Performance Tuning Guidelines for Windows Server 2008 R2

May 16, 2011

Abstract

This guide describes important tuning parameters and settings that you can adjust to improve the performance and energy efficiency of the Windows Server® 2008 R2 operating system. This guide describes each setting and its potential effect to help you make an informed decision about its relevance to your system, workload, and performance goals.

This paper is for information technology (IT) professionals and system administrators who need to tune the performance of a server that is running Windows Server 2008 R2.

This information applies to the Windows Server 2008 R2 operating system.

References and resources discussed here are listed at the end of this guide.

The current version of this guide is maintained on the Web at:
http://msdn.microsoft.com/windows/hardware/gg463392.aspx

Feedback: Please tell us whether this paper was useful to you. Submit comments at:
http://go.microsoft.com/fwlink/?LinkId=102585

***Disclaimer**: This document is provided “as-is”. Information and views expressed in this document, including URL and other Internet Web site references, may change without notice. You bear the risk of using it. Some examples depicted herein are provided for illustration only and are fictitious. No real association or connection is intended or should be inferred.*

This document does not provide you with any legal rights to any intellectual property in any Microsoft product. You may copy and use this document for your internal, reference purposes.

Document History

Date	Change
May 13, 2011	• "Performance Tuning for Web Servers" – Updated guidance to reflect that Http.sys manages connections automatically. • "Performance Tuning for File Servers" – Fixed typos in NFS Server tuning parameter registry keys. • "Performance Tuning for Virtualization Servers" – Added information about Dynamic Memory tuning. • "Performance Tuning for TPC-E Workload" – Clarified tuning guidance. • "Resources" – Updated references.
October 15, 2010	• Throughout the paper – Clarified some explanations; clarified energy consumption vs. power consumption. • "Interrupt Affinity" – Added recommendation to use device-specific mechanism for binding interrupts, if supported by the driver model. • "Network-Related Performance Counters" – Added IPv6 and TCPv6. • "Performance Tuning for the Storage Subsystem" – Various minor updates throughout. • "Performance Tuning for File Servers" –Added guidance for NtfsDisableLastAccessUpdate; added "Tuning Parameters for NFS Server", "File Server Tuning Example", and "File Client Tuning Example". • "Performance Tuning for Remote Desktop Session Host" – Added references to two new white papers on capacity planning. • "Monitoring and Data Collection" (multiple sections) – Updated the list of counters to monitor. • "Performance Tuning for File Server Workload (SPECsfs2008)" – New section. • "Performance Tuning for SAP Sales and Distribution Two-Tier Workload" – Substantial updates to the whole section. • "Performance Tuning for TPC-E Workload" – New section. • "Resources" – A few additions and updates.
November 18, 2009	• Throughout the paper – Updated the references to the Remote Desktop Session Host (previously called Terminal Server); various minor edits. • "Choosing a Network Adapter" – Fixed a typo in the RSS registry entries. • "Performance Tuning for File Servers" – Added MaxMpxCt parameter information; updated the default maximum payload for the SMB redirector to 64 KB per request; added MaxCmds parameter information. • "Performance Tuning for Remote Desktop Session Host" – Added information about the settings used when you choose a connection speed. • "Resources" – Provided additional resources.
June 25, 2009	First publication.

Contents

Introduction

Windows Server® 2008 R2 performs well out of the box while consuming the least energy possible for most customer workloads. However, you might have business needs that are not met by using the default server settings. You might need the lowest possible energy consumption, or the lowest possible latency, or the maximum possible throughput on your server. This guide describes how you can further tune the server settings and obtain incremental performance or energy efficiency gains, especially when the nature of the workload varies little over time.

To have the most impact, your tuning changes should consider the hardware, the workload, the power budgets, and the performance goals of your server. This guide describes important tuning considerations and settings that can result in improved performance or energy efficiency. This guide describes each setting and its potential effect to help you make an informed decision about its relevance to your system, workload, performance, and energy usage goals.

Since the release of Windows Server 2008, customers have become increasingly concerned about energy efficiency in the datacenter. To address this need, Microsoft and its partners invested a large amount of engineering resources in developing and optimizing the features, algorithms, and settings in Windows Server 2008 R2 to maximize energy efficiency with minimal effects on performance. This paper describes energy consumption considerations for servers and provides guidelines for meeting your energy usage goals. Although “power consumption” is a more commonly used term, “energy consumption” is more accurate because power is an instantaneous measurement (Energy = Power *Time). Power companies typically charge datacenters for both the energy consumed (megawatt-hours) and the peak power draw required (megawatts).

Note: Registry settings and tuning parameters changed significantly from Windows Server 2003 and Windows Server 2008 to Windows Server 2008 R2. Be sure to use the latest tuning guidelines to avoid unexpected results.

As always, be careful when you directly manipulate the registry. If you must edit the registry, back it up before you make any changes.

In This Guide

This guide contains key performance recommendations for the following components:

- Server Hardware
- Networking Subsystem
- Storage Subsystem

This guide also contains performance tuning considerations for the following server roles:

- Web Servers
- File Servers
- Active Directory Servers

- Remote Desktop Session Host
- Remote Desktop Gateway
- Virtualization Servers (Hyper-V)
- File Server Workload (NetBench)
- File Server Workload (SPECsfs2008)
- Network Workload (NTttcp)
- Remote Desktop Services Knowledge Worker Workload
- SAP Sales and Distribution Two-Tier Workload
- TCP-E Workload

Choosing and Tuning Server Hardware

It is important to select the proper hardware to meet your expected performance and power goals. Hardware bottlenecks limit the effectiveness of software tuning. This section provides guidelines for laying a good foundation for the role that a server will play.

It is important to note that there is a tradeoff between power and performance when choosing hardware. For example, faster processors and more disks will yield better performance but can also consume more energy. See “Choosing Server Hardware: Power Considerations” later in this guide for more details about these tradeoffs. Later sections of this paper provide tuning guidelines that are specific to a server role and include diagnostic techniques for isolating and identifying performance bottlenecks for certain server roles.

Choosing Server Hardware: Performance Considerations

Table 1 lists important items that you should consider when you choose server hardware. Following these guidelines can help remove artificial performance bottlenecks that might impede the server’s performance.

Table 1. Server Hardware Recommendations

Component	Recommendation
Processors	Choose 64-bit processors for servers. 64-bit processors have significantly more address space, and 64-bit processors are required for Windows Server 2008 R2. No 32-bit editions of the operating system will be provided, but 32-bit applications will run on the 64-bit Windows Server 2008 R2 operating system. To increase the computing resources in a server, you can use a processor with higher-frequency cores, or you can increase the number of processor cores. If CPU is the limiting resource in the system, a core with 2x frequency typically provides a greater performance improvement than two cores with 1x frequency. Multiple cores are not expected to provide a perfect linear scaling, and the scaling factor can be even less if hyper-threading is enabled because hyper-threading relies on sharing resources of the same physical core. It is important to match and scale the memory and I/O subsystem with the CPU performance and vice versa. Do not compare CPU frequencies across manufacturers and generations because the comparison can be a misleading indicator of speed.

Component	Recommendation
Cache	Choose large L2 or L3 processor caches. The larger caches generally provide better performance and often play a bigger role than raw CPU frequency.
Memory (RAM) and paging storage	Increase the RAM to match your memory needs. When your computer runs low on memory and needs more immediately, modern operating systems use hard disk space to supplement system RAM through a procedure called paging. Too much paging degrades overall system performance. You can optimize paging by using the following guidelines for pagefile placement: • Place the pagefile and operating system files on separate physical disk drives. • Place the pagefile on a drive that is not fault-tolerant. Note that, if the disk fails, a system crash is likely to occur. If you place the pagefile on a fault-tolerant drive, remember that fault-tolerant systems often experience slower data writes because they write data to multiple locations. • Use multiple disks or a disk array if you need additional disk bandwidth for paging. Do not place multiple pagefiles on different partitions of the same physical disk drive.
Peripheral bus	In Windows Server 2008 R2, the primary storage and network interfaces must be PCIe. Choose servers with PCIe buses. Also, to avoid bus speed limitations, use PCIe x8 and higher slots for Gigabit Ethernet adapters.
Disks	Choose disks with higher rotational speeds to reduce random request service times (~2 ms on average when you compare 7,200- and 15,000-RPM drives) and to increase sequential request bandwidth. However, there are cost, power, and other considerations associated with disks that have high rotational speeds. The latest generation of 2.5-inch enterprise-class disks can service a significantly larger number of random requests per second compared to equivalent 3.5-inch drives. Store “hot” data – especially sequentially accessed data – near the “beginning” of a disk because this roughly corresponds to the outermost (fastest) tracks. Be aware that consolidating small drives into fewer high-capacity drives can reduce overall storage performance. Fewer spindles mean reduced request service concurrency and therefore potentially lower throughput and longer response times (depending on the workload intensity).

Table 2 lists the recommended characteristics for network and storage adapters for high-performance servers. These settings can help prevent your networking or storage hardware from being the bottleneck when they are under heavy load.

Table 2. Networking and Storage Adapter Recommendations

Recommendation	Description
WHQL certified	The adapter has passed the Windows® Hardware Quality Labs (WHQL) certification test suite.
64-bit capability	Adapters that are 64-bit-capable can perform direct memory access (DMA) operations to and from high physical memory locations (greater than 4 GB). If the driver does not support DMA greater than 4 GB, the system double-buffers the I/O to a physical address space of less than 4 GB.

Recommendation	Description
Copper and fiber (glass) adapters	Copper adapters generally have the same performance as their fiber counterparts, and both copper and fiber are available on some Fibre Channel adapters. Certain environments are better suited to copper adapters, whereas other environments are better suited to fiber adapters.
Dual- or quad-port adapters	Multiport adapters are useful for servers that have a limited number of PCI slots. To address SCSI limitations on the number of disks that can be connected to a SCSI bus, some adapters provide two or four SCSI buses on a single adapter card. Fibre Channel disks generally have no limits to the number of disks that are connected to an adapter unless they are hidden behind a SCSI interface. Serial Attached SCSI (SAS) and Serial ATA (SATA) adapters also have a limited number of connections because of the serial nature of the protocols, but you can attach more attached disks by using switches. Network adapters have this feature for load-balancing or failover scenarios. Using two single-port network adapters usually yields better performance than using a single dual-port network adapter for the same workload. PCI bus limitation can be a major factor in limiting performance for multiport adapters. Therefore, it is important to consider placing them in a high-performing PCIe slot that provides enough bandwidth.
Interrupt moderation	Some adapters can moderate how frequently they interrupt the host processors to indicate activity or its completion. Moderating interrupts can often result in reduced CPU load on the host but, unless interrupt moderation is performed intelligently, the CPU savings might increase latency.
Offload capability and other advanced features such as message-signaled interrupt (MSI)-X	Offload-capable adapters offer CPU savings that yield improved performance. For more information, see “Choosing a Network Adapter” later in this guide.
Dynamic interrupt and deferred procedure call (DPC) redirection	Windows Server 2008 R2 has functionality that enables PCIe storage adapters to dynamically redirect interrupts and DPCs. This capability, originally called “NUMA I/O,” can help any multiprocessor system by improving workload partitioning, cache hit rates, and on-board hardware interconnect usage for I/O-intensive workloads.

Choosing Server Hardware: Power Considerations

Although this guide focuses primarily on how to obtain the best performance from Windows Server 2008 R2, you must also recognize the increasing importance of energy efficiency in enterprise and data center environments. High performance and low energy usage are often conflicting goals, but by carefully selecting server components you can achieve the correct balance between them.

Table 3 contains guidelines for power characteristics and capabilities of server hardware components.

Table 3. Server Hardware Energy Saving Recommendations

Component	Recommendation
Processors	Frequency, operating voltage, cache size, and process technology all affect the energy consumption of processors. Processors have a thermal design point (TDP) rating that gives a basic indication of energy consumption relative to other models. In general, opt for the lowest-TDP processor that will meet your performance goals. Also, newer generations of processors are generally more energy efficient and may expose more power states for the Windows power management algorithms, which enables better power management at all levels of performance.
Memory (RAM)	Memory accounts for an increasing fraction of total system power. Many factors affect the energy consumption of a memory DIMM, such as memory technology, error correction code (ECC), bus frequency, capacity, density, and number of ranks. Therefore, it is best to compare expected power ratings before purchasing large quantities of memory. Low-power memory is now available, but you must consider the performance and cost trade-offs. If your server will be paging, then you should also factor in the energy cost of the paging disks.
Disks	Higher RPM means increased energy consumption. Also, new 2.5-inch drives require less than half the power of older 3.5-inch drives. For more information about the energy cost for different RAID configurations, see "Performance Tuning for Storage Subsystem" later in this guide.
Network and storage adapters	Some adapters decrease energy consumption during idle periods. This is an important consideration for 10-Gb networking adapters and high-bandwidth (4-8-Gb) storage links. Such devices can consume significant amounts of energy.
Power supplies	Increasing power supply efficiency is a great way to reduce energy consumption without affecting performance. High-efficiency power supplies can save many kilowatt-hours per year, per server.
Fans	Fans, like power supplies, are an area where you can reduce energy consumption without affecting system performance. Variable-speed fans can reduce RPM as system load decreases, eliminating otherwise unnecessary energy consumption.
USB devices	Windows Server 2008 R2 enables selective suspend for USB devices by default. However, a poorly written device driver can still disrupt system energy efficiency by a sizeable margin. To avoid potential issues, disconnect USB devices, disable them in the BIOS, or choose servers that do not require USB devices.

Component	Recommendation
Remotely managed power strips	Power strips are not an integral part of server hardware, but they can make a large difference in the data center. Measurements show that volume servers that are plugged in but have been ostensibly powered off may still require up to 30 watts of power. To avoid wasting electricity, you can deploy a remotely managed power strip for each rack of servers to programmatically disconnect power from specific servers.

Power and Performance Tuning

Energy efficiency is increasingly important in enterprise and data center environments and it adds another set of tradeoffs to the mix of configuration options.

The out-of-the-box experience in Windows Server 2008 R2 is optimized for excellent energy efficiency with minimum performance impact across a wide range of customer workloads. This section describes energy-efficiency tradeoffs, to help you make informed decisions if you need to adjust the default power settings on your server.

Calculating Server Energy Efficiency

When you tune your server for energy savings, you must consider performance as well. Tuning affects both performance and power, sometimes in disproportionate amounts. For each possible adjustment, consider your power budget and performance goals to determine whether the trade-off is acceptable.

You can calculate your server's energy efficiency ratio for a useful metric that incorporates both power and performance information. Energy efficiency is the ratio of work that is done to the average power that is required during a specified amount of time. In equation form:

$$Energy\ Efficiency\ = \frac{Rate\ of\ Work\ Done}{Average\ Watts\ Of\ Power\ Required}$$

You can use this metric to set practical goals that respect the tradeoff between power and performance. In contrast, a goal of 10 percent energy savings across the datacenter fails to capture the corresponding effects on performance and vice versa. Similarly, if you tune your server to increase performance by 5 percent and that results in 10 percent higher energy consumption; the total result might or might not be acceptable for your business goals. The energy efficiency metric allows for more informed decision making than power or performance metrics alone.

Measuring System Energy Consumption

You should establish a baseline power measurement before you tune your server for energy efficiency.

If your server has the necessary support, you can use the power metering and budgeting features in Windows Server 2008 R2 to view system-level energy consumption through Performance Monitor (Perfmon). One way to determine whether your server has support for metering and budgeting is to review the Windows Server Catalog. (For a link to the Windows Server Catalog, see “Resources” later in this guide.) If your server model qualifies for the new Enhanced Power

Management additional qualification in the Windows Logo Program, it is guaranteed to support the metering and budgeting functionality.

Another way to check for metering support is to manually look for the counters in Performance Monitor. Open Performance Monitor, select **Add Counters**, and locate the **Power Meter** counter group. If named instances of power meters appear in the box labeled **Instances of Selected Object**, your platform supports metering. The **Power** counter that shows power in watts appears in the selected counter group. The exact derivation of the power data value is not specified. For example, it could be instantaneous power draw or average power draw over some time interval.

If your server platform does not support metering, you can use a physical metering device connected to the power supply input to measure system power draw or energy consumption.

To establish a baseline, you should measure the average power required at various system load points, from idle to 100 percent (maximum throughput). Such a baseline generates what is called a “load line”. Figure 1 shows load lines for three sample configurations.

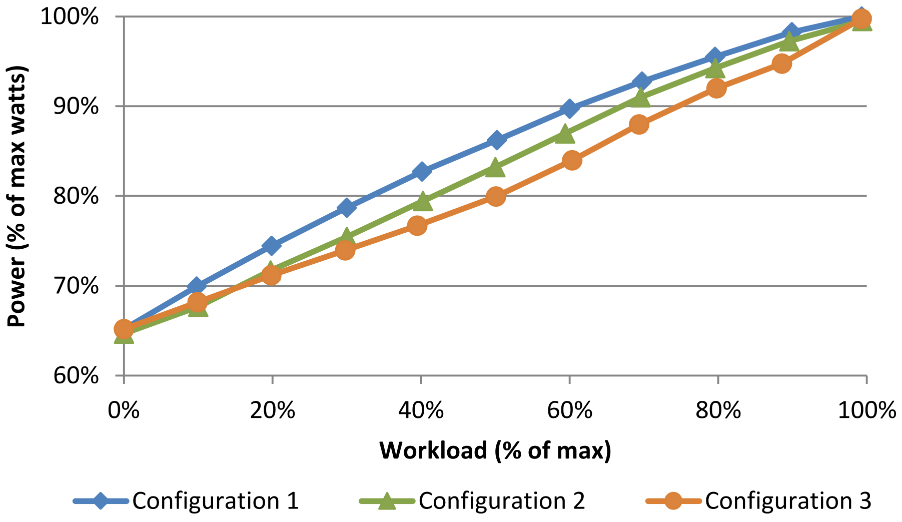


Figure 1. Sample load lines

You can use load lines to evaluate and compare the performance and energy consumption of different configurations at all load points.

You should measure system utilization and energy consumption on a regular basis and after changes in workloads, workload levels, or server hardware.

Diagnosing Energy Efficiency Issues

In Windows 7 and Windows Server 2008 R2, the Windows PowerCfg utility supports a new command-line option that you can use to analyze the energy efficiency of your server. When you run the **powercfg** command with the **/energy** option, the utility performs a 60-second test to detect potential energy efficiency issues. The utility generates a simple HTML report in the current directory. To ensure an accurate analysis, make sure that all local applications are closed before you run the **powercfg** command.

Shortened timer tick rates, drivers that lack power management support, and excessive CPU utilization are just a few of the behavioral problems that are detected by the **powercfg /energy** command. This tool provides a simple way to identify and fix power management problems, potentially resulting in significant cost savings in a large datacenter.

For more information on the **powercfg /energy** option, see "Resources" later in this guide.

Using Power Plans in Windows Server

Windows Server 2008 R2 has three built-in power plans, each designed to meet a different set of business needs. These plans provide a simple way for an administrator to customize a server to meet power or performance goals. Table 4 describes the plans, lists common scenarios in which to use each plan, and gives some implementation details for each plan.

Table 4. Built-in Server Power Plans

Plan	Description	Common applicable scenarios	Implementation highlights
Balanced (recommended)	Default setting. Highest energy efficiency with minimum performance impact.	• General computing.	Matches capacity to demand. Energy-saving features balance power and performance.
High Performance	Increases performance at the cost of high energy consumption. Should not be used unless absolutely necessary.	• Low latency. • Application code sensitive to processor frequency changes.	Processors are always locked at the highest performance state.
Power Saver	Limits performance to save energy and reduce operating cost.	• Deployments with limited power budgets. • Thermal constraints.	Caps processor frequency at a percentage of maximum (if supported), and enables other energy-saving features.

These plans exist in Windows for both AC (alternating current) and DC (direct current) powered systems, but in this paper we assume that servers are using AC power.

For more information on power plans, power policies, and power policy configuration, see "Resources" later in this guide.

Tuning Processor Power Management Parameters

Each power plan shown in Table 4 represents a combination of numerous underlying power management parameters. The built-in plans are three collections of recommended settings that cover a wide variety of workloads and scenarios. However, we recognize that these plans will not meet every customer's needs.

The following sections describe ways to tune some specific processor power management parameters to meet goals not addressed by the three built-in plans. If you need to understand a wider array of power parameters, you can read the document on power policy configuration listed in "Resources" later in this guide. That document provides a more detailed explanation of power plans and parameters, and it includes instructions for adjusting parameter values using the PowerCfg command-line tool.

Processor Performance Boost Policy

Intel Turbo Boost Technology is a feature that allows Intel processors to achieve additional performance when it is most useful (that is, at high system loads). However, this feature increases CPU core energy consumption, so we configure Turbo Boost based on the power policy that is in use and the specific processor implementation. Turbo Boost is enabled for High Performance power plans on all Intel processors and it is disabled for Power Saver power plans on all Intel processors. TurboBoost is disabled on Balanced power plans for some Intel processors. For future processors, this default setting might change depending on the energy efficiency of such features. To enable or disable the Turbo Boost feature, you must configure the Processor Performance Boost Policy parameter.

The Processor Performance Boost Policy is a percentage value from 0 to 100. The default value of this parameter is 49 percent on Balanced plans and 0 percent on Power Saver plans. Any value lower than 50 disables Turbo mode on some current Intel processors. To enable Turbo Mode, set this value to 50 or higher.

The following commands set Processor Performance Boost Policy to 100 on the current power plan. Specify the policy by using a GUID string, as shown below:

```
Powercfg -setacvalueindex scheme_current sub_processor 45bcc044-d885-
43e2-8605-ee0ec6e96b59 100
Powercfg -setactive scheme_current
```

Note that you must run the **powercfg -setactive** command to enable the new settings. You do not need to reboot the server.

To set this value for power plans other than the current selected plan, you can use aliases such as SCHEME_MAX (Power Saver), SCHEME_MIN (High Performance), and SCHEME_BALANCED (Balanced) in place of SCHEME_CURRENT. Replace "scheme current" in the **powercfg -setactive** commands shown above with the desired alias to enable that power plan. For example, to adjust the Boost Policy in the Power Saver plan and make Power Saver the current plan, run the following commands:

```
Powercfg -setacvalueindex scheme_max sub_processor 45bcc044-d885-43e2-
8605-ee0ec6e96b59 100
Powercfg -setactive scheme_max
```

Minimum and Maximum Processor Performance State

Processors change between performance states (“P-states”) very quickly to match supply to demand, delivering performance where necessary and saving energy when possible. If your server has specific high-performance or minimum-power-consumption requirements, you might consider configuring the Minimum or Maximum Processor Performance State parameter.

The values for both the Minimum and Maximum Processor Performance State parameters are expressed as a percentage of maximum processor frequency, with a value in the range 0 – 100.

If your server requires ultra-low latency, invariant CPU frequency, or the very highest performance levels, you might not want the processors switching to lower-performance states. For such a server, you can cap the minimum processor performance state at 100 percent by using the following commands:

```
Powercfg -setacvalueindex scheme_current sub_processor 893dee8e-2bef-
41e0-89c6-b55d0929964c 100
Powercfg -setactive scheme_current
```

If your server requires lower energy consumption, you might want to cap the processor performance state at a percentage of maximum. For example, you can restrict the processor to 75 percent of its maximum frequency by using the following commands:

```
Powercfg -setacvalueindex scheme_current sub_processor bc5038f7-23e0-
4960-96da-33abaf5935ec 75
Powercfg -setactive scheme_current
```

Note that capping processor performance at a percentage of maximum requires processor support. Check the processor documentation to determine whether such support exists, or view the Perfmon counter “% of maximum frequency” in the Processor group to see if any frequency caps were applied as desired.

Processor Performance Core Parking Maximum and Minimum Cores

Core parking is a new feature in Windows Server 2008 R2. The processor power management (PPM) engine and the scheduler work together to dynamically adjust the number of cores available to execute threads. The PPM engine chooses a minimum number of cores on which threads will be scheduled. Cores that are chosen to be “parked” will generally not have any threads scheduled on them and they will drop into very low power states when not processing interrupts or DPCs, or other strictly affinitized work. The remaining set of “unparked” cores is responsible for the remainder of the workload. Core parking can increase energy efficiency during lower usage periods on the server because parked cores can drop into deep low-power states.

For most servers, the default core-parking behavior provides the optimum balance of throughput and energy efficiency. If your server has specific core-parking requirements, you can control the number of cores available to park by using either the Processor Performance Core Parking Maximum Cores parameter or the Processor Performance Core Parking Minimum Cores parameter in Windows Server 2008 R2.

The values for these parameters are percentages in the range 0–100. The Maximum Cores parameter controls the maximum percentage of cores that can be unparked (available to run threads) at any time, while the Minimum Cores parameter controls the minimum percentage of cores that can be unparked. To turn off core parking, set the Minimum Cores parameter to 100 percent by using the following commands:

```
Powercfg -setacvalueindex scheme_current sub_processor bc5038f7-23e0-
4960-96da-33abaf5935ec 100
Powercfg -setactive scheme_current
```

To reduce the number of schedulable cores to 50 percent of the maximum count, set the Maximum Cores parameter to 50 as follows:

```
Powercfg -setacvalueindex scheme_current sub_processor bc5038f7-23e0-
4960-96da-33abaf5935ec 50
Powercfg -setactive scheme_current
```

Interrupt Affinity

The term “interrupt affinity” refers to the binding of interrupts from a specific device to one or more specific logical processors in a multiprocessor server. The binding forces interrupt processing to run on a specified logical processor or processors, unless the device specifies otherwise during its initialization. For some scenarios, such as a file server, the network connections and file server sessions remain on the same network adapter. In those scenarios, binding interrupts from a network adapter to a logical processor allows for processing incoming packets (SMB requests and data) on a specific set of logical processors, which improves locality and scalability.

