. If desired, you can disable them as follows:

  - Set the ESXi host power policy to **Maximum performance** (as described in "Host Power Management in ESXi" on page 24; this is the preferred method) or disable power management in the BIOS (as described in "Power Management BIOS Settings" on page 17).
  - Disable C1E and other C-states in BIOS (as described in "Power Management BIOS Settings" on page 17).
  - Enable Turbo Boost in BIOS (as described in "General BIOS Settings" on page 17).

- Tune the virtual network adapter.

For a majority of general purpose applications, and for moderately latency-sensitive applications, the VMXNET3 virtual network adapter, left at its default settings, will provide the best performance. Applications with low packet rate or few active threads might benefit by disabling VMXNET3 virtual interrupt coalescing for the desired NIC. In other cases—most notably applications with high numbers of outstanding network requests—disabling interrupt coalescing can reduce performance.

For further details about virtual interrupt coalescing, and instructions to change its configuration, see "Virtual Network Interrupt Coalescing" on page 43.

Host-Wide Performance Tuning

By default ESXi supports high consolidation ratios while still providing good response times for every virtual machine and every individual request. There are scenarios, however, where the ability to support very high consolidation ratios while keeping *average* response times low is more important than keeping *all* response times low. A feature introduced in vSphere 6.0, host-wide performance tuning (also called dense mode), allows system administrators to optimize individual ESXi hosts for such very high consolidation ratios (up to about 10% higher than without the feature).

When dense mode is enabled, ESXi monitors the number of virtual machines, number of vCPUs, and total CPU utilization; when all three exceed certain thresholds (numVMs >= numPCPUs, numvCPUs >= 2 * numPCPUs, and CPUUtil >= 50%) the dense mode optimizations are implemented.

The dense mode optimizations essentially batch packets more aggressively at various points. This primarily accomplishes two things:

- Reduces hypervisor overhead for packet processing

  Lower hypervisor overhead leaves more CPU resources for virtual machines. This is beneficial because when CPU utilization is high, average response time is greatly influenced by the availability of spare CPU resources.

- Reduces the number of interruptions to virtual machine execution

  Reducing the number of interruptions can increase the efficiency of the virtual machines.

Because dense mode increases batching of packets, it can increase the latency for individual requests. Under low CPU utilization conditions, or when some virtual machines consume a disproportionately large amount of network throughput, enabling dense mode might reduce system-wide throughput (though while still also reducing CPU utilization per unit of throughput). We therefore recommend enabling dense mode only on those hosts where its trade-offs will be useful, rather than simply enabling it on all hosts.

To enable or disable dense mode, use the vSphere Web Client (or `esxcli`) to change the `/Net/NetTuneHostMode` variable as follows:

- Dense mode disabled (the default configuration): `/Net/NetTuneHostMode = "default"`
- Dense mode enabled: `/Net/NetTuneHostMode = "dense"`

NOTE Even when dense mode is enabled, the dense mode optimizations are implemented only when a host's consolidation ratio and CPU utilization reach the thresholds described above.

Guest Operating Systems 3

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 ESXi. Refer to the VMware Compatibility Guide for a list (go to http://www.vmware.com/resources/compatibility then, in **What are you looking for:** choose **Guest OS**).

  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 ESXi 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 ESXi. Ballooning (described in "Memory Overcommit Techniques" on page 27) 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. Avoid scheduling them to run simultaneously in multiple virtual machines on the same ESXi 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, the VMware Tools time-synchronization option, or another timekeeping utility suitable for your operating system.

  NOTE As of the version included in ESXi 5.0, the VMware Tools time-synchronization option is a suitable choice. Versions prior to ESXi 5.0 were not designed for the same level of accuracy and do 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.

  For additional information about best practices for timekeeping within virtual machines, see VMware KB articles 1318 and 1006427.

Measuring Performance in Virtual Machines

Be careful when measuring performance from within virtual machines.

- vSphere allows you to take advantage of virtualized CPU performance counters to use performance tuning tools inside the guest operating system. For more information, see the *vSphere Virtual Machine Administration* guide.
- 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"). 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 within virtual machines can fail to take into account resources used by ESXi for tasks it offloads from the guest operating system, as well as resources consumed by virtualization overhead.

Measuring resource utilization using tools at the ESXi level, such as the vSphere Web 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:
  - Linux kernels after version 2.6 support the NO_HZ kernel configuration option (sometimes called "tickless timer") that uses a variable timer interrupt rate. Linux kernel versions 2.6 and earlier typically request a fixed timer rate, such as 100 Hz, 250 Hz, or 1000 Hz.
  - 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 (see "UP vs. SMP HALs/Kernels" on page 21 for more information on 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 (see "UP vs. SMP HALs/Kernels" on page 21 for more information on HALs/kernels).
  - 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.
  - 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*.

Virtual NUMA (vNUMA)

Virtual NUMA (vNUMA) exposes NUMA topology to the guest operating system, allowing NUMA-aware guest operating systems and applications to make the most efficient use of the underlying hardware's NUMA architecture.

NOTE For more information about NUMA, see "Non-Uniform Memory Access (NUMA)" on page 22.

Virtual NUMA, which requires virtual hardware version 8 or later, can in some cases provide significant performance benefits for wide virtual machines (as defined in "Non-Uniform Memory Access (NUMA)" on page 22), though the benefits depend heavily on the level of NUMA optimization in the guest operating system and applications.

- You can obtain the maximum performance benefits from vNUMA if your clusters are composed entirely of hosts with matching NUMA architecture.

  This is because the very first time a vNUMA-enabled virtual machine is powered on, its vNUMA topology is set based in part on the NUMA topology of the underlying physical host on which it is running. By default, once a virtual machine's vNUMA topology is initialized it doesn't change unless the number of vCPUs in that virtual machine is changed. This means that if a vNUMA virtual machine is moved to a host with a different NUMA topology, the virtual machine's vNUMA topology might no longer be optimal for the underlying physical NUMA topology, potentially resulting in reduced performance.

- When sizing your virtual machines, take into account the size of the physical NUMA nodes:

  NOTE Some multi-core processors have NUMA node sizes that are different than the number of cores per socket. For example, some 12-core processors have two six-core NUMA nodes per processor.

  - For the best performance, try to size your virtual machines to stay within a physical NUMA node. For example, if you have a host system with six cores per NUMA node, try to size your virtual machines with no more than six vCPUs.
  - When a virtual machine needs to be larger than a single physical NUMA node, try to size it such that it can be split evenly across as few physical NUMA nodes as possible.
  - Use caution when creating a virtual machine that has a vCPU count that exceeds the physical processor core count on a host. Because hyper-threads are considered logical threads, this is sometimes permissible, but will potentially create contention when used.

- As of vSphere 6.5, changing the `corespersocket` value no longer influences vNUMA or the configuration of the vNUMA topology. The configuration of vSockets and corespersocket now only affects the presentation of the virtual processors to the guest OS, something potentially relevant for software licensing. vNUMA will automatically determine the proper vNUMA topology to present to the guest OS based on the underlying ESXi host.

  For example: prior to vSphere 6.5, if you create a four-vSocket virtual machine and set `corespersocket` to four (for a total of 16 vCPUs) on a dual-socket physical ESXi host with 16 core per socket (for a total of 32 physical cores), vNUMA would have created four vNUMA nodes based on the `corespersocket` setting. In vSphere 6.5, the guest OS will still see four sockets and four cores per socket, but vNUMA will now create just one 16-core vNUMA node for the entire virtual machine because that virtual machine can be placed in a single physical NUMA node.

  This new decoupling of the `corespersocket` setting from vNUMA allows vSphere to automatically determine the best vNUMA topology.

To revert this behavior and directly control the vNUMA topology, see the `numa.vcpu.followcorespersocket` setting in the "Virtual NUMA Controls" section of the *vSphere 6.5 Resource Management Guide*.

NOTE Although corespersocket no longer directly sets the vNUMA topology, some corespersocket values could result in sub-optimal guest OS topologies; that is, topologies that are not efficiently mapped to the physical NUMA nodes, potentially resulting in reduced performance.

For more information about this, see VMware KB article 81383 and the Virtual Machine vCPU and vNUMA Rightsizing – Guidelines blog post. In addition, the Virtual Machine Compute Optimizer tool can provide guidance on configuring optimal topologies.

- By default, vNUMA is enabled only for virtual machines with more than eight vCPUs. This feature can be enabled for smaller virtual machines, however, while still allowing ESXi to automatically manage the vNUMA topology. This can be useful for wide virtual machines (that is, virtual machines with more vCPUs than the number of cores in each physical NUMA node) with eight or fewer vCPUs.

  You can enable vNUMA for virtual machines with eight or fewer vCPUs by adding to the `.vmx` file the line:
  `numa.vcpu.min = X`
  (where *X* is the number of vCPUs in the virtual machine).

  NOTE This change can be made through the vSphere Web Client:

  To change this variable through the vSphere Web Client:

  1. Select the virtual machine you wish to change.
  2. Under the **Configure** tab, expand **Settings**, then select **VM Options**
  3. Click **Edit** (in the upper right corner).
  4. Expand **Advanced**, then in **Configuration Parameters** click **Edit Configuration**.
  5. Look for **numa.vcpu.min** and configure it as you wish. If the variable isn't present, enter it as a new variable and click **Add**.
  6. Click **OK** to close the **Configuration Parameters** window.
  7. Click **OK** to close the **Edit Settings** window.

  Alternatively, you can take full manual control of a virtual machine's vNUMA topology using the `maxPerVirtualNode` option. For more details, see the "Virtual NUMA Controls" section of the *vSphere 6.5 Resource Management Guide*.

Guest Operating System Storage Considerations

The virtual storage adapter presented to the guest operating system can influence storage performance, as can that device driver, driver settings, and other factors within the guest. This section addresses those considerations.

- For most guest operating systems, the default virtual storage adapter in ESXi 6.5 is either LSI Logic Parallel or LSI Logic SAS, depending on the guest operating system and the virtual hardware version. However, ESXi also includes a paravirtualized SCSI storage adapter, PVSCSI (also called VMware Paravirtual). The PVSCSI adapter offers a significant reduction in CPU utilization as well as potentially increased throughput compared to the default virtual storage adapters, and is 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 or later, as described under "ESXi General Considerations" on page 19.

  NOTE PVSCSI adapters are supported for boot drives in only some operating systems. For additional information see VMware KB article 1010398.

- 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.
- vSphere 6.5 introduces a Non-Volatile Memory Express (NVMe) virtual storage adapter (virtual NVMe, or vNVMe). This allows recent guest operating systems that include a native NVMe driver to use that driver to access storage through ESXi, whether or not ESXi is itself using NVMe storage.

  Because the vNVMe virtual storage adapter has been designed for extreme low latency flash and non-volatile memory based storage, it isn't best suited for highly parallel I/O workloads and slow disk storage. For workloads that primarily have low outstanding I/O, especially latency-sensitive workloads, vNVMe will typically perform quite well. Compared to virtual SATA devices, vNVMe virtual storage adapters access local PCIe SSD devices with significantly lower CPU cost per I/O and significantly higher IOPS.
- 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

When a virtual machine is first created, ESXi allows selection of the virtual network adapter (vNIC) it will use. The available options are based on the specified guest operating system and the virtual hardware version. Because some operating systems don't have built-in drivers for the best-performing virtual network adapters, the list of available vNICs won't always include those that might provide the best performance. In some of these cases, though, a driver can later be installed in the guest operating system (typically as part of VMware Tools) and the vNIC then changed to a different type.

This section describes the various types of vNICs, how to select one, and how to obtain the best performance from it.

Types of Virtual Network Adapters

The possible virtual network adapters include three emulated types, three paravirtualized types, and a hybrid adapter.

