Performance Best Practices for VMware vSphere 8.0 Update 3

VMware ESXi 8.0 Update 3
vCenter Server 8.0 Update 3

vmware®
by Broadcom

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

https://docs.vmware.com/

The VMware website also provides the latest product updates.

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

docfeedback@vmware.com

Revision: 20240910

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

Contents

Tables

About This Book

This book, *Performance Best Practices for VMware vSphere 8.0 Update 3*, provides performance tips that cover the most performance-critical areas of VMware vSphere® 8.0 Update 3. It is not intended as a comprehensive guide for planning and configuring your deployments.

This book consists of the following chapters:

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 25, provides guidance regarding VMware ESXi™ software and the virtual machines that run in it.

Chapter 3, "Guest Operating Systems," on page 57, provides guidance regarding the guest operating systems running in vSphere virtual machines.

Chapter 4, "Virtual Infrastructure Management," on page 69, provides guidance regarding infrastructure management best practices.

NOTE For planning purposes we recommend reading this entire book before beginning a deployment. Material in the last chapter, Virtual Infrastructure Management, for example, might influence your hardware choices (covered in the first chapter, Hardware for Use with VMware vSphere).

Intended Audience

This book is intended for system administrators who are planning a VMware vSphere 8.0 Update 3 deployment and want to maximize its performance. The book assumes the reader is already familiar with VMware vSphere concepts and terminology.

VMware vSphere Documentation and Resources

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

http://docs.vmware.com

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

https://www.vmware.com/techpapers

You can read performance-related blog posts at the VMware VROOM! Performance blog:

https://blogs.vmware.com/performance

Hardware for Use with VMware vSphere 1

This chapter provides guidance on selecting and configuring hardware for use with VMware vSphere 8.0 Update 3.

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.

CPU Hardware Considerations

This section provides guidance regarding CPUs for use with vSphere 8.0 Update 3.

General CPU Considerations

When selecting hardware, it's 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 77, "VMware Distributed Resource Scheduler (DRS)" on page 80, and "VMware Fault Tolerance" on page 91.

Side-Channel Vulnerabilities

A class of security vulnerabilities collectively known as "side-channel vulnerabilities" have been discovered in many modern CPUs. These include vulnerabilities commonly called Spectre, Meltdown, Foreshadow, L1TF, and others. Some of the mitigation measures for these vulnerabilities can have a significant performance impact. Because of the complexity of this topic, as well as its ever-changing nature, we don't thoroughly address it in this book. For the latest information about these vulnerabilities as they relate to VMware products, see VMware KB articles 52245, 54951, and 55636, all of which are updated as needed.

These vulnerabilities can be mitigated in various ways and in various parts of the system. Depending on the variant of the vulnerability, mitigations might take place in hardware, through microcode, or in software.

Hardware Mitigations

Some recent CPU releases include hardware mitigations that can address some of these vulnerabilities with little or no performance impact. Thus, in addition to any performance gain from other aspects of these CPUs, choosing them for a new system might deliver even greater performance gains when compared to older processors that require microcode or software mitigations for certain vulnerabilities.

Microcode Mitigations

Some side-channel vulnerabilities can be mitigated in microcode. Microcode is a layer of code that runs on a CPU below the CPU's externally-visible instruction set. The microcode that ships with a CPU can be updated in the field either through BIOS updates or by an operating system.

Some versions of ESXi can update microcode to mitigate certain side-channel vulnerabilities.

Software Mitigations

Some side-channel vulnerabilities can be mitigated in software. Depending on the vulnerability, these mitigations might be at the hypervisor level or at the guest operating system level. These mitigations are addressed in "Side-Channel Vulnerability Mitigation in ESXi" on page 27 and "Side-Channel Vulnerability Mitigation in Guest Operating Systems" on page 59.

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 the white paper Software and Hardware Techniques for x86 Virtualization.

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, allowing trap-and-emulate style virtualization as well as providing assists to reduce the overhead involved in handling these traps.

NOTE Starting with version 6.7, ESXi no longer supports software CPU virtualization.

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.

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 in which guest operating systems 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" on page 37.

NOTE Starting with version 6.7, ESXi no longer supports software MMU virtualization.

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 49 and "Single Root I/O Virtualization (SR-IOV)" on page 49.

AES-NI Support

vSphere 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 activated in BIOS.

If a host that supports AES-NI, that support is exposed by default to virtual hardware version 7 or later 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 deactivating 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 45, "VMware vMotion Recommendations" on page 77, and "VMware vSAN" on page 93.

Memory Hardware Considerations

This section provides guidance regarding memory for use with vSphere 8.0 Update 3.

Persistent Memory (PMem)

Persistent memory, or PMem, is a type of non-volatile memory that fits in standard server DIMM slots but has various benefits over DRAM. Depending on the type of PMem, and how it's configured, these benefits can include some combination of:

- Persistence across reboots
- Higher capacities than DRAM
- Lower cost per GB than DRAM

Various types of PMem are available, including Intel® Optane™ Persistent Memory Modules and NVDIMM-N, both of which are discussed below, and each of which can be used only in servers designed to support it. Optane PMem can be configured in either of two modes: Memory Mode and App Direct Mode.

For information about PMem in NUMA systems, see "Persistent Memory (PMem) in NUMA Systems" on page 31. For information about using PMem in ESXi, see ""Persistent Memory (PMem)" on page 38.

Intel Optane Persistent Memory Modules

Optane PMem is a type of PMem available in larger capacities and at lower cost per GB than volatile DRAM, but with slightly slower access speeds. The higher capacities available with Optane PMem can potentially allow more total memory per system.

For information about vSphere support for Intel Optane Persistent Memory, see VMware KB article 67645.

For guidance on sizing Optane PMem, see Announcing VMware vSphere Support for Intel® Optane™ Persistent Memory Technology.

Optane Persistent Memory Modes

As mentioned above, Optane PMem can be configured in either of two modes, Memory Mode or App Direct Mode, or partially in each mode.

- In Memory Mode, the Optane PMem presents to the operating system as system main memory. In this mode, the system's DRAM is used as a cache for the larger (but slower) Optane PMem. This mode does not require changes to the operating system or application. Using Memory Mode can sometimes achieve higher VM density than a DRAM system of comparable cost. In this mode, the performance of a workload depends on the size of the working set in relation to the size of the DRAM cache and on the workload's access pattern.

  For performance insights about Optane PMem in Memory Mode, see the white paper Intel Optane DC Persistent Memory "Memory Mode" Virtualized Performance Study.

- In App Direct Mode the Optane PMem presents to the operating system as persistent memory, allowing a PMem-aware operating system (such as ESXi 8.0 Update 3) to use it directly. This in turn allows both legacy applications and new PMem-aware applications to use the Optane PMem in novel ways.

  An ESXi virtual machine can use Optane PMem in App Direct Mode as a regular disk attached to a virtual machine. This is called vPMEMDisk. In this case, there is no need for the guest operating system to be PMem aware.

  An ESXi virtual machine running on virtual hardware version 14 or later can also use Optane PMem in App Direct Mode as a virtual NVDIMM device. This is called vPMEM. In this configuration, unlike with vPMEMDisk, the ESXi storage stack is bypassed. In order to use a virtual NVDIMM device, the guest operating system must be PMem aware.

  For more performance information about Optane PMem in App Direct Mode, see the white paper Persistent Memory Performance in vSphere 6.7 with Intel Optane DC Persistent Memory.

To get the best performance from PMem, we recommend configuring it in App Direct Mode and using it as vPMEM. In order to get the best application performance, one would need a PMem-aware application, such as Microsoft SQL Server 2019.

NVDIMM-N Persistent Memory

NVDIMM-N is a type of PMem that operates at the same speeds as volatile DRAM, but allows its contents to be saved across reboots (or saved automatically on power failure).

NVDIMM-N is always presented to the operating system as PMem (it doesn't have a Memory Mode like Optane PMem), and thus requires a PMem-aware operating system, such as ESXi 8.0 Update 3.

ESXi can use NVDIMM-N in the same ways it uses Optane PMem in App Direct Mode, either as vPMEMDisk or vPMEM.

For information about NVDIMM-N PMem performance, see the white paper Persistent Memory Performance on vSphere 6.7.

Storage Hardware Considerations

This section provides guidance regarding storage hardware for use with vSphere 8.0 Update 3.

General Storage Hardware Considerations

Back-end storage configuration can greatly affect performance. For more information on storage configuration, refer to vSphere Storage.

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 77 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 33).

  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 43 for details about NVMe support in vSphere 8.0 Update 3 as well as use of the high-performance plug-in).
- 4K native (4Kn) and 512B emulation (512e) drives have some advantages and some limitations. In particular, such drives can perform poorly if your workloads don't issue mostly 4K-aligned I/Os. For more information on this subject, see VMware KB article 2091600 or the "Device Sector Formats" subsection of Viewing Storage Devices Available to an ESXi Host in the *vSphere Storage Guide*.
- Consider choosing storage hardware that supports vStorage APIs for Storage Awareness (VASA), thus allowing you to use vVols (as described in "VMware vSphere Virtual Volumes (vVols)" on page 96).

  Because vVols performance varies significantly between storage hardware vendors, make sure the storage hardware you choose will provide the vVols performance you expect.
- If you plan to use VMware vSAN, consider the advantages of an all-flash vSAN deployment versus a hybrid deployment (see "Hybrid versus All-Flash vSAN" on page 93).
- If you plan to use VMware vSAN, consider choosing NICS and network switches that support RDMA over Converged Ethernet version 2 (RoCE v2) (see "vSAN Network Considerations" on page 94).
- If you plan to use virtual machine encryption (see "vSphere Virtual Machine Encryption Recommendations" on page 45) consider choosing hardware that supports the AES-NI processor instruction set extensions. For more information, see "AES-NI Support" on page 13.
- 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 47 for a different approach to this issue.

- 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.
- Local storage performance might be improved with write-back cache. If your local storage has write-back cache installed, make sure it's activated 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), PCIe 3.0 (aka PCIe Gen 3), and PCIe 4.0 (aka PCIe Gen 4) 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 Gen 2 x8 or PCIe Gen 3 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 Gen 3 x8 slots (which are capable of a net maximum of 64Gb/s in each direction). Similarly, newer, faster cards might need PCIe Gen 4 to reach their full bandwidth potential.

  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.

Fibre Channel Considerations

- 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.

VMware vStorage APIs for Array Integration (VAAI)

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 40.

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.

NOTE 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 APIs: Array Integration (VAAI). For information about configuring the way ESXi uses VAAI, see "ESXi Storage Considerations" on page 39.

iSCSI and NFS Storage

- 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.

  Though primarily about how to diagnose problems once they start, the following two white papers might also be useful in avoiding issues in the first place: ESX IP Storage Troubleshooting Best Practice: Packet Capture and Analysis at 10G and ESXi NFS Read Performance: TCP Interaction between Slow Start and Delayed Acknowledgement.

- 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.

NVMe Storage

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.

- 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.

For information about using NVMe in ESXi, see "NVMe Recommendations" on page 43.

NVMe over Fabrics (NVMe-oF) Storage

ESXi supports NVMe over Fabrics (NVMe-oF) over Fibre Channel (FC), Remote Direct Memory Access (RDMA), and TCP/IP. Unlike NVMe, which is a protocol for accessing devices connected to the local system over a PCIe bus, NVMe-oF connects over a network to remote NVMe devices. NVMe-oF offers higher IOPS and lower latencies than many other remote storage options, with potentially lower CPU cost per I/O.