You can use the old Interrupt-Affinity Filter tool (IntFiltr) to change the CPU affinity of the interrupt service routine (ISR). The tool runs on most servers that run Windows Server 2008 R2, regardless of what logical processor or interrupt controller is used. For IntFiltr to work on some systems, you must set the MAXPROCSPERCLUSTER=0 boot parameter. However, on some systems with more than eight logical processors or for devices that use MSI or MSI-X, the tool is limited by the Advanced Programmable Interrupt Controller (APIC) protocol. The new Interrupt-Affinity Policy (IntPolicy) tool does not encounter this issue because it sets the CPU affinity through the affinity policy of a device. For more information about the Interrupt-Affinity Policy tool, see “Resources” later in this guide. You can use either tool to direct any device's ISR to a specific processor or to a set of processors (instead of sending interrupts to any of the CPUs in the system). Note that different devices can have different interrupt affinity settings. On some systems, directing the ISR to a processor on a different Non-Uniform Memory Access (NUMA) node can cause performance issues. Also, if an MSI or MSI-X device has multiple interrupt “messages,” each message can be affinitized to a different logical processor or set of processors.

We recommend that you use IntPolicy to bind interrupts only for devices whose driver models do *not* support affinitization functionality. For devices that support it, you should use the device-specific mechanism for binding interrupts. For example, most modern server NICs support Receive Side Scaling (RSS), which is the recommended method for controlling interrupts. Similarly, modern storage controllers implement multi-message MSI-X and take advantage of NUMA I/O optimization provided by the operating system (Windows Server 2008 and later). Regardless of device functionality, IRQ affinity specified by the operating system is only a suggestion that the device driver can choose to honor or not. IntPolicy has no effect on the synthetic devices within a VM in a Hyper-V server. You cannot use IntPolicy to distribute the synthetic interrupt load of a guest VM.

Performance Tuning for the Networking Subsystem

Figure 2 shows the network architecture, which covers many components, interfaces, and protocols. The following sections discuss tuning guidelines for some components of server workloads.

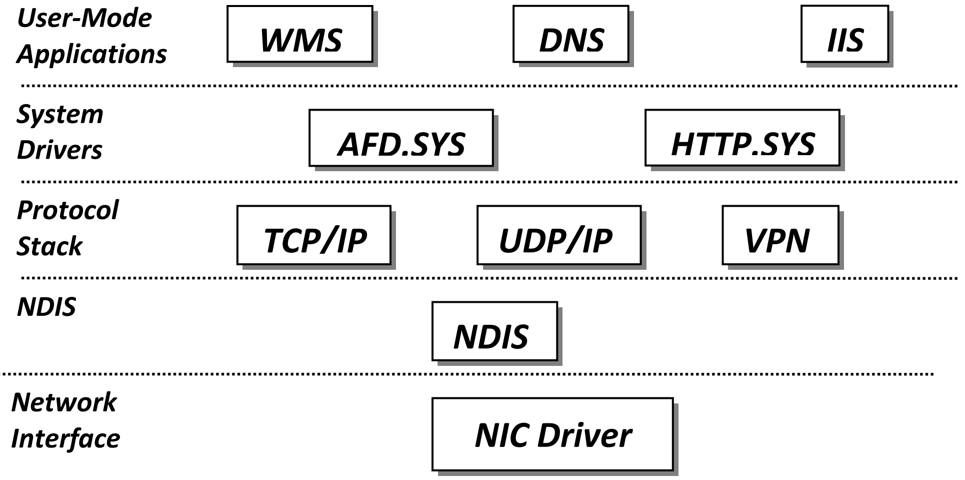


Figure 2. Network Stack Components

The network architecture is layered, and the layers can be broadly divided into the following sections:

- The network driver and Network Driver Interface Specification (NDIS).

  These are the lowest layers. NDIS exposes interfaces for the driver below it and for the layers above it such as TCP/IP.

- The protocol stack.

  This implements protocols such as TCP/IP and UDP/IP. These layers expose the transport layer interface for layers above them.

- System drivers.

  These are typically transport data interface extension (TDX) or Winsock Kernel (WSK) clients and expose interfaces to user-mode applications. The WSK interface was a new feature for Windows Server 2008 and Windows Vista® and is exposed by Afd.sys. The interface improves performance by eliminating the switching between user mode and kernel mode.

- User-mode applications.

  These are typically Microsoft solutions or custom applications.

Tuning for network-intensive workloads can involve each layer. The following sections describe some tuning recommendations.

Choosing a Network Adapter

Network-intensive applications require high-performance network adapters. This section covers some considerations for choosing network adapters.

Offload Capabilities

Offloading tasks can reduce CPU usage on the server, which improves overall system performance. The Microsoft network stack can offload one or more tasks to a network adapter if you choose one that has the appropriate offload capabilities. Table 5 provides more details about each offload capability.

Table 5. Offload Capabilities for Network Adapters

Offload type	Description
Checksum calculation	The network stack can offload the calculation and validation of both Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) checksums on sends and receives. It can also offload the calculation and validation of both IPv4 and IPv6 checksums on sends and receives.
IP security authentication and encryption	The TCP/IP transport can offload the calculation and validation of encrypted checksums for authentication headers and Encapsulating Security Payloads (ESPs). The TCP/IP transport can also offload the encryption and decryption of ESPs.
Segmentation of large TCP packets	The TCP/IP transport supports Large Send Offload v2 (LSOv2). With LSOv2, the TCP/IP transport can offload the segmentation of large TCP packets to the hardware.
TCP stack	The TCP offload engine (TOE) enables a network adapter that has the appropriate capabilities to offload the entire network stack.

Receive-Side Scaling (RSS)

Windows Server 2008 R2 supports Receive Side Scaling (RSS) out of the box, as does Windows Server 2008. RSS distributes incoming network I/O packets among processors so that packets that belong to the same TCP connection are on the same processor, which preserves ordering. This helps improve scalability and performance for receive-intensive scenarios that have fewer networking adapters than available processors. Research shows that distributing packets to logical processors that share the same physical processor (for example, hyper-threading) degrades performance. Therefore, packets are only distributed across physical processors. Windows Server 2008 R2 offers the following optimizations for improved scalability with RSS:

- NUMA awareness.

  RSS considers NUMA node distance (latency between nodes) when selecting processors for load balancing incoming packets.

- Improved initialization and processor selection algorithm.

  At boot time, the Windows Server 2008 R2 networking stack considers the bandwidth and media connection state when assigning CPUs to RSS-capable adapters. Higher-bandwidth adapters get more CPUs at startup. Multiple NICs with the same bandwidth receive the same number of RSS CPUs.

- More control over RSS on a per-NIC basis.

  Depending on the scenario and the workload characteristics, you can use the following registry parameters to choose on a per-NIC basis how many processors can be used for RSS, the starting offset for the range of processors, and which node the NIC allocates memory from:

- ***MaxRSSProcessors**

  ```
  HKLM\system\CurrentControlSet\Control\class\{XXXXX72-
  XXX}\<network adapter number>\(REG_SZ)
  ```

  The maximum number of RSS processors assigned to each NIC.

- ***RssBaseProcNumber**

  ```
  HKLM\system\CurrentControlSet\Control\class\{XXXXX72-
  XXX}\<network adapter number>\(REG_SZ)
  ```

  The first processor in the range of RSS processors assigned to each NIC.

- ***NumaNodeID**

  ```
  HKLM\system\CurrentControlSet\Control\class\{XXXXX72-
  XXX}\<network adapter number>\(REG_SZ)
  ```

  The NUMA node each NIC can allocate memory from.

Note: The asterisk (*) is part of the registry parameter.

For more information about RSS, see the document about Scalable Networking in "Resources" later in this guide.

Message-Signaled Interrupts (MSI/MSI-X)

Network adapters that support MSI/MSI-X can target their interrupts to specific processors. If the adapters also support RSS, then a processor can be dedicated to servicing interrupts and DPCs for a given TCP connection. This preserves the cache locality of TCP structures and greatly improves performance.

Network Adapter Resources

A few network adapters actively manage their resources to achieve optimum performance. Several network adapters let the administrator manually configure resources by using the **Advanced Networking** tab for the adapter. For such adapters, you can set the values of a number of parameters including the number of receive buffers and send buffers.

Interrupt Moderation

To control interrupt moderation, some network adapters either expose different interrupt moderation levels, or buffer coalescing parameters (sometimes separately for send and receive buffers), or both. You should consider buffer coalescing or batching when the network adapter does not perform interrupt moderation. Interrupt Moderation helps reduce overall CPU utilization by minimizing per-buffer processing cost, but the moderation of interrupts and buffer batching can have a negative impact on latency-sensitive scenarios.

Suggested Network Adapter Features for Server Roles

Table 6 lists high-performance network adapter features that can improve performance in terms of throughput, latency, or scalability for some server roles.

Table 6. Benefits from Network Adapter Features for Different Server Roles

Server role	Checksum offload	Segmentation offload	TCP offload engine (TOE)	Receive-side scaling (RSS)
File server	X	X	X	X
Web server	X	X	X	X
Mail server (short-lived connections)	X			X
Database server	X	X	X	X
FTP server	X	X	X	
Media server	X		X	X

Disclaimer: The recommendations in Table 6 are intended to serve as guidance only for choosing the most suitable technology for specific server roles under a deterministic traffic pattern. User experience can be different, depending on workload characteristics and the hardware that is used.

If your hardware supports TOE, then you must enable that option in the operating system to benefit from the hardware’s capability. You can enable TOE by running the following command:

```
netsh int tcp set global chimney = enabled
```

Tuning the Network Adapter

You can optimize network throughput and resource usage by tuning the network adapter, if any tuning options are exposed by the adapter. Remember that the correct tuning settings depend on the network adapter, the workload, the host computer resources, and your performance goals.

Enabling Offload Features

Turning on network adapter offload features is usually beneficial. Sometimes, however, the network adapter is not powerful enough to handle the offload capabilities at high throughput. For example, enabling segmentation offload can reduce the maximum sustainable throughput on some network adapters because of limited hardware resources. However, if the reduced throughput is not expected to be a limitation, you should enable offload capabilities even for such network adapters. Note that some network adapters require offload features to be independently enabled for send and receive paths.

Increasing Network Adapter Resources

For network adapters that allow for the manual configuration of resources such as receive and send buffers, you should increase the allocated resources. Some network adapters set their receive buffers low to conserve allocated memory from the host. The low value results in dropped packets and decreased performance. Therefore, for receive-intensive scenarios, we recommend that you increase the receive buffer value to the maximum. If the adapter does not expose manual resource configuration, then it either dynamically configures the resources or it is set to a fixed value that cannot be changed.

Enabling Interrupt Moderation

To control interrupt moderation, some network adapters expose different interrupt moderation levels, buffer coalescing parameters (sometimes separately for send and receive buffers), or both. You should consider interrupt moderation for CPU-bound workloads and consider the trade-off between the host CPU savings and latency versus the increased host CPU savings because of more interrupts and less latency. If the network adapter does not perform interrupt moderation but does expose buffer coalescing, then increasing the number of coalesced buffers allows for more buffers per send or receive, which improves performance.

Enabling RSS for Web Scenarios

RSS can improve Web scalability and performance when there are fewer NICs than processors on the server. When all the Web traffic is going through the RSS-capable NICs, incoming Web requests from different connections can be simultaneously processed across different CPUs. It is important to note that due to logic in RSS and HTTP for load distribution, performance can be severely degraded if a non-RSS-capable NIC accepts Web traffic on a server that has one or more RSS-capable NICs. We recommend that you either use RSS-capable-NICs or disable RSS from the **Advanced Properties** tab. To determine whether a NIC is RSS-capable, view the RSS information in the **Advanced Properties** tab for the device.

Binding Each Adapter to a CPU

The method to use for binding network adapters to a CPU depends on the number of network adapters, the number of CPUs, and the number of ports per network adapter. Important factors are the type of workload and the distribution of the interrupts across the CPUs. For a workload such as a Web server that has several networking adapters, partition the adapters on a processor basis to isolate the interrupts that the adapters generate.

TCP Receive Window Auto-Tuning

Starting with Windows Server 2008, one of the most significant changes to the TCP stack is TCP receive window auto-tuning. Previously, the network stack used a fixed-size receive-side window that limited the overall potential throughput for connections. You can calculate the total throughput of a single connection when you use this fixed size default as:

Total achievable throughput in bytes = TCP window * (1 / connection latency)

For example, the total achievable throughput is only 51 Mbps on a 1-GB connection with 10-ms latency (a reasonable value for a large corporate network infrastructure). With auto-tuning, however, the receive-side window is adjustable and can grow to meet the demands of the sender. It is entirely possible for a connection to achieve a full line rate of a 1-GB connection. Network usage scenarios that might have been limited in the past by the total achievable throughput of TCP connections now can fully use the network.

Remote file copy is a common network usage scenario that is likely to increase demand on the infrastructure because of this change. Many improvements have been made to the underlying operating system support for remote file copy that now let large file copies perform at disk I/O speeds. If many concurrent remote large file copies are typical within your network environment, your network infrastructure might be taxed by the significant increase in network usage by each file copy operation.

Windows Filtering Platform

The Windows Filtering Platform (WFP) that was introduced in Windows Vista and Windows Server 2008 provides APIs to third-party independent software vendors (ISVs) to create packet processing filters. Examples include firewall and antivirus software. Note that a poorly written WFP filter significantly decreases a server's networking performance. For more information about WFP, see "Resources" later in this guide.

TCP Parameters

The following registry keywords in Windows Server 2003 are *no longer supported* and are ignored in Windows Server 2008 and Windows Server 2008 R2:

- **TcpWindowSize**

  ```
  HKLM\System\CurrentControlSet\Services\Tcpip\Parameters
  ```

- **NumTcbTablePartitions**

  ```
  HKLM\system\CurrentControlSet\Services\Tcpip\Parameters
  ```

- **MaxHashTableSize**

  ```
  HKLM\system\CurrentControlSet\Services\Tcpip\Parameters
  ```

Network-Related Performance Counters

This section lists the counters that are relevant to managing network performance. Most of the counters are straightforward. We provide guidelines for the counters that typically require explanation.

IPv4

- Datagrams received per second.
- Datagrams sent per second.

IPv6

- Datagrams received per second.
- Datagrams sent per second.

Network Interface > [adapter name]

- Bytes received per second.
- Bytes sent per second.
- Packets received per second.
- Packets sent per second.

- Output queue length.

  This counter is the length of the output packet queue (in packets). If this is longer than 2, delays occur. You should find the bottleneck and eliminate it if you can. Because NDIS queues the requests, this length should always be 0.
- Packets received errors.

  This counter is the number of incoming packets that contain errors that prevent them from being deliverable to a higher-layer protocol. A zero value does not guarantee that there are no receive errors. The value is polled from the network driver, and it can be inaccurate.
- Packets outgoing errors.

Processor Information

- Percent of processor time.
- Interrupts per second.
- DPCs queued per second.

  This counter is an average rate at which DPCs were added to the processor's DPC queue. Each processor has its own DPC queue. This counter measures the rate at which DPCs are added to the queue, not the number of DPCs in the queue. It displays the difference between the values that were observed in the last two samples, divided by the duration of the sample interval.

TCPv4

- Connection failures.
- Segments sent per second.
- Segments received per second.
- Segments retransmitted per second.

TCPv6

- Connection failures.
- Segments sent per second.
- Segments received per second.
- Segments retransmitted per second.

Performance Tuning for the Storage Subsystem

Decisions about how to design or configure storage software and hardware usually consider performance. Performance is degraded or improved as a result of trade-offs with other factors such as cost, reliability, availability, power, or ease of use. Trade-offs are made all along the path between application and disk media. File cache management, file system architecture, and volume management translate application calls into individual storage access requests. These requests traverse the storage driver stack and generate streams of commands that are presented to the disk storage subsystem. The sequence and quantity of calls, and the subsequent translation, can improve or degrade performance.

Figure 3 shows the storage architecture, which covers many components in the driver stack.

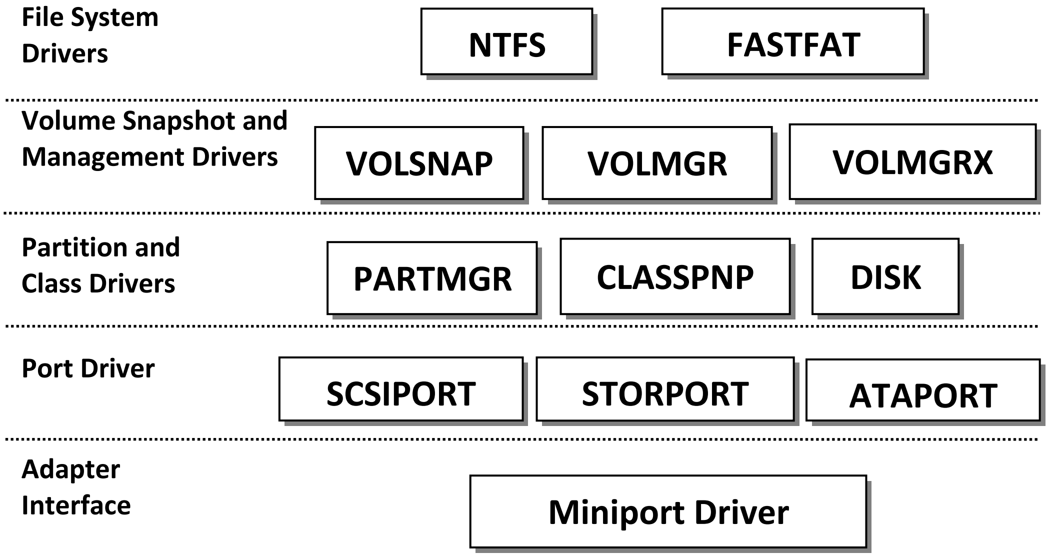


Figure 3. Storage Driver Stack

The layered driver model in Windows sacrifices some performance for maintainability and ease of use (in terms of incorporating drivers of varying types into the stack). The following sections discuss tuning guidelines for storage workloads.

Choosing Storage

The most important considerations in choosing storage systems include the following:

- Understanding the characteristics of current and future storage workloads.
- Understanding that application behavior is essential for both storage subsystem planning and performance analysis.
- Providing necessary storage space, bandwidth, and latency characteristics for current and future needs.
- Selecting a data layout scheme (such as striping), redundancy architecture (such as mirroring), and backup strategy.
- Using a procedure that provides the required performance and data recovery capabilities.
- Using power guidelines – That is, calculating the expected average power required in total and per-unit volume (such as watts per rack).

  When compared to 3.5-inch disks, 2.5-inch disks have greatly reduced power requirements, but they can also be packed more compactly into racks or servers, which can increase cooling requirements per rack or per server chassis. Note that enterprise disk drives are currently not built to withstand frequent power-up/power-down cycles. Attempts to save energy consumption by shutting down a server's internal or external storage should be carefully weighed against possible increases in lab operation costs or decreases in system data availability caused by a higher rate of disk failures. This issue might be alleviated in future enterprise disk designs or through the use of solid-state storage (for example, SSDs).

The better you understand the workloads on a specific server or set of servers, the more accurately you can plan. The following are some important workload characteristics:

- Read:write ratio
- Sequential vs. random access, including both temporal and spatial locality
- Request sizes
- Request concurrency, interarrival rates, and burstiness (that is, patterns of request arrival rates)

Estimating the Amount of Data to Be Stored

When you estimate how much data will be stored on a new server, consider these issues:

- How much data you will move to the new server from existing servers.
- How much replicated data you will store on the new file server if the server is a file server replica member.
- How much data you will store on the server in the future.

A general guideline is to assume that growth will be faster in the future than it was in the past. Investigate whether your organization plans to hire many employees, whether any groups in your organization are planning large projects that will require additional storage, and so on.

You must also consider how much space is used by operating system files, applications, RAID redundancy, log files, and other factors. Table 7 describes some factors that affect server storage capacity.

Table 7. Factors That Affect Server Storage Capacity

Factor	Required storage capacity
Operating system files	At least 15 GB. To provide space for optional components, future service packs, and other items, plan for an additional 3 to 5 GB for the operating system volume. Windows installation can require even more space for temporary files.
Paging file	For smaller servers, 1.5 times the amount of RAM, by default. For servers that have hundreds of gigabytes of memory, you might be able to eliminate the paging file; otherwise, the paging file might be limited because of space constraints (available disk capacity). The benefit of a paging file of larger than 50 GB is unclear.
Memory dump	Depending on the memory dump file option that you have chosen, as large as the amount of physical memory plus 1 MB. On servers that have very large amounts of memory, full memory dumps become intractable because of the time that is required to create, transfer, and analyze the dump file.
Applications	Varies according to the application. Example applications include backup and disk quota software, database applications, and optional components such as Recovery Console, Services for UNIX, and Services for NetWare.

Factor	Required storage capacity
Log files	Varies according to the applications that create the log file. Some applications let you configure a maximum log file size. You must make sure that you have enough free space to store the log files.
Data layout and redundancy	Varies depending on cost, performance, reliability, availability, and power goals. For more information, see "Choosing the Raid Level" later in this guide.
Shadow copies	10 percent of the volume, by default, but we recommend increasing this size based on frequency of snapshots and rate of disk data updates.

Choosing a Storage Array

There are many considerations in choosing a storage array and adapters. The choices include the type of storage communication protocols that you use, including the options shown in Table 8.

Table 8. Options for Storage Array Selection

Option	Description
Fibre Channel or SCSI	Fibre Channel enables long glass or copper cables to connect the storage array to the system and provides high bandwidth. SCSI provides high bandwidth, but it has cable length restrictions.
SAS or SATA	These serial protocols improve performance, reduce cable length limitations, and reduce cost. SAS and SATA drives are replacing much of the SCSI market. In general, SATA drives are built with higher capacity and lower cost targets than SAS drives; the premium associated with SAS is typically attributed to performance.
Hardware RAID capabilities	For maximum performance and reliability, the enterprise storage controllers should offer RAID capabilities. RAID levels 0, 1, 0+1, 5, and 6 are described in Table 9.
Maximum storage capacity	Total usable storage space.
Storage bandwidth	The maximum peak and sustained bandwidths at which storage can be accessed are determined by the number of physical disks in the array, the speed of the controllers, the type of bus protocol (such as SAS or SATA), the hardware-managed or software-managed RAID, and the adapters that are used to connect the storage array to the system. Of course, the more important values are the achievable bandwidths for the specific workloads to be executed on servers that access the storage.

Hardware RAID Levels

Most storage arrays provide some hardware RAID capabilities. Table 9 lists the common RAID levels.

Table 9. RAID Options

Option	Description
Just a bunch of disks (JBOD)	This is not a RAID level, but instead is the baseline against which to measure RAID performance, reliability, availability, cost, capacity, and energy consumption. Individual disks are referenced separately, not as a combined entity. In some scenarios, JBOD actually provides better performance than striped data layout schemes. For example, when serving multiple lengthy sequential streams, performance is best when a single disk services each stream. Also, workloads that are composed of small, random requests do not experience performance improvements when they are moved from JBOD to a striped data layout. JBOD is susceptible to static and dynamic “hot spots” (frequently accessed ranges of disk blocks) that reduce available storage bandwidth due to the resulting load imbalance between the physical drives. Any physical disk failure results in data loss in a JBOD configuration. However, the loss is limited to the failed drives. In some scenarios, JBOD provides a level of data isolation that can be interpreted as actually offering greater reliability than striped configurations.
Spanning	This is also not a RAID level, but instead is the simple concatenation of multiple physical disks into a single logical disk. Each disk contains one continuous set of sequential logical blocks. Spanning has the same performance and reliability characteristics as JBOD.
RAID 0 (striping)	RAID 0 is a data layout scheme in which sequential logical blocks of a specified size (the stripe unit) are laid out in a round-robin manner across multiple disks. It presents a combined logical disk that stripes disk accesses over a set of physical disks. For most workloads, a striped data layout provides better performance than JBOD if the stripe unit is appropriately selected based on server workload and storage hardware characteristics. The overall storage load is balanced across all physical drives. This is the least expensive RAID configuration because all of the disk capacity is available for storing the single copy of data. Because no capacity is allocated for redundant data, RAID 0 does not provide data recovery mechanisms such as those provided in the other RAID schemes. Also, the loss of any disk results in data loss on a larger scale than JBOD because the entire file system or raw volume spread across *n* physical disks is disrupted; every *n*th block of data in the file system is missing.

Option	Description
RAID 1 (mirroring)	RAID 1 is a data layout scheme in which each logical block exists on multiple physical disks (typically two). It presents a logical disk that consists of a set of two or more mirrored disks. RAID 1 often has worse bandwidth and latency for write operations when compared to RAID 0 (or JBOD). This is because data from each write request must be written to two or more physical disks. Request latency is based on the slowest of the two (or more) write operations that are necessary to update all copies of the updated data blocks. RAID 1 can provide faster read operations than RAID 0 because it can read from the least busy physical disk from the mirrored pair, or the disk that will experience the shortest mechanical positioning delays. RAID 1 is the most expensive RAID scheme in terms of physical disks because half (or more) of the disk capacity stores redundant data copies. RAID 1 can survive the loss of any single physical disk. In larger configurations it can survive multiple disk failures, if the failures do not involve all the disks of a specific mirrored disk set. RAID 1 has greater power requirements than a non-mirrored storage configuration. RAID 1 doubles the number of disks and therefore doubles the required amount of idle power. Also, RAID 1 performs duplicate write operations that require twice the power of non-mirrored write operations. RAID 1 is the fastest ordinary RAID level for recovery time after a physical disk failure. Only a single disk (the other part of the broken mirror pair) brings up the replacement drive. Note that the second disk is typically still available to service data requests throughout the rebuilding process.
RAID 0+1 (striped mirrors)	The combination of striping and mirroring is intended to provide the performance benefits of RAID 0 and the redundancy benefits of RAID 1. This option is also known as RAID 1+0 and RAID 10. Cost and power characteristics are similar to those of RAID 1.