The emulated virtual network adapters are:

- The Vlance virtual network adapter, which emulates an AMD 79C970 PCnet32 NIC. Drivers for this NIC are found in most 32-bit operating systems.
- The E1000 virtual network adapter, which emulates an Intel 82545EM NIC. Drivers for this NIC are found in many recent operating systems.
- The E1000e virtual network adapter, which emulates an Intel 82574 NIC. Drivers for this NIC are found in a smaller set of recent operating systems.

The VMXNET family of paravirtualized network adapters provide better performance in most cases than the emulated adapters. These paravirtualized network adapters implement an idealized network interface that passes network traffic between the virtual machine and the physical network interface card with minimal overhead. The VMXNET virtual network adapters (especially VMXNET3) also offer performance features not found in the other virtual network adapters. Drivers for VMXNET-family adapters are available for many guest operating systems supported by ESXi; for optimal performance these adapters should be used for any guest operating system that supports them.

The VMXNET family includes:

- The VMXNET virtual network adapter.
- The VMXNET2 virtual network adapter (also called "Enhanced VMXNET"). Based on the VMXNET adapter, this adapter adds a number of performance features.
- The VMXNET3 virtual network adapter (also called VMXNET Generation 3). This adapter has all the features of the VMXNET2, along with several new ones.

Lastly, there's a hybrid virtual network adapter:

- The Flexible virtual network adapter, which starts out emulating a Vlance adapter, but can function as a VMXNET adapter with the right driver.

In some cases, the virtual network adapter options will include **SR-IOV passthrough**. For information on this feature, see "Single Root I/O Virtualization (SR-IOV)" on page 41.

NOTE The network speeds reported by the guest network driver in the virtual machine do not 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 10Mb/s because the AMD PCnet card that ESXi is emulating is a 10Mb/s device. This is true even if the physical card on the server is 100Mb/s, 1Gb/s, or faster. However, ESXi is not limited to 10Mb/s in this situation and transfers network packets as fast as the resources on the physical server machine allow.

For more information on virtual network adapters, see VMware KB article 1001805 and *Virtual Machine Administration* for vSphere 6.5.

Selecting Virtual Network Adapters

This section describes how to select virtual network adapters for the best performance.

- 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 or later and, in some cases, requires 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/ESXi 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 or later (to maintain vMotion compatibility with older versions of ESX/ESXi, for example), the best performance can be obtained with the use of VMXNET2 for operating systems in which it is supported. This typically requires that VMware Tools be installed in the guest operating system.

  NOTE A virtual machine with a VMXNET2 device cannot vMotion to a host running ESX 3.0.x or earlier.

- For the operating systems in which VMXNET2 is not supported, use the flexible device type. In ESXi, 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.

Virtual Network Adapter Features and Configuration

This section describes various virtual network adapter features and how to configure the adapters for the best performance.

- When networking two virtual machines on the same ESXi host, try to connect them to the same vSwitch. When connected this way, their network speeds are not limited by the wire speed of any physical network card. Instead, they transfer network packets as fast as the host resources allow.
- The E1000, E1000e, VMXNET2, and VMXNET3 devices support jumbo frames. The use of jumbo frames can allow data to be transmitted using larger, and therefore fewer, packets. Because much of the CPU load from network traffic is per-packet overhead, a lower packet rate can reduce the CPU load and thus increase performance.

  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 ESXi TCP Segmentation Offload (TSO) is enabled by default in the VMkernel but is supported in virtual machines only when they are using the E1000 device, the E1000e device, the VMXNET2 device, or the VMXNET3 device. TSO can improve performance even if the underlying hardware does not support TSO.
- Similarly, in ESXi Large Receive Offload (LRO) is enabled by default in the VMkernel, but is supported in virtual machines only when they are using the VMXNET2 device or the VMXNET3 device. LRO might improve performance even if the underlying hardware does not support LRO.

  NOTE LRO support varies by operating system. Many Linux variants support LRO. In Windows virtual machines LRO is supported when the following prerequisites are met:

  - The virtual machine uses virtual hardware version 11 or later.
  - The virtual machine uses the VMXNET3 device.
  - The virtual machine is running Windows Server 2012 or later or Windows 8.0 or later.
  - The guest operating system uses VMXNET3 vNIC driver version 1.6.6.0 or later.
  - Receive Segment Coalescing (RSC) is globally enabled in the guest operating system.

  For more information about configuring LRO, search for "Large Receive Offload" in *vSphere Networking* for vSphere 6.5.

- In some cases, low receive throughput in a virtual machine can be due to insufficient receive buffers (also described as an overflowing receive ring), in the receiver network device. If the receive buffers in the guest operating system's network driver are insufficient, packets will be dropped in the VMkernel, degrading network throughput. A possible workaround is to increase the number of receive buffers, though this might 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 VMXNET2, 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 additional information see VMware KB article 1010071.

  For E1000, E1000e, and VMXNET3, the default number of receive and transmit buffers are controlled by the guest driver, with the maximum possible for both being 4096. In 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.

- Multiple receive and transmit queues (often referred to as receive-side scaling (RSS) or scalable I/O) allow network packets from a single NIC to be scheduled in parallel on multiple CPUs. Without multiple queues, network interrupts can be handled on only one CPU at a time. Multiple queues help throughput in cases where a single CPU would otherwise be saturated with network packet processing and become a bottleneck. To prevent out-of-order packet delivery, the driver schedules all of a flow's packets to the same CPU.

  The E1000e and VMXNET3 devices support multiple queues for many guest operating systems that natively support the feature, which include Windows Server 2003 SP2 or later, Windows 7 or later, and some Linux distributions.

  The VMXNET3 drivers included with the vSphere 5.0 and later versions of VMware Tools default to having multiple receive queues enabled, as does the VMXNET3 driver included with Linux kernel 2.6.37 and later. For more information, see VMware KB article 2020567.

  When multiple receive queues are enabled, the feature by default configures 1, 2, 4, or 8 receive queues in a virtual machine, choosing the largest of these values less than or equal to the number of vCPUs in that virtual machine.

  In Windows, regardless of the number of receive queues, the number of transmit queues defaults to 1. If the driver is also configured to use multiple transmit queues, the number of transmit queues will be the same as the number of receive queues.

  In Linux the number the number of transmit queues, by default, is the same as the number of receive queues.

  In order to obtain the maximum performance with your specific workloads and resource availability you can try out different values for the number of receive queues (which must be set to a power of 2 and can be a maximum of 8 or the number of vCPUs, whichever is lower), and you can try configuring the driver to use multiple transmit queues (except in Linux, where the number of transmit queues will already match the number of receive queues). These settings are changed on the advanced driver configuration tab within the guest operating system.

Virtual Infrastructure Management 4

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

ESXi 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, **Reservation**, **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, ESXi 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 (as described in "ESXi Memory Considerations" on page 26) 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.

- Whether run on virtual machines or physical systems, make sure you provide vCenter Server and the vCenter Server database with sufficient CPU, memory, and storage resources (both capacity and performance) for your deployment size. For additional information see the 6.5 version of *vSphere Installation and Setup*.
- Because of the importance of the database to vCenter performance, make sure to follow the recommendations in "VMware vCenter Database Considerations" on page 59.
- The performance of vCenter Server is dependent in large part on the number of managed entities (hosts and virtual machines) and the number of connected vSphere Web Clients. Exceeding the maximums specified in *Configuration Maximums for vSphere 6.5*, in addition to being unsupported, is thus likely to impact vCenter Server performance.
- To minimize the latency of vCenter operations, keep to a minimum the number of network hops between the vCenter Server system and the vCenter Server database.
- Be aware, also, that network latency between vCenter Server and the hosts it manages can impact the performance of operations involving those hosts.
- The VMware vSphere Update Manager, when deployed in Windows, can be run on the same system and use the same database as vCenter Server but, for maximum performance on heavily-loaded vCenter systems, consider running Update Manager on its own system and providing it with a dedicated database. (Note that this doesn't apply to vSphere Update Manager in Linux — that is, with the vCenter Server Appliance) For additional information see "VMware vSphere Update Manager" on page 81.
- During installation of VMware vCenter you will be asked to choose a target inventory size. This choice will be used to set Java virtual machine (JVM) maximum heap memory sizes for various services. The default values (detailed in the 6.5 version of *vSphere Installation and Setup*; search for "Tomcat Server settings") should provide good performance while avoiding unnecessary memory commitment. If you will have inventory sizes well into the large range (as defined in the 6.5 version of *vSphere Installation and Setup*; search for "vCenter Server for Windows Hardware Requirements"), however, you might obtain better performance by increasing one or more of these settings.

VMware vCenter Database Considerations

vCenter Server relies heavily on a database to store configuration information about inventory items and performance statistics data. It also stores data about alarms, events, and tasks. Due to the importance of this database to the reliability and performance of your vCenter Server, VMware recommends the database practices described in this section.

VMware vCenter Database Network and Storage Considerations

- 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.
- The hardware on which the vCenter database is stored, and the arrangement of the files on that hardware, can have a significant effect on vCenter performance:
  - The vCenter database performs best when its files are placed on high-performance storage.
  - The database data files generate mostly random write I/O traffic, while the database transaction logs generate mostly sequential write I/O traffic. For this reason, and because their traffic is often significant and simultaneous, vCenter performs best when these two file types are placed on separate storage resources that share neither disks nor I/O bandwidth.
- If the vCenter database is placed on thin-provisioned or lazy-zeroed thick-provisioned storage, vCenter startup might be slower than it would be if placed on eager-zeroed thick-provisioned storage. See "Virtual Disk Types" on page 33 for more information on these provisioning types.
- Large deployments can rapidly generate significant amounts of data. It's good practice to periodically monitor the database storage to avoid running out of storage space. For information about setting an alert for this in a vCenter Server Appliance, see VMware KB article 2058187.

VMware vCenter Database Configuration and Maintenance

- 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.

  When you do need higher levels (for interactive debugging, for example), increase the level only temporarily, and set it back to a lower level when done.

  For more a detailed discussion of this topic, see *VMware vCenter 5.1 Database Performance Improvements and Best Practices for Large-Scale Environments* (though this paper specifically addresses vSphere 5.1, most of the concepts are applicable to vSphere 6.5).
- To avoid frequent transaction log switches, ensure that your vCenter database logs are sized appropriately for your vCenter inventory. For example, with a large vCenter inventory running with an Oracle database, the size of each redo log should be at least 1GB.
- 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.

  To change the pool size, edit the `vpxd.cfg` file, adding:

```
<vpxd>
 <odbc>
  <maxConnections>xxx</maxConnections>
 </odbc>
</vpxd>
```

  (where *xxx* is the desired pool size).

  NOTE If you make this change to the vCenter pool size, you should also consider increasing your database's maximum allowed connections.

- Update statistics of the tables and indexes on a regular basis for better overall performance of the database.
- As part of the regular database maintenance activity, check the fragmentation of the index objects and recreate indexes if needed (i.e., if fragmentation is more than about 30%).
- Top N is a vCenter feature that allows you to view the top consumers of certain resources (CPU, memory, disk, and network) in chart form (in the vSphere Web Client select a cluster or host, choose the **Monitor** tab, choose **Performance**, click **Overview**, then select **Resource Pools and Virtual Machines**, **Hosts**, or **Virtual Machines**). The computation of Top N data imposes a small additional load on the vCenter database. While we expect this load to be negligible in most circumstances, the feature can be disabled if desired. Disabling the feature would result only in the Top N charts being blank. If you wish to disable it, follow these steps:
  - On Microsoft SQL Server: From the SQL Server Agent, disable the ***TOPN* tables** job.
  - On Oracle: Use the `DBMS_JOB.BROKEN` package to disable the ***TOPN* tables** job.

  NOTE There is no option to disable the Top N feature when using the vPostgres database (used by vCenter Server if you select the embedded database option during deployment).

  If you later wish to re-enable Top N, wait until all the database sessions are cleared and the database is back to a normal state, then re-enable the job.

Recommendations for Specific Database Vendors

This subsection describes database-specific recommendations.