For the best performance using NVMe over TCP/IP:

- Configure network links for jumbo frames.
- Increase the number of hardware queues for NetQueue RSS.
- Make sure to configure at least one distributed virtual port group per vmnic.

For additional information see What's New in vSphere 7 Core Storage (NVMe-oF was introduced in vSphere 7.0, so there's lots of explanation here even though it's describing a prior version) and What's New in vSphere 8 Core Storage (this describes the latest updates about NVMe-oF). The Performance Characterization of NVMe-oF in vSphere 7.0 U1 white paper compares the performance of NVMe over Fibre Channel to legacy Fibre Channel Protocol (SCSI FCP). For information about using NVMe-oF in ESXi, see "NVMe-oF Recommendations" on page 43.

Network Hardware Considerations

This section provides guidance regarding network hardware for use with vSphere 8.0 Update 3.

- 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 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)
  - Receive Side Scaling (RSS)
  - When using a virtualization encapsulation protocol, such as VXLAN or GENEVE:
    - The NICs should support offload of that protocol's encapsulated packets:

      For VXLAN:
      VXLAN Offload
      VXLAN Rx/Tx Filters

      For GENEVE:
      GENEVE Offload
      GENEVE Rx/Tx Filters

    - The NICs should support NetQueue and, along with it, inner (encapsulated) MAC and VXLAN Network ID (VNI) filtering.
  - Enhanced Datapath modes (designed to provide higher performance at lower CPU cost)

NOTE NIC features can be confirmed by looking under IO Devices in the *VMware Compatibility Guide*.

- Make sure network cards are installed in slots with enough bandwidth to support their maximum throughput. As described in "Storage Hardware Considerations" on page 16, 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; 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 single-port 40Gb/s Ethernet network adapters should use PCIe Gen 3 x8 slots (or higher), dual-port 40Gb/s adapters or single-port 100Gb/s adapters should use PCIe Gen 3 x16 slots (or higher), and dual-port 100Gb/s adapters should use PCIe Gen 4 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.
- Beginning with version 8.0, vSphere supports certain DPUs (data processing units, sometimes called SmartNICs). These DPUs are high-performance network interface cards with added embedded CPU cores and memory, running a hypervisor optimized for I/O activities (such as packet offloads, external management, and so on). This hypervisor runs on the SmartNIC independently from the ESXi hypervisor installed on the server.

  For information about the performance of DPUs in ESXi, see "Data Processing Units (DPUs, or SmartNICs)" on page 49. For information about using DPUs, see Introducing VMware vSphere Distributed Services Engine and Networking Acceleration by Using DPUs in *VMware ESXi Installation and Setup*.

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 activate all populated processor sockets and to activate all cores in each socket.
- Activate "Turbo Boost" in the BIOS if your processors support it.
- Make sure hyper-threading is activated in the BIOS for processors that support it.
- Some NUMA-capable systems provide an option in the BIOS to deactivate NUMA by activating node interleaving. In most cases you will get the best performance by deactivating node interleaving (in other words, leaving NUMA activated).
- Make sure any hardware-assisted virtualization features (VT-x, AMD-V, EPT, RVI, and so on) are activated 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.

- Deactivate 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 25 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 13), and your deployment will be able to make use of the feature (see "vSphere Virtual Machine Encryption Recommendations" on page 45, "VMware vMotion Recommendations" on page 77, and "VMware vSAN" on page 93):
  - 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 activated in BIOS.

Processor-Specific BIOS Settings

In addition to common BIOS settings, some processor families have their own special settings that can affect performance. These include:

- Snoop mode selection, described in "Snoop Mode Selection" on page 30.
- AMD EPYC Processor NUMA Settings, described in "AMD EPYC Processor NUMA Settings" on page 30.

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 31). 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.

  However, for a very few multithreaded workloads that are highly sensitive to I/O latency, such as financial platforms or media and entertainment, C-states (including C1E) can reduce performance. In these cases you might obtain better performance by deactivating them in the BIOS.
- C-states deeper than C1/C1E (typically C3 and/or C6) are managed by software and activate further power savings. In order to get the best performance per watt, you should activate 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 activated, C1E and deep halt states (for example, C3 and C6) can sometimes even increase the performance of certain lightly-threaded workloads (that is, workloads that leave some hardware threads idle).

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 deactivate 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

  Deactivating 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 deactivated.

  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 be impacted. This impact can be reduced by configuring the virtual machines to use ISO images instead of physical drives, and can be avoided entirely by deactivating optical drives in virtual machines when the devices are not needed.
- ESXi 8.0 Update 3 supports virtual hardware version 21. 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 8.0 Update 2, however, and thus if a cluster of ESXi hosts will contain some hosts running pre-8.0 Update 2 versions of ESXi, the virtual machines running on hardware version 21 will be constrained to run only on the ESXi 8.0 Update 2 or later 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 run 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 running instructions. Because in this case 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 full CPU commitment (running as many vCPUs on a host as the total number of physical processor cores in that host) and even significant levels of CPU overcommitment (running *more* vCPUs on a host than the total number of physical processor cores in that host) without impacting virtual machine performance.

  Caution is warranted, though. If an ESXi host is fully CPU committed, or overcommitted, and the virtual machines on that host consume all their allocated CPU resources, the host could become CPU saturated; that is, the virtual machines and other loads on the host could demand all the CPU resources the host has.

  NOTE It's usually best to leave some CPU resources for ESXi, rather than allowing all CPU resources to be consumed by virtual machines. This could mean either not fully committing physical processor cores to virtual machines or fully committing them but with the expectation that the virtual machines will rarely, if ever, simultaneously use all their allocated CPU resources.

  If an ESXi host becomes CPU saturated, performance of some or all of the virtual machines on that host could be significantly reduced. Latency-sensitive workloads in particular 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 Distributed Resource Scheduler (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 Client or by using `esxtop` or `resxtop`. When using either utility:
  - If the load average on the first line of the 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 Performance Monitoring Utilities: resxtop and esxtop in *vSphere Monitoring and Performance*.

- 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 59.)
- Maintaining a consistent memory view among multiple vCPUs can consume additional resources, both in the guest operating system and in ESXi.
- Most guest operating systems run an idle loop during periods of inactivity. Within this loop, most of these guest operating systems halt by running 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.

- 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.

Side-Channel Vulnerability Mitigation in ESXi

As described in "Side-Channel Vulnerabilities" on page 11, mitigation for some side-channel vulnerabilities takes place at the hypervisor level. This section provides a brief overview of those mitigations.

Speculative Execution Vulnerabilities (aka Spectre, Meltdown, and Foreshadow)

See VMware KB articles 52245, 54951, and 55636 for details about which Spectre, Meltdown, and Foreshadow security vulnerabilities can be mitigated by using the latest ESXi release.

NOTE As described in "Side-Channel Vulnerabilities" on page 11, some of these vulnerabilities are mitigated in recent CPU versions, potentially with less impact on performance.

Scheduler Options for L1 Terminal Fault (L1TF) Vulnerability Mitigation

One particular side-channel vulnerability, the L1 Terminal Fault (L1TF) vulnerability (CVE-2018-3646), requires you to make choices and take further action. This vulnerability potentially allows information leakage between virtual machines (or between VMs and ESXi) when hyper-threading is activated.

As described in VMware KB article 55806, updates that automatically address one part of this vulnerability (the sequential-context attack vector) without imposing a significant performance impact are available for many vSphere versions.

Those same updates *can* address the other part of this vulnerability (the concurrent-context attack vector), but they *don't do so by default*. Addressing this part of the vulnerability requires manually activating a new ESXi scheduler (called an ESXi Side-Channel-Aware Scheduler), with a potentially significant performance impact and possible limitations on running certain virtual machines.

An initial version of the Side-Channel-Aware Scheduler (SCAv1) was made available in updates for previous versions of ESXi. SCAv1 limits the scheduler to use only one thread per core. A second version (SCAv2) was released with ESXi 6.7 Update 2. SCAv2 activates multithreading, but never simultaneously schedules threads from different VMs on the same core. In nearly all cases SCAv2 imposes a lower performance penalty than SCAv1.

You can run the HTAware Mitigation Tool, described in VMware KB article 56931, to get an estimate of the performance impact of activating SCAv1 and a report of potential issues you might encounter when doing so. This tool can also be used to activate or deactivate SCAv1, SCAv2, and various associated settings and options.

For more information about schedulers, see VMware KB article 55806 and the white paper Performance of vSphere 6.7 Scheduling Options (though written about vSphere 6.7 Update 2, most of the information is still relevant).

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 activate 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 Activate 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 activated, 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 spread load 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 59.

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 deactivated, ESXi detects the system as NUMA and applies NUMA optimizations. If node interleaving is activated, 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 the Using NUMA Systems with ESXi section of *vSphere Resource Management*.

Virtual Topology

ESXi 8.0 introduced an enhanced virtual topology feature. This feature automatically selects optimal coresPerSocket values for virtual machines and optimal virtual L3 sizes. It also includes a new virtual motherboard layout to expose NUMA for virtual devices and vNUMA topology when CPU hotplug is enabled.

For more information on this topic, see the white paper VMware vSphere 8.0 Virtual Topology.

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 activated 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 59), 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 59).

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 five 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)
- Sub-NUMA cluster (SNC)

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).

Sub-NUMA cluster (SNC) is similar to cluster-on-die, but with differences in how it uses the last level cache (LLC).

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 cluster on die, see VMware KB article 2142499. For information about sub-NUMA cluster, see Intel® Xeon® Processor Scalable Family Technical Overview.

AMD EPYC Processor NUMA Settings

The second generation AMD EPYC processors have some new and updated NUMA-related BIOS settings, including NUMA Nodes per Socket (NPS) and CCX as NUMA Domain.

- `NUMA Nodes per Socket` (NPS) can be configured to change the memory interleaving policy, offering the following three possible values (though not all AMD EPYC processors offer all NPS options):
  - NPS-1 — presents each CPU as a single NUMA node,
  - NPS-2 — presents each CPU as two NUMA nodes,
  - NPS-4 — presents each CPU as four NUMA nodes.
- `CCX as NUMA Domain` can be activated or deactivated:
  - Activated — presents each Core Cache Complex (CCX) as a NUMA domain,
  - Deactivated — presents the CPUs as set by NPS.

  Note, however, that the CCX setting doesn't change the memory channel interleave policy, which is still controlled by the NPS setting.

Starting with vSphere 7.0 Update 2, when running on AMD EPYC processors the default values for both the vSphere settings and these BIOS settings (NPS set to NPS-1, CCX-as-NUMA deactivated) will provide the best performance in nearly all cases.

In some unusual cases it might be worth testing NPS-4. This would be for workloads that meet all these conditions:

- They require very high bandwidth (for example, high performance computing applications).
- They require the lowest possible memory latency.
- They are highly NUMA optimized.
- You can afford to pin/affinitize the virtual machines.

For more information on this topic, see the white paper "Performance Optimizations in VMware vSphere 7.0 U2 CPU Scheduler for AMD EPYC Processors (though written about a previous version, most of this information is still relevant)."

Persistent Memory (PMem) in NUMA Systems

As described in "Persistent Memory (PMem)" on page 14, PMem is non-volatile memory that fits in standard server DIMM slots.

For information about using PMem in ESXi, see "Persistent Memory (PMem)" on page 38.

For information about using vNUMA and PMem, see "vNUMA and PMem" on page 61.