Option	Description
RAID 5 (rotated parity)	RAID 5 presents a logical disk composed of multiple physical disks that have data striped across the disks in sequential blocks (stripe units) in a manner similar to RAID 0. However, the underlying physical disks have parity information scattered throughout the disk array, as Figure 4 shows. For read requests, RAID 5 has characteristics that resemble those of RAID 0. However, small RAID 5 writes are much slower than those of JBOD or RAID 0 because each parity block that corresponds to the modified data block must also be updated. This process requires three additional disk requests. Because four physical disk requests are generated for every logical write, bandwidth is reduced by approximately 75 percent. RAID 5 provides data recovery capabilities because data can be reconstructed from the parity. RAID 5 can survive the loss of any one physical disk, as opposed to RAID 1, which can survive the loss of multiple disks as long as an entire mirrored set is not lost. RAID 5 requires additional time to recover from a lost physical disk compared to RAID 1 because the data and parity from the failed disk can be re-created only by reading all the other disks in their entirety. Performance during the rebuilding period is severely reduced due not only to the rebuilding traffic but also because the reads and writes that target the data that was stored on the failed disk must read all disks (an entire “stripe”) to re-create the missing data. RAID 5 is less expensive than RAID 1 because it requires only an additional single disk per array, instead of double (or more) the total number of disks in an array. Power guidelines: RAID 5 might consume more or less energy than a mirrored configuration, depending on the number of drives in the array, the characteristics of the drives, and the characteristics of the workload. RAID 5 might use less energy if it uses significantly fewer drives. The additional disk adds to the required amount of idle power as compared to a JBOD array, but it requires less additional idle power versus a full mirrored set of drives. However, RAID 5 requires four accesses for every random write request in order to read the old data, read the old parity, compute the new parity, write the new data, and write the new parity. This means that the power needed beyond idle to perform the write operations is up to 4X that of JBOD or 2X that of a mirrored configuration. (Note that depending on the workload, there may be only two seeks, not four, that require moving the disk actuator.) Thus, though unlikely in most configurations, RAID 5 might have greater energy consumption. This might happen in the case of a heavy workload being serviced by a small array or an array of disks whose idle power is significantly lower than their active power.

Option	Description
RAID 6 (double-rotated redundancy)	Traditional RAID 6 is basically RAID 5 with additional redundancy built in. Instead of a single block of parity per stripe of data, two blocks of redundancy are included. The second block uses a different redundancy code (instead of parity), which enables data to be reconstructed after the loss of any two disks. Power guidelines: RAID 6 might consume more or less energy than a mirrored configuration, depending on the number of drives in the array, the characteristics of the drives, and the characteristics of the workload. RAID 6 might use less energy if it uses significantly fewer drives. The additional disk adds to the required amount of idle power as compared to a JBOD array, but it requires less additional idle power versus a full mirrored set of drives. However, RAID 6 requires six accesses for every random write request in order to read the old data, read the old redundancy data (two sets), compute the new redundancy data (two sets), write the new data, and write the new redundancy data (two sets). This means that the power needed beyond idle to perform the writes is up to 6X that of JBOD or 3X that of a mirrored configuration. (Note that depending on the workload, there may be only three seeks, not six, that require moving the disk actuator.) Thus, though unlikely in most configurations, RAID 6 might have greater energy consumption. This might happen in the case of a heavy workload being serviced by a small array or an array of disks whose required idle power is significantly lower than their active power. There are some hardware-managed arrays that use the term RAID 6 for other schemes that attempt to improve the performance and reliability of RAID 5. For example, the data disks can be arranged in a two-dimensional matrix, with both vertical and horizontal parity being maintained, but that scheme requires even more drives. This document uses the traditional definition of RAID 6.

Rotated redundancy schemes (such as RAID 5 and RAID 6) are the most difficult to understand and plan for. Figure 4 shows a RAID 5 example, where the sequence of logical blocks presented to the host is A0, B0, C0, D0, A1, B1, C1, E1, and so on.

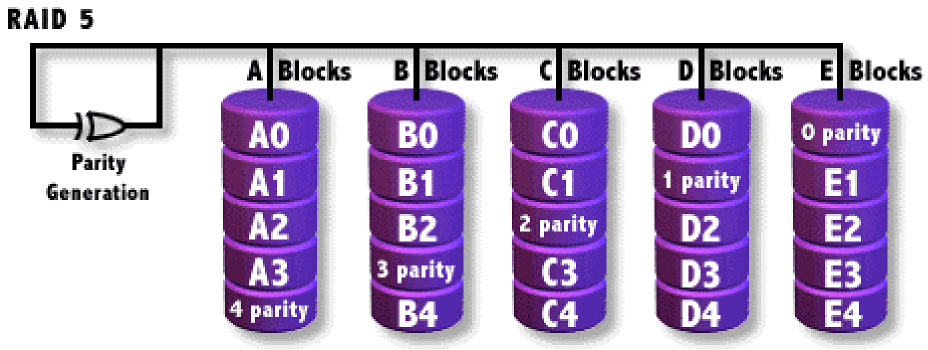


Figure 4. RAID 5 Overview

Choosing the RAID Level

Each RAID level involves a trade-off between the following factors:

- Performance
- Reliability

- Availability
- Cost
- Capacity
- Power

To determine the best RAID level for your servers, evaluate the read and write loads of all data types and then decide how much you can spend to achieve the performance and availability/reliability that your organization requires. Table 10 describes common RAID levels and their relative performance, reliability, availability, cost, capacity, and energy consumption.

Table 10. RAID Trade-Offs

Configuration	Performance	Reliability	Availability	Cost, capacity, and power
JBOD	**Pros:** • Concurrent sequential streams to separate disks. **Cons:** • Susceptibility to load imbalance.	**Pros:** • Data isolation; single loss affects one disk. **Cons:** • Data loss after one failure.	**Pros:** • Single loss does not prevent access to other disks.	**Pros:** • Minimum cost. • Minimum power.
RAID 0 (striping) **Requirements:** • Two-disk minimum.	**Pros:** • Balanced load. • Potential for better response times, throughput, and concurrency. **Cons:** • Difficult stripe unit size choice.	**Cons:** • Data loss after one failure. • Single loss affects the entire array.	**Cons:** • Single loss prevents access to entire array.	**Pros:** • Minimum cost. • Minimum power.

Configuration	Performance	Reliability	Availability	Cost, capacity, and power
RAID 1 (mirroring) **Requirements:** • Two-disk minimum.	**Pros:** • Two data sources for every read request (up to 100% performance improvement). **Cons:** • Writes must update all mirrors.	**Pros:** • Single loss and often multiple losses (in large configurations) are survivable.	**Pros:** • Single loss and often multiple losses (in large configurations) do not prevent access.	**Cons:** • Twice the cost of RAID 0 or JBOD. • Up to 2X power .
RAID 0+1 (striped mirrors) **Requirements:** • Four-disk minimum.	**Pros:** • Two data sources for every read request (up to 100% performance improvement). • Balanced load. • Potential for better response times, throughput, and concurrency. **Cons:** • Writes must update mirrors. • Difficult stripe unit size choice.	**Pros:** • Single loss and often multiple losses (in large configurations) are survivable.	**Pros:** • Single loss and often multiple losses (in large configurations) do not prevent access.	**Cons:** • Twice the cost of RAID 0 or JBOD. • Up to 2X power .

Configuration	Performance	Reliability	Availability	Cost, capacity, and power
RAID 5 (rotated parity) **Requirements:** • One additional disk required. • Three-disk minimum.	**Pros:** • Balanced load. • Potential for better read response times, throughput, and concurrency. **Cons:** • Up to 75% write performance reduction because of Read-Modify-Write. • Decreased read performance in failure mode. • All sectors must be read for reconstruction; major slowdown. • Danger of data in invalid state after power loss and recovery.	**Pros:** • Single loss survivable; "in-flight" write requests might still become corrupted. **Cons:** • Multiple losses affect entire array. • After a single loss, array is vulnerable until reconstructed.	**Pros:** • Single loss does not prevent access. **Cons:** • Multiple losses prevent access to entire array. • To speed reconstruction, application access might be slowed or stopped.	**Pros:** • Only one more disk to power. **Cons:** • Up to 4X the power for write requests (excluding the idle power).

Configuration	Performance	Reliability	Availability	Cost, capacity, and power
RAID 6 (two separate erasure codes) **Requirements:** • Two additional disks required. • Five-disk minimum.	**Pros:** • Balanced load. • Potential for better read response times, throughput, and concurrency. **Cons:** • Up to 83% write performance reduction because of multiple RMW. • Decreased read performance in failure mode. • All sectors must be read for reconstruction: major slowdown. • Danger of data in invalid state after power loss and recovery.	**Pros:** • Single loss survivable; "in-flight" write requests might still be corrupted. **Cons:** • More than two losses affect entire array. • After two losses, an array is vulnerable until reconstructed.	**Pros:** • Single loss does not prevent access. **Cons:** • More than two losses prevent access to entire array. • To speed reconstruction, application access might be slowed or stopped.	**Pros:** • Only two more disks to power. **Cons:** • Up to 6X the power for write requests (excluding the idle power).

The following are sample uses for various RAID levels:

- JBOD: Concurrent video streaming.
- RAID 0: Temporary or reconstructable data, workloads that can develop hot spots in the data, and workloads with high degrees of unrelated concurrency.
- RAID 1: Database logs, critical data, and concurrent sequential streams.
- RAID 0+1: A general purpose combination of performance and reliability for critical data, workloads with hot spots, and high-concurrency workloads.
- RAID 5: Web pages, semicritical data, workloads without small writes, scenarios in which capital and operating costs are an overriding factor, and read-dominated workloads.
- RAID 6: Data mining, critical data (assuming quick replacement or hot spares), workloads without small writes, scenarios in which cost or power is a major factor, and read-dominated workloads. RAID 6 might also be appropriate for massive datasets, where the cost of mirroring is high and double-disk failure is a

real concern (due to the time required to complete an array parity rebuild for disk drives greater than 1 TB).

If you use more than two disks, RAID 0+1 is usually a better solution than RAID 1.

To determine the number of physical disks that you should include in RAID 0, RAID 5, and RAID 0+1 virtual disks, consider the following information:

- Bandwidth (and often response time) improves as you add disks.
- Reliability, in terms of mean time to failure for the array, decreases as you add disks.
- Usable storage capacity increases as you add disks, but so does cost.
- For striped arrays, the trade-off is in data isolation (small arrays) and better load balancing (large arrays). For RAID 1 arrays, the trade-off is in better cost/capacity (mirrors—that is, a depth of two) and the ability to withstand multiple disk failures (shadows—that is, depths of three or even four). Read and write performance issues can also affect RAID 1 array size. For RAID 5 arrays, the trade-off is better data isolation and mean time between failures (MTBF) for small arrays and better cost/capacity/power for large arrays.
- Because hard disk failures are not independent, array sizes must be limited when the array is made up of actual physical disks (that is, a bottom-tier array). The exact amount of this limit is very difficult to determine.

The following is the array size guideline with no available hardware reliability data:

- Bottom-tier RAID 5 arrays should not extend beyond a single desk-side storage tower or a single row in a rack-mount configuration. This means approximately 8 to 14 physical disks for modern 3.5-inch storage enclosures. Smaller 2.5-inch disks can be racked more densely and therefore might require dividing into multiple arrays per enclosure.
- Bottom-tier mirrored arrays should not extend beyond two towers or rack-mount rows, with data being mirrored between towers or rows when possible. These guidelines help avoid or reduce the decrease in time between catastrophic failures that is caused by using multiple buses, power supplies, and so on from separate storage enclosures.

Selecting a Stripe Unit Size

The Windows volume manager stripe unit is fixed at 64 KB. Hardware solutions can range from 4 KB to 1 MB or more. Ideal stripe unit size maximizes the disk activity without unnecessarily breaking up requests by requiring multiple disks to service a single request. For example, consider the following:

- One long stream of sequential requests on JBOD uses only one disk at a time. To keep all striped disks in use for such a workload, the stripe unit should be at least 1/*n* where *n* is the request size.
- For *n* streams of small serialized random requests, if *n* is significantly greater than the number of disks and if there are no hot spots, striping does not increase performance over JBOD. However, if hot spots exist, the stripe unit size must maximize the possibility that a request will not be split while it minimizes the possibility of a hot spot falling entirely within one or two stripe units. You might

choose a low multiple of the typical request size, such as 5X or 10X, especially if the requests are on some boundary (for example, 4 KB or 8 KB).

- If requests are large and the average (or perhaps peak) number of outstanding requests is smaller than the number of disks, you might need to split some requests across disks so that all disks are being used. You can interpolate an appropriate stripe unit size from the previous two examples. For example, if you have 10 disks and 5 streams of requests, split each request in half (that is, use a stripe unit size equal to half the request size).
- Optimal stripe unit size increases with concurrency, burstiness, and typical request sizes.
- Optimal stripe unit size decreases with sequentiality and with good alignment between data boundaries and stripe unit boundaries.

Determining the Volume Layout

Placing individual workloads into separate volumes has advantages. For example, you can use one volume for the operating system or paging space and one or more volumes for shared user data, applications, and log files. The benefits include fault isolation, easier capacity planning, and easier performance analysis.

You can place different types of workloads into separate volumes on different physical disks. Using separate disks is especially important for any workload that creates heavy sequential loads such as log files, where a single set of physical disks (that compose the logical disk exposed to the operating system by the array controller) can be dedicated to handling the disk I/O that the updates to the log files create. Placing the paging file on a separate virtual disk might provide some improvements in performance during periods of high paging.

There is also an advantage to combining workloads on the same physical disks, if the disks do not experience high activity over the same time period. This is basically the partnering of hot data with cold data on the same physical drives.

The “first” partition on a volume usually uses the outermost tracks of the underlying disks and therefore provides better performance.

Storage-Related Parameters

You can adjust registry parameters on Windows Server 2008 R2 for high-throughput scenarios.

I/O Priorities

Windows Server 2008 and Windows Server 2008 R2 can specify an internal priority level on individual I/Os. Windows primarily uses this ability to de-prioritize background I/O activity and to give precedence to response-sensitive I/Os (for example, multimedia). However, extensions to file system APIs let applications specify I/O priorities per handle. The storage stack logic to sort out and manage I/O priorities has overhead, so if some disks will be targeted by only a single priority of I/Os (such as a SQL database disks), you can improve performance by disabling the I/O priority management for those disks by setting the following registry entry to zero:

```
HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\DeviceClasses
\{Device GUID}\DeviceParameters\Classpnp\IdlePrioritySupported
```

Storage-Related Performance Counters

The following sections describe performance counters that you can use for workload characterization, capacity planning, and identifying potential bottlenecks.

Logical Disk and Physical Disk

On servers that have heavy I/O workloads, you should enable the disk counters on a sampling basis or in specific scenarios to diagnose storage-related performance issues. Continuously monitoring disk counters can incur up to a 1 percent CPU overhead penalty.

The same counters are valuable in both the logical and physical disk counter objects. Logical disk statistics are tracked by the volume manager (or managers), and physical disk statistics are tracked by the partition manager.

The following counters are exposed through volume and partition managers:

- **% Disk Read Time, % Disk Time, % Disk Write Time, % Idle Time**

  These counters are of little value when multiple physical drives are behind logical disks. Imagine a subsystem of 100 physical drives presented to the operating system as five disks, each backed by a 20-disk RAID 0+1 array. Now imagine that the administrator spans the five disks to create one logical disk, volume *x*. One can assume that any serious system that needs that many physical disks has at least one outstanding request to volume *x* at any given time. This makes the volume appear to be 100% busy and 0% idle, when in fact the 100-disk array could be up to 99% idle with only a single request outstanding.

- **Average Disk Bytes / { Read | Write | Transfer }**

  This counter collects average, minimum, and maximum request sizes. If possible, you should observe individual or sub-workloads separately. You cannot differentiate multimodal distributions by using average values if the request types are consistently interspersed.

- **Average Disk Queue Length, Average Disk { Read | Write } Queue Length**

  These counters collect concurrency data, including burstiness and peak loads. Guidelines for queue lengths are given later in this guide. These counters represent the number of requests in flight below the driver that takes the statistics. This means that the requests are not necessarily queued but could actually be in service or completed and on the way back up the path. Possible in-flight locations include the following:

  - Waiting in an ATAport queue or a Storport queue.
  - Waiting in a queue in a miniport driver.
  - Waiting in a disk controller queue.
  - Waiting in an array controller queue.
  - Waiting in a hard disk queue (that is, on board a physical disk).
  - Actively receiving service from a physical disk.

  - Completed, but not yet back up the stack to where the statistics are collected.

- **Average Disk second / {Read | Write | Transfer}**

  These counters collect disk request response time data and possibly extrapolate service time data. They are probably the most straightforward indicators of storage subsystem bottlenecks. Guidelines for response times are given later in this guide. If possible, you should observe individual or sub-workloads separately. You cannot differentiate multimodal distributions by using Perfmon if the requests are consistently interspersed.

- **Current Disk Queue Length**

  This counter instantly measures the number of requests in flight and therefore is subject to extreme variance. Therefore, this counter is not useful except to check for the existence of many short bursts of activity.

- **Disk Bytes / second, Disk {Read | Write } Bytes / second**

  This counter collects throughput data. If the sample time is long enough, a histogram of the array's response to specific loads (queues, request sizes, and so on) can be analyzed. If possible, you should observe individual or sub-workloads separately.

- **Disk {Reads | Writes | Transfers } / second**

  This counter collects throughput data. If the sample time is long enough, a histogram of the array's response to specific loads (queues, request sizes, and so on) can be analyzed. If possible, you should observe individual or sub-workloads separately.

- **Split I/O / second**

  This counter is useful only if the value is not in the noise. If it becomes significant, in terms of split I/Os per second per physical disk, further investigation could be needed to determine the size of the original requests that are being split and the workload that is generating them.

Note: If the Windows standard stacked driver's scheme is circumvented for some controller, so-called "monolithic" drivers can assume the role of partition manager or volume manager. If so, the monolithic driver writer must supply the counters listed above through the Windows Management Instrumentation (WMI) interface, or the counters will not be available.

Processor Information

- **% DPC Time, % Interrupt Time, % Privileged Time**

  If interrupt time and DPC time are a large part of privileged time, the kernel is spending a long time processing I/Os. Sometimes, it is best to keep interrupts and DPCs affinitized to only a few CPUs on a multiprocessor system to improve cache locality. At other times, it is best to distribute the interrupts and DPCs among many CPUs to prevent the interrupt and DPC activity from becoming a bottleneck on individual CPUs.

- **DPCs Queued / second**

  This counter is another measurement of how DPCs are using CPU time and kernel resources.
- **Interrupts / second**

  This counter is another measurement of how interrupts are using CPU time and kernel resources. Modern disk controllers often combine or coalesce interrupts so that a single interrupt causes the processing of multiple I/O completions. Of course, there is a trade-off between delaying interrupts (and therefore completions) and amortizing CPU processing time.

Power Protection and Advanced Performance Option

There are two performance-related options for every disk under **Disk > Properties > Policies**:

- Enable write caching.
- Enable an "advanced performance" mode that assumes that the storage is protected against power failures.

Enabling write caching means that the storage hardware can indicate to the operating system that a write request is complete even though the data has not been flushed from the volatile intermediate hardware cache(s) to its final nonvolatile storage location. Note that with this action a period of time passes during which a power failure or other catastrophic event could result in data loss. However, this period is typically fairly short because write caches in the storage hardware are usually flushed during any period of idle activity. Cache flushes are also requested frequently by the operating system (or some applications) to explicitly force writes to be written to the final storage medium in a specific order. Alternately, hardware time-outs at the cache level might force dirty data out of the caches.

Other than cache flush requests, the only means of synchronizing writes is to tag them as "write-through." Storage hardware is supposed to guarantee that a write-through request's data has reached nonvolatile storage (such as magnetic media on a disk platter) before it indicates a successful request completion to the operating system. Some commodity disks or disk controllers may not honor write-through semantics. In particular, ATA/SATA/USB storage components may not support the ForceUnitAccess flag that is used to tag write-through requests in the hardware. Enterprise storage subsystems typically use battery-backed caches or use SAS/SCSI/FC hardware to correctly maintain write-through semantics.

The "advanced performance" disk policy option is available only when write caching is enabled. This option strips all write-through flags from disk requests and removes all flush-cache commands from the request stream. If you have power protection for all hardware write caches along the I/O path, you do not need to worry about those two pieces of functionality. By definition, any dirty data that resides in a power-protected write cache is safe and appears to have occurred "in-order" from the software's viewpoint. If power is lost to the final storage location while the data is being flushed from a write cache, the cache manager can retry the write operation after power has been restored to the relevant storage components.

Block Alignment (DISKPART)

NTFS aligns its metadata and data clusters to partition boundaries in increments of the cluster size (which is selected during file system creation or set by default to 4 KB). In releases of Windows prior to Windows Server 2008, the partition boundary offset for a specific disk partition could be misaligned when it was compared to array disk stripe unit boundaries. This caused small requests to be unintentionally split across multiple disks. To force alignment, you were required to use Diskpar.exe or Diskpart.exe at the time that the partition was created.

In Windows Server 2008 and Windows Server 2008 R2, partitions are created by default with a 1-MB offset, which provides good alignment for the power-of-two stripe unit sizes that are typically found in hardware. If the stripe unit size is set to a size that is greater than 1 MB, the alignment issue is much less of a problem because small requests rarely cross large stripe unit boundaries. Note that Windows Server 2008 and Windows Server 2008 R2 default to a 64-KB offset if the disk is smaller than 4 GB.

If alignment is still a problem even with the default offset, you can use Diskpart.exe to force alternative alignments when you create a partition.

Solid-State and Hybrid Drives

Previously, the cost of large quantities of nonvolatile memory was prohibitive for most server configurations. Exceptions included aerospace or military applications in which the shock and vibration tolerance of flash memory is highly desirable.

As the cost of flash memory continues to decrease, it becomes more possible to use nonvolatile memory (NVM) to improve storage subsystem response time on servers. The typical vehicle for incorporating NVM in a server is the solid-state disk (SSD). One cost-effective strategy is to place only the hottest data of a workload onto nonvolatile memory. In Windows Server 2008 R2, as in previous versions of Windows Server, partitioning can be performed only by applications that store data on an SSD; Windows does not try to dynamically determine what data should optimally be stored on SSDs versus rotating media.

Response Times

You can use tools such as Perfmon to obtain data on disk request response times. Write requests that enter a write-back hardware cache often have very low response times (less than 1 ms) because completion depends on dynamic RAM (DRAM) speeds instead of rotating disk speeds. The data is lazily written to disk media in the background. As the workload begins to saturate the cache, response times increase until the write cache’s only potential benefit is a better ordering of requests to reduce positioning delays.

For JBOD arrays, reads and writes have approximately the same performance characteristics. Writes can be slightly longer due to additional mechanical “settling” delays. With modern hard disks, positioning delays for random requests are 5 to 15 ms. Smaller 2.5-inch drives have shorter positioning distances and lighter actuators, so they generally provide faster seek times than comparable larger 3.5-inch drives. Positioning delays for sequential requests should be insignificant except for streams of write-through requests, where each positioning delay should approximate the required time for a complete disk rotation. (Write-through requests are typically identified by the ForceUnitAccess (FUA) flag on the disk request.)

Transfer times are usually less significant when they are compared to positioning delays, except for sequential requests and large requests (larger than 256 KB) that are instead dominated by disk media access speeds as the requests become larger or more sequential. Modern enterprise disks access their media at 50 to 150 MB/s depending on rotation speed and sectors per track, which varies across a range of blocks on a specific disk model. The outermost tracks can have up to twice the sequential throughput of innermost tracks.

If the stripe unit size of a striped array is well chosen, each request is serviced by a single disk—except for low-concurrency workloads. So, the same general positioning and transfer times still apply.

For mirrored arrays, a write completion must typically wait for both disks to complete the request. Depending on how the requests are scheduled, the two completions of the requests could take a long time. However, although writes for mirrored arrays generally should not take twice the time to complete, they are typically slower than for JBOD. Reads can experience a performance increase if the array controller is dynamically load balancing or factoring in spatial locality.

For RAID 5 arrays (rotated parity), small writes become four separate requests in the typical read-modify-write scenario. In the best case, this is approximately the equivalent of two mirrored reads plus a full rotation of the disks, if you assume that the read/write pairs continue in parallel. Traditional RAID 6 slows write performance even more because each RAID 6 write request becomes three reads plus three writes.

You must consider the performance effect of redundant arrays on read and write requests when you plan subsystems or analyze performance data. For example, Perfmon might show that 50 writes per second are being processed by volume x, but in reality this could mean 100 requests per second for a mirrored array, 200 requests per second for a RAID 5 array, or even more than 200 requests per second if the requests are split across stripe units.