PostgreSQL (vPostgres) Database Recommendations

If you are using a PostgreSQL (vPostgres) database (used by vCenter Server if you select the embedded database option during deployment), many optimizations are done automatically. However, the points in this section can improve vCenter Server performance.

NOTE Many vCenter Server Appliance configuration tasks can be accomplished using the vCenter Server Appliance Management Interface. This interface can be accessed using a web browser to access port 5480 on the appliance (`https://<appliance-IP-address>:5480`), then logging in as root (using the password set when the vCenter Server Appliance was deployed).

- The vCenter Server Appliance creates multiple virtual disks. For improved performance in large scale environments, make sure that the database virtual disk (`/storage/db`) and the virtual disks related to logs, statistics, events, and alarms (`/storage/dblog`, and `/storage/seat`) are all backed by different physical disks.
- The `/storage/dblog` virtual disk stores the database transaction log. While the performance of each partition is important to vCenter performance, vCenter is particularly sensitive to the latency and throughput of this partition. We thus recommend placing this partition on low latency, high throughput storage. VMware KB article 2126276, though primarily addressing resizing of virtual disks, also includes a table showing which VMDK corresponds to each mount point.
- Though PostgreSQL does its own data caching, data is also cached at the operating system level. Performance can be improved by increasing the size of the OS-level cache. This can be accomplished by increasing the amount of memory allocated to the vCenter Server Appliance virtual machine; a portion of this additional memory will be automatically used for the OS-level data cache.
- PostgreSQL periodically sets checkpoints to guarantee that preceding transactions have been flushed to disk. These checkpoints are triggered every time database writes cumulatively approach 90% of the size of the `/storage/dblog/` partition. Increasing the size of this partition beyond its default size of 5GB therefore increases the time between checkpoints. This reduces the overall I/O load on the system, especially for large vCenter deployments, but increases recovery time after a crash. For information about changing the size of the `/storage/dblog/` partition, see VMware KB article 2126276.

- In order to avoid having the vCenter Server Appliance database run out of disk space, you can set alarms using the `vpxd.vdb.space.errorPercent` and `vpxd.vdb.space.warningPercent` parameters under the advanced vCenter Server settings. In addition to displaying warnings in the vSphere Web Client, these alarms can also be configured to send email notifications.

Microsoft SQL Server Database Recommendations

If you are using a Microsoft SQL Server database, the following points can improve vCenter Server performance:

- Setting the transaction logs to **Simple** recovery mode significantly reduces the database logs' disk space usage as well as their storage I/O load. If it isn't possible to set this to **Simple**, make sure to have a high-performance storage subsystem.
- To further improve database performance for large vCenter Server inventories, place tempDB on a different disk than either the database data files or the database transaction logs.
- To ensure consistent database performance and avoid login failures, make sure the AUTO_CLOSE parameter is set to OFF. This will prevent SQL Server from releasing resources when there are no users, only to have to reserve them again as soon as another connection is made.

Oracle Database Recommendations

If you are using an Oracle database, the following points can improve vCenter Server performance:

- When using Automatic Memory Management (AMM) in Oracle 11g, or Automatic Shared memory Management (ASMM) in Oracle 10g, allocate sufficient memory for the Oracle database.
- Set appropriate **PROCESSES** and **SESSIONS** initialization 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 connections Oracle can accept. In large vSphere environments (as defined in the 6.5 version of *vSphere Installation and Setup*; search for "vCenter Server for Windows Hardware Requirements") we recommend setting both **PROCESSES** and **SESSIONS** to **`1000`**. In environments that approach the configuration maximums (as specified in *Configuration Maximums for vSphere 6.5*), it might be necessary to increase both of these values to **`1200`**.
- If database operations are slow, after checking that the statistics are up to date and the indexes are not fragmented, you should move the indexes to separate tablespaces (i.e., place tables and primary key (PK) constraint index on one tablespace and the other indexes (i.e., BTree) on another tablespace).
- For large vCenter Server inventories (i.e., those that approach the limits for the number of hosts or virtual machines), increase the number of threads available for writing by changing the `db_writer_processes` parameter to `6`. This change can be useful for applications that issue many disk writes, such as vpxd.

VMware vSphere Management

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

Large numbers of connected vSphere Web Clients and vSphere Web Services SDK Clients can affect the performance of a vCenter Server. Exceeding the maximums specified in *Configuration Maximums for vSphere 6.5* is not supported. Even if it seems to work, doing so is even more likely to affect vCenter Server performance.

vSphere Web Clients

The VMware vSphere Web Client can be used to control and configure vCenter Servers, ESXi hosts, and virtual machines. The back end, called the vSphere Web Client Server, is a Java application. The front end, called the vSphere Web Client, is an Adobe Flash application running in a web browser pointed to the Java-based Application Server.

vSphere Web Client Back-End Performance Considerations

The vSphere Web Client back-end consists of the vSphere Web Client Server, a Java application, which in turn interacts heavily with the vCenter Server. Both of these modules thus impact the vSphere Web Client back-end performance. This subsection describes how to obtain the best performance from the vSphere Web Client back-end.

NOTE In vSphere 6.5, the inventory service is replaced by `vpxd-svcs`, a lighter-weight service containing logic for authorization, roles, and privileges.

- For optimal performance, make sure you provide the vSphere Web Client Server with sufficient CPU, memory, and storage resources for your deployment size and usage patterns. These resource requirements are listed in the 6.5 version of *vSphere Installation and Setup* (included as part of the vCenter Server requirements) and are described during the installation process.
- Performance of the vSphere Web Client is also significantly influenced by the following factors:
  - The vCenter inventory size
  - The rate of operations (that is, tasks per hour) performed through vCenter
  - The number of vSphere Web Clients in use
  - The usage patterns of vSphere Web Client users
- Installing plug-ins in the vSphere Web Client Server might cause higher memory usage on the system running the vSphere Web Client Server. It's therefore a good practice, after any plug-ins are installed and enabled, to monitor the vSphere Web Client Server's memory usage. You can use Task Manager (in Windows) or `top` (in Linux) to make sure the memory usage of the Java process is either below or only slightly above its maximum heap size. (You can also use a Java profiler to see a more detailed view of the memory usage of the JVM.)

  NOTE You can determine the JVM's maximum heap size as follows:

  - In a vCenter Server Appliance, run:
    `cloudvm-ram-size -l vsphere-client`
  - In Windows, locate the file `cloudvm-ram-size.bat` (by default, it is in `C:\Program Files\VMware\vCenter Server\visl-integration\usr\sbin`) and run:
    `cloudvm-ram-size.bat -l vspherewebclientsvc`

  If you find that the amount of memory allocated to the vSphere Web Client Server is insufficient, increasing it can significantly improve performance.

The most straightforward way to increase the memory allocated to the vSphere Web Client Server is to increase the total amount of memory provisioned for the system on which it's running, then reboot the system. At boot-up, a dynamic memory algorithm will automatically reconfigure the amount of memory allocated to the various services, including the vSphere Web Client Server.

If you don't wish to increase the total memory, or if you find that the choices made by the dynamic memory algorithm aren't appropriate for your environment, you can manually change the vSphere Web Client Server maximum heap size. Be aware, though, that this will affect the amount of memory available for other services as well as for the operating system.

To manually change the vSphere Web Client Server maximum heap size:

- In a vCenter Server Appliance, run:
  `cloudvm-ram-size -C XXX vsphere-client`
  (where *XXX* is the desired heap size in MB).
  For more information, run `cloudvm-ram-size -h`.
- In Windows, locate the file `cloudvm-ram-size.bat` (by default, it is in `C:\Program Files\VMware\vCenter Server\visl-integration\usr\sbin`) and run:
  `cloudvm-ram-size.bat -C XXX vspherewebclientsvc`
  (where *XXX* is the desired heap size in MB).
  For more information, run `cloudvm-ram-size.bat -h`.

After changing the maximum heap size, you'll need to restart the vSphere Web Client service for the change to take effect:

- In a vCenter Server Appliance, run:
  ```
  service-control -stop vsphere-client
  service-control -start vsphere-client
  ```
- In Windows, open **Component Services** (the details vary slightly with Windows versions, but typically: **Start** menu > **All Programs** > **Administrative Tools** > **Component Services**), in the left pane select **Services**, scroll to **vsphere web client service**, right click on it and select **Stop**, then right click it again and select **Start**.

- A number of advanced configuration options can influence the performance of the vSphere Web Client. These options, listed in Table 4-1, can be configured in the `webclient.properties` file, found at the following locations:
  - In a vCenter Server Appliance:
    `/etc/vmware/vsphere-client`
  - In Windows:
    `%ProgramData%\VMware\vSphere Web Client`

Table 4-1. Advanced Configuration Options for the vSphere Web Client Back-End

Advanced option name	Description	Default Value
live.updates.enabled	Whether or not to enable live refresh in **Recent Tasks** view.	true
live.updates.alarms.enabled	Whether or not to enable live refresh in **Alarms** view.	true
live.updates.navtree.enabled	Whether or not to enable live refresh in four types of inventory trees (Hosts and Clusters, VMs and Templates, Storage, Networking).	true
live.updates.lists.enabled	Whether or not to enable live refresh in lists.	true

Table 4-1. Advanced Configuration Options for the vSphere Web Client Back-End

Advanced option name	Description	Default Value
live.updates.objectstate.enabled	Whether or not to enable live refresh in the **Summary** tab of the current object. This is not done for each property shown on each **Summary** tab, but only for a well-defined subset of properties for each type (the so-called "critical properties," that can roughly be described as the properties that determine the icon and the state of the corresponding object).	true
pagingThreshold	The initial number of retrieved children per type for a node in the inventory tree. The rest of the children will be loaded on demand.	2000
navigation.tabMode.convertSecondaryToToc	Whether or not to enable flattened view of Manage to Configure tab.	true
navigation.tabMode.convertSecondaryToToc.excludedPlugins	A denylist of plug-ins for which the Manage to Configure tab flattening cannot be applied.	com.vmware.srm.client,com.vmware.vShieldManager,com.vmware.vsphere.client.cmui,com.vmware.vum.client.updatemanagerui
navigation.tabMode.convertSecondaryToToc.supportedPlugins	An allowlist of plug-ins for which the Manage to Configure tab flattening can be applied.	com.vmware.*.client.*
show.allusers.tasks	Whether or not to enable All Users Tasks in the Recent Tasks view.	true
aggregationThreshold.VirtualMachine aggregationThreshold.HostSystem aggregationThreshold.Datastore aggregationThreshold.Network aggregationThreshold.OpaqueNetwork aggregationThreshold.DistributedVirtualPortgroup	The number of objects of each type the vSphere Web Client will display in its inventory tree before aggregating them to a single node with a count label. A value of 0 disables aggregation.	5000
session.timeout	Time (in minutes) before a user is automatically logged out.	120
refresh.rate	Time (in seconds) between automatic refreshes of the vSphere Web Client. Warning: Enabling auto-refresh prevents automatic logout of inactive users. A value of -1 disables automatic refresh.	-1

Table 4-1. Advanced Configuration Options for the vSphere Web Client Back-End

Advanced option name	Description	Default Value
alarms.refresh.rate	Time (in seconds) between automatic refreshes of the alarms list. Must be between 10 and 600. A value of -1 disables automatic refresh of alarms.	60
tasks.refresh.rate	Time (in seconds) between automatic refreshes of recent tasks lists. Must be between 10 and 600. A value of -1 disables automatic refresh of recent tasks lists.	60
tasks.display.count	The maximum number of tasks displayed in the recent tasks list. Must be between 1 and 100.	50
feature.facetedSearch.enabled	Whether or not the facet filter options will be available in lists.	true
portlets.collapsed	Whether or not the portlets shown on the summary page will be collapsed by default. When collapsed they request no data, thus potentially improving performance.	false
navigator.disableAnimation	Disables the sliding animation of the navigator pane	true
optimizedLogin.disabled	Disables the preloading of certain files. In some browsers, the preloading introduces a noticeable delay between a user typing their login credentials and those credentials appearing on the screen.	false

vSphere Web Client Front-End Performance Considerations

The vSphere Web Client front-end is an Apache Flex application running in a web browser on the user's system. This subsection describes how to obtain the best performance from the vSphere Web Client front-end

- The first vSphere Web Client login will typically be slower than subsequent logins due to various one-time initializations.
- For the best vSphere Web Client performance, make sure the user's system has sufficient CPU resources (we recommend at least two CPU cores; the faster the better) and memory (as an example, for client systems running Windows we recommend at least 4GB of RAM).
- Large network latencies can significantly reduce vSphere Web Client performance. For the best performance, the network latencies between the vSphere Web Client running on the user's system and the vSphere Web Client back end should be under 30ms.