Though each DIMM slot that contains PMem is local to a NUMA node, ESXi doesn't automatically associate PMem with its NUMA node. The association can be manually configured, however, as follows:

1 Run the following command:

```
memstats -r pmem-extent-stats -s numaNode|grep -w "[0-9]$"|sort -n|awk 'BEGIN {nvd=0;} {printf
               "nvdimm0:%d.node = \"%d\"\n", nvd, $0; nvd++}'
```

2 Using the output of the above command, configure the `nvdimm0:<devNum>.nodeAffinity` parameters, as described in VMware KB article 78094.

NOTE Due to implementation differences between vendors, virtual machines might not perform as expected if they are configured as described here and are then moved with vMotion to a different vendor's hardware.

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 86.

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 uses no power saving features, thus maximizing performance for a system where all CPUs are fully loaded.

- **Balanced**
  This power policy (the default) is designed to reduce host power consumption while maintaining power usage flexibility, thus typically incurring little or no reduction in performance. On a system where some CPUs are idle part of the time, this policy might actually increase performance by allowing active (non-idle) CPUs to enter faster Turbo Boost states than are available in the High Performance policy.
- **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, see the Host Power Management Policies section 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 22.

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 Client, select the host of interest.

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

3 In the left pane within the **Configure** tab, under **Hardware**, 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

The default power policy of **Balanced** will typically not impact the performance of CPU-intensive or memory-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 Workloads" on page 52.

ESXi Memory Considerations

This subsection provides guidance regarding memory considerations in ESXi.

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 33) 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 to dynamically reduce the amount of machine physical memory required for each virtual machine. These are page sharing, ballooning, memory compression, swap to host cache, and regular swapping.
  - **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 not shared by default *between* virtual machines for security reasons.

    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 activate the previous default behavior if desired, see "Memory Page Sharing" on page 34.
  - **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 transparent to the guest operating system and 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 the white papers Understanding Memory Resource Management in VMware vSphere 5.0 (though this paper specifically addresses vSphere 5.0, most of the concepts are still applicable).

- 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 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 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 Client, select the virtual machine in question, select **Monitor** > **Utilization**, expand the **Guest Memory** pane, then look at the values of **Swapped** and **Compressed**. Non-zero values here indicate swapping or compressing at the host level, which is a sign of 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 activated by default *within* virtual machines ("intra-VM sharing"), but is activated *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 for virtual machines.

Because of the prevalence of 2MB large memory pages (see "2MB Large Memory Pages" on page 37), which are not shared even when page sharing is activated, 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 activate inter-VM page sharing using one of these methods:

- To activate inter-VM page sharing between all virtual machines on an ESXi host, set the host configuration option `Mem.ShareForceSalting` to 0. This duplicates the original default behavior of pre-6.0 versions of vSphere.

  NOTE For more information about the host configuration option `Mem.ShareForceSalting` see VMware KB article 2097593.

- 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 above in "Memory Overcommit Techniques" on page 33 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 deactivate 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 36).
- 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 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 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).

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 activated 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). The system retains a small working set memory reservation, and does not swap under ordinary usage. Extraordinary usage (for example, taking screenshots every second, or collecting a support dump) might cause swapping.

Swapping carries a risk of deadlock when the datastore is backed by a virtual machine on the same host, which is not a recommended configuration. If necessary, VMX swap file creation can be prevented entirely by setting `sched.swap.vmxSwapEnabled` to `FALSE`, at the cost of significantly increased memory reservation.

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 33.

In addition, ESXi also provides optimizations, such as page sharing (see "Memory Overcommit Techniques" on page 33), 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 described in this section.

2MB Large Memory Pages

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; it 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, as described in "Guest Operating System Memory Considerations" on page 62). 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.

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 34 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.

Persistent Memory (PMem)

As described in "Persistent Memory (PMem)" on page 14, PMem is non-volatile memory that fits in standard server DIMM slots.

For information about PMem in NUMA systems, see "Persistent Memory (PMem) in NUMA Systems" on page 31.

- On supported hardware, the ESXi `memstats` tool can be used to view host-wide bandwidth and latency statistics for DRAM and PMem, as well as the DRAM miss rate, using this command:
  `memstats -r hw-stats`

  On certain hosts using PMem in Memory Mode, `memstats` can be used to view virtual machine level statistics for DRAM and PMem using this command:
  `memstats -r vm-stats -b`

  For additional information use this command:
  `memstats -h`

  NOTE In vSphere 8.0 and later the `memstats -r dcpmm-stats` option is no longer supported.

- If desired, PMem can be preallocated as part of virtual machine startup using `sched.pmem.prealloc`, as described in VMware KB article 78094. This prevents allocation overhead when the virtual machine first uses a PMem page.
- For guidance on sizing Optane PMem, see Announcing VMware vSphere Support for Intel® Optane™ Persistent Memory Technology.

  NOTE For more details on monitoring active memory in your environment, see VMware KB article 1002604.

- vSphere Memory Monitoring and Remediation (vMMR) collects data and displays performance statistics for both PMem and DRAM at both the host and virtual machine level. This can help determine if a workload's performance is affected by a host's use and configuration of Intel Optane PMem in Memory Mode. You can see this information in the Advanced Memory section under Performance in vCenter.

  vMMR will try to raise alarms on critical memory conditions. You can also create custom alerts for thresholds acceptable for your workloads. For more information, see the Solutions for Performance Issues With Memory Mode section of *vSphere Monitoring and Performance*.

ESXi Storage Considerations

This subsection provides guidance regarding storage considerations in ESXi.

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 "Storage Hardware Considerations" on page 16) 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.

- 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.
- Refer to your array vendor's VASA Provider sizing guide to determine the resource requirements (such as vCPUs and memory) they recommend for the best performance.

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 the Raw Device Mapping section of *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 vSphere Virtual Volumes (vVols)" on page 96) 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 "Storage Hardware Considerations" on page 16) 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. The first-write performance for lazy-zeroed thick disks was significantly improved starting in vSphere 7.0 Update 2, though eager-zeroed thick disks still have better first-write performance.

    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. Just like with lazy-zeroed thick disks, there is reduced performance during the first write to an unwritten file block, but that was significantly improved starting in vSphere 7.0 Update 2. 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 Client (right-click the virtual machine, select **Edit Settings**, choose the **Virtual Hardware** tab, click **ADD NEW DEVICE**, select **Hard Disk**, 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 more information see the Using vmkfstools section of *vSphere Storage* 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.

Automatic Space Reclamation (UNMAP)

On VMFS6 datastores (supported by vSphere 6.5 and later), vSphere can perform automatic space reclamation (i.e., UNMAP) for thin-provisioned disks. vSphere 6.5 allowed the UNMAP priority to be set; vSphere 6.7 added the ability to set the UNMAP rate. Using these options you can reclaim storage space much more quickly on fast arrays (especially useful for all-flash arrays) or limit the potential performance impact of performing an unthrottled burst of UNMAP operations (especially useful for slower arrays).

The vSphere Client offers limited configuration options for UNMAP. For the full range of options, use the `esxcli` command.

For more information on the configuration options available for UNMAP and how to use them, see the Storage Space Reclamation section of *vSphere Storage*.

NOTE VMFS5 and earlier file systems do not unmap free space automatically, but you can use the `esxcli storage vmfs unmap` command to reclaim space manually. When you use the command, be aware that it might send many unmap requests at once. This action can lock some of the resources during the operation. For more information, see VMware KB article 2057513.

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 can have reduced performance when the partition is unaligned. Using the vSphere 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 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.

Unified Data Transport (UDT)

- vSphere 8.0 introduced vSphere Unified Data Transport (UDT). We strongly recommend activating this feature in order to dramatically increase performance of cloning as well as vMotion operations on powered-off virtual machines and virtual machines with snapshots. UDT comes into play only when both the source and destination ESXi hosts are running vSphere 8.0 or later and have the provisioning service activated.

For more information about UDT, including how to activate it, see vSphere vMotion Unified Data Transport and the video vSphere 8 Enhancements for VM Cold Migrations, vMotion, and Cloning.

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. These suggestions apply to SCSI, FC, and NVMe-oF arrays.

- 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), activating 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 Using ESXi with a SAN section of *vSphere Storage* 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 88).

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 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

- When possible, it's beneficial to create an independent network connection for iSCSI and NFS between the client and server.

  If an independent connection isn't practical, consider isolating iSCSI and NFS traffic from other network traffic by using a VLAN. Note, however, that the use of VLANs does not guarantee Quality of Service.
- Most iSCSI and NFS deployments will perform best with 10Gb/s or faster links. In NFS deployments, link aggregation can also be used to provide more bandwidth.
- Consider using jumbo frames when possible.
- Packet loss on a network with NFS or iSCSI traffic might interact with ESXi-side TCP delayed ACKs and lead to poor NFS or iSCSI performance. For more information see VMware KB article 59548 (NFS) or 1002598 (iSCSI). Additional detail on this topic (for NFS) can be found in the ESXi NFS Read Performance white paper.

NVMe Recommendations

Non-Volatile Memory Express (NVMe) is a high-performance storage protocol described in "NVMe Storage" on page 19.

Note: vSphere also supports virtual NVMe (vNVMe), which is discussed in "Guest Operating System Storage Considerations" on page 63.

NVMe performance considerations:

- To maximize throughput, activate the high-performance plug-in (HPP) on your ESXi hosts. For more information about the HPP, see VMware High Performance Plug-In and Path Selection Schemes in *vSphere Storage*.
- With HPP, increasing the maximum number of outstanding disk requests with the `Disk.SchedNumReqOutstanding` parameter might increase performance (see KB article 1268).
- NVMe devices are highly parallelized; their maximum throughput is generally achieved when they are backing multiple virtual disks and/or multiple virtual machines.
- 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 63 for details.)

For additional information, see the About VMware NVMe Storage section of *vSphere Storage*.

NVMe-oF Recommendations

Non-Volatile Memory Express over Fabrics (NVMe-oF) connects over a network to remote NVMe devices, as described in "NVMe over Fabrics (NVMe-oF) Storage" on page 19.

NVMe-oF performance considerations:

- The default plugin for NVMe-oF is the High-Performance Plug-in (HPP). For more information about the HPP, see VMware High Performance Plug-In and Path Selection Schemes in *vSphere Storage*.

- Review the requirements and prerequisites in both What's New in vSphere 7 Core Storage (NVMe-oF was introduced in vSphere 7.0, so there's lots of explanation here) and What's New in vSphere 8 Core Storage (this describes the latest updates about NVMe-oF).

For additional information, see the About VMware NVMe Storage section of *vSphere Storage*.

vSphere Virtual Machine Encryption Recommendations

Virtual machine encryption 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 will see no significant performance impact. However, when sufficient CPU resources are not available (as might be possible, for example, with a workload driving high IOPS on faster storage devices), activating VM encryption can begin to impact performance.
- To significantly minimize additional CPU utilization due to encryption, choose processors that support AES-NI (see "AES-NI Support" on page 13), especially some recent processor versions in which the AES-NI implementation is further optimized. Make sure AES-NI is activated in BIOS. In some cases a BIOS upgrade might be required for AES-NI support.
- Because with VM encryption 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 vSAN" on page 93.)

Further detail on this topics can be found in the VMware vSphere Virtual Machine Encryption Performance white paper (though written about a previous version, much of the material still applies) and the Virtual Machine Encryption Best Practices section of *vSphere Security*.