Use the following are response-time guidelines if no workload details are available:

- For a lightly loaded system, average write response times should be less than 25 ms on RAID 5 or RAID 6 and less than 15 ms on non-RAID 5 or non-RAID 6 disks. Average read response times should be less than 15 ms regardless.
- For a heavily loaded system that is not saturated, average write response times should be less than 75 ms on RAID 5 or RAID 6 and less than 50 ms on non-RAID 5 or non-RAID 6 disks. Average read response times should be less than 50 ms.

Queue Lengths

Several opinions exist about what constitutes excessive disk request queuing. This guide assumes that the boundary between a busy disk subsystem and a saturated one is a persistent average of two requests per physical disk. A disk subsystem is near saturation when every physical disk is servicing a request and has at least one queued-up request to maintain maximum concurrency—that is, to keep the data pipeline flowing. Note that in this guideline, disk requests split into multiple requests (because of striping or redundancy maintenance) are considered multiple requests.

This rule has caveats, because most administrators do not want all physical disks constantly busy. Because disk workloads are often bursty, this rule is more likely applied over shorter periods of (peak) time. Requests are typically not uniformly spread among all hard disks at the same time, so the administrator must consider deviations between queues—especially for bursty workloads. Conversely, a longer queue provides more opportunity for disk request schedulers to reduce positioning delays or optimize for full-stripe RAID 5 writes or mirrored read selection.

Because hardware has an increased capability to queue up requests—either through multiple queuing agents along the path or merely agents with more queuing capability—increasing the multiplier threshold might allow more concurrency within the hardware. This creates a potential increase in response time variance, however. Ideally, the additional queuing time is balanced by increased concurrency and reduced mechanical positioning times.

Use the following queue length targets when few workload details are available:

- For a lightly loaded system, the average queue length should be less than one per physical disk, with occasional spikes of 10 or less. If the workload is write heavy, the average queue length above a mirrored controller should be less than 0.6 per physical disk and the average queue length above a RAID 5 or RAID 6 controller should be less than 0.3 per physical disk.
- For a heavily loaded system that is not saturated, the average queue length should be less than 2.5 per physical disk, with infrequent spikes up to 20. If the workload is write heavy, the average queue length above a mirrored controller should be less than 1.5 per physical disk and the average queue length above a RAID 5 or RAID 6 controller should be less than 1.0 per physical disk.
- For workloads of sequential requests, larger queue lengths can be tolerated because services times and therefore response times are much shorter than those for a random workload.

For more details on Windows storage performance, see “Resources” later in this guide.

Performance Tuning for Web Servers

Selecting the Proper Hardware for Performance

It is important to select the proper hardware to satisfy the expected Web load, considering average load, peak load, capacity, growth plans, and response times. Hardware bottlenecks limit the effectiveness of software tuning. “Choosing and Tuning Server Hardware” earlier in this guide provides recommendations for hardware to avoid the following performance constraints:

- Slow CPUs offer limited processing power for ASP, ASP.NET, and SSL scenarios.
- A small L2 processor cache might adversely affect performance.

- A limited amount of memory affects the number of sites that can be hosted, how many dynamic content scripts (such as ASP.NET) can be stored, and the number of application pools or worker processes.
- Networking becomes a bottleneck because of an inefficient networking adapter.
- The file system becomes a bottleneck because of an inefficient disk subsystem or storage adapter.

Operating System Practices

If possible, do a clean installation of the operating system software. Upgrading can leave outdated, unwanted, or suboptimal registry settings and previously installed services and applications that consume resources if they are started automatically. If another operating system is installed and must be kept, you should install the new operating system on a different partition. Otherwise, the new installation overwrites the settings under Program Files\Common Files.

To reduce disk access interference, place the system pagefile, operating system, Web data, ASP template cache, and Internet Information Services (IIS) log on separate physical disks if possible.

To reduce contention for system resources, install SQL and IIS on different servers if possible.

Avoid installing nonessential services and applications. In some cases, it might be worthwhile to disable services that are not required on a system.

Tuning IIS 7.5

Internet Information Services (IIS) 7.5 is the version that ships as part of Windows Server 2008 R2. It uses a process model similar to that of IIS 6.0. A kernel-mode HTTP listener (Http.sys) receives and routes HTTP requests and can even satisfy requests from its response cache. Worker processes register for URL subspaces, and Http.sys routes the request to the appropriate process (or set of processes for application pools).

The IIS 7.5 process relies on the kernel-mode Web driver, Http.sys. Http.sys is responsible for connection management and request handling. The request can be either served from the Http.sys cache or handed to a worker process for further handling (see Figure 5). Multiple worker processes can be configured, which provides isolation at a reduced cost.

Http.sys includes a response cache. When a request matches an entry in the response cache, Http.sys sends the cache response directly from kernel mode. Figure 5 shows the request flow from the network through Http.sys and potentially up to a worker process. Some Web application platforms, such as ASP.NET, provide mechanisms to enable any dynamic content to be cached in the kernel cache. The static file handler in IIS 7.5 automatically caches frequently requested files in Http.sys.

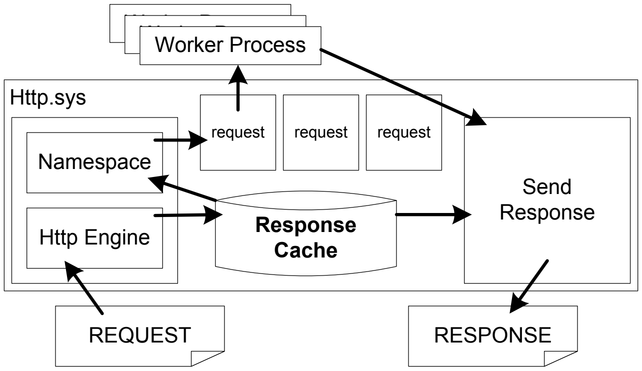


Figure 5. Request Handling in IIS 7.5

Because a Web server has a kernel-mode and a user-mode component, both components must be tuned for optimal performance. Therefore, tuning IIS 7.5 for a specific workload includes configuring the following:

- Http.sys (the kernel-mode driver) and the associated kernel-mode cache.
- Worker processes and user-mode IIS, including application pool configuration.
- Certain tuning parameters that affect performance.

The following sections discuss how to configure the kernel-mode and user-mode aspects of IIS 7.5.

Kernel-Mode Tunings

Performance-related Http.sys settings fall into two broad categories: cache management, and connection and request management. All registry settings are stored under the following entry:

```
HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\Http\Parameters
```

If the HTTP service is already running, you must restart it for the changes to take effect.

Cache Management Settings

One benefit that Http.sys provides is a kernel-mode cache. If the response is in the kernel-mode cache, you can satisfy an HTTP request entirely from kernel mode, which significantly lowers the CPU cost of handling the request. However, the kernel-mode cache of IIS 7.5 is a physical memory–based cache and the cost of an entry is the memory that it occupies.

An entry in the cache is helpful only when it is used. However, the entry always consumes physical memory, whether the entry is being used or not. You must evaluate the usefulness of an item in the cache (the savings from being able to serve it from the cache) and its cost (the physical memory occupied) over the lifetime of the entry by considering the available resources (CPU and physical memory) and the workload requirements. Http.sys tries to keep only useful, actively accessed items in the cache, but you can increase the performance of the Web server by tuning the Http.sys cache for particular workloads.

The following are some useful settings for the Http.sys kernel-mode cache:

- **UriEnableCache.** Default value 1.

  A nonzero value enables the kernel-mode response and fragment cache. For most workloads, the cache should remain enabled. Consider disabling the cache if you expect very low response and fragment cache usage.

- **UriMaxCacheMegabyteCount.** Default value 0.

  A nonzero value specifies the maximum memory that is available to the kernel cache. The default value, 0, enables the system to automatically adjust how much memory is available to the cache. Note that specifying the size sets only the maximum and the system might not let the cache grow to the specified size.

- **UriMaxUriBytes.** Default value 262144 bytes (256 KB).

  This is the maximum size of an entry in the kernel-mode cache. Responses or fragments larger than this are not cached. If you have enough memory, consider increasing the limit. If memory is limited and large entries are crowding out smaller ones, it might be helpful to lower the limit.

- **UriScavengerPeriod.** Default value 120 seconds.

  The Http.sys cache is periodically scanned by a scavenger, and entries that are not accessed between scavenger scans are removed. Setting the scavenger period to a high value reduces the number of scavenger scans. However, the cache memory usage might increase because older, less frequently accessed entries can remain in the cache. Setting the period to too low a value causes more frequent scavenger scans and might result in too many flushes and cache churn.

Request and Connection Management Settings

In Windows Server 2008 R2, Http.sys manages connections automatically. The following registry keys that were used in earlier releases are considered deprecated and are not necessary in Windows Server 2008 R2:

- **MaxConnections**

  ```
  HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\Http\
  Parameters\MaxConnections
  ```

- **IdleConnectionsHighMark**

  ```
  HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\Http\
  Parameters\IdleConnectionsHighMark
  ```

- **IdleConnectionsLowMark**

  ```
  HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\Http\
  Parameters\IdleConnectionsLowMark
  ```

- **IdleListTrimmerPeriod**
  ```
  HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\Http\
  Parameters\IdleListTrimmerPeriod
  ```

- **RequestBufferLookasideDepth**
  ```
  HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\Http\
  Parameters\RequestBufferLookasideDepth
  ```

- **InternalRequestLookasideDepth**
  ```
  HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\Http\
  Parameters\InternalRequestLookasideDepth
  ```

User-Mode Settings

The settings in this section affect the IIS 7.5 worker process behavior. Most of these settings can be found in the %SystemRoot%\system32\inetsrv\config \applicationHost.config XML configuration file. Use either Appcmd.exe or the IIS 7.5 management console to change them. Most settings are automatically detected and do not require a restart of the IIS 7.5 worker processes or Web Application Server.

User-Mode Cache Behavior Settings

This section describes the settings that affect caching behavior in IIS 7.5. The user-mode cache is implemented as a module that listens to the global caching events that the integrated pipeline raises. To completely disable the user-mode cache, remove the FileCacheModule (Cachfile.dll) module from the list of installed modules in the system.webServer/globalModules configuration section in applicationHost.config.

system.webServer/caching

Attribute	Description	Default
Enabled	Disables the user-mode IIS cache when set to false. When the cache hit rate is very small, you can disable the cache completely to avoid the overhead that is associated with the cache code path. Disabling the user-mode cache does not disable the kernel-mode cache.	True
enableKernelCache	Disables the kernel-mode cache when set to false.	True
maxCacheSize	Limits the IIS user-mode cache size to the specified size in megabytes. IIS adjusts the default depending on available memory. Choose the value carefully based on the size of the hot set (the set of frequently accessed files) versus the amount of RAM or the IIS process address space, which is limited to 2 GB on 32-bit systems.	0
maxResponseSize	Lets files up to the specified size be cached. The actual value depends on the number and size of the largest files in the dataset versus the available RAM. Caching large, frequently requested files can reduce CPU usage, disk access, and associated latencies. The default value is 256 KB.	262144

Compression Behavior Settings

In Windows Server 2008 R2, IIS 7.5 compresses static content by default. Also, compression of dynamic content is enabled by default when the

DynamicCompressionModule is installed. Compression reduces bandwidth usage but increases CPU usage. Compressed content is cached in the kernel-mode cache if possible. IIS 7.5 lets compression be controlled independently for static and dynamic content. Static content typically refers to content that does not change, such as GIF or HTM files. Dynamic content is typically generated by scripts or code on the server, that is, ASP.NET pages. You can customize the classification of any particular extension as static or dynamic.

To completely disable compression, remove StaticCompressionModule and DynamicCompressionModule from the list of modules in the system.webServer/globalModules section in applicationHost.config.

system.webServer/httpCompression

Attribute	Description	Default
staticCompression-EnableCpuUsage, staticCompression-DisableCpuUsage, dynamicCompression-EnableCpuUsage, dynamicCompression-DisableCpuUsage	Enables or disables compression if the current percentage CPU usage goes above or below specified limits. IIS 7.5 automatically disables compression if steady-state CPU increases above the disable threshold. Compression is re-enabled if CPU drops below the enable threshold.	50, 100, 50, and 90 respectively
directory	Specifies the directory in which compressed versions of static files are temporarily stored and cached. Consider moving this directory off the system drive if it is accessed frequently. The default value is %SystemDrive%\inetpub\temp \IIS Temporary Compressed Files.	See Description column
doDiskSpaceLimiting	Specifies whether a limit exists for how much disk space all compressed files, which are stored in the compression directory that is specified by directory, can occupy.	True
maxDiskSpaceUsage	Specifies the number of bytes of disk space that compressed files can occupy in the compression directory. This setting might need to be increased if the total size of all compressed content is too large.	100 MB

system.webServer/urlCompression

Attribute	Description	Default
doStaticCompression	Specifies whether static content is compressed.	True
doDynamicCompression	Specifies whether dynamic content is compressed.	True *(changed in Windows Server 2008 R2)*

Note: For IIS 7.5 servers that have low average CPU usage, consider enabling compression for dynamic content, especially if responses are large. This should first be done in a test environment to assess the effect on the CPU usage from the baseline.

Tuning the Default Document List

The default document module handles HTTP requests for the root of a directory and translates them into requests for a specific file, such as Default.htm or Index.htm. On average, around 25 percent of all requests on the Internet go through the default document path. This varies significantly for individual sites. When an HTTP request does not specify a file name, the default document module linearly walks the list of allowed default documents, searching for each one in the file system. This can adversely affect performance, especially if reaching the content requires making a network round trip or touching a disk.

You can avoid the overhead by selectively disabling default documents and by reducing or ordering the list of documents. For Web sites that use a default document, you should reduce the list to only the default document types that are used. Additionally, order the list so that it begins with the most frequently accessed default document file name. Finally, you can selectively set the default document behavior on particular URLs by using custom configuration inside a location tag in applicationHost.config or by inserting a Web.config file directly in the content directory. This allows a hybrid approach, which will enable default documents only where they are necessary and will set the list to the correct file name for each URL.

To disable default documents completely, remove DefaultDocumentModule from the list of modules in the system.webServer/globalModules section in applicationHost.config.

system.webServer/defaultDocument

Attribute	Description	Default
enabled	Specifies that default documents are enabled.	True
<files> element	Specifies the file names that are configured as default documents. The default list is Default.htm, Default.asp, Index.htm, Index.html, Iisstart.htm, and Default.aspx.	See Description column

Central Binary Logging

Binary IIS logging reduces CPU usage, disk I/O, and disk space usage. Central binary logging is directed to a single file in binary format, regardless of the number of hosted sites. Parsing binary-format logs requires a post-processing tool.

You can enable central binary logging by setting the *centralLogFileMode* attribute to CentralBinary and setting the *enabled* attribute to True. Consider moving the location of the central log file off the system partition and onto a dedicated logging partition to avoid contention between system activities and logging activities.

system.applicationHost/log

Attribute	Description	Default
centralLogFileMode	Specifies the logging mode for a server. Change this value to CentralBinary to enable central binary logging.	Site

system.applicationHost/log/centralBinaryLogFile

Attribute	Description	Default
enabled	Specifies whether central binary logging is enabled.	False
directory	Specifies the directory where log entries are written. *The default directory is: %SystemDrive%\inetpub\logs\LogFiles*	See Description column

Application and Site Tunings

The following settings relate to application pool and site tunings.

system.applicationHost/applicationPools/applicationPoolDefaults

Attribute	Description	Default
queueLength	Indicates to the Universal Listener how many requests are queued for an application pool before future requests are rejected. When the value for this property is exceeded, IIS rejects subsequent requests with a 503 error. Consider increasing this for applications that communicate with high-latency back-end data stores if 503 errors are observed.	1000
enable32BitAppOnWin64	When True, enables a 32-bit application to run on a computer that has a 64-bit processor. Consider enabling 32-bit mode if memory consumption is a concern. Because pointer sizes and instruction sizes are smaller, 32-bit applications use less memory than 64-bit applications. The drawback to running 32-bit applications on a 64-bit machine is that user-mode address space is limited to 4 GB.	False

system.applicationHost/sites/VirtualDirectoryDefault

Attribute	Description	Default
allowSubDirConfig	Specifies whether IIS looks for Web.config files in content directories lower than the current level (True) or does not look for Web.config files in content directories lower than the current level (False). When configuration is queried in the IIS 7.5 pipeline, it is not known whether a URL (/<name>.htm) is a reference to a directory or a file name. By default, IIS 7.5 must assume that /<name>.htm is a reference to a directory and search for configuration in a /<name>.htm/web.config file. This results in an additional file system operation that can be costly. By imposing a simple limitation, which allows configuration only in virtual directories, IIS 7.5 can then know that unless /<name>.htm is a virtual directory it should not look for a configuration file. Skipping the additional file operations can significantly improve performance to Web sites that have a very large set of randomly accessed static content.	True

Managing IIS 7.5 Modules

IIS 7.5 has been refactored into multiple, user-pluggable modules to support a more modular structure. This refactorization has a small cost. For each module present, the integrated pipeline must call into the module for every event that is relevant to the module. This happens regardless of whether the module must do any work. You can conserve CPU cycles and memory by removing all modules that are not relevant to a particular Web site.

A Web server that is tuned only for simple static files might include only the following five modules: UriCacheModule, HttpCacheModule, StaticFileModule, AnonymousAuthenticationModule, and HttpLoggingModule.

To remove modules from applicationHost.config, remove all references to the module from the system.webServer/handlers and system.webServer/modules sections in addition to the module declaration in system.webServer/globalModules.

Classic ASP Settings

The following settings apply only to classic ASP pages and do not affect ASP.NET settings. For performance recommendations on ASP.NET, see the article about high-performance Web applications in "Resources" later in this guide.

system.webServer/asp/cache

Attribute	Description	Default
diskTemplateCacheDirectory	This attribute contains the name of the directory that ASP uses to store compiled ASP templates to disk after overflow of the in-memory cache. Recommendation: If possible, set to a platter not in heavy use, for example, not shared with the operating system, pagefile, IIS log, or other frequently accessed content. The default directory is: %SystemDrive%\inetpub\temp \ASP Compiled Templates	See Description column
maxDiskTemplateCacheFiles	This attribute specifies the maximum number of compiled ASP templates that can be stored. Recommendation: Set to the maximum value of 0x7FFFFFFF.	2000
scriptFileCacheSize	This attribute specifies the number of precompiled script files to cache. Recommendation: Set to as many ASP templates as memory limits allow.	500
scriptEngineCacheMax	This attribute specifies the maximum number of scripting engines that ASP pages will keep cached in memory. Recommendation: Set to as many script engines as memory limits allow.	250

system.webServer/asp/limits

Attribute	Description	Default
processorThreadMax	Specifies the maximum number of worker threads per processor that ASP can create. Increase if the current setting is insufficient to handle the load, possibly causing errors when it is serving some requests or under-usage of CPU resources.	25

system.webServer/asp/comPlus

Attribute	Description	Default
executeInMta	Set to True if errors or failures are detected while it is serving some ASP content. This can occur, for example, when hosting multiple isolated sites in which each site runs under its own worker process. Errors are typically reported from COM+ in the Event Viewer. This setting enables the multithreaded apartment model in ASP.	False

ASP.NET Concurrency Setting

By default, ASP.NET limits request concurrency to reduce steady-state memory consumption on the server. High concurrency applications might need to adjust some settings to improve overall performance. These settings are stored under the following registry entry:

```
HKEY_LOCAL_MACHINE\Software\Microsoft\ASP.NET\2.0.50727.0\Parameters
```

The following setting is useful for fully using resources on a system:

- **MaxConcurrentRequestPerCpu.** Default value 12.

  This setting limits the maximum number of concurrently executing ASP.NET requests on a system. The default value is conservative to reduce memory consumption of ASP.NET applications. Consider increasing this limit on systems that run applications that perform long, synchronous I/O operations. Otherwise, users can experience high latency because of queuing or request failures from exceeding queue limits under high load with the default setting.

Worker Process and Recycling Options

The options for recycling IIS worker processes under the IIS Administrator user interface provide practical solutions to acute situations or events without requiring intervention, a service reset, or even a computer reset. Such situations and events include memory leaks, increasing memory load, or unresponsive or idle worker processes. Under ordinary conditions, recycling options might not be needed and can be turned off or the system can be configured to recycle very infrequently.

You can enable process recycling for a particular application by adding attributes to the recycling/periodicRestart element. The recycle event can be triggered by several events including memory usage, a fixed number of requests, and a fixed time period. When a worker process is recycled, the queued and executing requests are drained and a new process is simultaneously started to service new requests. The periodicRestart element is per-application, meaning that each attribute in the table below will be partitioned on a per-application basis.

system.applicationHost/applicationPools/ApplicationPoolDefaults/recycling/periodicRestart

Attribute	Description	Default
memory	Enable process recycling if virtual memory consumption exceeds the specified limit in kilobytes. This is a useful setting for 32-bit machines that have a small, 2-GB address space to avoid failed requests because of out-of-memory errors.	0
privateMemory	Enable process recycling if private memory allocations exceed a specified limit in kilobytes.	0
requests	Enable process recycling after a certain number of requests.	0
time	Enable process recycling after a specified time period. (The default is 29 hours.)	29:00:00

Secure Sockets Layer Tuning Parameters

The use of secure sockets layer (SSL) imposes additional CPU cost. The most expensive component of SSL is the session establishment cost (involving a full handshake), and then reconnection cost and encryption/decryption cost. For better SSL performance, do the following:

- Enable keep-alives for SSL sessions. This eliminates the session establishment costs.
- Reuse sessions when appropriate, especially with non-keep-alive traffic.
- Note that larger keys provide more security but also use more CPU time.
- Note that not all components of your page might need to be encrypted. However, mixing plain HTTP and HTTPS might result in a pop-up warning on the client browser that not all content on the page is secure.

ISAPI

No special tuning parameters are needed for the Internet Server API (ISAPI) applications. If writing a private ISAPI extension, make sure that you code it efficiently for performance and resource use. See also “Other Issues that Affect IIS Performance” later in this guide.

Managed Code Tuning Guidelines

The new integrated pipeline model in IIS 7.5 enables a high degree of flexibility and extensibility. Custom modules that are implemented in native or managed code can be inserted into the pipeline or can replace existing modules. Although this extensibility model offers convenience and simplicity, you should be careful before you insert new managed modules that hook into global events. Adding a global managed module means that all requests, including static file requests, must touch managed code. Custom modules are susceptible to events such as garbage collection in addition to adding significant CPU cost because of marshaling data between native and managed code. If possible, you should implement global modules in native (C/C++) code.

When you first deploy an ASP.NET Web site, make sure that you precompile all scripts. You can do this by calling one .NET script in each directory. Reset IIS after compilation is complete. Recompile after changes to Machine.config, Web.config, or any .aspx script.

If session state is not needed, make sure that you turn it off for each page.

When you run multiple hosts that contain ASP.NET scripts in isolated mode (one application pool per site), monitor the memory usage. Make sure that the server that runs has enough RAM for the expected number of concurrently running application pools. Consider using multiple application domains instead of multiple isolated processes.

For performance recommendations on ASP.NET, see the article about high-performance web applications in "Resources" later in this guide.

Other Issues that Affect IIS Performance

The following issues affect IIS performance:

- Installation of filters that are not cache-aware.

  The installation of a filter that is not HTTP-cache-aware causes IIS to completely disable caching, which results in poor performance. Old ISAPI filters that were written before IIS 6.0 can cause this behavior.

- Common Gateway Interface (CGI) requests.

  For performance reasons, the use of CGI applications for serving requests is not recommended under IIS. The frequent creation and deletion of CGI processes involves significant overhead. Better alternatives include the use of ISAPI application and ASP or ASP.NET scripts. Isolation is available for each of these options.

NTFS File System Setting

Under HKLM\System\CurrentControlSet\Control\FileSystem\ is NtfsDisableLastAccessUpdate (REG_DWORD) 1.

This system-global switch reduces disk I/O load and latencies by disabling the updating of the date and time stamp for the last file or directory access. This key is set to 1 by default. Clean installations of Windows Server 2008 and Windows Server 2008 R2 set this key by default and you do not need to adjust it. Earlier versions of Windows operating systems did not set this key. If your server is running an earlier version of Windows or was upgraded to Windows Server 2008 or Windows Server 2008 R2, you should set this key to 1.

Disabling the updates is effective when you are using large data sets (or many hosts) that contain thousands of directories. We recommend that you use IIS logging instead if you maintain this information only for Web administration.

Caution: Some applications such as incremental backup utilities rely on this update information and do not function correctly without it.

Networking Subsystem Performance Settings for IIS

See “Performance Tuning for Networking Subsystem” earlier in this guide.

Performance Tuning for File Servers

Selecting the Proper Hardware for Performance

You should select the proper hardware to satisfy the expected file server load, considering average load, peak load, capacity, growth plans, and response times. Hardware bottlenecks limit the effectiveness of software tuning. "Choosing and Tuning Server Hardware" earlier in this guide provides recommendations for hardware. The sections on networking and storage subsystems also apply to file servers.

Server Message Block Model

This following sections provide information about the Server Message Block (SMB) model for client-server communication, including the SMB 1 and SMB 2 protocols.

SMB Model Overview

The SMB model consists of two entities: the client and the server.

On the client, applications perform system calls by requesting operations on remote files. These requests are handled by the redirector subsystem (Rdbss.sys) and the SMB mini-redirector (Mrxsmb.sys), which translate them into SMB protocol sessions and requests over TCP/IP. Starting with Windows Vista, the SMB 2 protocol is supported. The Mrxsmb10.sys driver handles legacy SMB traffic, and the Mrxsmb20.sys driver handles SMB 2 traffic.