NOTE When it's not possible to establish a low-latency connection, an alternate option would be to place a virtual or physical system near the vSphere Web Client back-end (such that it has a low-latency connection to the back-end), and run the vSphere Web Client front-end on that system, remotely accessing it from the more distant location using RDP or another remote protocol.

- Using a 64-bit browser to view the vSphere Web Client allows the allocation of more memory, potentially improving performance and avoiding the Flash player running out of memory.
- When possible, use the search function instead of navigating to managed objects. The vSphere Web Client is designed for the use of inventory search to find managed objects (clusters, hosts, virtual machines, datastores, and so on). Though you can access managed objects through the inventory tree or inventory list views, the inventory search function will typically provide better performance than navigating among these objects.
- Close the vSphere Web Client browser window occasionally (approximately daily). Closing and restarting the browser window in which the vSphere Web Client is running will prevent client sessions from consuming more memory than required.

vSphere Web Services SDK Clients

The VMware 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 vSphere API and SDK Documentation.

For examples of good programming practices, see code samples from the VMware Communities sample code page (http://communities.vmware.com/community/vmtn/developer/codecentral).

To use the SDK to obtain cluster performance data by aggregating information from individual hosts, see the vCenter Cluster Performance Tool fling (https://labs.vmware.com/flings/vcenter-cluster-performance-tool).

VMware vMotion and Storage vMotion

This section provides performance best practices for vMotion™, Storage vMotion, and Cross-host Storage vMotion.

VMware vMotion Recommendations

The following performance recommendations apply to vMotion:

- ESXi 6.5 introduces virtual hardware version 13. Because virtual machines running on hardware version 13 can't run on prior versions of ESX/ESXi, such virtual machines can be moved using VMware vMotion only to other ESXi 6.5 hosts. ESXi 6.5 is also backward compatible with virtual machines running on earlier virtual hardware versions, however, and these virtual machines can be moved using VMware vMotion to hosts running earlier ESX/ESXi versions with which they are compatible.
- vSphere 6.5 introduces encrypted vMotion. This feature encrypts vMotion traffic when both the source and destination hosts are capable of supporting encrypted vMotion (that is, when both hosts are running ESXi 6.5). If a virtual machine running on ESXi 6.5 is migrated to a host running an earlier ESX/ESXi version, vMotion traffic is not encrypted.

  Encryption performance is significantly higher on hosts that support AES-NI (Intel's Advanced Encryption Standard New Instruction Set). Without AES-NI encrypted vMotion operations might have unacceptable performance.

  Encrypted virtual machines are always moved with encrypted vMotion. For virtual machines that are not encrypted, vMotion encryption can be changed from **Opportunistic** (the default) to **Disabled** or **Required** (right-click the virtual machine, select **Edit Settings**, select **VM Options**, click **Encryption**, select an option from the **Encrypted vMotion** drop-down menu).
- vMotion performance will increase as additional network bandwidth is made available to the vMotion network. Consider provisioning 10Gb/s or faster vMotion network interfaces for maximum vMotion performance. Multiple vMotion vmknics can provide a further increase in the network bandwidth available to vMotion.

  All vMotion vmknics on a host should share a single vSwitch. Each vmknic's portgroup should be configured to leverage a different physical NIC as its active vmnic. In addition, all vMotion vmknics should be on the same vMotion network.
- To achieve full line rate on a 40Gb/s vMotion network, we recommend configuring three vMotion vmknics on the 40Gb/s NIC. The use of multiple vMotion vmknics enables vMotion to create multiple vMotion worker threads, thus spreading the CPU load across multiple cores rather than saturating a single core and bottlenecking network bandwidth. For further information see VMware KB article 2108824.
- While a vMotion operation is in progress, ESXi opportunistically reserves CPU resources on both the source and destination hosts in order to ensure the ability to fully utilize the network bandwidth. ESXi will attempt to use the full available network bandwidth regardless of the number of vMotion operations being performed. The amount of CPU reservation thus depends on the number of vMotion NICs and their speeds; 10% of a processor core for each 1Gb/s network interface, 100% of a processor core for each 10Gb/s network interface, and a minimum total reservation of 30% of a processor core. 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.
- vMotion performance might be reduced if host-level swap files are placed on local storage (whether SSD or hard drive). For more information on host-level swap files, see "Memory Swapping Optimizations" on page 29.
- During vMotion of a vFRC-enabled virtual machine (described in "vSphere Flash Read Cache (vFRC)" on page 32), the vFRC cache is migrated by default. Because this cache can potentially be quite large, migrating it can increase the time required to perform a vMotion operation on the virtual machine and can consume more network bandwidth in the process.

If desired, migration of the vFRC cache during vMotion can be manually disabled, causing the cache file to be discarded at the source host and recreated at the destination host. This can decrease vMotion time and save network bandwidth, it might also temporarily reduce the vFRC performance gains in the virtual machine after the vMotion operation is complete, while the cache gets repopulated.

The choice to disable vFRC cache migration should be made after taking into account the relative importance of application performance, network bandwidth utilization, and vMotion duration.

- To obtain the best vMotion performance with EMC VPLEX Metro Distributed Virtual Volume hardware, we recommend the following:
  - Provision hardware TSO-capable NICs for the vMotion network.
  - Add the VMX option (`extension.converttonew = "FALSE"`) to virtual machine's .vmx files. This option optimizes the opening of virtual disks during virtual machine power-on and thereby reduces switch-over time during vMotion. While this option can also be used in other situations, it is particularly helpful on VPLEX Metro deployments.
  - Because virtual machine snapshots slow the switchover process, avoid unneeded snapshots.

  For further information on this topic, see VMware KB article 1026692.

VMware Storage vMotion Recommendations

The following performance recommendations apply to Storage vMotion:

- VMware Storage vMotion performance depends strongly on the available storage infrastructure bandwidth between the ESXi host where the virtual machine is running and both the source and destination data stores.

  During a Storage vMotion operation the virtual disk to be moved is being read from the source data store and written to the destination data store. At the same time the virtual machine continues to read from and write to the source data store while also writing to the destination data store.

  This additional traffic takes place on storage that might also have other I/O loads (from other virtual machines on the same ESXi host or from other hosts) that can further reduce the available bandwidth.
- 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 can perform up to four simultaneous disk copies per Storage vMotion operation. Storage vMotion will involve each datastore in no more than one disk copy at any one time, however. This means, for example, that moving four VMDK files from datastore A to datastore B will happen serially, but moving four VMDK files from datastores A, B, C, and D to datastores E, F, G, and H will happen in parallel.

  For performance-critical Storage vMotion operations involving virtual machines with multiple VMDK files, you can use anti-affinity rules to spread the VMDK files across multiple datastores, thus ensuring simultaneous disk copies.
- During a Storage vMotion operation, the benefits of moving to a faster data store will be seen only when the migration has completed. However, the impact of moving to a slower data store will gradually be felt as the migration progresses.
- Storage vMotion will often have significantly better performance on VAAI-capable storage arrays (described in "Hardware Storage Considerations" on page 13).

VMware Cross-Host Storage vMotion Recommendations

Cross-host Storage vMotion allows a virtual machine to be moved simultaneously across both hosts and datastores. The following performance recommendations apply to Cross-host Storage vMotion:

- Cross-host Storage vMotion has performance optimizations that depend on a 1MB file system block size. While 1MB has been the default VMFS block size since the early VMFS versions, certain VMFS versions (VMFS 3.x, for example) allowed users to choose a different block size. If your VMFS file systems don't use 1MB block sizes, you can obtain significant reductions in the vMotion migration time by switching to the latest VMFS 5.x version.

  NOTE In-place upgrades of VMFS datastores with non-1MB block sizes to VMFS 5.x leave the block size unchanged. Thus in this situation it would be necessary to create a new VMFS 5.x datastore to obtain the performance benefits of 1MB block size.

- When using a 40Gb/s NIC, Cross-host Storage vMotion performs best with three vMotion vmknics, as described for vMotion in "VMware vMotion Recommendations" on page 67.
- Cross-host Storage vMotion can perform up to four simultaneous disk copies, as described for Storage vMotion in "VMware Storage vMotion Recommendations" on page 68.
- Cross-host Storage vMotion has a variety of options it can use to migrate virtual disk contents between datastores.

  For most powered-on virtual machines, Cross-host Storage vMotion typically uses the vMotion network. In a number of circumstances, however, it instead uses the following alternative methods (listed most-preferred first):

  - If both the source and destination datastores are on the same VAAI-capable array, and the source host has access to the destination datastore, Cross-host Storage vMotion will offload to the array via VAAI the task of copying the disk content.
  - If the source host has access to the destination datastore, vMotion will use the source host's storage interface to transfer the disk content, thus reducing vMotion network utilization and host CPU utilization.

  To migrate powered-off virtual machines and, in some cases, to migrate "cold" data (the base disk, and any snapshots other than the current one, in a virtual machine that has a snapshot), Cross-host Storage vMotion will use the Network File Copy (NFC) service. As in the case for powered-on virtual machines, the NFC service will similarly preferentially use VAAI or the source host's storage interface. If neither of these approaches are possible, the NFC service will use the network designated for NFC traffic.

  NOTE Prior to vSphere 6.0, NFC traffic could use only the management network (sometimes called the "provisioning network"). Starting with vSphere 6.0, NFC traffic still uses the management network by default, but can optionally be routed over a network dedicated to NFC traffic. This allows NFC traffic to be separated from management traffic, giving more flexibility in network provisioning.

  In any case, we recommend that NFC traffic be provided at least 1Gb/s of network bandwidth.

VMware Distributed Resource Scheduler (DRS)

This section lists Distributed Resource Scheduler (DRS) practices and configurations recommended by VMware for optimal performance.

DRS in General

- Watch for manual recommendations generated by DRS, especially if you've set any manual overrides. In some cases unaddressed manual recommendations might prevent DRS from generating recommendations.
- When DRS affinity rules are set, watch for DRS faults generated by rule violations. Addressing such faults can often significantly help with load balancing.

DRS Cluster Configuration Settings

- 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 improves performance predictability and stability.

  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.

- 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.

- The more vMotion compatible ESXi 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 1Gb/s 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 *vSphere vCenter Server and Host Management* for further details.

- When possible, make sure every host in a DRS cluster has connectivity to the full set of datastores accessible by the other hosts in that cluster. Such full connectivity allows DRS to make better decisions when balancing loads among hosts.
- 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 balancing migrations across the ESXi hosts before the virtual machines that are not in automatic mode.
- Powered-on virtual machines consume memory resources—and typically consume some CPU resources—even when idle. Thus even idle virtual machines, though their 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.

- 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.
- DRS affinity rules can keep two or more virtual machines on the same ESXi host ("VM/VM affinity") or make sure they are always on different hosts ("VM/VM anti-affinity"). 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 ESXi 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, from the vSphere Web Client select a cluster, click the **Configure** tab, expand **Configuration**, click **VM/Host 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 affinity setting types are:

  - **Keep Virtual Machines Together**
    This affinity type can improve performance due to lower latencies of communication between machines.
  - **Separate Virtual Machines**
    This affinity type can maintain 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.
  - **Virtual Machines to Hosts** (including **Must run on...**, **Should run on...**, **Must not run on...**, and **Should not run on...**)
    These affinity types 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 those hosts.
- DRS Doctor, an unofficial tool from VMware (https://labs.vmware.com/flings/drsdoctor), can provide insight into DRS and the actions it performs. This can be useful when troubleshooting DRS issues.