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 activate 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 88), 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, activating 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 activated, 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 Workloads

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. Workloads 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. Deactivate them as follows:

  - Set the ESXi host power policy to **Maximum performance** (as described in "Host Power Management in ESXi" on page 31; this is the preferred method) or deactivate power management in the BIOS (as described in "Power Management BIOS Settings" on page 22). This will, of course, tend to increase power consumption.
  - Deactivate C1E and other C-states in BIOS (as described in "Power Management BIOS Settings" on page 22). This, too, will tend to increase power consumption.
  - Activate Turbo Boost in BIOS (as described in "General BIOS Settings" on page 22).

- If you are using an LSILogic vHBA or a PVSCSI vHBA (with hardware version 11 or later) 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.)

  Reducing the reqCallThreshold value will typically increase CPU usage. It also, perhaps counter-intuitively, has the potential to reduce the maximum possible throughput on the virtual disk(s).

  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 workloads, the white paper Best Practices for Performance Tuning of Latency-Sensitive Workloads in vSphere VMs, though now somewhat dated, and addressing primarily network latency-sensitive workloads, 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 workloads 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.

Enhanced Data Path

Enhanced Data Path (EDP) is a networking stack mode which, when configured, provides superior network performance. Verify a NIC's compatibility with EDP by looking under IO Devices in the *VMware Compatibility Guide*. For more information about EDP, see Enhanced Data Path in the *NSX Installation Guide*.

Network Traffic Shaping

Network traffic shaping in ESXi allows the configuration of average bandwidth, peak bandwidth, and burst size on vSphere switches and port groups. This feature shapes outbound network traffic on standard switches and port groups; it shapes both inbound and outbound traffic on distributed switches and port groups.

For instructions to configure traffic shaping, see What is Traffic Shaping Policy in *vSphere Networking*.

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, vSAN 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 8.0 Update 3, 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.

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 80) load-balances virtual machines across hosts, it will include NetIOC bandwidth reservations in its calculations.

For further information about NetIOC, see the white paper Performance Evaluation of Network I/O Control in VMware vSphere 6.0 and the vSphere Network I/O Control section of *vSphere Networking*.

Network I/O Control Configuration

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

NOTE NetIOC controls only transmit traffic; it does not limit receive traffic. Traffic shaping (described in "Network Traffic Shaping" on page 47), a mechanism to bound the bandwidth of a vNIC, can be configured for transmit or receive traffic separately.

- Shares can be used to allocate to a resource pool or virtual NIC 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 NIC 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 NIC. The total bandwidth reservations are limited to 75% of the capacity of the underlying physical adapter. If a resource pool or virtual NIC 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 NIC'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

Beginning in vSphere 7.0 Update 2, Network I/O Control by default automatically scales the number of hardware transmit queues based on link speed (one hardware queue on the NIC for each 12.5Gb/s of link speed). This allows multiple virtual NICs or multiple vmknics to distribute traffic across multiple hardware transmit queues on a single physical NIC, and can potentially improve performance, especially in environments with small packets and high packet rate.

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

1 Activate the feature, called HClock Multiqueue, by running the following command:
`esxcli system settings advanced set –i x –o /Net/NetSchedHClkMQ`
(where a value for *x* of `0` deactivates the feature, `1` activates the feature without automatic scaling, and `2` (the default) activates the feature with automatic scaling).

2 If you set it to 1 (activated, but without auto-scaling), 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).

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

Once this feature is activated (with or without automatic scaling), you can use NetSchedHClkMaxHwQueue to determine how many hardware transmit queues can be used.

Data Processing Units (DPUs, or SmartNICs)

DPUs (data processing units, also called SmartNICs) can improve networking performance while also reducing CPU load. DPUs provide many of the advantages of DirectPath I/O (described in "DirectPath I/O" on page 49) and SR-IOV (described in "Single Root I/O Virtualization (SR-IOV)" on page 49), while also allowing the use of vSphere high-availability and migration features.

DPUs offer two data path modes, MUX mode and UPTv2 mode:

- MUX mode provides higher flexibility, but continues to use some of the host's CPU resources.
- UPTv2 mode provides higher performance by completely offloading all network processing to the DPU, but requires virtual hardware version 20 or later, a VMXNET3 vNIC guest driver version that supports the feature, and full reservation and pinning of guest memory.

Both MUX and UPTv2 modes provide accelerated networking services for VLAN and overlay networking, checksum offload, TCP Segmentation Offload (TSO), and Receive Side Scaling (RSS).

Although configuring a data path mode requires NSX Manager, some of these configuration options don't require an NSX license.

For more information about DPUs, see VMware vSphere Distributed Services Engine and Networking Acceleration by Using DPUs in *VMware ESXi Installation and Setup*.

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 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. However, for workloads that are both very networking intensive (especially those with high packet rates) and that don't need the core virtualization features mentioned above, 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 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 Single Root I/O Virtualization (SR-IOV) section 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 vMotion, physical NIC sharing, snapshots, suspend/resume, Fault Tolerance, NetIOC, memory overcommit, VMSafe, or NSX network virtualization. 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 activated 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 activated 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 activated if desired, as described in "Activating or Deactivating SplitRx Mode for an Individual Virtual NIC" on page 50.

Deactivating SplitRx Mode for an Entire ESXi Host

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

The possible values for this variable are:

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

To change this variable through the vSphere 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.

Activating or Deactivating 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 deactivates splitRx mode for `ethernet`*X*.
- `ethernet`*X*`.emuRxMode = "1"`
  This value activates splitRx mode for `ethernet`*X*.

To change this variable through the vSphere 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 **ethernet*X*.emuRxMode** (where ***X*** is the number of the desired NIC) and configure it as you wish. If the variable isn't present, click **ADD CONFIGURATION PARAMS** and enter it as a new parameter.
4. Click **OK** to close the **Configuration Parameters** window.
5. 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 faster 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, some configurations have this feature deactivated by default.

You can activate this feature as follows:

- For Intel 82599, X450, and X550 NICs:

  Load the driver with the following command:
  `esxcli system module parameters set –a –m ixgben –p RSS="1"`

  To activate the feature on multiple Intel 82599, X450, or X550 NICs, include another comma-separated `1` for each additional NIC (for example, to activate the feature on three such NICs, you'd replace `RSS="1"` in the above command with `RSS="1,1,1"`).

- For Mellanox Technologies ConnectX-4, ConnectX-5, or ConnectX-6 NICs:

  The driver supports Default Queue RSS (DRSS) and NetQueue RSS by default. Additional information can be found using the command:
  `esxcli system module parameters list –m nmlx5_core`

For any of these NIC types, after activating RSS as described above, 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. Depending on the workload, this might increase network latency by anywhere from a few hundred microseconds to a few milliseconds. Many workloads, however, are not impacted by this amount of network latency, 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 activated for all virtual NICs in ESXi. For VMXNET3 virtual NICs, however, this feature can be set to one of three schemes or deactivated by changing the `ethernet`*`X`*`.coalescingScheme` variable (where *X* is the number of the virtual NIC to configure).

- The feature will be activated (the default) if the `ethernet`*`X`*`.coalescingScheme` variable is not present in the `.vmx` file or if the variable is present and set to **`rbc`** (which stands for rate-based coalescing)

  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 `ethernet`*`X`*`.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 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 deactivated by setting `ethernetX.coalescingScheme` to **`disabled`**. Deactivating 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 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, click **ADD CONFIGURATION PARAMS** and enter it as a new parameter.
   - 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, click **ADD CONFIGURATION PARAMS** and enter it as a new parameter.
4. Click **OK** to close the **Configuration Parameters** window.
5. Click **OK** to close the **Edit Settings** window.

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

Running Network Latency Sensitive Workloads

By default, the ESXi network stack is configured to drive high network throughput 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 workloads that are highly sensitive to network latency (that is, workloads that are impacted by latency on the order of a few tens of microseconds). This section describes those options.

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

  vSphere allows you to change the network latency sensitivity of a virtual machine from **Normal** (the default) to **High** or **High with Hyperthreading**. (**High with Hyperthreading** is available only for virtual machines using virtual hardware version 20 or later.)
  - Leaving this set to **Normal** allows ESXi to assign the vCPUs in a virtual machine to any hyper-thread in any physical CPU core on the host system. In most cases this allows ESXi to provide the best overall performance for the virtual machines on that host.
  - Setting this to **High** causes the scheduler to attempt to grant each vCPU in the virtual machine exclusive access to one physical CPU core (the entire CPU core, thus effectively deactivating hyper-threading for that core). This can potentially increase the performance of that one virtual machine, but possibly at the cost of reduced performance for other virtual machines on the same host.

- Setting this to **High with Hyperthreading** causes the scheduler to grant each vCPU in the virtual machine exclusive access to one hyper-thread of a physical CPU core and to ensure that consecutive even-odd pairs of vCPUs within a virtual machine are assigned to the pair of hyper-threads on a single physical core (that is, vCPUs 0 and 1 are assigned to one physical CPU core, vCPUs 2 and 3 are assigned to another CPU core, and so on). ESXi then exposes this hyper-thread topology to the guest OS. While this option can be quite helpful for certain workloads, it is not a general-purpose setting, and should be used only where it would be expected to improve performance, as detailed below.

  By guaranteeing placement on the physical cores, and exposing that placement to the guest OS, **High with Hyperthreading** allows the guest OS and applications running within it to take advantage of cache locality. This is primarily useful for the very few guest workloads that can use awareness of the underlying hyper-threading topology to improve performance by allowing multiple threads to share the same cache footprint.

  For the majority of workloads, however, **High with Hyperthreading** has the potential to provide no performance benefit, *or even to reduce performance for those workloads*. Even though many workloads benefit from hyper-threading, those workloads typically obtain the full benefit from ESXi scheduling them on the hyper-threads. And because ESXi is aware of any other virtual machines it's running, it can also make scheduling decisions that benefit all of those VMs. But by providing each vCPU with one physical hyper-thread, **High with Hyperthreading**, essentially gives that vCPU half as much physical CPU resources as **High** (because **High** gives each vCPU an entire physical core). And by locking vCPUs to specific hyper-threads, **High with Hyperthreading** prevents ESXi from scheduling vCPUs on otherwise idle physical CPU cores (as it would if set to **Normal**), also potentially limiting performance. For these reasons, we recommend using **High with Hyperthreading** only with workloads known to be aware of and benefit from being exposed to the underlying topology.

With either **High** or **High with Hyperthreading**, 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** or **High with Hyperthreading**, consider these additional important points:

- Before network latency sensitivity can be set to **High** or **High with Hyperthreading** 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.

  NOTE Even with 100% CPU reservation, the VM might not acquire exclusive assignment of CPU cores if there are no NUMA nodes with enough non-exclusive CPU cores when the virtual machine powers on.

  If a virtual machine uses a virtual hardware version prior to version 14, 100% CPU reservation is not required in order to set latency sensitivity to **High**, but 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.

  If a virtual machine uses virtual hardware version 14 or later and has latency sensitivity set to **High** or **High with Hyperthreading**, 100% CPU reservation is actually a requirement to power it on. (This can be changed by setting `"latency.enforceCpuMin" = FALSE`.)
- The reservations for both CPU cores and memory, as well as the latency sensitivity 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 settings of **High** or **High with Hyperthreading** (as described above), fully reserving memory can help reduce latency even when latency sensitivity is set to **Normal**.

- Fully reserve CPUs.

In addition to allowing latency sensitivity set to **High** to work reliably (as described above), making a full CPU reservation can help reduce latency even when latency sensitivity is set to **Normal**.