On the server, SMB connections are accepted and SMB requests are processed as local file system operations through NTFS and the local storage stack. The Srv.sys driver handles legacy SMB traffic, and the Srv2.sys driver handles SMB 2 traffic. The Srvnet.sys component implements the interface between networking and the file server for both SMB protocols. File system metadata and content can be cached in memory through the system cache in the kernel (Ntoskrnl.exe).

Figure 6 summarizes the different layers that a user request on a client machine must pass through to perform file operations over the network on a remote SMB file server that uses SMB 2.

Windows 7 and Windows Server 2008 R2 introduce SMB 2.1. The new protocol version has optimizations to reduce network chattiness, which improves overall performance.

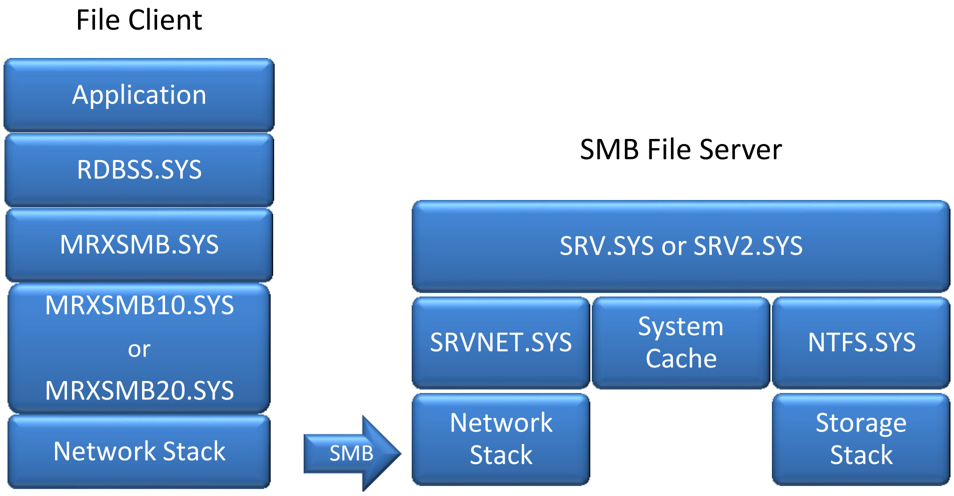


Figure 6. Windows SMB Communication Model Components

SMB Configuration Considerations

Do not enable any services or features that your particular file server and file clients do not require. These might include SMB signing, client-side caching, file system minifilters, search service, scheduled tasks, NTFS encryption, NTFS compression, IPSEC, firewall filters, Teredo, and antivirus features.

Ensure that any BIOS and operating system power management mode is set as needed, which might include High Performance mode. Ensure that the latest and fastest storage and networking device drivers are installed.

Copying files is one of the common operations performed on a file server. Windows has several built-in file copy utilities that you can run in a command shell, including **xcopy** and **robocopy**. When you use **xcopy**, we recommend adding the **/q** and **/k** options to your existing parameters, when applicable, to maximize performance. The former option reduces CPU overhead by reducing console output and the latter reduces network traffic. When using **robocopy**, the **/mt** option (new to Windows Server 2008 R2) can significantly improve speed on remote file transfers by using multiple threads when copying multiple small files. We also recommend the **/log** option to reduce console output by redirecting to NUL or to a file.

Previous releases of Windows Server sometimes benefitted from tools that limit the working-set size of the Windows file cache. These tools are not necessary on most servers running Windows Server 2008 R2. You should reevaluate your use of such tools.

Tuning Parameters for SMB File Servers

The following registry tuning parameters can affect the performance of SMB file servers:

- **NtfsDisable8dot3NameCreation**

  ```
  HKLM\System\CurrentControlSet\Control\FileSystem\REG_DWORD)
  ```

  The default is 0. This parameter determines whether NTFS generates a short name in the 8.3 (MS-DOS®) naming convention for long file names and for file names that contain characters from the extended character set. If the value of this entry is 0, files can have two names: the name that the user specifies and the short name that NTFS generates. If the user-specified name follows the 8.3 naming convention, NTFS does not generate a short name.

Changing this value does not change the contents of a file, but it avoids the short-name attribute creation for the file, which also changes how NTFS displays and manages the file. For most SMB file servers, the recommended setting is 1.

Starting with Windows Server 2008 R2, you can disable 8.3 name creation on a per-volume basis without using the global NtfsDisable8dot3NameCreation setting. You can do this with the built-in **fsutil** tool. For example, to disable 8.3 name creation on the d: volume, run **fsutil 8dot3name set d: 1** from a command prompt window. You can view help text by using the command **fsutil 8dot3name**.

- **TreatHostAsStableStorage**

  ```
  HKLM\System\CurrentControlSet\Services\LanmanServer
  \Parameters\(REG_DWORD)
  ```

  The default is 0. This parameter disables the processing of write flush commands from clients. If the value of this entry is 1, the server performance and client latency for power-protected servers can improve. Workloads that resemble the NetBench file server benchmark benefit from this behavior.

- **AsynchronousCredits**

  ```
  HKLM\System\CurrentControlSet\Services\LanmanServer
  \Parameters\(REG_DWORD)
  ```

  The default is 512. This parameter limits the number of concurrent asynchronous SMB commands that are allowed on a single connection. Some file clients such as IIS servers require a large amount of concurrency, with file change notification requests in particular. The value of this entry can be increased to support these clients.

- **Smb2CreditsMin** and **Smb2CreditsMax**

  ```
  HKLM\System\CurrentControlSet\Services\LanmanServer
  \Parameters\(REG_DWORD)
  ```

  The defaults are 64 and 1024, respectively. These parameters allow the server to throttle client operation concurrency dynamically within the specified boundaries. Some clients might achieve increased throughput with higher concurrency limits. One example is file copy over high-bandwidth, high-latency links.

- **AdditionalCriticalWorkerThreads**

  ```
  HKLM\System\CurrentControlSet\Control\Session
  Manager\Executive\(REG_DWORD)
  ```

  The default is 0, which means that no additional critical kernel worker threads are added to the default number. This value affects the number of threads that the file system cache uses for read-ahead and write-behind requests. Raising this value can allow for more queued I/O in the storage subsystem and can improve I/O performance, particularly on systems with many processors and powerful storage hardware.

- **MaximumTunnelEntries**

  ```
  HKLM\System\CurrentControlSet\Control\FileSystem\(REG_DWORD)
  ```

The default is 1024. Reduce this value to reduce the size of the NTFS tunnel cache. This can significantly improve file deletion performance for directories that contain a large number of files. Note that some applications depend on NTFS tunnel caching.

- **MaxThreadsPerQueue**

  ```
  HKLM\System\CurrentControlSet\Services\LanmanServer\Parameters\
  (REG_DWORD)
  ```

  The default is 20. Increasing this value raises the number of threads that the file server can use to service concurrent requests. When a large number of active connections need to be serviced and hardware resources (such as storage bandwidth) are sufficient, increasing the value can improve server scalability, performance, and response times.

- **RequireSecuritySignature**

  ```
  HKLM\system\CurrentControlSet\Services\LanmanServer\Parameters
  \(REG_DWORD)
  ```

  The default is 0. Changing this value to 1 prevents SMB communication with machines where SMB signing is disabled. In addition, a value of 1 causes SMB signing to be used for all SMB communication. SMB signing can increase CPU cost and network round trips. If SMB signing is not required, ensure that the registry value is 0 on all clients and servers.

- **MaxMpxCt (not applicable with SMB 2 clients)**

  ```
  HKLM\System\CurrentControlSet\Services\LanmanServer
  \Parameters\(REG_DWORD)
  ```

  The default is 50. This parameter suggests a limit on the maximum number of outstanding requests that an SMB 1 client can send. Increasing the value can use more memory, but can improve performance for some client applications by enabling deeper request pipelining. Increasing the value in conjunction with MaxCmds can also eliminate errors encountered due to large numbers of outstanding long-term file requests, such as FindFirstChangeNotification calls. This parameter does not affect connections with SMB 2 clients.

- **NtfsDisableLastAccessUpdate**

  ```
  HKLM\System\CurrentControlSet\Control\FileSystem\(REG_DWORD)
  ```

  The default is 1. In versions of Windows prior to Windows Vista and Windows Server 2008, the default is 0 (do not disable last access). A value of 0 can reduce performance because the system performs additional storage I/O when files and directories are accessed to update date and time information.

The following parameters are *no longer required*:

- **NoAliasingOnFileSystem**

  ```
  HKLM\System\CurrentControlSet\Services\LanmanServer
  \Parameters\(REG_DWORD)
  ```

- **PagedPoolSize**

  ```
  HKLM\System\CurrentControlSet\Control\SessionManager
  \MemoryManagement\(REG_DWORD)
  ```

- **NumTcbTablePartitions**

  ```
  HKLM\system\CurrentControlSet\Services\Tcpip\Parameters\(REG_DWORD)
  ```

- **TcpAckFrequency**

```
HKLM\system\CurrentControlSet\Services\Tcpip\Parameters\Interfaces
```

SMB Server Tuning Example

The following settings can optimize a machine for file server performance in many cases. The settings are not optimal or appropriate on all machines. You should evaluate the impact of individual settings before applying them.

Parameter	Value
NtfsDisable8dot3NameCreation	1
TreatHostAsStableStorage	1
AdditionalCriticalWorkerThreads	64
MaximumTunnelEntries	32
MaxThreadsPerQueue	64
RequireSecuritySignature	0
MaxMpxCt (not applicable with SMB 2 clients)	32768

Services for NFS Model

The following sections provide information about the Microsoft Services for Network File System (NFS) model for client-server communication.

Services for NFS Model Overview

Microsoft Services for NFS provides a file-sharing solution for enterprises that have a mixed Windows and UNIX environment. This communication model consists of two entities: the client and the server (see Figure 7). Applications on the client request files located on the server through the redirector (Rdbss.sys and NFS mini-redirector Nfsrdr.sys). The mini-redirector uses the NFS protocol to send its request through TCP/IP. The server receives multiple requests from the clients through TCP/IP and routes the requests to the local file system (Ntfs.sys), which accesses the storage stack.

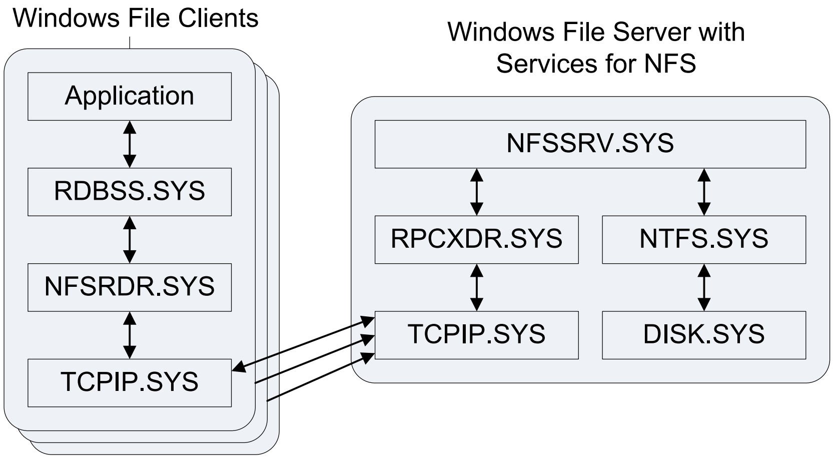


Figure 7. Microsoft Services for Network File System (NFS) Model for Client-Server Communication

Tuning Parameters for NFS Server

The following registry-tuning parameters can affect the performance of NFS file servers:

- **DefaultNumberOfWorkerThreads**

  ```
  HKLM\System\CurrentControlSet\Services\RpcXdr\Parameters\
  (REG_DWORD)
  ```

  Default is 16. Specifies the number of threads that are used to handle incoming NFS requests. Minimum is 1 and maximum is 4096. Determine the appropriate number of threads based on the workload profile. You can increase the number of threads if the server is not responsive.

- **OptimalReads**

  ```
  HKLM\System\CurrentControlSet\Services\NfsServer\Parameters\
  (REG_DWORD)
  ```

  Default is 0. Determines whether files are opened for FILE_RANDOM_ACCESS as opposed to FILE_SEQUENTIAL_ONLY, depending on the workload I/O characteristics. Set this value to 1 to force files to be opened for FILE_RANDOM_ACCESS. FILE_RANDOM_ACCESS prevents the file system and cache manager from performing prefetching. For more information on File Access Services, see "Resources" later in this guide.

- **RdWrHandleLifeTime**

  ```
  HKLM\System\CurrentControlSet\Services\NfsServer\Parameters\
  (REG_DWORD)
  ```

  Default is 5. Controls the lifetime of an NFS cache entry in the file handle cache. This parameter refers to cache entries that have an associated open NTFS file handle. Actual lifetime is approximately equal to RdWrHandleLifeTime multiplied by RdWrThreadSleepTime. Minimum is 1 and maximum is 60.

- **RdWrNfsHandleLifeTime**

  ```
  HKLM\System\CurrentControlSet\Services\NfsServer\Parameters\
  (REG_DWORD)
  ```

  Default is 5. Controls the lifetime of an NFS cache entry in the file handle cache. This parameter refers to cache entries that do not have an associated open NTFS file handle. The Server for NFS uses these cache entries to store file attributes for a file without keeping an open handle with the file system. Actual lifetime is approximately equal to RdWrNfsHandleLifeTime multiplied by RdWrThreadSleepTime. Minimum is 1 and maximum is 60.

- **RdWrNfsReadHandlesLifeTime**

  ```
  HKLM\System\CurrentControlSet\Services\NfsServer\Parameters\
  (REG_DWORD)
  ```

  Default is 5. Controls the lifetime of an NFS read cache entry in the file handle cache. Actual lifetime is approximately equal to RdWrNfsReadHandlesLifeTime multiplied by RdWrThreadSleepTime. Minimum is 1 and maximum is 60.

- **RdWrThreadSleepTime**

  ```
  HKLM\System\CurrentControlSet\Services\NfsServer\Parameters\
  (REG_DWORD)
  ```

Default is 5. Controls the wait interval before running the cleanup thread on the file handle cache. Value is in ticks and is non-deterministic. A tick is equivalent to approximately 100 nanoseconds. Minimum is 1 and maximum is 60.

- **FileHandleCacheSizeinMB**

  ```
  HKLM\System\CurrentControlSet\Services\NfsServer\Parameters\
  (REG_DWORD)
  ```

  Default is 4. Specifies the maximum memory to be consumed by file handle cache entries. Minimum is 1 and maximum is 1*1024*1024*1024 (1073741824).

- **LockFileHandleCacheInMemory**

  ```
  HKLM\System\CurrentControlSet\Services\NfsServer\Parameters\
  (REG_DWORD)
  ```

  Default is 0. Specifies whether the physical pages allocated for the cache size specified by FileHandleCacheSizeInMB are locked in memory. Setting this value to 1 enables this activity. Pages are locked in memory (that is, they are not paged to disk), which improves the performance of resolving file handles but reduces the memory available to applications.

- **MaxIcbNfsReadHandlesCacheSize**

  ```
  HKLM\System\CurrentControlSet\Services\NfsServer\Parameters\
  (REG_DWORD)
  ```

  Default is 64. Specifies the maximum number of handles per volume for the read data cache. Read cache entries are created only on systems that have more than 1 GB of memory. Minimum is 0 and maximum is 0xFFFFFFFF.

- **SecureHandleLevel**

  ```
  HKLM\System\CurrentControlSet\Services\NfsServer\Parameters\
  (REG_DWORD)
  ```

  Default is 6. Controls the strength of the security on handles given out by NFS Server. You can assign the following values in the bitmask:

  - 0x0 – Disable all security checks on the NFS handles.
  - 0x1 – Add checksum to the client for tamper detection.
  - 0x2 – Use the IP address of the client, in addition to other data, to sign the handle.
  - 0x4 – Validate that the parent path of the NTFS field embedded in the handle is exported when the handle is exported.
  - 0x6 –Use the IP Address of the client, in addition to other data, to sign the handle, and also validate that the parent path of the NTFS field embedded in the handle is exported when the handle is exported.

- **RdWrNfsDeferredWritesFlushDelay**

  ```
  HKLM\System\CurrentControlSet\Services\NfsServer\Parameters\
  (REG_DWORD)
  ```

  Default is 60. Soft timeout that controls the duration of NFS V3 UNSTABLE write data caching. Minimum is 1 and maximum is 600. Actual lifetime is approximately equal to RdWrNfsDeferredWritesFlushDelay multiplied by RdWrThreadSleepTime.

- **CacheAddFromCreateAndMkDir**

  ```
  HKLM\System\CurrentControlSet\Services\NfsServer\Parameters\
  (REG_DWORD)
  ```

  Default is 1 (enabled). Controls whether handles opened during NFS V2 and V3 CREATE and MKDIR RPC procedure handlers are retained in the file handle cache. Set this to value 0 to disable adding entries to the cache in CREATE and MKDIR code paths.

- **AdditionalDelayedWorkerThreads**

  ```
  HKLM\SYSTEM\CurrentControlSet\Control\SessionManager\Executive\
  (REG_DWORD)
  ```

  Increases the number of delayed worker threads created for the specified work queue. Delayed worker threads process work items that are not considered time-critical and can have their memory stack paged out while waiting for work items. An insufficient number of threads reduces the rate at which work items are serviced; a value that is too high consumes system resources unnecessarily.

- **NtfsDisable8dot3NameCreation**

  ```
  HKLM\System\CurrentControlSet\Control\FileSystem\ (REG_DWORD)
  ```

  Default is 0. Determines whether NTFS generates a short name in the 8.3 (MS-DOS) naming convention for long file names and for file names that contain characters from the extended character set. If the value of this entry is 0, files can have two names: the name that the user specifies and the short name that NTFS generates. If the name that the user specifies follows the 8.3 naming convention, NTFS does not generate a short name.

  Changing this value does not change the contents of a file, but it avoids the short-name attribute creation for the file and also changes how NTFS displays and manages the file. For most file servers, the recommended setting is 1.

- **NtfsDisableLastAccessUpdate**

  ```
  HKLM\System\CurrentControlSet\Control\FileSystem\(REG_DWORD)
  ```

  Default is 1. This system-global switch reduces disk I/O load and latencies by disabling the updating of the date and time stamp for the last file or directory access.

General Tuning Parameters for Client Computers

The following registry-tuning parameters can affect the performance of client computers that interact with SMB or NFS file servers:

- **DisableBandwidthThrottling**

  ```
  HKLM\system\CurrentControlSet\Services\LanmanWorkstation\Parameters
  \(REG_DWORD)
  ```

  Windows Vista and Windows 7 clients only.

  The default is 0. By default, the SMB redirector throttles throughput across high-latency network connections in some cases to avoid network-related timeouts. Setting this registry value to 1 disables this throttling, enabling higher file transfer throughput over high-latency network connections.

- **DisableLargeMtu**

  ```
  HKLM\system\CurrentControlSet\Services\LanmanWorkstation\Parameters
  \(REG_DWORD)
  ```

  Windows Vista and Windows 7 clients only.

  The default is 1. By default, the SMB redirector does not transfer payloads larger than approximately 64 KB per request. Setting this registry value to 0 enables larger request sizes, which can improve file transfer speed.

- **EnableWsd**

  ```
  HKLM\System\CurrentControlSet\Services\Tcpip\Parameters\(REG_DWORD)
  ```

  Windows Vista and Windows 7 clients only.

  The default is 1 for client operating systems. By default, Windows Scaling Diagnostics (WSD) automatically disables TCP receive window autotuning when heuristics suspect a network switch component might not support the required TCP option (scaling). Setting this registry setting to 0 disables this heuristic and allows autotuning to stay enabled. When no faulty networking devices are involved, applying the setting can enable more reliable high-throughput networking via TCP receive window autotuning. For more information about disabling this setting, see “Resources” later in this guide.

- **RequireSecuritySignature**

  ```
  HKLM\system\CurrentControlSet\Services\LanmanWorkstation\Parameters
  \(REG_DWORD)
  ```

  Windows Vista and Windows 7 clients only.

  The default is 0. Changing this value to 1 prevents SMB communication with machines where SMB signing is disabled. In addition, a value of 1 causes SMB signing to be used for all SMB communication. SMB signing can increase CPU cost and network round trips. If SMB signing is not required, ensure that this registry value is 0 on all clients and servers.

- **FileInfoCacheEntriesMax**

  ```
  HKLM\System\CurrentControlSet\Services\LanmanWorkstation\Parameters
  \(REG_DWORD)
  ```

  Windows Vista and Windows 7 clients only.

  The default is 64 with a valid range of 1 to 65536. This value is used to determine the amount of file metadata that can be cached by the client. Increasing the value can reduce network traffic and increase performance when a large number of files are accessed.

- **DirectoryCacheEntriesMax**

  ```
  HKLM\System\CurrentControlSet\Services\LanmanWorkstation\Parameters
  \(REG_DWORD)
  ```

  Windows Vista and Windows 7 clients only.

  The default is 16 with a valid range of 1 to 4096. This value is used to determine the amount of directory information that can be cached by the client. Increasing the value can reduce network traffic and increase performance when large directories are accessed.

- **FileNotFoundCacheEntriesMax**

  ```
  HKLM\System\CurrentControlSet\Services\LanmanWorkstation\Parameters
  \(REG_DWORD)
  ```

  Windows Vista and Windows 7 clients only.

  The default is 128 with a valid range of 1 to 65536. This value is used to determine the amount of file name information that can be cached by the client. Increasing the value can reduce network traffic and increase performance when a large number of file names are accessed.

- **MaxCmds**

  ```
  HKLM\System\CurrentControlSet\Services\LanmanWorkstation\Parameters
  \(REG_DWORD)
  ```

  Windows Vista and Windows 7 clients only.

  The default is 15. This parameter limits the number of outstanding requests on a session. Increasing the value can use more memory, but can improve performance by enabling deeper request pipelining. Increasing the value in conjunction with MaxMpxCt can also eliminate errors encountered due to large numbers of outstanding long-term file requests, such as FindFirstChangeNotification calls. This parameter does not affect connections with SMB 2 servers.

- **DormantFileLimit**

  ```
  HKLM\system\CurrentControlSet\Services\LanmanWorkstation
  \Parameters\(REG_DWORD)
  ```

  Windows XP client computers only. By default, this registry key is not created.

  This parameter specifies the maximum number of files that should be left open on a share after the application has closed the file.

- **ScavengerTimeLimit**

  ```
  HKLM\system\CurrentControlSet\Services\LanmanWorkstation
  \Parameters\(REG_DWORD)
  ```

  Windows XP client computers only.

  This is the number of seconds that the redirector waits before it starts scavenging dormant file handles (cached file handles that are currently not used by any application).

- **DisableByteRangeLockingOnReadOnlyFiles**

  ```
  HKLM\System\CurrentControlSet\Services\LanmanWorkstation
  \Parameters\(REG_DWORD)
  ```

  Windows XP client computers only.

  Some distributed applications that lock parts of a read-only file as synchronization across clients require that file-handle caching and collapsing behavior be off for all read-only files. This parameter can be set if such applications will not be run on the system and collapsing behavior can be enabled on the client computer.

File Client Tuning Example

The following settings for parameters previously described in the "General Tuning Parameters for Client Computers" section can optimize a machine for accessing remote file shares in many cases, particularly over some high-latency networks. The settings are not optimal or appropriate on all machines. You should evaluate the impact of individual settings before applying them.

Parameter	Value
DisableBandwidthThrottling	1
EnableWsd	0
RequireSecuritySignature	0
FileInfoCacheEntriesMax	32768
DirectoryCacheEntriesMax	4096
FileNotFoundCacheEntriesMax	32768
MaxCmds	32768
DormantFileLimit [Windows XP only]	32768
ScavengerTimeLimit [Windows XP only]	60
DisableByteRangeLockingOnReadOnlyFiles [Windows XP only]	1

Performance Tuning for Active Directory Servers

You can improve the performance of Active Directory®, especially in large environments, by following these tuning steps:

- Increase address space by using 64-bit processors.

  64-bit architecture is preferred for running Active Directory. The large address space makes it possible to equip the server with enough RAM to cache all or most of the Active Directory database in memory. It also provides room for expansion to add RAM if the database size grows. For more information, see the article about Active Directory performance on 64-bit processors in "Resources" later in this guide. Note that Windows Server 2008 R2 is supported only on 64-bit processors.

- Use an appropriate amount of RAM.

  Active Directory uses the server's RAM to cache as much of the directory database as possible. This reduces disk access and improves performance. The Active Directory cache in Windows Server 2008 R2 is permitted to grow. However, it is still limited by the virtual address space and how much physical RAM is on the server.

  To determine whether more RAM is needed for the server, monitor the percentage of Active Directory operations that are being satisfied from the cache by using the Reliability and Performance Monitor. Examine the lsass.exe instance (for Active Directory Domain Services) or Directory instance (for Active Directory Lightweight Directory Services) of the Database\Database Cache % Hit performance counter. A low value indicates that many operations are not being satisfied from the cache. Adding more RAM might improve the cache hit rate and the performance of Active Directory. You should examine the counter after Active Directory has been running for several hours under a typical workload. The cache starts out empty when the Active Directory service is restarted or the machine is rebooted, so the initial hit rate is low.

The use of the Database Cache % Hit counter is the preferred way to assess how much RAM a server needs. Or, a guideline is that when the RAM on a server is twice the physical size of the Active Directory database on disk, it likely gives sufficient room for caching the entire database in memory. However, in many scenarios this is an overestimation because the actual part of the database frequently used is only a fraction of the entire database.

- Use a good disk I/O subsystem.