DRS Cluster Sizing and Resource Settings

- Exceeding the maximum number of hosts, virtual machines, or resource pools for each DRS cluster specified in *Configuration Maximums for vSphere 6.5* is not supported. Even if it seems to work, doing so could adversely affect vCenter Server or DRS performance.
- 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 for virtual machine memory overhead need to be accounted for when sizing resources in the cluster.

If the overall cluster capacity might not meet the needs of all virtual machines during peak hours, you can assign relatively higher shares to virtual machines or resource pools hosting mission-critical applications to reduce the performance interference from less-critical virtual machines.

- As of vSphere 6.0, DRS includes NetIOC ("Network I/O Control (NetIOC)" on page 39) bandwidth reservations in its calculations.
- If you will be using vMotion, it's a good practice to leave some unused (and unreserved) CPU capacity in your cluster. As described in "VMware vMotion Recommendations" on page 67, when a vMotion operation is started, ESXi reserves some CPU resources for that operation.

DRS Performance Tuning

- The migration threshold for fully automated DRS allows the administrator to control the aggressiveness of the DRS algorithm. In many cases, the default setting of the migration threshold, representing a medium level of aggressiveness, will work well.

  The migration threshold should be set to more aggressive levels when the following conditions are satisfied:

  - The hosts in the cluster are relatively homogeneous.
  - The virtual machines' resource utilization does not vary much over time.

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

  NOTE If the most conservative threshold is chosen, DRS will not perform load balancing; it will only apply move recommendations that must be taken either to satisfy hard constraints, such as affinity or anti-affinity rules, or to evacuate virtual machines from a host entering maintenance or standby mode.

- In order to simplify cluster management, vSphere 6.5 provides three new advanced options that provide simple customizations to DRS behavior to better suit varied cluster needs.

  - The **VM Distribution** option causes DRS to consider distributing VMs evenly across the hosts in the cluster for availability purposes.
  - The **Memory Metric for Load Balancing** option causes DRS to consider consumed memory usage for load balancing rather than its default of considering active memory usage.
  - The **CPU Over-Commitment** option allows you to specify how much CPU overcommitment (as a percentage of total cluster CPU capacity) DRS should consider. This can be useful when you need to consolidate your workloads for better utilization of hardware resources.

  These options are available in the vSphere Web Client (select a cluster, click the **Configure** tab, expand **Services**, select **vSphere DRS**, click the **Edit** button, and expand **Additional Options**).

  For more information about these options, refer to *vSphere 6.5 DRS Performance*.

- An advanced option, `AggressiveCPUActive`, affects how DRS predicts CPU demand of virtual machines in the DRS cluster.

  When `AggressiveCPUActive` is set to **`0`** (the default), DRS predicts CPU demand using the 5-minute average of CPU activity.

  When `AggressiveCPUActive` is set to **`1`**, DRS predicts CPU demand using the larger of *either* the 5-minute average of CPU activity *or* the 80th percentile of the last five 1-minute average values of CPU activity (in other words, the second highest 1-minute average).

  The more aggressive DRS behavior when this value is set to **1** can better detect spikes in CPU ready time and thus better predict CPU demand in some deployments.

- There are a variety of reasons that a DRS cluster might be shown as **Load imbalanced**. These include:

  - Migrations being filtered out due to affinity/anti-affinity rules.
  - Migrations being filtered out due to VM-host incompatibilities.

- The cost estimate for potential migrations (based on the costs of previous migrations) exceeding the expected benefits of the potential migrations.
- All the hosts in the cluster are network saturated.

If achieving a "load balanced" cluster is critical for your specific environment you can relax some rules or adjust the migration threshold.

On the other hand, if all the virtual machines are receiving 100% of their entitled resources, it might be acceptable to have a slightly imbalanced cluster.

- A set of scripts with which advanced DRS users can conduct more proactive cluster load balancing is available on the *Scripts for Proactive DRS* VMware community page, at http://communities.vmware.com/docs/DOC-10231.
- Starting with vSphere 6.5, predictive DRS can be configured to receive predictions from VMware vRealize® Operations™ and use this information to make load-balancing moves.

NOTE This requires vRealize Operations version 6.4 or later. For more detail, see https://blogs.vmware.com/management/2016/11/david-davis-vrealize-operations-post-34-new-predictive-drs-vrealize-operations-6-4.html.

For more information on DRS, refer to *vSphere 6.5 Resource Management*.

VMware Distributed Power Management (DPM)

VMware Distributed Power Management (DPM) conserves power when hosts in a cluster are underutilized. It does this by consolidating virtual machines to a subset of the hosts in the cluster, then putting the remaining hosts into standby mode. DPM keeps sufficient host capacity powered-on to meet the needs of the virtual machines in the cluster. When demand increases, DPM powers-on additional hosts and migrates virtual machines to them, keeping the cluster's load balanced across the powered-on hosts.

DPM is most appropriate for clusters in which composite virtual machine demand varies significantly 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 70) are also relevant to DPM.

NOTE While DPM powers-off hosts to save power, a very different power-saving technique, Host Power Management, can be used to reduce the power consumption of individual ESXi hosts while they are powered on. This is described in "Host Power Management in ESXi" on page 24.

DPM and Host Power Management can be used together for the best power conservation.

DPM Configuration and Modes of Operation

- DPM is complementary to host power management policies (described in "Host Power Management in ESXi" on page 24). DPM saves power at the cluster scale by putting underutilized hosts into standby mode, thus eliminating their idle power consumption. Host-level power management policies allow efficient use of the hosts in the cluster that remain powered on. Using DPM and host power management together can offer greater power savings than those obtained when either solution is used alone.
- The hosts in a DPM-enabled cluster inherit the automation level (automatic or manual) set at the cluster level. The automation level can also be set for individual hosts, overriding the inherited automation level. When a host is enabled for manual DPM, vCenter requests user approval before putting that host into or taking it out of standby mode.

  DPM has the most flexibility and the potential for maximum power savings when all hosts are enabled for automatic DPM. DPM also preferentially chooses hosts in automatic mode over hosts in manual mode to put into or take out of standby mode.
- If desired, DPM can also be disabled for individual hosts in a cluster even when DPM is enabled for that cluster. This might be done for hosts running mission-critical virtual machines, for example. VM/Host affinity rules can then be used to ensure that those virtual machines are not migrated away from these hosts.

Tuning the DPM Algorithm

- DPM considers historical demand in determining how much capacity to keep powered on and keeps some excess capacity available for increases in demand. DPM will also power on additional hosts as needed for unexpected increases in the demand of existing virtual machines or to allow new virtual machines to be admitted to the cluster.
- The aggressiveness of the DPM algorithm can be tuned by adjusting the **DPM Threshold** in the cluster settings menu. This parameter is similar to the DRS imbalance threshold in that it controls how aggressively DPM recommends migrations and putting hosts into and taking them out of standby mode. The default setting for the threshold is 3 (medium aggressiveness).
- Hosts are not put into standby mode unless the memory demand of the virtual machines in the cluster is low. Before vSphere 5.5, the memory demand of virtual machines was considered low when their active memory was low, even when their consumed memory was high.

vSphere 5.5 introduced a change in the default behavior of DPM designed to make the feature less aggressive. When estimating memory demand, DPM now takes into account a percentage of the idle-consumed memory (25% by default). This behavior can help prevent performance degradation for virtual machines when active memory is low but consumed memory is high.

If desired, this behavior can be adjusted using the variable in the advanced options for DPM, `PercentIdleMBInMemDemand`. As mentioned above, the default value of this variable is `25`. Setting it to `0` would revert to the more aggressive DPM behavior found in previous releases of vSphere. Setting it to `100` would make DPM very conservative by equating estimates of memory demand to consumed memory.

- For clusters that often have unexpected spikes in virtual machine resource demands, two variables in the advanced options for DPM can be used to set the minimum CPU and memory capacities that DPM will always keep powered on, `MinPoweredOnCpuCapacity` (default: `1`, unit: MHz) and `MinPoweredOnMemCapacity` (default: `1`, unit: MB).

Scheduling DPM and Running DPM Proactively

- DPM can be enabled or disabled on a predetermined schedule using **Scheduled Tasks** in vCenter Server. When DPM is disabled, all standby ESXi hosts in the 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 or to reduce the likelihood of some hosts being left in standby mode for extended periods of time.

  NOTE Similarly, when the vCenter server or the vCenter vpxd service is restarted, all standby ESXi hosts managed by that vCenter server will be powered on.

- When predictive DRS is enabled and is configured to receive predictions from vRealize Operations, DPM can make use of these predictions to proactively bring hosts back from standby mode even before the workload demand increases. For example, if workload demand increases every morning at 9am, DPM can use predictions to bring hosts out of standby by 8am to accommodate the expected demand.

  NOTE This requires vRealize Operations version 6.4 or later. For more detail, see https://blogs.vmware.com/management/2016/11/david-davis-vrealize-operations-post-34-new-predictive-drs-vrealize-operations-6-4.html.

- The VMware Community page *Scripts for "Proactive DPM"* (http://communities.vmware.com/docs/DOC-10230) provides a set of Perl scripts with which advanced DPM users can conduct more proactive power management.

Using DPM With VMware High Availability (HA)

- DPM respects VMware High Availability (HA) settings and takes them into account when making recommendations to put a host into or take it out of standby mode. In a cluster with HA enabled, DPM maintains excess powered-on capacity to meet the HA requirements.

  This could mean that additional virtual machines might not be admitted even if the cluster seems to have available resources (just as in a DRS cluster with HA enabled). Also, in order to reserve the additional capacity for HA, fewer hosts might be put into standby mode than if the cluster did not have HA enabled. These factors should be considered when configuring HA and DPM.

- If VMware HA is enabled in a cluster, and one or more virtual machines are powered on, DPM keeps a minimum of two hosts powered on. This is true even if HA admission control is disabled.

For more information on DPM performance tuning, see *VMware Distributed Power Management Concepts and Use*.

VMware vSphere Storage I/O Control

VMware vSphere® Storage I/O Control is a feature that allows an entire datastore's resources to be allocated as desired between the various virtual machines accessing the datastore. Beginning with vSphere 6.5, Storage I/O Control now works with SSD datastores.

NOTE To control how an individual ESXi host allocates the storage I/O resources it receives (rather than allocating an entire datastore's I/O resources), see "Storage I/O Resource Allocation" on page 35.

With Storage I/O Control disabled (the default), each host accessing a datastore gets a portion of that datastore's resources corresponding to the proportion of the datastore's total I/O workload coming from that host.

Even with Storage I/O Control enabled, no action is taken until Storage I/O Control is triggered, which happens automatically when datastore congestion is detected. Storage I/O Control tunes the trigger threshold to adapt to the datacenter environment in which it's running.

The datastore's I/O resources can be allocated based on the following settings:

- IOPS **shares** designate what proportion of a datastore's IOPS a virtual machine will receive. Shares are evaluated globally and the portion of the datastore's resources each host receives is the sum of the shares of the virtual machines running on that host divided by the sum of the shares of all the virtual machines accessing that datastore.
- IOPS **reservations**, new in vSphere 6.5, set a lower bound on the IOPS a virtual machine will receive. Storage I/O Control will guarantee this minimum provided the hardware can support it.
- IOPS **limits** set an upper bound on the IOPS a virtual machine will receive.

To enable Storage I/O Control, from the vSphere Web Client in the **Navigator** pane, select the **Storage** tab, select a datastore, select the **Configure** tab, select **General**, to the right of **Datastore Capabilities** click the **Edit...** button, select the **Enable Storage I/O Control** check box, then click **OK**.

For more information about Storage I/O Control performance, especially with SSD storage, see *Performance Implications of Storage I/O-Enabled SSD Datastores in VMware vSphere 6.5*.