- 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 49) or SR-IOV ("Single Root I/O Virtualization (SR-IOV)" on page 49). 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 64).

- Adjust the host power management settings.

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

  - Set the ESXi host power policy to **Maximum performance** (as described in "Host Power Management in ESXi" on page 31; this is the preferred method) or deactivate power management in the BIOS (as described in "Power Management BIOS Settings" on page 22).
  - Deactivate C1E and other C-states in BIOS (as described in "Power Management BIOS Settings" on page 22).
  - Activate Turbo Boost in BIOS (as described in "General BIOS Settings" on page 22).

- Tune the virtual network adapter.

  For a majority of general purpose workloads, and for moderately latency-sensitive workloads, the VMXNET3 virtual network adapter, left at its default settings, will provide the best performance. Workloads with low packet rate or few active threads might benefit by deactivating VMXNET3 virtual interrupt coalescing for the desired NIC. In other cases—most notably workloads with high numbers of outstanding network requests—deactivating 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 51.

- Network Function Virtualization (NFV) workloads that depend on high packet throughput for their responsiveness, such as telco, 5G, and IoT-based deployments, might benefit from Enhanced Network Stack (ENS; also referred to as Enhanced Datapath) in NSX. For more information, see the Enhanced Datapath section in the *NSX Administration Guide*.

For further information about running latency-sensitive workloads, the white paper Best Practices for Performance Tuning of Latency-Sensitive Workloads in vSphere VMs, though now somewhat dated, might prove helpful.

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 activated, 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 reduces hypervisor overhead for packet processing, which 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.

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, activating dense mode might reduce system-wide throughput (though while still also reducing CPU utilization per unit of throughput). We therefore recommend activating dense mode only on those hosts where its trade-offs will be useful, rather than simply activating it on all hosts.

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

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

NOTE Even when dense mode is activated, 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 33) will not work if VMware Tools is not installed.

- Deactivate screen savers and Window animations in virtual machines. On Linux, if using an X server is not required, deactivate 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 physical CPU cores, 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.

  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.

Microsoft Virtualization-Based Security (VBS)

Microsoft virtualization-based security (VBS) is a feature first available in Windows 10 and Windows Server 2016. It uses virtualization to increase operating system security by isolating a portion of memory from the rest of the operating system.

Beginning with vSphere 6.7, and using virtual hardware version 14 or later, ESXi can support VBS in the guest operating system. However, because VBS uses virtualization, this results in a virtualization layer within a virtualization layer, sometimes called "nested virtualization," which can impact performance.

For the best performance using VBS, consider these points:

- If Hypervisor-Based Code Integrity will be activated in the guest OS, performance can be improved if the underlying hardware has Mode-Based Execution (MBX), a feature introduced in Intel Kaby Lake and Sky Lake processors.
- VMCS shadowing, a feature introduced in Intel Haswell processors, reduces the impact of nested virtualization.

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 Monitoring Guest Operating System Performance in *vSphere Monitoring and Performance* and Configure CPU Resources of a Virtual Machine in *vSphere Virtual Machine Administration* (then scroll down to "Activate Virtual CPU Performance Counters").
- 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 the white paper Timekeeping in VMware Virtual Machines.

- 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 Client, VMware Tools, `esxtop`, or `resxtop`) can avoid these problems. For information about using `esxtop` and `resxtop` see Performance Monitoring Utilities: resxtop and esxtop in *vSphere Monitoring and Performance*.

Guest Operating System CPU Considerations

This section addresses CPU considerations in the guest operating system.

- 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.)

Side-Channel Vulnerability Mitigation in Guest Operating Systems

As described in "Side-Channel Vulnerabilities" on page 11, mitigation for some side-channel vulnerabilities takes place in the guest operating system. These mitigations address serious security vulnerabilities, but they can also have a significant impact on performance, especially on systems that are CPU resource constrained.

If this performance impact might be an issue for your environment, you can reduce it by choosing recent CPUs that mitigate some of these vulnerabilities in hardware (again, as described in "Side-Channel Vulnerabilities" on page 11).

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 29.

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 29), 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-activated 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.

- Beginning with 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 cores 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 and later, 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 *vSphere Resource Management*.

  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 (go to https://community.broadcom.com/flings/home and search for "VMCO").

- By default, vNUMA is activated only for virtual machines with more than eight vCPUs. This feature can be activated 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 activate 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 To change this variable through the vSphere 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 **numa.vcpu.min** and configure it as you wish. If the variable isn't present, click **ADD CONFIGURATION PARAMS** and enter it as a new parameter.
  4. Click **OK** to close the **Configuration Parameters** window.
  5. 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 *vSphere Resource Management*.

- CPU Hot Add is a feature that allows the addition of vCPUs to a running virtual machine. Activating this feature, however, deactivates vNUMA for that virtual machine, resulting in the guest OS seeing a single vNUMA node. Without vNUMA support, the guest OS has no knowledge of the CPU and memory virtual topology of the ESXi host. This in turn could result in the guest OS making sub-optimal scheduling decisions, leading to reduced performance for applications running in large virtual machines.

For this reason, activate CPU Hot Add only if you expect to use it. Alternatively, plan to power down the virtual machine before adding vCPUs, or configure the virtual machine with the maximum number of vCPUs that might be needed by the workload. If choosing the latter option, note that unused vCPUs incur a small amount of unnecessary overhead. Unused vCPUs could also cause the guest OS to make poor scheduling decisions within the virtual machine, again with the potential for reduced performance.

For additional information see VMware KB article 2040375.

vNUMA and PMem

The performance of Persistent Memory (PMem; described in "Persistent Memory (PMem)" on page 14) in NUMA systems can potentially be improved if vNUMA nodes are manually pinned to physical NUMA nodes. This way a virtual machine running in a particular vNUMA node is guaranteed to remain on the same physical NUMA node, and thus to have local access to the same PMem.

This also depends on the PMem being correctly associated with a physical NUMA node. For more information about this, see "Persistent Memory (PMem) in NUMA Systems" on page 31.

Guest Operating System Memory Considerations

As described in "2MB Large Memory Pages" on page 37, ESXi can make large memory pages available to the guest operating system.

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.

Guest Operating System Storage Considerations

The virtual storage adapter presented to the guest operating system can influence storage performance, as can that device's 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 8.0 Update 3 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 25.

  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 introduced 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 extremely low latency flash and non-volatile memory based storage, it isn't best suited for slow disk-based storage.

  Compared to virtual SATA devices, the vNVMe virtual storage adapter accesses local PCIe SSD devices with much 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 size I/O requests 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.
- If your storage subsystem uses 4KB native (4Kn) or 512B emulation (512e) drives, you can obtain the best storage performance if your workload issues mostly 4K-aligned I/Os. For more information on this subject, see VMware KB article 2091600 or the "Device Sector Formats" subsection of Viewing Storage Devices Available to an ESXi Host in the *vSphere Storage Guide*.

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; for optimal performance these adapters should be used for any guest operating system that supports them.

The VMXNET family of paravirtualized virtual network adapters includes:

- The VMXNET virtual network adapter.
- The VMXNET2 virtual network adapter (also called "Enhanced VMXNET"). This adapter is based on the VMXNET adapter, but 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 adapter, along with several new ones.

The PVRDMA virtual network adapter, introduced in ESXi 6.5, supports remote direct memory access (RDMA) between virtual machines on the same ESXi host or—if both hosts have compatible network hardware—between virtual machines on different ESXi hosts.

Lastly, there's a hybrid virtual network adapter, the Flexible virtual network adapter. This adapter 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 49.

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 the Virtual Machine Network Configuration section of *Virtual Machine Administration*.

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 Because Microsoft Windows doesn't include a VMXNET3 driver, you'll need to use a different virtual network adapter (typically E1000e or E1000) during operating system installation. After the OS is installed, you should install VMware Tools (which includes a VMXNET3 driver), then add a VMXNET3 virtual network adapter.

- For guest operating systems in which VMXNET3 is not supported, we recommend using the E1000e virtual network adapter.
- If E1000e is not an option, 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 activate 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 activated by default in the VMkernel but is supported in virtual machines only when they are using the E1000, E1000e, VMXNET2, or VMXNET3 device. TSO can improve performance even if the underlying hardware does not support TSO.
- Similarly, in ESXi Large Receive Offload (LRO) is activated by default in the VMkernel, but is supported in virtual machines only when they are using the VMXNET2 or 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 activated in the guest operating system.

  For more information about configuring LRO, see the Large Receive Offload section of *vSphere Networking*.

- 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 and later, Windows 7 and 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 activated, 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 activated, 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 both Windows and 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). This setting is changed on the advanced driver configuration tab within the guest operating system.

- Virtual hardware version 17 (supported by vSphere 7.0 and later) includes the VMXNET3 version 6 virtual NIC. This new release of the VMXNET3 NIC now supports up to 32 receive and transmit queues (increased from a maximum of 8 in the previous version), though the default when RSS is activated is 8 or the number of vCPUs in the virtual machine, whichever is lower. (Note, however, that with this new vNIC the number of queues no longer has to be a power of two.)

  To use this new feature, in addition to the virtual machine running on virtual hardware version 17 or later, the guest OS needs to be using the VMXNET3 version 6 driver, currently supported only in Linux.

  To provide more than 8 queues, configure the `.vmx` file values `ethernet`*`X`*`.maxTxQueues` and `ethernet`*`X`*`.maxRxQueues` (where *X* is the number of the virtual NIC to configure). The maximum number of queues can be no greater than the number of vCPUs in the virtual machine.

Increasing the number of queues also increases processor overhead and, for most environments, the VMXNET3 default of no more 8 queues provides the best performance, so this configuration should be used only after other performance options have been tried. However, in unusual cases, where the number of queues are identified as a bottleneck for performance, these options can be used to increase the maximum number of queues. In these situations we recommend increasing the number of queues gradually (for example, from 8 to 12, then 16, then 20, and so on), then evaluating the performance at each setting, rather than increasing directly to 32.

4 Virtual Infrastructure Management

This chapter provides guidance regarding infrastructure management best practices. Most of the suggestions included in this section can be implemented using a vSphere 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 white paper VMware vCenter Server Performance and Best Practices. (Though written for vSphere 6.0, much of the information in this paper is still relevant.)

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 33) and migration overhead. In a DRS-activated 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 vCenter Server 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 71.
- 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 Clients. Exceeding the maximums specified in Configuration Maximums for vSphere 8.0 Update 3, 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.
- 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 should provide good performance while avoiding unnecessary memory commitment. If, however, you will have inventory sizes well into the large environment range (as defined in the System Requirements for the vCenter Server Appliance section of *vCenter Server Installation and Setup*), 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

- If the vCenter appliance 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 40 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 deactivate 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 still applicable).

- 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 `/etc/vmware-vpx/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.

- To connect to the VCDB database (for example, to check the size of the tables), follow the instructions in VMware KB article 2147285.
- If you observe query slowness, performance might be improved by running Vacuum and Analyze on the VCDB database as follows:

  a Connect to the VCDB database (as described in VMware KB article 2147285).

  b Update the table statistics within VCDB by running the following command:
  VACUUM ANALYZE;

  c Exit the VCDB database (also as described in VMware KB article 2147285).

PostgreSQL (vPostgres) Database Recommendations

vCenter Server uses a PostgreSQL (vPostgres) database. Though many database optimizations are done automatically, 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 Client, these alarms can also be configured to send email notifications.
- A vCenter Server plug-in, `pgtop`, can be used to monitor the PostgreSQL database and can help diagnose performance issues. For more information see vSphere Blog post Getting Comfortable with vPostgres and the vCenter Server Appliance – Part 3.