  Ideally, the server is equipped with sufficient RAM to be able to cache the “hot” parts of the database entirely in memory. However, the on-disk database must still be accessed to initially populate the memory cache, when it accesses uncached parts of the database, and when it writes updates to the directory. Therefore, appropriate selection of storage is also important to Active Directory performance.

  We recommend that the Active Directory database folder be located on a physical volume that is separate from the Active Directory log file folder. In the Active Directory Lightweight Directory Services installation wizard, these are known as data files and data recovery files. Both folders should be on a physical volume that is separate from the operating system volume. The use of drives that support command queuing, especially Serial Attached SCSI or SCSI, might also improve performance.

Considerations for Read-Heavy Scenarios

The typical directory workload consists of more query operations than update operations. Active Directory is optimized for such a workload. To obtain the maximum benefit, the most important performance tuning step is to make sure that the server has sufficient RAM to be able to cache the most frequently used part of the database in memory. Query performance on a freshly rebooted server, or after the Active Directory service has been restarted, might initially be low until the cache is populated. Active Directory automatically populates the cache as queries visit parts of the directory.

Considerations for Write-Heavy Scenarios

Write-heavy scenarios do not benefit as much from the Active Directory cache. To guarantee the transactional durability of data that is written to the directory, Active Directory does not cache disk writes. It commits all writes to the disk before it returns a successful completion status for an operation, unless explicitly requested not to do this. Therefore, fast disk I/O is important to the performance of writes to Active Directory. The following are hardware recommendations that might improve performance for these scenarios:

- Hardware RAID controllers.
- Low-latency/high-RPM disks.
- Battery-backed write caches on the controller.

To determine whether disk I/O is a bottleneck, monitor the Physical Disk\Average Disk Queue Length counter for the volumes on which the Active Directory database and logs are located. A high queue length indicates a large amount of concurrent disk I/O activity. Choosing a storage system to improve write performance on those volumes might improve Active Directory performance.

Using Indexing to Improve Query Performance

Indexing of attributes is useful when you search for objects that have the attribute name in the filter. Indexing can reduce the number of objects that must be visited when you evaluate the filter. However, this reduces the performance of write operations because the index must be updated when the corresponding attribute is modified or added. It also increases the size of your directory database. You can use logging to find the expensive and inefficient queries and consider indexing some attributes that are used in the corresponding queries to improve the search performance. For more information on Active Directory event logging on servers, see "Resources" later in this guide.

Optimizing Trust Paths

Trusts are a way to enable users to authenticate across different forests or domains. If the trust path between the resource and the user is long, then the user might experience high latency because the authentication request must travel through the trust path and return. For example, if a user from the grandchild of a domain tries to log on from a different grandchild in the same forest, the authentication request must travel up the chain from the grandchild to the root and then take the path to the other grandchild. To avoid this, you can create a shortcut trust directly between the two grandchild domains that avoids the long path. However, the administrator must manage trusts. Therefore you must consider how frequently a given trust will be used before you create it. You can create “external trusts” to reduce the trust path when authenticating between inter-forest domains.

Active Directory Performance Counters

You can use several resources to conduct performance diagnosis of a domain controller that is not performing as expected.

You can use the following Reliability and Performance Monitor (Perfmon) counters to track and analyze a domain controller’s performance:

- If you notice slow write or read operations, check the following disk I/O counters under the Physical Disk category to see whether many queued disk operations exist:
  - Avg. Disk Queue Length
  - Avg. Disk Read Queue Length
  - Avg. Disk Write Queue Length
- If lsass.exe uses lots of physical memory, check the following Database counters under the Database category to see how much memory is used to cache the database for Active Directory Domain Services. These counters are located under

the lsass.exe instance, whereas for Active Directory Lightweight Directory Services they are located under the Directory instance:

  - Database Cache % Hit
  - Database Cache Size (MB)

- If lsass.exe uses lots of CPU, check Directory Services\ATQ Outstanding Queued Requests to see how many requests are queued at the domain controller. A high level of queuing indicates that requests are arriving at the domain controller faster than they can be processed. This can also lead to a high latency in responding to requests.

You can also use the Data Collector Sets tool that is included with Windows Server 2008 R2 to see the activity inside the domain controller. On a server on which the Active Directory Domain Services or Active Directory Lightweight Directory Services role has been installed, you can find the collector template in Reliability and Performance Monitor under **Reliability and Performance > Data Collector Sets > System > Active Directory Diagnostics**. To start it, click the **Play** icon.

The tool collects data for five minutes and stores a report under **Reliability and Performance > Reports > System > Active Directory Diagnostics**. This report contains information about CPU usage by different processes, Lightweight Directory Access Protocol (LDAP) operations, Directory Services operations, Kerberos Key Distribution Center operations, NT LAN Manager (NTLM) authentications, Local Security Authority/Security Account Manager (LSA/SAM) operations, and averages of all the important performance counters. This report identifies the workload that is being placed on the domain controller, identifies the contribution of different aspects of that workload to the overall CPU usage, and locates the source of that workload such as an application sending a high rate of requests to the domain controller. The CPU section of the report indicates whether lsass.exe is the process that is taking highest CPU percentage. If any other process is taking more CPU on a domain controller, you should investigate it.

Performance Tuning for Remote Desktop Session Host (formerly Terminal Server)

This section discusses the selection of Remote Desktop Session Host (RD Session Host) hardware, tuning the host, and tuning applications. The following white papers describe the most relevant factors that influence the capacity of a given RD Session Host deployment, methodologies to evaluate capacity for specific deployments, and a set of experimental results for different combinations of usage scenarios and hardware configurations:

- "RD Session Host Capacity Planning in Windows Server 2008 R2"
- "RD Virtualization Host Capacity Planning in Windows Server 2008 R2"

For links to these papers, see "Resources" later in this guide.

Selecting the Proper Hardware for Performance

In an RD Session Host, the choice of hardware is governed by the application set and how the users exercise it. The key factors that affect the number of users and their experience are CPU, memory, disk, and graphics. Earlier in this guide was a discussion on server hardware guidelines. Although these guidelines still apply in this role, this section contains additional guidelines that are specific to RD Session Host Servers, mostly related to the multiuser environment of RD Session Host Servers.

CPU Configuration

CPU configuration is conceptually determined by multiplying the required CPU to support a session by the number of sessions that the system is expected to support, while maintaining a buffer zone to handle temporary spikes. Multiple processors and cores can help reduce abnormal CPU congestion situations, which are usually caused by a few overactive threads that are contained by a similar number of cores. Therefore, the more cores on a system, the lower the cushion margin that must be built into the CPU usage estimate, which results in a larger percentage of active load per CPU. One important factor to remember is that doubling the number of CPUs does not double CPU capacity. For more considerations, see “Choosing and Tuning Server Hardware” earlier in this guide.

Processor Architecture

The 64-bit processor architecture provides a significantly higher kernel virtual address space, which makes it much more suitable for systems that need large amounts of memory. Specifically, the x64 version of the 64-bit architecture is the more workable option for RD Session Host deployments because it provides very small overhead when it runs 32-bit processes. The most significant performance drawback when you migrate to 64-bit architecture is significantly greater memory usage.

Memory Configuration

It is difficult to predict the memory configuration without knowing the applications that users employ. However, the required amount of memory can be estimated by using the following formula:

$$\text{TotalMem} = \text{OSMem} + \text{SessionMem} * \text{NS}$$

OSMem is how much memory the operating system requires to run (such as system binary images, data structures, and so on), SessionMem is how much memory processes running in one session require, and NS is the target number of active sessions. The amount of required memory for a session is mostly determined by the private memory reference set for applications and system processes that are running inside the session. Shared pages (code or data) have little effect because only one copy is present on the system.

One interesting observation is that, assuming the disk system that is backing the pagefile does not change, the larger the number of concurrent active sessions the system plans to support, the bigger the per-session memory allocation must be. If the amount of memory that is allocated per session is not increased, the number of page faults that active sessions generate increases with the number of sessions and eventually overwhelms the I/O subsystem. By increasing the amount of memory that is allocated per session, the probability of incurring page faults decreases, which helps reduce the overall rate of page faults.

Disk

Storage is one of the aspects most often overlooked when you configure an RD Session Host system, and it can be the most common limitation on systems that are deployed in the field.

The disk activity that is generated on a typical RD Session Host system affects the following three areas:

- System files and application binaries
- Pagefiles
- User profiles and user data

Ideally, these three areas should be backed by distinct storage devices. Using striped RAID configurations or other types of high-performance storage further improves performance. We highly recommend that you use storage adapters with battery-backed caches that allow writeback optimizations. Controllers with writeback caches offer improved support for synchronous disk writes. Because all users have a separate hive, synchronous disk writes are significantly more common on an RD Session Host system. Registry hives are periodically saved to disk by using synchronous write operations. To enable these optimizations, from the Disk Management console, open the **Properties** dialog box for the destination disk and, on the **Policies** tab, select the **Enable write caching on the disk** and **Enable advanced performance** check boxes.

For more specific storage tunings, see the guidelines in "Performance Tuning for the Storage Subsystem" earlier in this guide.

Network

Network usage includes two main categories:

- RD Session Host connections traffic in which usage is determined almost exclusively by the drawing patterns exhibited by the applications that are running inside the sessions and the redirected devices I/O traffic.

  For example, applications handling text processing and data input consume bandwidth of approximately 10 to 100 kilobits per second, whereas rich graphics and video playback cause significant increases in bandwidth usage. We do not recommend video playback over RD Session Host connections because desktop remoting is not optimized to support the high frame rate rendering that is associated with video playback. Frequent use of device redirection features such as file, clipboard, printer, or audio redirection also significantly increases network traffic. Generally, a single 1-gigabit adapter is satisfactory for most systems.

- Back-end connections such as roaming profiles, application access to file shares, database servers, e-mail servers, and HTTP servers.

  The volume and profile of network traffic is specific to each deployment.

Tuning Applications for Remote Desktop Session Host

Most of the CPU usage on an RD Session Host system is driven by applications. Desktop applications are usually optimized toward responsiveness with the goal of minimizing how long it takes an application to respond to a user request. However, in a server environment it is equally important to minimize the total amount of CPU that is used to complete an action to avoid adversely affecting other sessions.

Consider the following suggestions when you configure applications to be used on an RD Session Host system:

- Minimize background/Idle loop processing.

  Typical examples are disabling background grammar/spell checking, data indexing for search, and background saves.

- Minimize how often an application polls to do a state check or update.

  Disabling such behaviors or increasing the interval between polling iterations and timer firing significantly benefits CPU usage because the CPU effect of such activities is quickly amplified for many active sessions. Typical examples are connection status icons and status bar information updates.

- Minimize resource contention between applications by reducing their synchronization frequency with that resource.

  Examples of such resources include registry keys and configuration files. Examples of such application components and features are status indicator (like shell notifications), background indexing or change monitoring, and offline synchronization.

- Disable unnecessary processes that are registered to be started at user logon or session startup.

  These processes can significantly contribute to the CPU cost of creating a new session for the user, which generally is a CPU-intensive process and can be very expensive in morning scenarios. Use MsConfig.exe or MsInfo32.exe to obtain a list of processes that are started at user logon.

- When possible, avoid multimedia application components for RD Session Host deployments.

  Video playback causes high bandwidth usage for the RD Session Host connection, and audio playback causes high bandwidth usage on the audio redirection channel. Also, multimedia processing (encoding and decoding, mixing, and so on) has a significant CPU usage cost.

For memory consumption, consider the following suggestions:

- Verify that DLLs that applications load are not relocated at load.

  If DLLs are relocated, it is impossible to share their code across sessions, which significantly increases the footprint of a session. This is one of the most common memory-related performance problems in RD Session Host.

- For common language runtime (CLR) applications, use Native Image Generator (Ngen.exe) to increase page sharing and reduce CPU overhead.

  When possible, apply similar techniques to other similar execution engines.

Remote Desktop Session Host Tuning Parameters

Pagefile

Insufficient pagefile size can cause memory allocation failures either in applications or system components. A general guideline is that the combined size of the pagefiles should be two to three times larger than the physical memory size. You can use the Memory\Committed Bytes performance counter to monitor how much committed virtual memory is on the system. When the value of this counter reaches close to the total combined size of physical memory and pagefiles, memory allocation begins to fail. Because of significant disk I/O activity that pagefile access generates, consider using a dedicated storage device for the pagefile, ideally a high-performance one such as a striped RAID array.

For more specific storage tuning guidelines, see "Performance Tuning for the Storage Subsystem" earlier in this guide.

Antivirus and Antispyware

Installing antivirus and antispyware software on an RD Session Host server greatly affects overall system performance, especially CPU usage. We highly recommend that you exclude from the active monitoring list all the folders that hold temporary files, especially those that services and other system components generate.

Task Scheduler

Task Scheduler (which can be accessed under **All Programs** > **Accessories** > **System Tools**) lets you examine the list of tasks that are scheduled for different events. For RD Session Host, it is useful to focus specifically on the tasks that are configured to run on idle, at user logon, or on session connect and disconnect. Because of the specifics of the deployment, many of these tasks might be unnecessary.

Desktop Notification Icons

Notification icons on the desktop can have fairly expensive refreshing mechanisms. You can use **Customize Notifications Icons** to examine the list of notifications that are available in the system. Generally, it is best to disable unnecessary notifications by either removing the component that registers them from the startup list or by changing the configuration on applications and system components to disable them.

You can implement the following tuning parameters by opening the MMC snap-in for Group Policy (Gpedit.smc) and making the respective changes under **Computer Configuration** > **Administrative Templates** > **Windows Components** > **Remote Desktop Services**:

- Color depth.

  Color depth can be adjusted under **Remote Session Environment** > **Limit Maximum Color Depth** with possible values of 8, 15, 16, and 32 bit. The default value is 16 bit, and increasing the bit depth increases memory and bandwidth consumption. Or, the color depth can be adjusted from TSConfig.exe by opening the **Properties** dialog box for a specific connection and, on the **Client**

Setting tab, changing the selected value in the drop-down box under **Color Depth**. The **Limit Maximum Color Depth** check box must be selected.

- Remote Desktop Protocol compression.

  Remote Desktop Protocol (RDP) compression can be configured under **Remote Session Environment** > **Set compression algorithm for RDP data**. Three values are possible:

  - **Optimized to use less memory** is the configuration that matches the default Windows Server 2003 configuration. This uses the least amount of memory per session but has the lowest compression ratio and therefore the highest bandwidth consumption.
  - **Balances memory and network bandwidth** is the default setting for Windows Server 2008 R2. This has reduced bandwidth consumption while marginally increasing memory consumption (approximately 200 KB per session).
  - **Optimized to use less network bandwidth** further reduces network bandwidth usage at a cost of approximately 2 MB per session. This memory is allocated in the kernel virtual address space and can have a significant effect on 32-bit processor–based systems that are running a fairly large number of users. Because 64-bit systems do not have these issues, this setting is recommended if the additional memory cost is considered acceptable. If you want to use this setting, you should assess the maximum number of sessions and test to that level with this setting before placing a server in production.

- Device redirection.

  Device redirection can be configured under **Device and Resource Redirection**. Or, it can be configured through Remote Desktop Session Host Configuration by opening the properties for a specific connection such as **RDP-Tcp** and, on the **Client Settings** tab, changing **Redirection** settings.

  Generally, device redirection increases how much network bandwidth RD Session Host connections use because data is exchanged between devices on the client machines and processes that are running in the server session. The extent of the increase is a function of the frequency of operations that are performed by the applications that are running on the server against the redirected devices.

  Printer redirection and Plug and Play device redirection also increase logon CPU usage. You can redirect printers in two ways:

  - Matching printer driver-based redirection when a driver for the printer must be installed on the server. Earlier releases of Windows Server used this method.
  - Easy Print printer driver redirection, introduced in Windows Server 2008, uses a common printer driver for all printers.

  We recommend the Easy Print method because it causes less CPU usage for printer installation at connection time. The matching driver method causes increased CPU usage because it requires the spooler service to load different drivers. For bandwidth usage, the Easy Print method causes slightly increased network bandwidth usage, but not significant enough to offset the other performance, manageability, and reliability benefits.

Audio redirection is disabled by default because using it causes a steady stream of network traffic. Audio redirection also enables users to run multimedia applications that typically have high CPU consumption.

Client Experience Settings

The Remote Desktop Connection (RDC) Client provides control over a range of settings that influence network bandwidth performance for the Remote Desktop Services (RDS) connection. You can access them either through the RDC Client user interface on the **Experience** tab or as settings in the RDP file:

- **Disable wallpaper** (RDP file setting: disable wallpaper:i:0) suppresses the display of desktop wallpaper on redirected connections. It can significantly reduce bandwidth usage if desktop wallpaper consists of an image or other content with significant drawing cost.
- **Font smoothing** (RDP file setting: allow font smoothing:i:0) controls ClearType font rendering support. Although this improves the rendering quality for fonts when it is enabled, it does affect network bandwidth consumption significantly.
- **Desktop composition** is supported only for a remote session to Windows Vista and has no relevance for server systems.
- **Show contents of windows while dragging** (RDP file setting: disable full window drag:i:1), when it is disabled, reduces bandwidth by displaying only the window frame instead of all the contents when dragged.
- **Menu and window animation** (represented by two distinct RDP file settings: disable menu anims:i:1 and disable cursor setting:i:1), when it is disabled, reduces bandwidth by disabling animation on menus (such as fading) and cursors.
- **Visual styles** (RDP file setting: disable themes:i:1), when it is disabled, reduces bandwidth by simplifying theme drawings that use the classic theme.
- **Bitmap cache** (RDP file setting: bitmapcachepersistenable:i:1), when it is enabled, creates a client-side cache of bitmaps that are rendered in the session. It is a significant improvement on bandwidth usage and should always be enabled (except for security considerations).

Using the **Experience** tab within the Remote Desktop Connection Client you can choose your connection speed to influence network bandwidth performance. The following list indicates which options are chosen if you change the connection speed within the **Experience** tab of the Remote Desktop Connection Client:

- **Modem** (56 Kbps)
  - Persistent bitmap caching
- **Low Speed Broadband** (256 Kbps - 2 Mbps)
  - Persistent bitmap caching, visual styles
- **Cellular/Satellite** (2Mbps - 16 MBps)
  - Desktop composition; persistent bitmap caching; visual styles; desktop background

- **High Speed Broadband** (2 Mbps – 10 Mbps)
    - Desktop composition; show contents of windows while dragging; menu and window animation; persistent bitmap caching; visual styles; desktop background
- **WAN** (10 Mbps or higher with high latency)
    - Desktop composition; show contents of windows while dragging; menu and window animation; persistent bitmap caching; visual styles; desktop background (all)
- **LAN** (10 Mbps or higher)
    - Desktop composition; show contents of windows while dragging; menu and window animation; persistent bitmap caching; themes; desktop background

When the RDP connection profile is saved, it creates an *xxx.rdp* file where *xxx* is the friendly name chosen by the user (the default is *default.rdp*). The speed optimization settings in the *xxx.rdp* configuration file are attributed as follows:

- Modem=1
- LowSpeedBroadband=2
- Cellular with latency=3
- HighSpeedBroadband=4
- WAN with latency=5
- LAN=6

Desktop Size

Desktop size for remote sessions can be controlled either through the RDC client user interface (on the **Display** tab under **Remote desktop size** settings) or the RDP file (desktopwidth:i:1152 and desktopheight:i:864). The larger the desktop size, the greater the memory and bandwidth consumption that is associated with that session. The current maximum desktop size that a server accepts is 4096 x 2048.

Windows System Resource Manager

Windows System Resource Manager (WSRM) is an optional component that is available in Windows Server 2008 R2. WSRM supports an “equal per session” built-in policy that keeps CPU usage equally distributed among all active sessions on the system. Although enabling WSRM adds some CPU usage overhead to the system, we recommend that you enable it because it helps limit the effect that high CPU usage in one session has on the other sessions on the system. This helps improve user experience and also lets you run more users on the system because of a reduced need for a large cushion in CPU capacity to accommodate random CPU usage spikes.

Performance Tuning for Remote Desktop Gateway

This section describes the performance-related parameters that help improve the performance of a customer deployment and the tunings that rely on their network usage patterns. At its core, the Remote Desktop Gateway (RD Gateway) performs many packet forwarding operations between the Remote Desktop Connection Client instances and the RD Session Host Server instances within the customer's network. IIS and RD Gateway export the following registry parameters to help improve system performance in the RD Gateway role:

- Thread tunings
  - **MaxIoThreads**

    ```
    HKLM\Software\Microsoft\Terminal Server Gateway\ (REG_DWORD)
    ```

    The default value is 5. It specifies the number of threads that the RDGateway service creates to handle incoming requests.

  - **MaxPoolThreads**

    ```
    HKLM\System\CurrentControlSet\Services\InetInfo\Parameters
    \(REG_DWORD)
    ```

    The default value is 4. It specifies the number of Internet Information Services (IIS) pool threads to create per processor. The IIS pool threads watch the network for requests and process all incoming requests. The **MaxPoolThreads** count does not include threads that the RD Gateway service consumes.

- Remote procedure call tunings for RD Gateways

  The following parameters can help tune the remote procedure call (RPC) receive windows on the RDC Client and RD Gateway machines. Changing the windows helps throttle how much data is flowing through each connection and can improve performance for RPC over HTTP v2 scenarios:

  - **ServerReceiveWindow**

    ```
    HKLM\Software\Microsoft\Rpc\ (REG_DWORD)
    ```

    The default value is 64 KB. This value specifies the receive window that the server uses for data that is received from the RPC proxy. The minimum value is set to 8 KB, and the maximum value is set at 1 GB. If the value is not present, then the default value is used. When changes are made to this value, IIS must be restarted for the change to take effect.

  - **ClientReceiveWindow**

    ```
    HKLM\Software\Microsoft\Rpc\ (REG_DWORD)
    ```

    The default value is 64 KB. This value specifies the receive window that the client uses for data that is received from the RPC proxy. The minimum valid value is 8 KB, and the maximum value is 1 GB. If the value is not present, then the default value is used.

Monitoring and Data Collection

The following list of performance counters is considered a base set of counters when you monitor the resource usage on the RD Gateway:

\Terminal Service Gateway*
\RPC/HTTP Proxy*
\RPC/HTTP Proxy Per Server*
\Web Service*
\W3SVC_W3WP*
\IPv4*
\Memory*
\Network Interface(*)*
\Process(*)*
\Processor Information(*)*
\Synchronization(*)*
\System*
\TCPv4*

Note: If applicable, add the “\IPv6*” and “\TCPv6*” objects.

Performance Tuning for Virtualization Servers

Hyper-V™ is the virtualization server role in Windows Server 2008 R2. Virtualization servers can host multiple virtual machines (VMs) that are isolated from each other but share the underlying hardware resources by virtualizing the processors, memory, and I/O devices. By consolidating servers onto a single machine, virtualization can improve resource usage and energy efficiency and reduce the operational and maintenance costs of servers. In addition, VMs and the management APIs offer more flexibility for managing resources, balancing load, and provisioning systems.

The following sections define the virtualization terminology that is used in this guide and suggest best practices that yield increased performance on Hyper-V servers.

Terminology

This section summarizes key terminology specific to VM technology that is used throughout this performance tuning guide:

child partition
Any partition (VM) that is created by the root partition.

device virtualization
A mechanism that lets a hardware resource be abstracted and shared among multiple consumers.

emulated device
A virtualized device that mimics an actual physical hardware device so that guests can use the typical drivers for that hardware device.

enlightenment
An optimization to a guest operating system to make it aware of VM environments and tune its behavior for VMs.

guest

Software that is running in a partition. It can be a full-featured operating system or a small, special-purpose kernel. The hypervisor is “guest-agnostic.”

hypervisor

A layer of software that sits just above the hardware and below one or more operating systems. Its primary job is to provide isolated execution environments called partitions. Each partition has its own set of hardware resources (CPU, memory, and devices). The hypervisor is responsible for controls and arbitrates access to the underlying hardware.

logical processor (LP)

A processing unit that handles one thread of execution (instruction stream). There can be one or more logical processors per core and one or more cores per processor socket.

passthrough disk access

A representation of an entire physical disk as a virtual disk within the guest. The data and commands are “passed through” to the physical disk (through the root partition’s native storage stack) with no intervening processing by the virtual stack.

root partition

A partition that is created first and owns all the resources that the hypervisor does not, including most devices and system memory. It hosts the virtualization stack and creates and manages the child partitions.

synthetic device

A virtualized device with no physical hardware analog so that guests might need a driver (virtualization service client) to that synthetic device. The driver can use VMBus to communicate with the virtualized device software in the root partition.

virtual machine (VM)

A virtual computer that was created by software emulation and has the same characteristics as a real computer.

virtual processor (VP)

A virtual abstraction of a processor that is scheduled to run on a logical processor. A VM can have one or more virtual processors.

virtualization service client (VSC)

A software module that a guest loads to consume a resource or service. For I/O devices, the virtualization service client can be a device driver that the operating system kernel loads.

virtualization service provider (VSP)

A provider exposed by the virtualization stack in the root partition that provides resources or services such as I/O to a child partition.

virtualization stack

A collection of software components in the root partition that work together to support VMs. The virtualization stack works with and sits above the hypervisor. It also provides management capabilities.

Hyper-V Architecture

Hyper-V features a hypervisor-based architecture that is shown in Figure 7. The hypervisor virtualizes processors and memory and provides mechanisms for the virtualization stack in the root partition to manage child partitions (VMs) and expose services such as I/O devices to the VMs. The root partition owns and has direct access to the physical I/O devices. The virtualization stack in the root partition provides a memory manager for VMs, management APIs, and virtualized I/O devices. It also implements emulated devices such as Integrated Device Electronics (IDE) and PS/2 but supports synthetic devices for increased performance and reduced overhead.