VMware Storage Distributed Resource Scheduler (Storage DRS)

Storage Distributed Resource Scheduler (Storage DRS) provides I/O load balancing and space balancing across datastores within a datastore cluster as well as providing out-of-space avoidance. This can avoid storage performance bottlenecks or address them if they occur.

This section lists Storage DRS practices and configurations recommended by VMware for optimal performance.

- When deciding which datastores to group into a datastore cluster, try to choose datastores that are as homogeneous as possible in terms of host interface protocol (i.e., FCP, iSCSI, NFS), RAID level, and performance characteristics. We recommend not mixing SSD and hard disks in the same datastore cluster.
- Don't configure into a datastore cluster more datastores or virtual disks than the maximum allowed in *Configuration Maximums for vSphere 6.5*.
- While a datastore cluster can have as few as two datastores, the more datastores a cluster has, the more flexibility Storage DRS has to better balance that cluster's I/O load and capacity usage.
- When possible, make sure every datastore in a datastore cluster can be accessed by the full set of hosts that can access the other datastores in that cluster. Such full connectivity allows Storage DRS to make better decisions when addressing high I/O loads or out-of-space issues.
- As you add workloads you should monitor datastore I/O latency in the performance chart for the datastore cluster, particularly during peak hours. If most or all of the datastores in a datastore cluster consistently operate with latencies close to the congestion threshold used by Storage I/O Control (set to 30ms by default, but sometimes tuned to reflect the needs of a particular deployment), this might be an indication that there aren't enough spare I/O resources left in the datastore cluster. In this case, consider adding more datastores to the datastore cluster or reducing the load on that datastore cluster.

  NOTE Make sure, when adding more datastores to increase I/O resources in the datastore cluster, that your changes do actually add resources, rather than simply creating additional ways to access the same underlying physical disks.

- By default, Storage DRS affinity rules keep all of a virtual machine's virtual disks on the same datastore (using intra-VM affinity). However you can give Storage DRS more flexibility in I/O load balancing, potentially increasing performance, by overriding the default intra-VM affinity rule. This can be done for either a specific virtual machine or for the entire datastore cluster.
- Inter-VM anti-affinity rules can be used to keep the virtual disks from two or more different virtual machines from being placed on the same datastore, potentially improving performance in some situations. They can be used, for example, to separate the storage I/O of multiple workloads that tend to have simultaneous but intermittent peak loads, preventing those peak loads from combining to stress a single datastore.
- If a datastore cluster contains thin-provisioned LUNs, make sure those LUNs don't run low on backing disk space. If many thin-provisioned LUNs in a datastore cluster simultaneously run low on backing disk space (quite possible if they all share the same backing store), this could cause excessive Storage vMotion activity or limit the ability of Storage DRS to balance datastore usage.

VMware vSphere High Availability

VMware vSphere® High Availability (HA) minimizes virtual machine downtime by monitoring hosts, virtual machines, or applications within virtual machines, then, in the event a failure is detected, restarting virtual machines on alternate hosts.

VMware High Availability in General

- When vSphere HA is enabled in a cluster, all active hosts (those not in standby mode, maintenance mode, or disconnected) participate in an election to choose the primary host for the cluster; all other hosts become secondary hosts. The primary has a number of responsibilities, including monitoring the state of the hosts in the cluster, protecting the powered-on virtual machines, initiating failover, and reporting cluster health state to vCenter Server. The primary is elected based on the properties of the hosts, with preference being given to the one connected to the greatest number of datastores. Serving in the role of primary will have little or no effect on a host's performance.
- When the primary host can't communicate with a secondary host over the management network, the primary uses datastore heartbeating to determine the state of that secondary host. By default, vSphere HA uses two datastores for heartbeating, resulting in very low false failover rates. In order to reduce the chances of false failover even further—at the potential cost of a very slight performance impact—you can use the advanced option `das.heartbeatdsperhost` to change the number of datastores used for heartbeating (up to a maximum of five) and can configure vCenter Server to give preference to datastores that don't share a point of failure with the network used by HA.
- 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.

For further details about HA, see *vSphere Availability*.

Virtual Machine Component Protection (VMCP)

- Virtual Machine Component Protection (VMCP) is a feature that allows HA to detect failure of FC, iSCSI, or NFS storage connectivity and provide automated recovery for affected virtual machines. VMCP is disabled by default; when the feature is enabled, the HA agent consumes slightly more CPU cycles.

For further details about VMCP, including how to enable the feature, see the *vSphere Availability* guide for ESXi 6.5 and vCenter Server 6.5.

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 vSphere High Availability" on page 78) are relevant to FT as well.

When FT is enabled in a virtual machine, that virtual machine is called the FT primary virtual machine, or primary. Each such FT primary virtual machine is paired with another virtual machine, called the FT secondary virtual machine, or secondary. The roles of primary and secondary are dynamic: the virtual machines can swap roles when there is a failover.

- FT requires some additional resources. Before turning on FT for a virtual machine, make sure you have the following resources available:
  - **Storage**: The secondary virtual machine consumes storage for its configuration file, as well as its VMDK files, each of which are complete copies of the primary's VMDK files. Make sure you have space for the secondary virtual machine; this space can be on the same datastore as the primary or a different datastore. As long as a virtual machine is FT protected, changes to the primary's VMDK files will be mirrored to the secondary's VMDK files.
  - **Memory**: The primary and secondary virtual machines automatically receive a full memory reservation, ensuring ballooning or swapping are never necessary on either virtual machine. Make sure that the hosts on which the primary and secondary virtual machines will run have enough memory for this reservation.

    When the primary virtual machine is powered on, both the primary and secondary virtual machines consume additional overhead memory. The amount depends on virtual machine memory size, but is typically in the range of 1GB-2GB each.
  - **Network**: Make sure the FT logging traffic is carried by at least a 10Gb/s NIC. While the amount of network traffic depends on the workload, even one multi-vCPU virtual machine can require several Gb/s.
  - **CPU**: The secondary virtual machine requires some additional CPU cycles to support the synchronization between the primary and secondary. This is in proportion to how active the primary VM is, and is generally low.
- Before a virtual machine is FT protected, the primary and secondary must undergo an initial synchronization of their memory and VMDK states. This is done through a live memory and disk migration that occurs while the FT virtual machine is running. The disk synchronization, in particular, can be a lengthy operation, so expect a delay before it completes and the virtual machine becomes FT protected.

  The initial synchronization happens in the following cases. Because this process can be resource intensive, avoid performing these operations more often than necessary.
  - When FT is turned on for a running virtual machine.
  - When an FT virtual machine changes from powered-off to powered-on.
  - When the **Resume Fault Tolerance** operation is executed on a running FT virtual machine that has had the **Suspend Fault Tolerance** operation performed on it.
  - When FT protection is reestablished following the failure of either the primary or secondary. This could be caused by a hardware failure or intentionally triggered with the **Test Failover** or **Test Restart Secondary** operations. In either case the initial synchronization will execute to re-establish FT protection.
- The live migration that takes place for initial synchronization can briefly saturate the vMotion network link and can also cause spikes in CPU utilization.
- Avoid using the same network link for both FT logging traffic and other network traffic (such as vMotion traffic); one type of traffic can negatively affect the other, especially when multiple FT virtual machines are running on the same host.

- FT logging traffic is asymmetric; the majority of the traffic flows from primary to secondary. Congestion of logging traffic can therefore be reduced by distributing primaries across multiple hosts. For example on a cluster with two ESXi hosts and two FT virtual machines, placing one of the primary virtual machines on each of the hosts allows FT logging traffic to utilize the network bandwidth bidirectionally.
- Avoid placing more than four FT virtual machines on a single host, or having more than eight total FT vCPUs on a host. In addition to reducing the likelihood of saturating the FT logging NIC, 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 runs out of resources (such as FT logging bandwidth, storage bandwidth, or CPU cycles) that the primary virtual machine has plenty of, then ESXi might slow the primary to allow the secondary to keep up.

  For example, if the host on which the secondary is running also has many other secondaries saturating the FT logging bandwidth, but the host on which the primary is running has only one FT virtual machine, then that primary might be slowed to accommodate the lack of resources on the secondary. The following recommendations help avoid this situation:

  - Make sure the hosts on which the primary and secondary virtual machines run are relatively closely matched in terms of CPU performance, virtual machine load (including and especially FT virtual machine load), FT logging bandwidth and load, and datastore bandwidth and load. Bottlenecks in any of these areas on the primary or secondary can be detrimental to the performance of the other virtual machine.
  - Make sure that power management scheme settings (both in the BIOS and in ESXi) 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.
- When FT is enabled for a virtual machine, that virtual machine can use only hardware CPU and MMU virtualization (for information about hardware-assisted virtualization, see "Hardware-Assisted Virtualization" on page 11 and "Configuring ESXi for Hardware-Assisted Virtualization" on page 23).
- Although FT will work on many earlier generation CPUs (for minimum requirements, see the *vSphere Availability* guide for ESXi 6.5 and vCenter Server 6.5), for the best performance we recommend the following CPUs:
  - Intel Platform:
    Intel "Haswell" generation (such as Intel Xeon 2600-v3 Series) or later.
  - AMD Platform:
    AMD "Piledriver" generation (such as AMD Opteron 6300 Series) or later.

VMware vSphere Update Manager

VMware vSphere 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, VMware Tools and virtual hardware, and so on.

In addition to the material presented here, more information about vSphere Update Manager performance can be found in the white paper *VMware vSphere Update Manager Performance and Best Practices*.

Update Manager Deployed in Windows

For deployments of vCenter Server in Windows, Update Manager can be run either on the same host as the vCenter Server or on a different host and can be configured to share a database with vCenter Server or to have its own database. The following recommendations apply to these Windows deployments:

- When there are more than 300 virtual machines or more than 30 hosts, separate the Update Manager database from the vCenter Server database.
- When there are more than 1000 virtual machines or more than 100 hosts, separate the Update Manager server from the vCenter Server *and* separate 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 ESXi hosts.
- In order to cache frequently used patch files in memory, make sure the Update Manager server host has at least 2GB of RAM.

Update Manager Deployed in Linux (with vCenter Server Appliance)

Beginning with vSphere 6.5, vCenter Server Appliance (vCSA) deployments share both a Linux host and a database with Update Manager. The Update Manager service `updatemgr` is enabled and started by default once the appliance starts. The following recommendations apply to these Linux deployments:

NOTE Despite the shared resources, Linux deployments of Update Manager support the same maximum configurations as Windows deployments.

- Make sure the database has enough space for both the vCenter inventory and the Update Manager data. Update Manager initialization needs about 150MB, and its database usage can increase by up to about 100MB per month.
- The amount of memory consumed by Update Manager is primarily affected by the size of the inventory, especially the number of virtual machines. Periodically monitor the resource usage by Update Manager, as well as other services, using `top`, `htop`, or other tools, especially when the system is under heavy load. Update the resources or configuration when necessary.
- An Update Manager plug-in (about 6MB in size) will be automatically installed into the vSphere Web Client the first time the Web Client is connected to a vCenter Server Appliance running the Update Manager service. (In contrast, this plug-in must be manually installed when the Web Client is used with Update Manager in Windows.)

Update Manager General Recommendations

- 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.
- Upgrading VMware Tools is faster if the virtual machine is already powered on. Otherwise, Update Manager must power on the virtual machine before the VMware Tools upgrade, which could increase the overall latency.

- Upgrading virtual machine hardware is faster if the virtual machine is already powered off. Otherwise, Update Manager must power off the virtual machine before upgrading the virtual hardware, which could increase the overall latency.

  NOTE Because VMware Tools must be up to date before virtual hardware is upgraded, Update Manager might need to upgrade VMware Tools before upgrading virtual hardware. In such cases the process is faster if the virtual machine is already powered-on.