VMware vSphere Management

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

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

vSphere Clients

The vSphere Client can be used to control and configure vCenter Servers, ESXi hosts, and virtual machines.

The vSphere Client back end, called the vSphere Client Server, is a Java application. The vSphere Client front end, called simply the vSphere Client, is an HTML5-based client running in a web browser pointed to the Java-based Application Server.

vSphere Client Back-End Performance Considerations

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

NOTE Starting with vSphere 6.5, the inventory service was replaced by `vpxd-svcs`, a lighter-weight service that contains logic for authorization, roles, and privileges.

- For optimal performance, make sure you provide the vSphere Client Server with sufficient CPU, memory, and storage resources for your deployment size and usage patterns. These resource requirements are included as part of the vCenter Server requirements listed in the System Requirements for the vCenter Server Appliance section of *vCenter Server Installation and Setup*. The requirements are also described during the installation process.
- Performance of the vSphere 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 Clients in use
  - The usage patterns of vSphere Client users
- Installing plug-ins in the vSphere Client Server might cause higher memory usage on the system running the vSphere Client Server. It's therefore a good practice, after any plug-ins are installed and activated, to monitor the vSphere 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-ui`
  - 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 vsphere-ui`

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

The most straightforward way to increase the memory allocated to the vSphere 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 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 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 Client Server maximum heap size:

- In a vCenter Server Appliance, run:
  `cloudvm-ram-size -C XXX vsphere-ui`
  (where *XXX* is the desired heap size in MB).
  For more information, run `cloudvm-ram-size -h`.

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

- In a vCenter Server Appliance, run:
  ```
  service-control -stop vsphere-ui
  service-control -start vsphere-ui
  ```

- Keep in mind that there are limits to the number of concurrent operations at both the vCenter Server level and the ESXi host level.
- A number of advanced configuration options can influence the performance of the vSphere 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-ui`
  - In Windows:
    ```
    %ALLUSERSPROFILE%\VMware\vCenterServer\cfg\vsphere-ui
    %ProgramData%\VMware\vsphere client
    ```

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

Advanced option name	Description	Default Value
live.updates.enabled	Whether or not to activate live refresh in **Recent Tasks** view.	true
live.updates.alarms.enabled	Whether or not to activate live refresh in **Alarms** view.	true
live.updates.navtree.enabled	Whether or not to activate 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 activate live refresh in lists.	true

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

Advanced option name	Description	Default Value
live.updates.objectstate.enabled	Whether or not to activate 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
session.timeout	Time (in minutes) before a user is automatically logged out.	120
alarms.refresh.rate	Time (in seconds) between automatic refreshes of the alarms list. Must be between 10 and 600. A value of -1 deactivates automatic refresh of alarms.	60
dataservice.timeoutSeconds	Time (in seconds) before an error is displayed in the UI if a data adapter fails to respond.	120
dataservice.connectionTimeoutSeconds	Time (in seconds) before an error is displayed in the UI if the dataservice connection is lost.	10
sso.pending.password.expiration.notification.days	Number of days before expiration of an SSO password that a notification is displayed in the UI.	30

vSphere Client Front-End Performance Considerations

The vSphere Client front-end is an HTML5-based application running in a web browser on the user's system. This subsection describes how to obtain the best performance from the vSphere Client front-end.

- For improved performance—as well as increased security—install Certificate Authority (CA) signed SSL certificates in the system from which the vSphere Client is run. For further information see VMware KB article 2111219.
- For the best vSphere 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 Client performance. For the best performance, the network latencies between the vSphere Client running on the user's system and the vSphere 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 Client back-end (such that the system has a low-latency connection to the back-end), and run the vSphere Client front-end on that system, remotely accessing it from the more distant location using a remote protocol such as RDP.

- Using a 64-bit browser to view the vSphere Client allows the allocation of more memory, potentially improving performance.

- When possible, use the search function instead of navigating to managed objects. The vSphere Client is designed for the use of inventory search to find managed objects (clusters, hosts, virtual machines, datastores, tags, 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.

Tagging in vSphere

vSphere tags can be used to organize inventory objects such as virtual machines, hosts, datastores, and so on. If you'll be using vSphere tags, you can find performance guidance in the blog post vSphere 7.0 U1 Tagging Performance Best Practices and the associated white paper VMware vSphere 7.0 U2 Tagging Best Practices. (Though both of these were written about a prior vSphere version, the information remains relevant.)

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.

VMware vMotion and Storage vMotion

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

VMware vMotion Recommendations

Consider the recommendations in this section for the best vMotion performance. For more detail about vMotion architecture and performance, see the white paper vMotion Innovations in VMware vSphere 7.0 U1. (Though written about a previous version, much of the material still applies.)

- ESXi 8.0 Update 3 supports virtual hardware version 21. Because virtual machines running on hardware version 21 can't run on prior versions of ESXi, such virtual machines can be moved using VMware vMotion only to other hosts running ESXi 8.0 Update 2 or later. ESXi 8.0 Update 2 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 ESXi versions with which they are compatible.
- vSphere 8.0 introduced vSphere Unified Data Transport (UDT). We strongly recommend activating this feature in order to dramatically increase performance of cold migration operations on powered-off virtual machines and virtual machines with snapshots as well as cloning of virtual machines. UDT comes into play only when both the source and destination ESXi hosts are running vSphere 8.0 or later and have a provisioning network assigned.

  For more information about UDT, including how to activate it, see vSphere vMotion Unified Data Transport and the video vSphere 8 Enhancements for VM Cold Migrations, vMotion, and Cloning.
- Starting with version 6.5, vSphere supports 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 or later). If a virtual machine running on ESXi 6.5 or later is migrated to a host running an earlier ESX/ESXi version, vMotion traffic will not be 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).

  For more detail about encrypted vMotion architecture and performance, see the white paper VMware vSphere Encrypted vMotion Architecture, Performance, and Best Practice (though written about a previous version, much of the material still applies).
- vMotion performance will increase as additional network bandwidth is made available to the vMotion and provisioning networks. This is especially true with the new UDT feature introduced in vSphere 8.0 (described above), that improves utilization of available network bandwidth. Consider providing 10Gb/s or faster vMotion network and provisioning network interfaces for maximum vMotion performance.

  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.
- We'd previously recommended configuring three vMotion vmknics on networks faster than 10Gb/s in order to achieve full line rate for vMotion operations. As of vSphere 7.0 Update 2, this is no longer necessary; the vMotion process automatically creates the number of streams needed in order to fully utilize the bandwidth of these faster physical NICs. For further information, see the blog post Faster vMotion Makes Balancing Workloads Invisible.
- Keep in mind that there are limits to the number of concurrent vMotion operations. For information about these limits, see the blog post vCenter Limits for Concurrent vMotion.

- 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 35.
- 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.
  - 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

Consider the recommendations in this section for the best Storage vMotion performance.

- 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 "Storage Hardware Considerations" on page 16).

VMware Cross-Host Storage vMotion Recommendations

Cross-host Storage vMotion allows a virtual machine to be moved simultaneously across both hosts and datastores. Consider the recommendations in this section for the best Cross-host Storage vMotion performance.

NOTE The performance recommendations in this section also apply to Cross vCenter vMotion, a feature that allows virtual machines to be moved from one vCenter Server instance to another.

- 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.

- Cross-host Storage vMotion can perform up to four simultaneous disk copies, as described for Storage vMotion in "VMware Storage vMotion Recommendations" on page 78.
- 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 potentially some snapshots), Cross-host Storage migration will use UDT or the Network File Copy (NFC) service. NFC will use VAAI or the source host's storage interface (as described above for powered-on virtual machines) if either of these are available. If they're not available, NFC will use the network designated for management or provisioning; UDT will use the network designated for provisioning if UDT is activated for that network.

  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

- vSphere Cluster Services (vCLS) is a feature designed to ensure that cluster services, such as vSphere DRS and vSphere HA, are available to maintain the resources and health of the workloads running in the clusters regardless of the availability of the vCenter Server instance. As of vSphere 8.0 Update 3, cluster services still require *both* the vCLS deployed agent virtual machines (vCLS VMs) and the vCenter Server, as not all functionality has moved to the vCLS VMs.

  NOTE If you attempt to activate DRS on a cluster where there are issues with the vCLS VMs, or if issues with the vCLS VMs develop after DRS is activated, a warning message will be displayed on the Cluster Summary page. These issues must be resolved before DRS can be activated or, if already activated, for DRS to operate.

- 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.
- As of vSphere 7.0, the cluster-level balance metric, which indicated a cluster's balance or imbalance, has been replaced by a **Cluster DRS Score**, which is an indicator of cluster health rather than cluster load balance (for more information on this, see "DRS Performance Tuning" on page 84).
- When a cluster includes hosts that contain Persistent Memory in Memory Mode (described in "Persistent Memory (PMem)" on page 14) and has DRS activated and fully automated, DRS will monitor memory usage to detect potential memory performance issues.

  If the active memory utilization of the host as a percentage of the size of the DRAM cache reaches a threshold level, DRS will proactively attempt to find suitable migrations that would avoid memory performance issues and then move some VMs in order to balance the load. If DRS doesn't find suitable migrations, the relevant preconfigured default alarms will be raised.

  For information about vSphere Memory Monitoring and Remediation (vMMR), where memory utilization thresholds and alarms can be configured, see "Persistent Memory (PMem)" on page 38.

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 CPUs. Thus with 'incompatible CPU' heterogeneous systems, the opportunities DRS has to improve the usable resource availability for virtual machines 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 KB article 1003212.

- The more vMotion compatible ESXi hosts DRS has available, the more choices it has to recommend vMotions to improve usable resource availability for virtual machines in 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 VMware 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 computing vMotion recommendations.

- Just as in previous versions of vSphere, 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.

  Starting in vSphere 7.0, however, when computing vMotion recommendations DRS considers granted memory (that is, the total RAM available to a virtual machine) when evaluating a virtual machine's memory demand. This means that now over-provisioning a virtual machine's memory could even more significantly constrain the DRS migration options than in previous vSphere versions.

- If a cluster is in DRS fully automated mode, only virtual machines that are *also* in DRS fully automated mode will be considered for recommended migrations. Thus setting virtual machines on such clusters to DRS fully automated mode provides DRS a broader range of recommendation options.

- Powered-on virtual machines consume memory resources—and typically 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. In order to allow DRS to best manage resource pools, especially in deployments with varying inventory, we recommend activating a new option introduced in vSphere 7.0, **Scalable Shares**. This option, which can be activated at the cluster or resource pool level, brings dynamic and relative entitlements to resource pools and virtual machines, based on their share value settings. This allows resource pools and virtual machines to be made siblings in a hierarchy without creating a dilution problem.

  When Scalable Shares is *deactivated*, however, we recommended 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 with Scalable Shares deactivated, resource pools are assigned default 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 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 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 Cluster Sizing and Resource Settings

- On a new installation of vSphere 7.0 Update 1 or later (or when a vCenter Server deployment is upgraded to vSphere 7.0 Update 1 or later), vCLS automatically creates agent virtual machines (vCLS VMs).

  The version of vCLS used prior to vSphere 8.0 Update 3 is now called External vCLS. vSphere 8.0 Update 3 introduces Embedded vCLS, which is used instead of External vCLS when the vCenter version is 8.0 Update 3 or later and at least one host in the cluster is running ESXi 8.0 Update 3 or later.