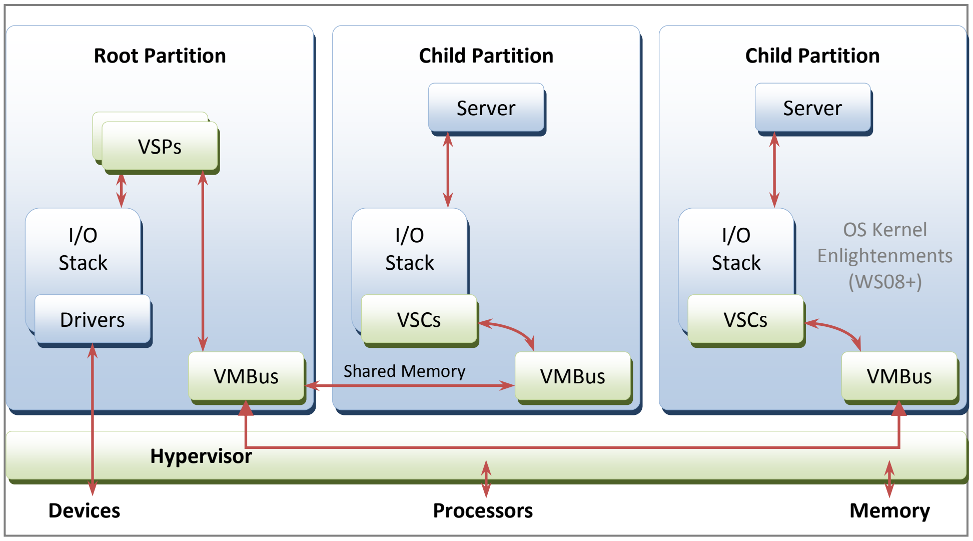


Figure 7. Hyper-V Hypervisor-Based Architecture

The synthetic I/O architecture consists of VSPs in the root partition and VSCs in the child partition. Each service is exposed as a device over VMBus, which acts as an I/O bus and enables high-performance communication between VMs that use mechanisms such as shared memory. Plug and Play enumerates these devices, including VMBus, and loads the appropriate device drivers (VSCs). Services other than I/O are also exposed through this architecture.

Windows Server 2008 and Windows Server 2008 R2 feature enlightenments to the operating system to optimize its behavior when it is running in VMs. The benefits include reducing the cost of memory virtualization, improving multiprocessor scalability, and decreasing the background CPU usage of the guest operating system.

Server Configuration

This section describes best practices for selecting hardware for virtualization servers and installing and setting up Windows Server 2008 R2 for the Hyper-V server role.

Hardware Selection

The hardware considerations for Hyper-V servers generally resemble those of non-virtualized servers, but Hyper-V servers can exhibit increased CPU usage, consume more memory, and need larger I/O bandwidth because of server consolidation. For more information, refer to "Choosing and Tuning Server Hardware" earlier in this guide.

- Processors.

  Hyper-V in Windows Server 2008 R2 presents the logical processors as one or more virtual processors to each active virtual machine. You can achieve additional run-time efficiency by using processors that support Second Level Address Translation (SLAT) technologies such as EPT or NPT.

  Hyper-V in Windows Server 2008 R2 adds support for deep CPU idle states, timer coalescing, core parking, and guest idle state. These features allow for better energy efficiency over previous versions of Hyper-V.

- Cache.

  Hyper-V can benefit from larger processor caches, especially for loads that have a large working set in memory and in VM configurations in which the ratio of virtual processors to logical processors is high.

- Memory.

  The physical server requires sufficient memory for the root and child partitions. Hyper-V first allocates the memory for child partitions, which should be sized based on the needs of the expected load for each VM. Having additional memory available allows the root to efficiently perform I/Os on behalf of the VMs and operations such as a VM snapshot.

- Networking.

  If the expected loads are network intensive, the virtualization server can benefit from having multiple network adapters or multiport network adapters. Each network adapter is assigned to its own virtual switch, allowing each virtual switch to service a subset of virtual machines. When hosting multiple VMs, using multiple network adapters allows for distribution of the network traffic among the adapters for better overall performance.

  To reduce the CPU usage of network I/Os from VMs, Hyper-V can use hardware offloads such as Large Send Offload (LSOv1), TCPv4 checksum offload, Chimney, and VMQ.

  For details on network hardware considerations, see "Performance Tuning for the Networking Subsystem" earlier in this guide.

- Storage.

  The storage hardware should have sufficient I/O bandwidth and capacity to meet current and future needs of the VMs that the physical server hosts. Consider these requirements when you select storage controllers and disks and choose the RAID configuration. Placing VMs with highly disk-intensive workloads on different physical disks will likely improve overall performance. For example, if four VMs share a single disk and actively use it, each VM can yield only 25 percent of the bandwidth of that disk. For details on storage hardware considerations and discussion on sizing and RAID selection, see “Performance Tuning for the Storage Subsystem” earlier in this guide.

Server Core Installation Option

Windows Server 2008 and Windows Server 2008 R2 feature the Server Core installation option. Server Core offers a minimal environment for hosting a select set of server roles including Hyper-V. It features a smaller disk, memory profile, and attack surface. Therefore, we highly recommend that Hyper-V virtualization servers use the Server Core installation option. Using Server Core in the root partition leaves additional memory for the VMs to use (approximately 80 MB for commit charge on 64-bit Windows).

Server Core offers a console window only when the user is logged on, but Hyper-V exposes management features through WMI so administrators can manage it remotely (for more information, see "Resources" later in this guide).

Dedicated Server Role

The root partition should be dedicated to the virtualization server role. Additional server roles can adversely affect the performance of the virtualization server, especially if they consume significant CPU, memory, or I/O bandwidth. Minimizing the server roles in the root partition has additional benefits such as reducing the attack surface and the frequency of updates.

System administrators should consider carefully what software is installed in the root partition because some software can adversely affect the overall performance of the virtualization server.

Guest Operating Systems

Hyper-V supports and has been tuned for a number of different guest operating systems. The number of virtual processors that are supported per guest depends on the guest operating system. See “Resources” later in this guide or the documentation provided with Hyper-V for a list of the supported guest operating systems and the number of virtual processors supported for each operating system.

CPU Statistics

Hyper-V publishes performance counters to help characterize the behavior of the virtualization server and break out the resource usage. The standard set of tools for viewing performance counters in Windows includes Performance Monitor (Perfmon.exe) and Logman.exe, which can display and log the Hyper-V performance counters. The names of the relevant counter objects are prefixed with “Hyper-V.”

You should always measure the CPU usage of the physical system by using the Hyper-V Hypervisor Logical Processor performance counters. The CPU utilization counters that Task Manager and Performance Monitor report in the root and child partitions do not accurately capture the CPU usage. Use the following performance counters to monitor performance:

- \Hyper-V Hypervisor Logical Processor (*) \% Total Run Time

The counter represents the total non-idle time of the logical processor(s).

- \Hyper-V Hypervisor Logical Processor (*) \% Guest Run Time

  The counter represents the time spent executing cycles within a guest or within the host.

- \Hyper-V Hypervisor Logical Processor (*) \% Hypervisor Run Time

  The counter represents the time spent executing within the hypervisor.

- \Hyper-V Hypervisor Root Virtual Processor (*) \ *

  The counters measure the CPU usage of the root partition.

- \Hyper-V Hypervisor Virtual Processor (*) \ *

  The counters measure the CPU usage of guest partitions.

Processor Performance

The hypervisor virtualizes the physical processors by time-slicing between the virtual processors. To perform the required emulation, certain instructions and operations require the hypervisor and virtualization stack to run. Moving a workload into a VM increases the CPU usage, but this guide describes best practices for minimizing that overhead.

VM Integration Services

The VM Integration Services include enlightened drivers for the synthetic I/O devices, which significantly reduces CPU overhead for I/O compared to emulated devices. You should install the latest version of the VM Integration Services in every supported guest. The services decrease the CPU usage of the guests, from idle guests to heavily used guests, and improves the I/O throughput. This is the first step in tuning a Hyper-V server for performance. For the list of supported guest operating systems, see the documentation that is provided with the Hyper-V installation.

Enlightened Guests

The operating system kernels in Windows Vista SP1, Windows 7, Windows Server 2008, and Windows Server 2008 R2 feature enlightenments that optimize their operation for VMs. For best performance, we recommend that you use Windows Server 2008 R2 or Windows Server 2008 as a guest operating system. The enlightenments present in Windows Server 2008 R2 and Windows Server 2008 decrease the CPU overhead of Windows that runs in a VM. The integration services provide additional enlightenments for I/O. Depending on the server load, it can be appropriate to host a server application in a Windows Server guest for better performance.

Virtual Processors

Hyper-V in Windows Server 2008 R2 supports a maximum of four virtual processors per VM. VMs that have loads that are not CPU intensive should be configured to use one virtual processor. This is because of the additional overhead that is associated with multiple virtual processors, such as additional synchronization costs in the guest operating system. More CPU-intensive loads should be placed in 2-VP to 4-VP VMs if the VM requires more than one CPU of processing under peak load. The documentation that is provided with the Hyper-V installation lists the supported guest operating systems and the number of virtual processors supported for each operating system.

Windows Server 2008 R2 features enlightenments to the core operating system that improve scalability in multiprocessor VMs. Workloads can benefit from the scalability improvements in Windows Server 2008 R2 if they run in 2-VP to 4-VP VMs.

Background Activity

Minimizing the background activity in idle VMs releases CPU cycles that can be used elsewhere by other VMs or saved to reduce energy consumption. Windows guests typically use less than 1 percent of one CPU when they are idle. The following are several best practices for minimizing the background CPU usage of a VM:

- Install the latest version of the VM Integration Services.
- Remove the emulated network adapter through the VM settings dialog box (use the Microsoft synthetic adapter).
- Remove unused devices such as the CD-ROM and COM port, or disconnect their media.
- Keep the Windows guest at the logon screen when it is not being used.
- Use Windows Server 2008 or Windows Server 2008 R2 for the guest operating system.
- Disable the screen saver.
- Disable, throttle, or stagger periodic activity such as backup and defragmentation.
- Review the scheduled tasks and services that are enabled by default.
- Improve server applications to reduce periodic activity (such as timers).
- Use the Balanced power plan instead of the High Performance power plan.

The following are additional best practices for configuring a *client version* of Windows in a VM to reduce the overall CPU usage:

- Disable background services such as SuperFetch and Windows Search.
- Disable scheduled tasks such as Scheduled Defrag.
- Disable AeroGlass and other user interface effects (through the System application in Control Panel).

Weights and Reserves

Hyper-V supports setting the weight of a virtual processor to grant it a larger or smaller share of CPU cycles than average and specifying the reserve of a virtual processor to make sure that it gets a minimal percentage of CPU cycles. The CPU that a virtual processor consumes can also be limited by specifying usage limits. System administrators can use these features to prioritize specific VMs, but we recommend the default values unless you have a compelling reason to alter them.

Weights and reserves prioritize or de-prioritize specific VMs if CPU resources are overcommitted. This makes sure that those VMs receive a larger or smaller share of the CPU. Highly intensive loads can benefit from adding more virtual processors instead, especially when they are close to saturating an entire physical CPU.

Tuning NUMA Node Preference

On Non-Uniform Memory Access (NUMA) hardware, each VM has a default NUMA node preference. Hyper-V uses this NUMA node preference when assigning physical memory to the VM and when scheduling the VM's virtual processors. A VM performs optimally when its virtual processors and memory are on the same NUMA node.

By default, the system assigns the VM to its preferred NUMA node every time the VM is run. An imbalance of NUMA node assignments might occur depending on the memory requirements of each VM and the order in which each VM is started. This can lead to a disproportionate number of VMs being assigned to a single NUMA node.

Use Perfmon to check the NUMA node preference setting for each running VM by examining the \Hyper-V VM Vid Partition (*)\ NumaNodeIndex counter.

You can change NUMA node preference assignments by using the Hyper-V WMI API. For information on the WMI calls available for Hyper-V and for a blog post on NUMA node balancing, see "Resources" later in this guide. To set the NUMA node preference for a VM, set the NumaNodeList property of the Msvm_VirtualSystemSettingData class.

Memory Performance

The hypervisor virtualizes the guest physical memory to isolate VMs from each other and provide a contiguous, zero-based memory space for each guest operating system. In general, memory virtualization can increase the CPU cost of accessing memory. On non-SLAT-based hardware, frequent modification of the virtual address space in the guest operating system can significantly increase the cost.

Enlightened Guests

Windows Server 2008 R2 and Windows Server 2008 include kernel enlightenments and optimizations to the memory manager to reduce the CPU overhead from Hyper-V memory virtualization. Workloads that have a large working set in memory can benefit from using Windows Server 2008 R2 or Windows Server 2008 as a guest. These enlightenments reduce the CPU cost of context switching between processes and accessing memory. Additionally, they improve the multiprocessor (MP) scalability of Windows Server guests.

Correct Memory Sizing for Child Partitions

You should size VM memory as you typically do for server applications on a physical machine. You must size it to reasonably handle the expected load at ordinary and peak times because insufficient memory can significantly increase response times and CPU or I/O usage.

You can enable Dynamic Memory to allow Windows to size VM memory dynamically. The recommended initial memory size for Windows Server 2008 R2 guests is at least

512 MB. With Dynamic Memory, if applications in the VM experience launching problems, you can increase the pagefile size for the VM. To increase the VM pagefile size, navigate to **Control Panel > System > Advanced System Settings > Advanced.** From this tab, navigate to **Performance Settings > Advanced > Virtual memory**. For the **Custom size** selection, configure the **Initial Size** to the amount of memory assigned to the VM by Hyper-V Dynamic Memory when VM reaches its steady state, and set the **Maximum Size** to three times the **Initial Size.** For more information about Dynamic Memory configuration, see "Resources" later in this guide.

When running Windows in the child partition, you can use the following performance counters within a child partition to identify whether the child partition is experiencing memory pressure and is likely to perform better with a higher VM memory size:

Performance counter	Suggested threshold value
Memory – Standby Cache Reserve Bytes	Sum of Standby Cache Reserve Bytes and Free and Zero Page List Bytes should be 200 MB or more on systems with 1 GB, and 300 MB or more on systems with 2 GB or more of visible RAM.
Memory – Free & Zero Page List Bytes	Sum of Standby Cache Reserve Bytes and Free and Zero Page List Bytes should be 200 MB or more on systems with 1 GB, and 300 MB or more on systems with 2 GB or more of visible RAM.
Memory – Pages Input/Sec	Average over a 1-hour period is less than 10.

Correct Memory Sizing for Root Partition

The root partition must have sufficient memory to provide services such as I/O virtualization, snapshot, and management to support the child partitions. The root partition should have at least 512 MB available. When Dynamic Memory is enabled, the root reserve is calculated automatically based on root physical memory and NUMA architecture. This logic applies for supported scenarios with no applications running in the root.

A good standard for the memory overhead of each VM is 32 MB for the first 1 GB of virtual RAM plus another 8 MB for each additional GB of virtual RAM. This should be factored in the calculations of how many VMs to host on a physical server. The memory overhead varies depending on the actual load and amount of memory that is assigned to each VM.

Storage I/O Performance

Hyper-V supports synthetic and emulated storage devices in VMs, but the synthetic devices generally can offer significantly better throughput and response times and reduced CPU overhead. The exception is if a filter driver can be loaded and reroutes I/Os to the synthetic storage device. Virtual hard disks (VHDs) can be backed by three types of VHD files or raw disks. This section describes the different options and considerations for tuning storage I/O performance.

For more information, refer to "Performance Tuning for the Storage Subsystem" earlier in this guide, which discusses considerations for selecting and configuring storage hardware.

Synthetic SCSI Controller

The synthetic storage controller provides significantly better performance on storage I/Os with less CPU overhead than the emulated IDE device. The VM Integration Services include the enlightened driver for this storage device and are required for the guest operating system to detect it. The operating system disk must be mounted on the IDE device for the operating system to boot correctly, but the VM integration services load a filter driver that reroutes IDE device I/Os to the synthetic storage device.

We strongly recommend that you mount the data drives directly to the synthetic SCSI controller because that configuration has reduced CPU overhead. You should also mount log files and the operating system paging file directly to the synthetic SCSI controller if their expected I/O rate is high.

For highly intensive storage I/O workloads that span multiple data drives, each VHD should be attached to a separate synthetic SCSI controller for better overall performance. In addition, each VHD should be stored on separate physical disks.

Virtual Hard Disk Types

There are three types of VHD files. We recommend that production servers use fixed-sized VHD files for better performance and also to make sure that the virtualization server has sufficient disk space for expanding the VHD file at run time. The following are the performance characteristics and trade-offs between the three VHD types:

- Dynamically expanding VHD.

  Space for the VHD is allocated on demand. The blocks in the disk start as zeroed blocks but are not backed by any actual space in the file. Reads from such blocks return a block of zeros. When a block is first written to, the virtualization stack must allocate space within the VHD file for the block and then update the metadata. This increases the number of necessary disk I/Os for the write and increases CPU usage. Reads and writes to existing blocks incur both disk access and CPU overhead when looking up the blocks' mapping in the metadata.

- Fixed-size VHD.

  Space for the VHD is first allocated when the VHD file is created. This type of VHD is less apt to fragment, which reduces the I/O throughput when a single I/O is split into multiple I/Os. It has the lowest CPU overhead of the three VHD types because reads and writes do not need to look up the mapping of the block.

- Differencing VHD.

  The VHD points to a parent VHD file. Any writes to blocks never written to before result in space being allocated in the VHD file, as with a dynamically expanding VHD. Reads are serviced from the VHD file if the block has been written to. Otherwise, they are serviced from the parent VHD file. In both cases, the metadata is read to determine the mapping of the block. Reads and writes to this VHD can consume more CPU and result in more I/Os than a fixed-sized VHD.

Snapshots of a VM create a differencing VHD to store the writes to the disks since the snapshot was taken. Having only a few snapshots can elevate the CPU usage of storage I/Os, but might not noticeably affect performance except in highly I/O-intensive server workloads.

However, having a large chain of snapshots can noticeably affect performance because reading from the VHD can require checking for the requested blocks in many differencing VHDs. Keeping snapshot chains short is important for maintaining good disk I/O performance.

Passthrough Disks

The VHD in a VM can be mapped directly to a physical disk or logical unit number (LUN), instead of a VHD file. The benefit is that this configuration bypasses the file system (NTFS) in the root partition, which reduces the CPU usage of storage I/O. The risk is that physical disks or LUNs can be more difficult to move between machines than VHD files.

Large data drives can be prime candidates for passthrough disks, especially if they are I/O intensive. VMs that can be migrated between virtualization servers (such as quick migration) must also use drives that reside on a LUN of a shared storage device.

Disabling File Last Access Time Check

Windows Server 2003 and earlier Windows operating systems update the last-accessed time of a file when applications open, read, or write to the file. This increases the number of disk I/Os, which further increases the CPU overhead of virtualization. If applications do not use the last-accessed time on a server, system administrators should consider setting this registry key to disable these updates.

NTFSDisableLastAccessUpdate

```
HKLM\System\CurrentControlSet\Control\FileSystem\ (REG_DWORD)
```

By default, Windows Server 2008 R2 disables the last-access time updates.

Physical Disk Topology

VHDs that I/O-intensive VMs use generally should not be placed on the same physical disks because this can cause the disks to become a bottleneck. If possible, they should also not be placed on the same physical disks that the root partition uses. For a discussion on capacity planning for storage hardware and RAID selection, see "Performance Tuning for the Storage Subsystem" earlier in this guide.

I/O Balancer Controls

The virtualization stack balances storage I/O streams from different VMs so that each VM has similar I/O response times when the system's I/O bandwidth is saturated. The following registry keys can be used to adjust the balancing algorithm, but the virtualization stack tries to fully use the I/O device's throughput while providing reasonable balance. The first path should be used for storage scenarios, and the second path should be used for networking scenarios:

```
HKLM\System\CurrentControlSet\Services\StorVsp\<Key> = (REG_DWORD)
HKLM\System\CurrentControlSet\Services\VmSwitch\<Key> = (REG_DWORD)
```

Both storage and networking have three registry keys at the preceding StorVsp and VmSwitch paths, respectively. Each value is a DWORD and operates as follows. We do not recommend this advanced tuning option unless you have a specific reason to use it. Note that *these registry keys might be removed in future releases*:

- **IOBalance_Enabled**

  The balancer is enabled when set to a nonzero value and disabled when set to 0. The default is enabled for storage and disabled for networking. Enabling the balancing for networking can add significant CPU overhead in some scenarios.

- **IOBalance_KeepHwBusyLatencyTarget_Microseconds**

  This controls how much work, represented by a latency value, the balancer allows to be issued to the hardware before throttling to provide better balance. The default is 83 ms for storage and 2 ms for networking. Lowering this value can improve balance but will reduce some throughput. Lowering it too much significantly affects overall throughput. Storage systems with high throughput and high latencies can show added overall throughput with a higher value for this parameter.

- **IOBalance_AllowedPercentOverheadDueToFlowSwitching**

  This controls how much work the balancer issues from a VM before switching to another VM. This setting is primarily for storage where finely interleaving I/Os from different VMs can increase the number of disk seeks. The default is 8 percent for both storage and networking.

Network I/O Performance

Hyper-V supports synthetic and emulated network adapters in the VMs, but the synthetic devices offer significantly better performance and reduced CPU overhead. Each of these adapters is connected to a virtual network switch, which can be connected to a physical network adapter if external network connectivity is needed.

For how to tune the network adapter in the root partition, including interrupt moderation, refer to "Performance Tuning for the Networking Subsystem" earlier in this guide. The TCP tunings in that section should be applied, if required, to the child partitions.

Synthetic Network Adapter

Hyper-V features a synthetic network adapter that is designed specifically for VMs to achieve significantly reduced CPU overhead on network I/O when it is compared to the emulated network adapter that mimics existing hardware. The synthetic network adapter communicates between the child and root partitions over VMBus by using shared memory for more efficient data transfer.

The emulated network adapter should be removed through the VM settings dialog box and replaced with a synthetic network adapter. The guest requires that the VM integration services be installed.

Perfmon counters representing the network statistics for the installed synthetic network adapters are available under the counter set \Hyper-V Virtual Network Adapter (*) \ *.

Install Multiple Synthetic Network Adapters on Multiprocessor VMs

Virtual machines with more than one virtual processor might benefit from having more than one synthetic network adaptor installed into the VM. Workloads that are network intensive, such as a Web server, can make use of greater parallelism in the virtual network stack if a second synthetic NIC is installed into a VM.

Offload Hardware

As with the native scenario, offload capabilities in the physical network adapter reduce the CPU usage of network I/Os in VM scenarios. Hyper-V currently uses LSOv1 and TCPv4 checksum offload. The offload capabilities must be enabled in the driver for the physical network adapter in the root partition. For details on offload capabilities in network adapters, refer to "Choosing a Network Adapter" earlier in this guide.

Drivers for certain network adapters disable LSOv1 but enable LSOv2 by default. System administrators must explicitly enable LSOv1 by using the driver **Properties** dialog box in Device Manager.

Network Switch Topology

Hyper-V supports creating multiple virtual network switches, each of which can be attached to a physical network adapter if needed. Each network adapter in a VM can be connected to a virtual network switch. If the physical server has multiple network adapters, VMs with network-intensive loads can benefit from being connected to different virtual switches to better use the physical network adapters.

Perfmon counters representing the network statistics for the installed synthetic switches are available under the counter set \Hyper-V Virtual Switch (*) \ *.

Interrupt Affinity

System administrators can use the IntPolicy tool to bind device interrupts to specific processors.

VLAN Performance

The Hyper-V synthetic network adapter supports VLAN tagging. It provides significantly better network performance if the physical network adapter supports NDIS_ENCAPSULATION_IEEE_802_3_P_AND_Q_IN_OOB encapsulation for both large send and checksum offload. Without this support, Hyper-V cannot use hardware offload for packets that require VLAN tagging and network performance can be decreased.

VMQ

Windows Server 2008 R2 introduces support for VMQ-enabled network adapters. These adapters can maintain a separate hardware queue for each VM, up to the limit supported by each network adapter.

As there are limited hardware queues available, you can use the Hyper-V WMI API to ensure that the VMs that are using the network bandwidth are assigned a hardware queue.

VM Chimney

Windows Server 2008 R2 introduces support for VM Chimney. Network connections with long lifetimes will see the most benefit due to the increase in CPU required for connection establishment when VM Chimney is enabled.

Live Migration

Live migration allows you to transparently move running virtual machines from one node of the failover cluster to another node in the same cluster without a dropped network connection or perceived downtime. In addition, failover clustering requires shared storage for the cluster nodes.

The process of moving a running virtual machine can be broken down in to two major phases. The first phase is the copying of the memory contents on the VM from the current host to the new host. The second phase is the transfer of the VM state from the current host to the new host. The durations of both phases is greatly determined by the speed at which data can be transferred from the current host to the new host.

Providing a dedicated network for Live Migration traffic helps to minimize the time required to complete a Live Migration and it ensures consistent migration times.