Update Manager Cluster Remediation

- Limiting the remediation concurrency level (i.e., the maximum number of hosts that can be simultaneously updated) to half the number of hosts in the cluster can reduce vMotion intensity, often resulting in better overall host remediation performance. (This option can be set using the cluster remediate wizard.)
- When all hosts in a cluster are ready to enter maintenance mode (that is, they have no virtual machines powered on), concurrent host remediation will typically be faster than sequential host remediation.
- Cluster remediation is most likely to succeed when the cluster is no more than 80% utilized. Thus for heavily-used clusters, cluster remediation is best performed during off-peak periods, when utilization drops below 80%. If this is not possible, it is best to suspend or power-off some virtual machines before the operation is begun.

Update Manager Bandwidth Throttling

- During remediation or staging operations, hosts download patches. On slow networks you can prevent network congestion by configuring hosts to use bandwidth throttling. By allocating comparatively more bandwidth to some hosts, those hosts can more quickly finish remediation or staging.
- To ensure that network bandwidth is allocated as expected, the sum of the bandwidth allocated to multiple hosts on a single network link should not exceed the bandwidth of that link. Otherwise, the hosts will attempt to utilize bandwidth up to their allocation, resulting in bandwidth utilization that might not be proportional to the configured allocations.
- Bandwidth throttling applies only to hosts that are downloading patches. If a host is not in the process of patch downloading, any bandwidth throttling configuration on that host will not affect the bandwidth available in the network link.

VMware Virtual SAN (vSAN)

The VMware Virtual SAN (vSAN) allows storage resources attached directly to ESXi hosts to be used for distributed storage and accessed by multiple ESXi hosts. Follow the recommendations in this section for the best performance.

Hybrid versus All-Flash vSAN

vSAN deployments can be "hybrid" or all-flash:

- In hybrid vSAN deployments, a layer of fast SSD storage is used as a "caching tier" to provide a read and write cache for a larger (but slower) magnetic disk layer (the "capacity tier").
- In all-flash vSAN deployments, SSD storage is used for both the caching tier (which, in all-flash vSAN, provides just a write cache) and the capacity tier (which provides the persistent storage).

While hybrid vSAN deployments are very fast, all-flash deployments are faster still. In addition, all-flash vSAN supports RAID-5, RAID-6, deduplication, and compression. By reducing storage requirements, these features can reduce the effective cost difference between hybrid and all-flash vSAN.

vSAN Hardware Selection and Layout

Hardware Selection and Layout for Hybrid vSAN

- In a hybrid vSAN, all writes go first to the SSDs that make up the caching tier and vSAN read cache hits (which ideally make up a large proportion of the reads) also come from that tier.

  Thus the caching tier (SSD) to capacity tier (HDD) ratio has a significant effect on hybrid vSAN performance. A higher SSD to HDD ratio typically improves performance, but at a higher cost.

Hardware Selection and Layout for All-Flash vSAN

- In an all-flash vSAN, all vSAN writes go first to the caching tier SSDs and are then destaged to the capacity tier SSDs; reads come directly from the capacity tier SSDs (except reads of data not yet destaged, which come from the caching tier SSDs).

  The performance of the caching tier SSDs is an important factor in the overall performance of an all-flash vSAN. Using faster, advanced SSDs — such as NVMe disk drives — for the caching tier can significantly improve performance, even when the capacity tier uses lower-performance SSDs.

Hardware Selection and Layout for vSAN in General

- vSAN performs best when disk resources are distributed relatively evenly across the ESXi hosts that make up the vSAN cluster.
- Disks in a vSAN are organized into disk groups, with each disk group consisting of one cache disk and one or more capacity disks. Generally, the more disk groups per host, the better vSAN performs.

vSAN Network Considerations

- While small vSAN deployments can perform well with 1Gb/s Ethernet links between the ESXi hosts in the vSAN cluster, most deployments will perform best with 10Gb/s or faster links.
- Jumbo frames, while they shouldn't hurt performance if properly configured, won't provide much improvement in vSAN performance.

For more information about vSAN network configuration, see the *VMware Virtual SAN 6.2 Network Design Guide* (though written for vSAN 6.2, most of the content applies to vSAN 6.5).

vSAN Configuration and Use

- A number of the VM Storage Policies in vSAN can affect vSAN performance. These include:

- Number of disk stripes per object
- Flash Read Cache reservation
- Number of failures to tolerate
- Object space reservation

In most cases, these options allow a trade-off between resource usage and performance. These are described in more detail in the document referenced below.

- End-to-end software checksum is enabled by default for vSAN, but can be disabled on a per virtual machine or per object basis. Disabling vSAN checksum will typically only be done when checksum functionality is already provided by an application layer. While the checksum operation does introduce some overhead, its performance impact is small, due in part to a dedicated in-memory cache.
- Just as disk resources should generally be distributed relatively evenly across vSAN hosts, for the best performance virtual machines should also be distributed relatively evenly across those hosts. VMware DRS can help to achieve this (see "VMware Distributed Resource Scheduler (DRS)" on page 70).
- Deduplication and compression, a pair of features available only on all-flash vSAN deployments, can dramatically reduce storage requirements, but with a moderate performance penalty. Because the performance impact is least significant with low-throughput workloads and small block sizes, workloads with these characteristics are especially well-suited to use this feature.

  For further information about vSAN performance with deduplication and compression, see the *VMware Virtual SAN 6.2 Performance with Online Transaction Processing Workloads* paper (though written for vSAN 6.2, most of the content applies to vSAN 6.5).
- RAID-5 and RAID-6 (also known collectively as erasure coding), features available only on all-flash vSAN deployments, can provide data protection with a relatively small performance penalty (due to high throughput capability of the all-flash storage). Replication (that is, RAID-1), however, still offers the best performance, but at the cost of significantly lower space efficiency.

  For further information about vSAN performance with erasure coding, see the *VMware Virtual SAN 6.2 Performance with Online Transaction Processing Workloads* paper (though written for vSAN 6.2, most of the content applies to vSAN 6.5).

For more details about vSAN configuration options and other vSAN performance tips, see the *VMware Virtual SAN 6.2 Network Design Guide* (though written for vSAN 6.2, most of the content applies to vSAN 6.5) and the *VMware Virtual SAN Design and Sizing Guide*.

vSAN Encryption

New in vSphere 6.5, vSAN Encryption encrypts data when it is written to a storage device, but transfers it unencrypted. (To encrypt the data *before* it is transferred, see "vSphere Virtual Machine Encryption Recommendations" on page 36.) This allows vSAN to deduplicate and/or compress the data, operations that are difficult or impossible to perform on encrypted data.

- The resource overhead of vSAN encryption is primarily in CPU utilization. Thus, with sufficient CPU resources, typical deployments (that is, those that don't use ultra-fast storage) will see no significant performance impact.
- To significantly minimize additional CPU utilization due to VM encryption, choose processors that support AES-NI (see "AES-NI Support" on page 12), especially some recent processor versions in which the AES-NI implementation is further optimized.
- Make sure AES-NI is enabled in BIOS. In some cases a BIOS upgrade might be required for AES-NI support.

VMware Virtual Volumes (VVols)

VMware Virtual Volumes (VVols) uses communication between VMware software and SAN or NAS storage arrays to allow finer control over virtual machine storage policies.

In addition to the recommendations in this section, it's also a good idea to read your storage vendor's documentation, as each vendor might have their own recommendations or best practices.

VVol Hardware Considerations

- VVols work only on storage hardware that supports vStorage APIs for Storage Awareness (VASA) version 2.0 or later with support for the Virtual Volumes API profile.
- VVol performance varies significantly between storage hardware vendors. Before choosing storage hardware, confirm that it will provide the VVol performance you expect.
- When choosing hardware, be mindful of the limitations imposed by the array implementation. A VVol-based VM, for instance, requires a number of VVols (config, swap, data, potentially snapshots, etc.) and the limit on the number of VVols an array can manage might be as important as the limit on its storage capacity.
- VVols require a VASA provider (sometimes called a "storage provider" or abbreviated "VP"), a software layer that presents the storage array's capabilities to the ESXi host and allow ESXi to perform virtual machine and VVol-related management operations. For some storage arrays the VASA provider runs on the array, for others it runs in a virtual machine or on a dedicated physical server.

  NOTE If your VASA provider runs in a virtual machine:

  - Be careful not to migrate it to VVol storage.
  - Consider using vSphere HA (see "VMware vSphere High Availability" on page 78) to protect it.
  - Implement an appropriate backup plan for it.

- VVol performance is heavily dependent on bandwidth between the ESXi host, the VASA provider, and the array. For the best performance, I/O operation traffic and management operation traffic should be on separate links, with the I/O link having at least 10Gb/s bandwidth.

  If I/O operation traffic and management operation traffic must share the same physical network link, they should be routed through separate virtual NICs.

VVol Workload Performance

VVol workloads fall into two major categories: management workloads and I/O workloads.

VVol Management Operation Performance

The following performance recommendations apply to VVol management operations:

- Virtual machine provisioning operations on VVol range in performance relative to native NFS or SAN:
  - Clone operations on VVol are significantly faster than on native NFS or SAN. Clones are offloaded to arrays and can be implemented by the array in a space-efficient manner from the moment they're created (since the array is free to share the underlying storage blocks).
  - Power operations (power on and power off) on VVol perform similarly to native NFS or SAN.
  - Destroy operations on VVol typically take slightly longer than on native NFS or SAN, though this varies somewhat with different array vendors.
- Storage migration performance between two VVol datastores is faster than migration involving one or more non-VVol datastores.
- VVol performance on snapshot operations varies between creation and deletion:

- Snapshot creation performs similarly between VVol and native NFS or SAN.
- Snapshot deletions on VVols are offloaded to the array and, since there's no requirement to re-issue writes previously directed to the redo log file, they will typically be of more predictable duration and faster than on native NFS or SAN.

VVol I/O Operation Performance

The following performance recommendations apply to VVol I/O operations:

- Because the I/O path of VVol is nearly the same as the I/O path for non-VVol datastores, the throughput and latency on a VVol datastore is very similar to that for an otherwise-similar non-VVol datastore.
- I/O to VVols is done through Protocol Endpoints (PEs), and vendors might have specific advice on configuring the number or type of PE LUNs for their array. There's little general advice that can be given on this topic, since the performance aspects rely heavily on the particular array implementation (for example, queueing of commands at the PE level vs. dispatching them directly to a per-VVol queue, depth of queues on the array side, how does this interact with any QoS features, etc.). Note that the Core Storage layer permits four times more I/Os to be enqueued for Protocol Endpoint (PE) LUNs than for regular (data) LUNs.
- Snapshots on VVols are offloaded to the array, which creates the snapshot using array-side native means, such as automatically doing its own Copy-on-Write (COW). VVol snapshots typically become comparable to LUN snapshots on the array. For this reason, VVol performance on disks with snapshots doesn't degrade as snapshot levels increase, as it does on native NFS or SAN (where vSphere creates and manages snapshots itself using redo log files). This can lead to VVols having dramatically better performance for snapshots than native NFS or SAN.

  This means that consolidating a snapshot is now a fast operation, even if a VM has been running with a snapshot in place, since the array usually doesn't require the data accumulated in a redo log to be re-written to the original disk. It also means you can use snapshots liberally (for backups on running VMs, for example) since the array handles the copy-on-write or other operations needed to maintain the snapshot, and the I/O from the VM to the disk is therefore unaffected. There's no longer a performance penalty for running a VM with a snapshot in place.

VVol Configuration Recommendations

- Because with VVols each of your virtual machines' virtual disks is stored as an individual data virtual volume, it becomes possible to assign distinct Storage Policy Based Management (SPBM) storage polices to each virtual disk, thus allowing the performance of each disk to be optimized. For example, a database disk might require a different storage policy than a database log disk.

  This also allows services to be enabled for specific virtual disks. For example, it might make sense to enable de-duplication (if available on your array) on a system disk but not on a picture storage disk.