  Either version of vCLS will have only a minimal impact on overall cluster capacity:

  - External vCLS creates a maximum of three vCLS VMs per cluster and consumes no more than about 400MB of memory and 400MHz of CPU per cluster. Table 4-2 provides specifications for each External vCLS VM.

    Table 4-2. External vCLS Agent Virtual Machine Specifications

Resource	Allocated to Each vCLS VM
Memory	128MB
Memory reservation	100MB
Swap size	256MB
vCPUs	1
CPU reservation	100 MHz
Hard disk	2GB (thin provisioned)
Network interfaces	0 (the VM has no NIC)
VMDK size	Approximately 245MB
Storage space	Approximately 480MB

  - Embedded vCLS creates a maximum of two vCLS VMs per cluster and consumes no more than about 400MB of memory and 200MHz of CPU per cluster. Table 4-3 provides specifications for each Embedded vCLS VM.

    Table 4-3. Embedded vCLS Agent Virtual Machine Specifications

Resource	Allocated to Each vCLS VM
Maximum memory	200MB
Memory reservation	160MB
Swap size	0MB
vCPUs	1
CPU reservation	100 MHz
Hard disk	0GB
Network interfaces	0 (the VM has no NIC)
VMDK size	0MB
Storage space	0.1MB

- Exceeding the maximum number of hosts, virtual machines, or resource pools specified in VMware Configuration Maximums for each DRS cluster 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 recommend vMotions.
  - 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.

- DRS includes NetIOC ("Network I/O Control (NetIOC)" on page 47) 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 77, when a vMotion operation is started, ESXi reserves some CPU resources for that operation.

DRS Performance Tuning

- Prior to vSphere 7.0, the migration threshold for fully-automated DRS adjusted the aggressiveness of the DRS algorithm. Starting in vSphere 7.0, however, the DRS migration threshold is workload focused, with each threshold tuned for a different kind of workload:
  - Level 1 — DRS recommends only mandatory moves; that is, vMotions to fix rule violations or when a host is entering Maintenance Mode. At this level DRS does not recommend vMotions to improve resource availability for the virtual machines.
  - Level 2 — Tuned for highly stable workloads; DRS will recommend vMotions only when the cluster is at or close to contention.
  - Level 3 — The default level; tuned to work well for mostly stable workloads.
  - Level 4 — For bursty workloads, when occasional spikes in the virtual machine resource utilization might require DRS to react to sudden load changes.
  - Level 5 — For dynamic workloads, where virtual machine resource utilization varies significantly and DRS needs to react to the workload changes every time.
- In order to simplify cluster management, vSphere has some advanced options that customize DRS behavior to better suit varied cluster needs:
  - The **VM Distribution** option causes DRS to consider distributing virtual machines evenly across the hosts in the cluster for availability purposes.
  - Starting in vSphere 7.0, in addition to CPU and memory usage, DRS now also considers network bandwidth usage to compute vMotion recommendations. The advanced option **Network Congestion** can be used to tune how aggressively DRS attempts to balance network bandwidth utilization.

  These options are available in the vSphere Client (select a cluster, click the **Configure** tab, expand **Services**, select **vSphere DRS**, click the **EDIT** button, and click the **Additional Options** tab).
- As mentioned above, vSphere now uses **Cluster DRS Score** as an indication of cluster health. Higher values for **Cluster DRS Score** are better, though the exact value is not critical; instead, the range is what matters. Scores above 80% indicate that there is no performance impact, scores below 80% indicate potential performance degradation in one or more cluster workloads. Lower scores indicate a higher degree of potential performance impact for the workloads.

  To achieve a higher Cluster DRS Score, you could relax some rules, adjust the migration threshold, or reduce the resource demands on the cluster.

  There are a variety of reasons that a DRS cluster might have a low Cluster DRS Score. 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 being network saturated.
- The cluster not having enough usable resources to meet virtual machine demands.

- Predictive DRS can be configured to receive predictions from VMware vRealize® Operations™ and use this information to make vMotion decisions.

NOTE This requires vRealize Operations version 6.4 or later. For more detail, see the blog post Predictive DRS with vRealize Operations 6.4.

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 there is a significant variation over time in the composite virtual machine resource demands; 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.

Starting with vSphere 7.0, DPM calculates virtual machine memory demand based on granted memory, which is typically stable over time. Thus, though DPM considers both CPU and memory demands, it largely responds to changes in CPU demand.

Because DPM uses DRS, most DRS best practices (described in "VMware Distributed Resource Scheduler (DRS)" on page 80) 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 31.

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 31). 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-activated 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 activated 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 activated 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 deactivated for individual hosts in a cluster even when DPM is activated 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 those 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 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 put into standby mode only if the memory demand of the virtual machines in the cluster is low. Starting with vSphere 7.0, the memory demand of virtual machines is considered low only if their granted memory is low.
- 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

- An advanced option for DPM introduced in vSphere 7.0, `DPMPowerOffInterval`, sets the minimum time DPM waits between calculations to evaluate if hosts should be considered for placement into standby mode. The default is 30 minutes, the maximum is 12 hours; this option is configured independently for each cluster.
- DPM can be activated or deactivated on a predetermined schedule using **Scheduled Tasks** in vCenter Server. When DPM is deactivated, 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 activated 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 the blog post Predictive DRS with vRealize Operations 6.4.

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 activated, 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 activated). 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 activated. These factors should be considered when configuring HA and DPM.
- If VMware HA is activated 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 deactivated.

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.

NOTE Storage I/O Control will be deprecated in a future vSphere release. See the notice "Deprecation of Storage DRS Load Balancer and Storage I/O Control (SIOC)" in the VMware vCenter Server 8.0 Update 3 Release Notes and additional detail in Storage DRS & Storage IO Control Deprecations.

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 42.

With Storage I/O Control deactivated (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 activated, 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** 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 activate Storage I/O Control, from the vSphere 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 and statistics collection** 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 Control-Enabled SSD Datastores (though written for a prior version of vSphere, some of the concepts in this paper are still relevant).

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.

NOTE Storage DRS will be deprecated in a future vSphere release. See the notice "Deprecation of Storage DRS Load Balancer and Storage I/O Control (SIOC)" in the VMware vCenter Server 8.0 Update 3 Release Notes and additional detail in Storage DRS & Storage IO Control Deprecations.

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.
- 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, datastores, virtual machines, or applications within virtual machines, then, in the event a failure is detected, restarting virtual machines on alternate hosts or resetting them on the same host.

VMware High Availability in General

- When vSphere HA is activated 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 using an algorithm that takes into account various properties of the hosts, such as the number of datastores to which they are connected. 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.
- Activating 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 activated, 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.
- In the event of a failure, HA queries DRS running on vCenter for its recommendation about which hosts the virtual machines should be restarted on. If DRS doesn't supply a recommendation, HA falls back to a simpler algorithm.

For further details about HA, see the Creating and Using vSphere HA Clusters section of *vSphere Availability*.

Virtual Machine Component Protection (VMCP)

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

For further details about VMCP, including how to activate the feature, see the VM Component Protection section of *vSphere Availability*.

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 90) are relevant to FT as well.

When FT is activated 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 performed 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 run 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.
- Be careful when placing more than four FT virtual machines on a single host, or having more than eight total FT vCPUs on a host. Doing so increases the possibility of saturating the FT logging NIC and increases 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.
  - Activate 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.

VMware vSAN

VMware 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.

NOTE For a more in-depth discussion of setting up a vSAN deployment, see the VMware vSAN Design Guide.

vSphere 8.0 introduced vSAN Express Storage Architecture (vSAN ESA) while also still supporting vSAN Original Storage Architecture (vSAN OSA). When configuring vSAN in a new cluster, vSphere 8.0 Update 3 allows you to choose vSAN OSA or vSAN ESA (in either case provided you're running on supported hardware). In-place conversion from OSA to ESA is not supported. We recommend vSAN ESA when running on hardware that supports it, but the performance of vSAN OSA in vSphere 8.0 and later has also been improved relative to previous versions.

For more information about vSAN ESA, see An Introduction to the vSAN Express Storage Architecture. For a collection of short articles on vSAN ESA, see Blog Posts on vSAN ESA. For the most up-to-date performance guidance about vSAN ESA, see Performance Recommendations for vSAN ESA.

Hybrid versus All-Flash vSAN

NOTE vSAN ESA supports only all-flash deployments.

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

- In hybrid vSAN OSA 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 OSA 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 OSA deployments are very fast, all-flash deployments are faster still. In addition, all-flash vSAN supports RAID-5, RAID-6, deduplication, and compression. By improving storage efficiency, 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 hybrid vSAN OSA, 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

NOTE For guidance on hardware selection for vSAN ESA, see the VMware vSAN Design Guide.

- In an all-flash vSAN OSA deployment, 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 high performance class NVMe drives—for the caching tier can significantly improve performance, especially for large I/O workloads, even when the capacity tier uses lower-performance SSDs.

Hardware Selection and Layout for vSAN in General

- vSAN performs most consistently when disk resources are distributed relatively evenly across the ESXi hosts that make up the vSAN cluster.
- Disks in vSAN OSA are organized into disk groups, with each disk group consisting of one cache disk and one or more capacity disks. Increasing the number of disk groups will typically increase vSAN performance.

vSAN Network Considerations

- vSAN over RDMA can improve network CPU efficiency compared to vSAN over TCP. This can free up CPU resources that could then be used for other purposes, such as virtual machines. However, RDMA support for vSAN should be activated only if your environment has the correct network hardware and configurations for RoCE v2 and you can ensure that the RoCE v2 traffic is lossless; otherwise, vSAN storage performance could be significantly reduced. For more information, see the blog post vSAN 7 Update 2 RDMA Support and the Designing the vSAN Network section of *vSAN Planning and Deployment*.
- Deployments using vSAN OSA require 10Gb/s or faster Ethernet. Deployments using vSAN ESA require 25Gb/s or faster Ethernet.
- A smoothly functioning network is important to obtain the best vSAN performance. Network issues, such as pause frames and dropped packets, can significantly reduce vSAN performance.
- Jumbo frames can help vSAN performance in some cases:
  - Because jumbo frames allow 4K and 8K blocks to fit in a single packet, workloads with a high proportion of these smaller block sizes might benefit from jumbo frames.
  - For larger block sizes jumbo frames typically won't offer much improvement in vSAN performance but, as long as the network is properly configured, they shouldn't hurt performance either.

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 (relevant only for hybrid vSAN OSA)
  - 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 VMware vSAN Design Guide.
- End-to-end software checksum is activated by default for vSAN, but can be deactivated on a per virtual machine or per object basis. Deactivating 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 80).
- In vSAN OSA, deduplication and compression (both 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, read-heavy workloads, and small block sizes, workloads with these characteristics are especially well-suited to use these features.

- In vSAN ESA, compression is activated by default. It can compress data as much as 8:1 and do so with relatively small CPU cost. Thus with vSAN ESA and compressible data, leaving compression activated can actually benefit write throughput. However, for applications that do their own compression (VMware Tanzu™ Greenplum®, video, etc.) it might be desirable to save even that small CPU cost by deactivating compression.
- In vSAN OSA, 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.
- In vSAN ESA, RAID-1 is not recommended except for 2-node clusters (only because 2-node clusters don't support RAID-5/6). This is because with vSAN ESA, RAID-5 is actually faster than RAID-1, and RAID-6 (tolerating up to two failures; FTT=2) is faster than FTT=2 RAID-1. Thus with vSAN ESA, RAID-5/6 provides both the best performance *and* the best space efficiency, making it a clear choice without trade-offs.
- A few very specific workloads (certain configurations of Hadoop, for example) have some data replication built in. When running such workloads, you might choose to set Failures to Tolerate (FTT) to 0 (in other words, perform no data replication in vSAN).

  vSphere 6.7 introduced a new vSAN Host Pinning storage policy for this purpose. The new policy is available only by request, however, and is not included in the standard version of vSAN.

vSAN Encryption

NOTE This section addresses encryption in vSAN OSA. For information about encryption in vSAN ESA, see Cluster Level Encryption with the vSAN Express Storage Architecture and vSAN Encryption Services.