  For this reason it's a good idea to look into the exact capabilities offered by the array you're using. Some might offer QoS on individual VVol disks, some might offer specific data services (e.g., de-duplication, compression, encryption) on individual VVol disks, and so on, all configured through SPBM.
- VVol storage containers are not physical objects in the storage array; they're simply an allocation quota of certain types of storage with certain capabilities enabled. Because you can change the size limit of a container at any time, there's no need to create LUNs that are somewhat larger than your actual anticipated needs. Instead a storage administrator can create VVol storage containers of the exact size limit desired, then later change that limit without the need to migrate the VVols, reformat the storage container, or take any other action visible from the consumer side.

For this reason it's also better to structure storage containers on logical, organizational/management boundaries instead of on the basis of LUN configuration. There's no need to have separate RAID-1/RAID-5/RAID-6 LUNs or (with a storage array supporting VVol 2.0) LUNs configured for replication and LUNs that are not replicated. Instead, it's better to offer all these storage types in the single storage container for a management unit (say, "Finance" or "HR") so that changing storage types (say, moving from RAID-1 to RAID-6 with replication as a VM moves from development to production) are simply back-end operations on the array and not storage migrations between LUNs.

VMware vCenter Single Sign-On Server

The VMware vCenter Single Sign-On Server offers users single sign-on access across the vSphere management stack. Follow the recommendations in this section for the best performance.

- Single Sign-On Server performance depends on the performance of the back-end identity sources. For the best identity source performance, follow the relevant best practices. For example:
  - For Windows Active Directory, see the Windows Server 2008 performance tuning guidelines: http://msdn.microsoft.com/en-us/windows/hardware/gg463394
  - For OpenLDAP, see the OpenLDAP Faq-O-Matic: Performance Tuning: http://www.openldap.org/faq/data/cache/190.html
- When using Active Directory as the identity source, we recommend using native AD rather than AD-over-LDAP. When using native AD, the Single Sign-On Server stack can process groups much more efficiently than when using AD-over-LDAP.
- In order to minimize search overhead, make sure only the necessary identity source servers are added to the Single Sign-On Server.

  You can browse and manage the identity sources attached to the vSphere Single Sign-On Server using the vSphere Single Sign-On Configuration user interface in the vSphere Web Client.
- When using the vSphere Web Client to add an Identity Source to the Single Sign-On Server, setting **Base DN for Users** and **Base DN for Groups** to the common portion of the domain name for users and groups, respectively, will significantly increase the performance of browsing and searching for users in large identity sources.
- For higher throughput and higher availability, consider deploying multiple vCenter Single Sign-On Servers behind a network load balancer.

For more information about deploying and configuring Single Sign-On Server, see the *VMware vCenter Server 6.0 Deployment Guide*. (Though this paper is about vCenter Server 6.0, the Single Sign-On Server information it contains applies to vCenter Server 6.5 as well.)

VMware vSphere Content Library

The VMware vSphere Content Library provides an easy way for vSphere administrators to manage virtual machine templates, vApps, ISO images, and scripts. Follow the recommendations in this section for the best performance.

- For the best library sync file transfer performance and reduced load on the vCenter servers, consider configuring your vCenter servers in Enhanced Linked Mode.

  When transferring content between Content Libraries on vCenter servers that are joined using Enhanced Linked Mode, files stored on datastores mounted on ESXi hosts are transferred between those ESXi hosts instead of between the vCenter servers. This significantly shortens the path taken by the data, improving performance and reducing load on the vCenter servers.

- For the best performance, place the Content Library on a datastore that supports VMware vStorage APIs for Array Integration (VAAI) (for more information about VAAI, see "Hardware Storage Considerations" on page 13).

  VAAI-capable datastores allow many Content Library operations, such as creating new virtual machines from Content Library templates, to take place largely on the datastore, dramatically improving performance while also freeing CPU and I/O resources on the hosts.

- The vCenter Server Appliance offers a reverse proxy that handles incoming HTTPS connections, offloading from the vCenter web server the process of encrypting and decrypting HTTPS traffic.

  In environments where the additional security provided by HTTPS connections is not required, file transfer speed from the Content Library can be significantly increased by using plain HTTP (instead of HTTPS), or by disabling the reverse proxy.

  NOTE Making either of these changes (using HTTP instead of HTTPS or disabling the reverse proxy) have the potential to expose sensitive data or metadata. Only make these changes if you are confident that you know the risks.

- Synchronizing content across a WAN can be slow, due to both the speed of the WAN and the use of HTTP.

  If you have subscribed content libraries that frequently synchronize content across a WAN, consider creating a mirror server to cache files at the remote site. This can significantly decrease user wait time while also avoiding transferring the same files multiple times across the slow network.

- The Content Library provides an option allowing custom transfer of content into a new library. This feature, called Custom Replication Support, can be useful if your storage infrastructure supports hardware-based replication, if your network bandwidth is insufficient to sync data over HTTPS or NFS, or if you want to make a copy that can be physically transferred to a different location.

- Because the Content Library shares a network link with other vCenter components, and perhaps with applications, you might want to limit the Content Library's network usage in order to leave sufficient network bandwidth for those other components of applications. This can be accomplished using the Content Library's global network throughput throttling control for file transfer bandwidth. This setting affects the network usage of all Content Library file transfer operations, including library sync, deploy VM, capture VM, file download, and file upload.

  To configure this setting: From the vSphere Web Client, use **Administration** > **System Configuration** > **Services** > **Transfer Service** > **Maximum Bandwidth Consumption**.

- The Content Library has two settings that limit the maximum number of simultaneous content transfers:

  - **Library Maximum Concurrent Sync Items** is the maximum number of sync threads all subscribed libraries can simultaneously have in a Content Library service instance to transfer content from published libraries.

  - **Maximum Number of Concurrent Transfers** is the maximum total number of file transfers allowed on a Content Library service instance. Content transfers include synchronizations, upload/download, library item copy, OVF template deployment from content library, and so on.

If you have a high speed network, or if the network throughput of your Content Library file transfers is lower than expected, increasing these maximum settings might increase the network throughput for Content Library file transfers.

Glossary

A

Active memory
Active memory is an estimate of the amount of guest memory that has been accessed (either read from or written to) within the past one minute. See also consumed memory.

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 AMD processors, also called AMD I/O Virtualization or IOMMU. See also I/O MMU.

B

Ballooning
A technique used in VMware ESXi 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.

Consumed memory
Consumed memory, in the context of a virtual machine, is the amount of host memory used to back the guest memory, whether or not that memory is active. Consumed memory, however, does not include overhead memory. See also active memory.

Core
A processing unit. Often used to refer to multiple processing units in one package (a so-called "multi-core CPU").

D

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

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 Scheduler (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 ESXi. The E1000 adapter emulates an Intel E1000 device. See also E1000e, Vlance, and VMXNET.

E1000e
One of the virtual network adapters available in a virtual machine running in ESXi. The E1000e adapter emulates an Intel E1000e device. See also E1000, Vlance, and VMXNET.

Enhanced VMXNET
A virtual network adapter type. See VMXNET2.

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

F

Fault Tolerance (FT)
A feature 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 a 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 a hypervisor. 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 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).

Host Power Management
Host power management reduces the power consumption of ESXi hosts while they are running. See also Distributed Power Management.

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, supported by some 64-bit Intel processors, also called Intel Virtualization Technology for Directed I/O. See also I/O MMU.

I/O MMU
The input/output memory management unit is a processor feature that remaps I/O DMA transfers and device interrupts. This feature 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. Jumbo frames are supported by the VMXNET2 and the VMXNET3 virtual network adapters (but not by the normal VMXNET adapter). Not all Ethernet cards or switches support jumbo frames, however.

K

Kernel
The heart of an operating system. The kernel usually includes the functionality of a Hardware Abstraction Layer (HAL).

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. Though some processors support 1GB memory pages, in this book the term "large pages" is used only to refer to 2MB pages.

Linked Clone
A copy of a virtual machine that must have access to the parent virtual machine's virtual disk(s). The linked clone stores only the differential changes to the parent's 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

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

MMU (Memory Management Unit)
Part of a computer's hardware that acts as an interface between the CPU core 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 10Gb/s 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 feature that allows the allocation of network bandwidth to network resource pools. These pools can be selected from among the seven predefined pools or can be user-defined.

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 might 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. See Hardware Virtual MMU.

P

Pacifica
The original name for AMD's version of virtualization assist (later called AMD-V), included in some 64-bit AMD processors. See AMD Virtualization.

Paravirtualization
A technique in which the guest operating system, or some component of the guest operating system (such as a device driver) interacts with the hypervisor directly, for better guest-host communication and for improved performance for some operations.

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

PCI-X (PCI Extended)
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 paravirtualized virtual storage adapter using the SCSI protocol. PVSCSI offers a significant reduction in CPU utilization as well as potentially increased throughput compared to the default virtual storage adapters. See Paravirtualization.

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. See Hardware Virtual MMU.

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.

Shadow Page Tables
A set of page tables maintained by ESXi 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.

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

SplitRx Mode
A feature in ESXi that can significantly improve network performance for some workloads.

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 DRS
A feature that provides I/O load balancing across datastores within a datastore cluster. This load balancing can avoid storage performance bottlenecks or address them if they occur.

Storage I/O Control
A feature that allows an entire datastore's I/O resources to be allocated among 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.

Swap to host cache
A feature in ESXi that uses a relatively small amount of solid-state drive (SSD) storage to significantly reduce the performance impact of host-level memory swapping.

Symmetric Multiprocessor (SMP)
A multiprocessor architecture in which two or more processors (cores, to use current terminology) 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 template is a primary copy of a virtual machine that can be used to create and provision other virtual machines. Templates can't 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 inadvertently disabled.

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, E1000e, VMXNET2, and VMXNET3 virtual network adapters (but not by the normal VMXNET adapter).

U

Uniprocessor (UP)
A single-processor architecture (single-core architecture, to use current terminology). See also Symmetric Multiprocessor (SMP).

V

Vanderpool
The original name for Intel's version of virtualization assist (later called VT), included in some 64-bit Intel processors. See also Virtualization Technology.

Virtual CPU (vCPU)
A processor within a virtual machine. See also Symmetric Multiprocessor (SMP).

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 NUMA (vNUMA)
A feature in ESXi that exposes NUMA topology to the guest operating system, allowing NUMA-aware guest operating systems and applications to make the most efficient use of the underlying hardware's NUMA architecture.

Virtual SMP
Multiple virtual CPUs (vCPUs) 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 is required in order for 64-bit operating systems to be run in virtual machines and which can improve the performance of both 32-bit and 64-bit guest operating systems. 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 might 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 ESXi. 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, E1000e, and VMXNET.

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

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/ESXi hosts or vCenter servers. Renamed vSphere Client in vSphere 4.0.

VMware vSphere 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, VMware Tools, virtual hardware, virtual appliances, and so on.

VMware vStorage APIs for Array Integration (VAAI)
A set of APIs that can improve storage scalability by offloading to VAAI-capable storage hardware a number of operations instead of performing those operations in ESXi.

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.

VMX Swap
A feature allowing ESXI to swap to disk some of the memory it reserves for the virtual machine executable (VMX) process.

VMXNET
One of the virtual network adapters available in a virtual machine running in ESXi. The VMXNET adapter is a high-performance paravirtualized device with drivers (available in VMware Tools) for many guest operating systems. See also VMXNET2, VMXNET3, E1000, E1000e, Vlance, and NIC Morphing.

VMXNET2

Also called Enhanced VMXNET; one of the virtual network adapters available in a virtual machine running in ESXi. The VMXNET2 adapter is a high-performance paravirtualized device with drivers (available in VMware Tools) for many guest operating systems. See also VMXNET, VMXNET3, E1000, E1000e, Vlance, and NIC Morphing.

VMXNET3 (VMXNET Generation 3)

The latest in the VMXNET family of paravirtualized network drivers. Requires virtual hardware version 7 or later.

vSphere Web Client.

A browser-based user interface used to manage ESX/ESXi hosts and vCenter servers.

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

W

X