In vSAN OSA 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 45.) 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 from vSAN encryption.
- AES-NI (see "AES-NI Support" on page 13) can significantly minimize additional CPU utilization due to encryption. Nearly all modern processors not only support AES-NI, but include an optimized implementation. It can still be necessary, however, to activate AES-NI in BIOS. In some cases a BIOS upgrade might be required for AES-NI support.
- When a host reboots, encrypted vSAN disk groups aren't mounted until a key is received from the key management server.

VMware vSphere Virtual Volumes (vVols)

VMware vSphere 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.

vVols 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.
- vVols performance varies significantly between storage hardware vendors. Before choosing storage hardware, confirm that it will provide the vVols performance you expect.
- When choosing hardware, be mindful of the limitations imposed by the array implementation. A vVols-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 vVols-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.

  If your VASA provider runs in a virtual machine:

  - Don't store it on vVols storage and make sure it can't migrate to vVols storage.
  - Consider using vSphere HA (see "VMware vSphere High Availability" on page 90) to protect it.
  - Implement an appropriate backup plan for it.
- Don't store your vCenter Server on vVols storage; to avoid circular dependencies it should instead be on a VMFS or NFS datastore.
- vVols using iSCSI should follow iSCSI best practices, such as:
  - Use port binding when appropriate.
  - Avoid configurations that require network routing to reach the iSCSI array.
  - Configure a dedicated VLAN or subnet.
  - Provide redundant host interfaces.
  - If it's necessary to share network links with non-iSCSI traffic, make sure the network has sufficient performance (both bandwidth and latency) for both the iSCSI traffic and its other traffic.
- As of vSphere 8.0, vVols supports NVMe over Fibre Channel arrays, though only for arrays that have been qualified with VMware.
- vVols 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.

vVols Workload Performance

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

vVols Management Operation Performance

The following performance recommendations apply to vVols management operations:

- Virtual machine provisioning operations on vVols range in performance relative to native NFS or VMFS on SAN:
  - Clone operations on vVols are significantly faster than on native NFS or VMFS on 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).
  - Destroy operations on vVols typically take slightly longer than VMFS on SAN, though this varies somewhat with different array vendors.
- vVols performance on snapshot operations varies between creation and deletion:
  - Snapshot creation on vVols will typically be similar to or slightly slower than on native NFS or VMFS on 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 for VMFS on SAN. The space freed by the snapshot deletion typically becomes available almost instantly on vVols, rather than gradually, as with VMFS, though this can vary by array vendor.

vVols I/O Operation Performance

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

- Although vVols are thin provisioned, they have similar performance to eager-zeroed thick-provisioned VMFS datastores; this means they have much higher throughput and lower latency for first writes than either thin-provisioned VMFS datastores or lazy-zeroed thick-provisioned VMFS datastores.
- 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-vVols 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). vVols snapshots typically become comparable to LUN snapshots on the array. For this reason, vVols 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.

vVols 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 activated for specific virtual disks. For example, it might make sense to activate deduplication (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 vVols disks, some might offer specific data services (for example deduplication, compression, or encryption) on individual vVols disks, and so on, all configured through SPBM.

- vVols storage containers are not physical objects in the storage array; they're simply an allocation quota of certain types of storage with certain capabilities activated. 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 vVols 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 vVols 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 containers or datastores.

- One of the performance benefits of vVols is that the I/O path for all a host's vVols uses one or two Protocol Endpoints (PEs). This means that, unlike when adding additional LUNs (which require a SCSI rescan), when adding additional vVols a rescan is not required because the PE will already have been scanned. This can also reduce boot times; instead of a potentially large number of LUNs, all of which need to be scanned at boot, with vVols only one or two PEs need to be scanned.

VMware vSphere Lifecycle Manager

VMware vSphere Lifecycle Manager enables centralized and simplified lifecycle management for VMware ESXi hosts. It incorporates the capabilities of vSphere Update Manager (which was part of prior vSphere versions) and adds both new capabilities and a new update model based on the desired state principle. Lifecycle Manager can be used to apply patches, updates, and upgrades to VMware ESXi hosts, firmware, VMware Tools and virtual hardware, and so on.

In addition to the material presented here, more information about vSphere Lifecycle Manager performance can be found in the VMware vSphere Update Manager Performance and Best Practices white paper. (Though written about a previous product, much of the information in this paper is still relevant.)

Lifecycle Manager General Recommendations

For the best performance from Lifecycle Manager, follow these recommendations:

- For the best Lifecycle Manager performance, make sure to have at least a 1Gb/s network connection between vCenter Server and the ESXi hosts.
- Make sure the vCenter Server Appliance (vCSA) database has enough space for both the vCenter inventory and the Lifecycle Manager data. Lifecycle Manager initialization needs about 150MB, and its database usage can increase by up to about 100MB per month.
- The amount of memory consumed by Lifecycle Manager is primarily affected by the size of the inventory, especially the number of virtual machines. Periodically monitor the resource usage by Lifecycle Manager (look for the `updatemgr` process and for any processes run as user `updatemgr`), 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.
- A Lifecycle Manager plug-in (about 7MB in size) will automatically be installed into the vSphere Client the first time the Client is connected to a vCenter Server Appliance running the Lifecycle Manager service.
- Upgrading VMware Tools is faster if the virtual machine is already powered on. Otherwise, Lifecycle Manager must power on the virtual machine before the VMware Tools upgrade, which could increase the overall latency.
- On the other hand, upgrading virtual machine *hardware* is faster if the virtual machine is already powered off. Otherwise, Lifecycle 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, Lifecycle 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.

- To configure ESXi properties at the cluster level, it can be most efficient to use vSphere Configuration Profiles. For details see "Using vSphere Configuration Profiles to Manage Host Configuration at a Cluster Level" in the *Managing Host and Cluster Lifecycle* document.

Lifecycle Manager Quick Boot Option

Lifecycle Manager offers a Quick Boot option on compatible hardware that reboots ESXi without also performing firmware and device initialization. To save time when remediating compatible ESXi hosts, make sure to activate this option.

To check if your system is compatible with Quick Boot, from a shell on an ESXi host run this command:
`/usr/lib/vmware/loadesx/bin/loadESXCheckCompat.py`

For more information about Quick Boot, see VMware KB article 52477.

Lifecycle Manager Cluster Remediation

Lifecycle Manager continues to support baselines and baseline groups as used by Update Manager in previous vSphere releases. Lifecycle Manager in vSphere 7.0, however, introduced the use of images to collectively manage hosts within a cluster. These update models are mutually exclusive; a cluster can have one or the other, but not both. A cluster using baselines can be converted to use images, but the reverse (conversion from using images to using baselines) is not supported.

- Cluster remediation is most likely to succeed when the cluster is not heavily utilized. Thus for heavily-used clusters, cluster remediation is best performed during off-peak periods. If this is not possible, it is best to suspend or power-off some virtual machines before beginning the operation.
- It's a good idea to generate a remediation pre-check report before remediation. Doing so will identify issues that might prevent successful remediation.

Lifecycle 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 for that host will not affect the bandwidth available in the network link.

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.

Starting with vSphere 7.0, VMware has changed its authentication mechanism recommendations, and as of vSphere 8.0 Update 1, vSphere now supports Okta federation. Okta or Active Directory Federation Services (ADFS) are preferred as the most secure authentication mechanisms, followed by AD over LDAP. Integrated Windows Authentication (IWA), while still supported, has been deprecated and will be removed in the next major release.

If you are using Okta or ADFS, ensure that the network path to, and responsiveness of, the OpenID Connect (OIDC) identity provider is optimized.

If you are using AD over LDAP or IWA, these recommendations will help performance:

- Single Sign-On Server performance depends heavily on the performance of the back-end identity sources. For the best identity source performance, follow the relevant best practices for the source you're using.
- In order to minimize search overhead, limit the number of identity source servers added to the Single Sign-On Server. Ideally, you would specify just a primary and a secondary for each domain.
- When adding 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.

In all cases, minimizing the total number of automated and scripted logins will reduce the load on your vCenter Server and, in most cases, speed up your scripts. To that end:

- When possible, reuse sessions or tokens.
- Audit logins to your vCenter Server to determine if they're all necessary or even still relevant. Look for things like:
  - a system logging in much more often than needed (for example, once per minute, when once per hour would be sufficient for its purpose),
  - an old or forgotten script, no longer serving its original purpose, but still logging in (or attempting to),
  - a forgotten (or restored) system attempting to log in with old credentials, failing, and entering a constant retry loop.
- Don't run scripts that immediately retry following a login failure. Instead make sure that scripts throttle login retries on failure.
- Configure backup software to log in only as often as necessary and, when feasible, to use local SSO accounts instead of AD.

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 "Storage Hardware Considerations" on page 16).

  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.

- Synchronizing content across a WAN can be slow. 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, called **Maximum Bandwidth Consumption**, affects the network usage of all Content Library file transfer operations, including library sync, deploy VM, capture VM, file download, and file upload.

- 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.

Kubernetes in vSphere

vSphere supports running Kubernetes workloads natively alongside virtual machines. This section provides performance guidance for deploying Kubernetes workloads on vSphere.

- Although on many Kubernetes platforms a CPU is defined as a *logical* core (that is, a hyper-thread), within vSphere Supervisor Clusters CPU refers to a *physical* CPU core. When moving Kubernetes objects from a Kubernetes cluster that uses the logical core definition to vSphere, this difference should be addressed by appropriately adjusting the CPU requests and limits for Kubernetes objects to achieve the same pod density.
- When deploying a pod on the Supervisor Cluster, vSphere with Kubernetes translates the CPU requests for the containers in that pod into an equivalent CPU frequency usage. This frequency usage, together with the CPU capacity, is considered when making pod admission and placement decisions. The CPU capacity of a Kubernetes node (that is, an ESXi host) on a Supervisor Cluster is calculated by simply aggregating the base frequency of all the cores in the host. Depending on the system's load, thermal, and cooling factors, there's often extra CPU capacity provided by features like frequency boost that's ignored by vSphere in these capacity calculations.

  If you deploy native pods on ESXi in the Kubernetes Guaranteed QoS class, you'll be able to reserve only the conservatively calculated CPU capacity described above and—because Guaranteed QoS class doesn't allow CPU overcommitment—achieve limited pod density. Kubernetes Burstable and BestEffort QoS classes, on the other hand, do allow CPU overcommitment, and therefore allow full utilization of CPU capacity and a higher pod density. As long as there is sufficient additional capacity provided by features like frequency boost, this overcommitment is unlikely to cause significant performance degradation.
- Similarly, if only the Guaranteed QoS class is used for the node definitions when deploying Tanzu Kubernetes Grid (TKG) cluster nodes on vSphere with Kubernetes, these nodes might not be able to use the additional CPU capacity provided by features like frequency boost. Choosing the BestEffort QoS class can help achieve better node density.