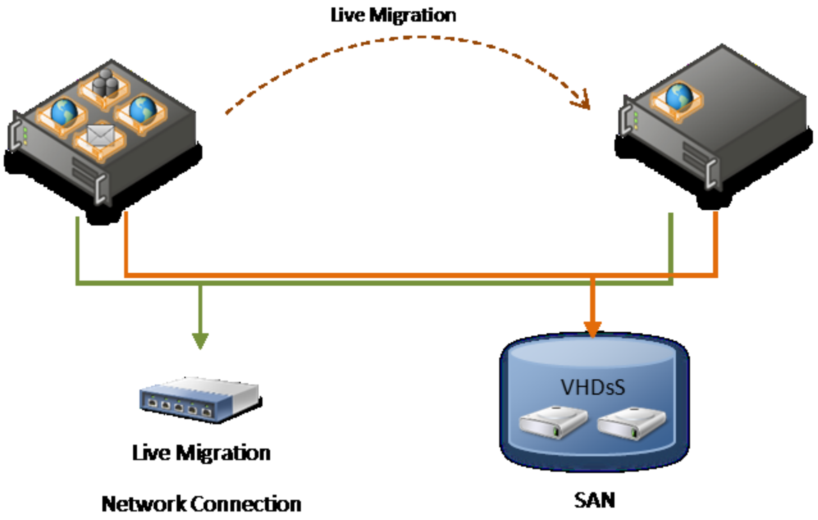


Figure 8. Example Hyper-V Live Migration Configuration

In addition, increasing the number of receive and send buffers on each network adapter involved in the migration can improve migration performance. For more information, see “Performance Tuning for the Networking Subsystem” earlier in this guide.

Performance Tuning for File Server Workload (NetBench)

NetBench 7.02 is an eTesting Labs workload that measures the performance of file servers as they handle network file requests from clients. NetBench gives you an overall I/O throughput score and average response time for your server and with individual scores for the client computers. You can use these scores to measure, analyze, and predict how well your server can handle file requests from clients.

To make sure of a fresh start, the data volumes should always be formatted between tests to flush and clean up the working set. For improved performance and scalability, we recommend that client data be partitioned over multiple data volumes. The networking, storage, and interrupt affinity sections contain additional tuning information that might apply to specific hardware.

Registry Tuning Parameters for Servers

The following registry tuning parameters can affect the performance of file servers:

- **NtfsDisable8dot3NameCreation**

  ```
  HKLM\System\CurrentControlSet\Control\FileSystem\ (REG_DWORD)
  ```

  The default is 0. This parameter determines whether NTFS generates a short name in the 8.3 (MS-DOS) naming convention for long file names and for file names that contain characters from the extended character set. If the value of this entry is 0, files can have two names: the name that the user specifies and the short name that NTFS generates. If the name that the user specifies follows the 8.3 naming convention, NTFS does not generate a short name.

  Changing this value does not change the contents of a file, but it avoids the short-name attribute creation for the file and also changes how NTFS displays and manages the file. For most file servers, the recommended setting is 1.

- **TreatHostAsStableStorage**

  ```
  HKLM\System\CurrentControlSet\Services\LanmanServer
  \Parameters\(REG_DWORD)
  ```

  The default is 0. This parameter disables the processing of write flush commands from clients. If you set the value of this entry to 1, you can improve the server performance and client latency for power-protected servers.

Registry Tuning Parameters for Client Computers

The following registry tuning parameters can affect the performance of client computers:

- **DormantFileLimit**

  ```
  HKLM\system\CurrentControlSet\Services\lanmanworkstation
  \parameters\ (REG_DWORD)
  ```

  Windows XP client computers only.

  This parameter specifies the maximum number of files that should be left open on a share after the application has closed the file.

- **ScavengerTimeLimit**

  ```
  HKLM\system\CurrentControlSet\Services\lanmanworkstation
  \parameters\ (REG_DWORD)
  ```

  Windows XP client computers only.

  This parameter is the number of seconds that the redirector waits before it starts scavenging dormant file handles (cached file handles that are currently not used by any application).

- **DisableByteRangeLockingOnReadOnlyFiles**

  ```
  HKLM\System\CurrentControlSet\Services\LanmanWorkStation
  \Parameters\ (REG_DWORD)
  ```

  Windows XP client computers only.

  Some distributed applications lock parts of a read-only file as synchronization across clients. Such applications require that file-handle caching and collapsing behavior be off for all read-only files. This parameter can be set if such applications will not be run on the system and collapsing behavior can be enabled on the client computer.

Performance Tuning for File Server Workload (SPECsfs2008)

SPECsfs2008 is a file server benchmark suite from Standard Performance Evaluation Corporation that measures file server throughput and response time, providing a standardized method for comparing performance across different vendor platforms. SPECsfs2008 results summarize the server's capabilities with respect to the number of operations that can be handled per second, and the overall latency of the operations.

To ensure accurate results, you should format the data volumes between tests to flush and clean up the working set. For improved performance and scalability, we recommend that you partition client data over multiple data volumes. The networking, storage, and interrupt affinity sections of this paper contain additional tuning information that might apply to specific hardware.

Registry-Tuning Parameters for NFS Server

You can tune the following registry parameters to enhance the performance of NFS servers:

Parameter	Recommended Value
AdditionalDelayedWorkerThreads	16
NtfsDisable8dot3NameCreation	1
NtfsDisableLastAccessUpdate	1
DefaultNumberOfWorkerThreads	128
OptimalReads	1
RdWrHandleLifeTime	10
RdWrNfsHandleLifeTime	60
RdWrNfsReadHandlesLifeTime	10
RdWrThreadSleepTime	60
FileHandleCacheSizeinMB	1*1024*1024*1024 (1073741824)
LockFileHandleCacheInMemory	1
MaxIcbNfsReadHandlesCacheSize	30000
SecureHandleLevel	0
RdWrNfsDeferredWritesFlushDelay	60
CacheAddFromCreateAndMkDir	1

Performance Tuning for Network Workload (NTttcp)

Tuning for NTttcp

NTttcp is a Winsock-based port of **ttcp** to Windows. It helps measure network driver performance and throughput on different network topologies and hardware setups. It provides the customer a multithreaded, asynchronous performance workload for measuring achievable data transfer rate on an existing network setup. For more information, see "Resources" later in this guide.

Options include the following:

- A single thread should be sufficient for optimal throughput.
- Multiple threads are required only for single to many clients.
- Posting enough user receive buffers (by increasing the value passed to the **-a** option) reduces TCP copying.
- You should not excessively post user receive buffers because the first ones that are posted would return before you need to use other buffers.
- It is best to bind each set of threads to a processor (the second delimited parameter in the **-m** option).
- Each thread creates a socket that connects (listens) on a different port.

Table 11. Example Syntax for NTttcp Sender and Receiver

Syntax	Details
Example Syntax for a Sender NTttcps –m 1,0,10.1.2.3 –a 2	Single thread. Bound to CPU 0. Connecting to a computer that uses IP 10.1.2.3. Posting two send-overlapped buffers. Default buffer size: 64 K. Default number of buffers to send: 20 K.
Example Syntax for a Receiver NTttcpr –m 1,0,10.1.2.3 –a 6 –fr	Single thread. Bound to CPU 0. Binding on local computer to IP 10.1.2.3. Posting six receive-overlapped buffers. Default buffer size: 64 KB. Default number of buffers to receive: 20 K. Posting full-length (64-K) receive buffers.

Network Adapter

Make sure that you enable all offloading features.

TCP/IP Window Size

For 1-GB adapters, the settings shown in Table 11 should provide you good throughput because NTttcp sets the default TCP window size to 64 K through a specific socket option (SO_RCVBUF) for the connection. This provides good performance on a low-latency network. In contrast, for high-latency networks or for 10-GB adapters, NTttcp's default TCP window size value yields less than optimal performance. In both cases, you must adjust the TCP window size to allow for the larger bandwidth delay product. You can statically set the TCP window size to a large value by using the **-rb** option. This option disables TCP Window Auto-Tuning, and we recommend its use only if the user fully understands the resultant change in TCP/IP behavior. By default, the TCP window size is set at a sufficient value and adjusts only under heavy load or over high-latency links.

Receive-Side Scaling (RSS)

Windows Server 2008 R2 supports RSS out of the box. RSS enables multiple DPCs to be scheduled and executed on concurrent processors, which improves scalability and performance for receive-intensive scenarios that have fewer networking adapters than available processors. Note that, because of hardware limitations on some adapters and other functionality constraints, not all adapters can support concurrently processing DPCs on all processors on the server. DPCs are also not scheduled on hyperthreading processors because of an adverse effect on performance. Therefore, DPCs in RSS are scheduled only on logical and physical processors regardless of how many cores or sockets are on the server.

Tuning for IxChariot

IxChariot is a networking workload generator from Ixia. It stresses the network to help predict networked application performance.

You can use the High_Performance_Throughput script workload of IxChariot to simulate the NTttcp workload. The tuning considerations for this workload are the same as those for NTttcp.

For more information on IxChariot, see "Resources" later in this guide.

Performance Tuning for Remote Desktop Services Knowledge Worker Workload

Windows Server 2008 R2 Remote Desktop Services (RDS) capacity planning tools include automation framework and application scripting support that enable the simulation of user interaction with RDS. Be aware that the following tunings apply only for a synthetic RDS knowledge worker workload and are not intended as turnings for a server that is not running this workload. This workload is built with these tools to emulate common usage patterns for knowledge workers.

The RDS knowledge worker workload uses Microsoft Office applications and Microsoft Internet Explorer. It operates in an isolated local network that has the following infrastructure:

- Domain controller (Active Directory, Domain Name System—DNS, and Dynamic Host Configuration Protocol —DHCP).
- Microsoft Exchange Server for e-mail hosting.
- IIS for Web hosting.
- Load Generator (a test controller) for creating a distributed workload.
- A pool of Windows XP–based test systems to execute the distributed workload, with no more than 60 simulated users for each physical test system.
- RDS (Application Server) with Microsoft Office installed.

Note: The domain controller and the load generator could be combined on one physical system without degrading performance. Similarly, IIS and Exchange Server could be combined on another computer system.

Table 12 provides guidelines for achieving the best performance on the RDS workload and suggestions as to where bottlenecks might exist and how to avoid them.

Table 12. Hardware Recommendations for RDS Workload

Hardware limiting factor	**Recommendation**
Processor usage	• Use 64-bit processors to expand the available virtual address space. • Use multicore systems (at least two or four sockets and dual-core or quad-core 64-bit CPUs).
Physical disks	• Separate the operating system files, pagefile, and user profiles (user data) to individual physical partitions. • Choose the appropriate RAID configuration. (Refer to “Choosing the RAID Level” earlier in this guide.) • If applicable, set the write-through cache policy to 50% reads and 50% writes. • If applicable, select **Enable write caching on the disk** through the Microsoft Management Console (MMC) disk management snap-in (Diskmgmt.msc). • If applicable, select **Enable Advanced Performance** through the MMC disk management snap-in (Diskmgmt.msc).
Memory (RAM)	The amount of RAM and physical memory access times affect the response times for the user interactions. On NUMA-type computer systems, make sure that the hardware configuration uses the NUMA, which is changed by using system BIOS or hardware partitioning settings.
Network bandwidth	Allow enough bandwidth by using network adapters that have high bandwidths such as 1-GB Ethernet.

Recommended Tunings on the Server

After you have installed the operating system and added the RDS role, apply the following changes:

- Navigate to **Control Panel > System > Advanced System Settings > Advanced** tab and set the following:
  - Navigate to **Performance Settings > Advanced > Virtual memory** and set one or more fixed-size pagefiles (**Initial Size** equal to **Maximum Size**) with a total pagefile size at least two to three times the physical RAM size to minimize paging. For servers that have hundreds of gigabytes of memory, the complete elimination of the paging file is possible. Otherwise, the paging file might be limited because of constraints in available disk space. There are no clear benefits of a paging file larger than 100 GB. Make sure that no system-managed pagefiles are in the **Virtual memory** on the Application Server.
  - Navigate to **Performance Settings > Visual Effects** and select the **Adjust for best performance** check box.

- Allow for the workload automation to run by opening the MMC snap-in for Group Policy (Gpedit.msc) and making the following changes to **Local Computer Policy > User Configuration > Administrative Templates**:
    - Navigate to **Control Panel** > **Display**, and disable **Screen Saver and Password protected screen saver**.
    - Under **Start Menu** and **Taskbar**, enable **Force Windows Classic Start Menu**.
    - Navigate to **Windows Components** > **Internet Explorer**, and enable **Prevent Performance of First Run Customize settings** and select **Go directly to home page**.
    - Navigate to **Start > All Programs > Administrative Tools > System Configuration Tools** tab, disable User Account Control (UAC) by selecting **Disable UAC**, and then reboot the system.
- Allow for the workload automation to run by opening the registry, adding the ProtectedModeOffForAllZones key, and setting it to 1 under:

```
HKLM\SOFTWARE\Microsoft\Internet Explorer\Low Rights\ (REG_DWORD)
```

- Minimize the effect on CPU usage when you are running many RDS sessions by opening the MMC snap-in for Group Policy (Gpedit.msc) and making the following changes under **Local Computer Policy > User Configuration > Administrative Templates**:
    - Under **Start Menu and Taskbar**, enable **Do not keep history of recently opened documents**.
    - Under **Start Menu and Taskbar**, enable **Remove Balloon Tips on Start Menu items**.
    - Under **Start Menu and Taskbar**, enable **Remove frequent program list from Start Menu**.
- Minimize the effect on the memory footprint and reduce background activity by disabling certain Microsoft Win32® services. The following are examples from command-line scripts to do this:

Service name	Syntax to stop and disable service
Desktop Window Manager Session Manager	sc config UxSms start= disabled sc stop UxSms
Windows Error Reporting service	sc config WerSvc start= disabled sc stop WerSvc
Windows Update	sc config wuauserv start= disabled sc stop wuauserv

- Minimize background traffic by opting out of diagnostics feedback programs. Under **Start > All Programs > Administrative Tools > Server Manager**, go to **Resources and Support**:
    - Opt out of participating in the **Customer Experience Improvement Program (CEIP)**.
    - Opt out of participating in **Windows Error Reporting (WER)**.
- Apply the following changes from the Remote Desktop Session Host Configuration MMC snap-in (Tsconfig.msc):

- Set the maximum color depth to **24 bits per pixel (bpp)**.
- Disable all device redirections.
- Navigate to **Start > All Programs > Administrative Tools > Remote Desktop Services > Remote Desktop Session Host Configuration** and change the **Client Settings** from the **RDP-Tcp** properties as follows:
  - Limit the Maximum Color Depth to 24 bpps.
  - Disable redirection for all available devices such as **Drive, Windows Printer**, **LPT Port**, **COM Port**, **Clipboard**, **Audio**, **Supported Plug and Play Devices**, and **Default to main client printer.**

Monitoring and Data Collection

The following list of performance counters is considered a base set of counters when you monitor the resource usage on the RDS workload. Log the performance counters to a local, raw (blg) performance counter log. It is less expensive to collect all instances (‘*’ wide character) and then extract particular instances while post-processing by using Relog.exe:

```
\Cache\*
\IPv4\*
\LogicalDisk(*)\*
\Memory\*
\Network Interface(*)\*
\Paging File(*)\*
\PhysicalDisk(*)\*
\Print Queue(*)\*
\Process(*)\*
\Processor Information(*)\*
\Synchronization(*)\*
\System\*
\TCPv4\*
```

Note: If applicable, add the \IPv6* and \TCPv6* objects.

Stop unnecessary ETW loggers by running **logman.exe stop -ets <provider name>**. To view providers on the system, run **logman.exe query -ets**.

Use Logman.exe to collect performance counter log data instead of using Perfmon.exe, which enables logging providers and increases CPU usage.

Performance Tuning for SAP Sales and Distribution Two-Tier Workload

SAP AG has developed several standard application benchmarks. The Sales and Distribution (SD) workload represents one of the important classes of workloads that are used for benchmarking SAP enterprise resource planning (ERP) installations. For more information on obtaining the benchmark kit, see the link to the SAP web page in “Resources” later in this guide.

SAP updated the SAP SD workload in January, 2009. The updates include added requirements such as subsecond response time and a Unicode codepage. For more information, see the link to the SAP web page in "Resources" later in this guide.

You can perform multidimensional tuning of the operating system level, application server, database server, network, and storage to achieve optimal throughput and good response times as the number of concurrent SD users increases before capping out because of resource limitations.

The following sections provide guidelines that can benefit the two-tier setup specifically for SAP ERP SD benchmarks on Windows Server 2008 R2. Some of these recommendations might not apply to the same degree for production systems.

Operating System Tunings on the Server

- Navigate to **Control Panel > System > Advanced System Settings > Advanced** tab and configure the following:
    - Navigate to **Performance Settings > Advanced > Virtual memory** and set one or more fixed-size pagefiles (**Initial Size** equal to **Maximum Size**). The pagefile size should meet the total virtual memory requirements of the workload. Make sure that no system-managed pagefiles are in the **Virtual memory** on the Application Server.
    - Navigate to **Performance Settings > Visual Effects** and select the **Adjust for best performance** check box.
- To enable SQL to use large pages, enable the **Lock pages in memory** user right assignment for the account that will run the SQL and SAP services.

  From the Group Policy MMC snap-in (Gpedit.msc), navigate to **Computer Configuration > Windows Settings > Security Settings > Local Policies > User Rights Assignment**. Double-click **Lock pages in memory** and add the accounts that have credentials to run Sqlservr.exe and SAP services.
- Disable User Account Control.

  Navigate to **Start > All Programs > Administrative Tools > System Configuration > Tools** tab, select **Disable UAC**, and then reboot the system. This setting can be used for benchmarking environments, but enabling UAC might be a security compliance requirement in production environments.

Tunings on the Database Server

When the database server is SQL Server®, consider setting the following SQL Server configuration options with *sp_configure*. For detailed information on the *sp_configure* stored procedure, see the information about setting server configuration options in "Resources" later in this guide.

- Apply CPU affinity for the SQL Server process: Set an affinity mask to partition the SQL process on specific cores. If required, use the affinity64 mask to set the affinity on more than 32 cores. Starting with SQL Server 2008 R2, you can apply equivalent settings for configuring CPU affinity on as many as 256 logical processors by using the ALTER SERVER CONFIGURATION SET PROCESS AFFINITY Data Definition Language (DDL) TSQL statement as the sp_configure affinity mask options are announced for deprecation. For more information on DDL, see

"Resources" later in this guide. For the current two-tier SAP SD benchmarks, it is typically sufficient to run SQL Server on one-eighth or fewer of the existing cores.

- Set a fixed amount of memory that the SQL Server process will use. For example, set the **max server memory** and **min server memory** equal and large enough to satisfy the workload (2500 MB is a good starting value).

On NUMA-class hardware, you can do the following:

- To further subdivide the CPUs in a hardware NUMA node to more CPU nodes (known as Soft-NUMA), see the information about configuring SQL Server to use Soft-NUMA in "Resources" later in this guide.
- To provide NUMA node locality for SQL Server, set preferred NUMA node hints (applies to Windows Server 2008 R2 and later). For the commands below, use the service name. The [server] parameter is optional, the other parameters are required:
    - Use the following command to set the preferred NUMA node:

      ```
      %windir%\system32\sc.exe [server] preferrednode <SQL Server
      service name> <NUMA node number>
      ```

      You need administrator permissions to set the preferred node. Use **%windir%\system32\sc.exe preferrednode** to display help text.
    - Use the following command to query the setting:

      ```
      %windir%\system32\sc.exe [server] qpreferrednode <SQL Server
      service name>
      ```

      This command fails if the service has no preferred node settings. Use **%windir%\system32\sc.exe qpreferrednode** to display help text.
    - Use the following command to remove the setting:

      ```
      %windir%\system32\sc.exe [server] preferrednode <SQL Server
      service name> -1
      ```

On a two-tier ERP SAP setup, consider enabling and using only the Named Pipes protocol and disabling the rest of the available protocols from the SQL Server Configuration Manager for the local SQL connections.

Tunings on the SAP Application Server

- The ratio between the number of Dialog (D) processes versus Update (U) processes in the SAP ERP installation might vary, but usually a ratio of 1D:1U or 2D:1U per logical processor is a good start for the SD workload. Ensure that in a SAP dialog instance, the number of worker processes and users does not exceed the capacity of the SAP dispatcher for that dialog instance (the current maximum is approximately 2,000 users per instance). On NUMA-class hardware, consider installing one or more SAP dialog instances per NUMA node (depending on the number of logical processors per NUMA node that you want to use with SAP worker processes). The D:U ratio, and the overall number of SAP dialog instances per NUMA node or system wide, might be improved based on the analysis of previous experiments.
- To further partition within an SAP instance, use the processor affinity capabilities in the SAP instance profiles to partition each worker process to a subset of the available logical processors and achieve better CPU and memory locality. Affinity

setting in the SAP instance profiles is supported for as many as 64 logical processors.

- Use the FLAT memory model that SAP AG released on November 23, 2006, with the SAP Note No. 1002587 “Flat Memory Model on Windows” for SAP kernel 7.00 Patch Level 87.
- Windows Server 2008 R2 supports more than 64 logical processors. On such NUMA-class systems, consider setting preferred NUMA nodes in addition to setting hard affinities by using the following steps:
  1. Set the preferred NUMA node for the SAP Win32 service and SAP Dialog Instance services (processes instantiated by Sapstartsrv.exe). When you enter commands on the local system, you can omit the server parameter. For the commands below, use the service short name:
     - Use the following command to set the preferred NUMA node:

       ```
       %windir%\system32\sc.exe [server] preferrednode <service
       name> <NUMA node number>
       ```

       You need administrator permissions to set the preferred node. Use **%windir%\system32\sc.exe preferrednode** to display help text.
     - Use the following command to query the setting:

       ```
       %windir%\system32\sc.exe [server] qpreferrednode <service
       name>
       ```

       This command fails if the service has no preferred node settings. Use **%windir%\system32\sc.exe qpreferrednode** to display help text.
     - Use the following command to remove the setting:

       ```
       %windir%\system32\sc.exe [server] preferrednode <service
       name> -1
       ```
  2. To allow each SAP worker process in a dialog instance to inherit the ideal NUMA node from its Win32 service, create registry key entries under the following key for each of the Sapstartsrv.exe, Msg_server.exe, Gwrd.exe, and Disp+work.exe images and set the "NodeOptions"=dword:00000100 value:

     ```
     HKLM\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Image File
     Execution Options\ (IMAGE NAME)\ (REG_DWORD)
     ```
  3. If the preferred NUMA node is used without hard affinity settings for SAP worker processes, or if time measurement issues are observed as described by SAP Note No. 532350 released on November 29, 2004, apply the recommendation to let SAP processes use the Query Performance Counter (QPC) timer to stabilize the benchmark environment. Set the following system environment variable:

     ```
     %windir%\system32\setx.exe /M SAP_USE_WIN_TIMER YES
     ```
  4. If applicable, use the IntPolicy tool as described in the “Interrupt Affinity” section earlier in this guide to set an optimal interrupt affinity for storage or network devices.

  You can use the Coreinfo tool from Windows Sysinternals to provide topology details about logical and physical processors, processor sockets, NUMA nodes, and processor cache. For more information, see “Resources” later in this guide.

Monitoring and Data Collection

The following list of performance counters is considered a base set of counters when you monitor the resource usage of the Application Server while you are running the two-tier SAP ERP SD workload. Log the performance counters to a local, raw (blg) performance counter log. It is less expensive to collect all instances (‘*’ wide character) and then extract particular instances while post-processing by using Relog.exe:

\Cache*
\IPv4*
\LogicalDisk(*)*
\Memory*
\Network Interface(*)*
\Paging File(*)*
\PhysicalDisk(*)*
\Process(*)*
\Processor Information(*)*
\Synchronization(*)*
\System*
\TCPv4*
\SQLServer:Buffer Manager\Lazy writes/sec

Note: If applicable, add the \IPv6* and \TCPv6* objects.

Performance Tuning for TPC-E Workload

TPC-E online transaction processing (OLTP) is one of the primary database workloads used to evaluate SQL Server and Windows Server performance. TPC-E uses a central database that executes transactions related to a brokerage firm’s customer accounts. The primary metric for TPC-E is Trade-Result transactions per second (tpsE). Note that Trade-Result transactions account for 10% of the transaction mix. For more information about the TPC-E benchmark, see the TPC-E website listed in “Resources” later in this guide.

A non-clustered TPC-E benchmark setup consists of two parts: a set of client systems and the server under test (SUT). To achieve maximum system utilization and throughput, you can tune the operating system, SQL Server, storage, memory, processors, and network. This section describes configuration guidelines for achieving optimal TPC-E performance.

Server Under Test (SUT) Tunings

Use the following SUT tunings:

- Set the power scheme to High Performance.
- Configure pagefiles for best performance:
  - Navigate to **Performance Settings > Advanced > Virtual memory** and configure one or more fixed-size pagefiles with Initial Size equal to Maximum Size. The pagefile size should be equal to the total virtual memory requirement of the workload. Make sure that no system-managed pagefiles are in the virtual memory on the application server.

    - Navigate to **Performance Settings > Visual Effects** and select **Adjust for best performance**.

- To enable SQL Server to use large pages, enable the Lock pages in memory user right assignment for the account that will run the SQL Server:

    - From the Group Policy MMC snap-in (Gpedit.msc), navigate to **Computer Configuration > Windows Settings > Security Settings > Local Policies > User Rights Assignment**. Double-click **Lock pages in memory** and add the accounts that have credentials to run SQL Server.

- Configure network devices:

    - The number of network devices is determined from previous runs. Network device utilization should not be higher than 65%-75% of total NIC bandwidth. Use 1-Gbps NICs at minimum.

    - From the Device Manager MMC snap-in (Devmgmt.msc), navigate to **Network Adapters** and determine the network devices to be used. Disable devices that are not being used.

    - If interrupt partitioning is necessary in high interrupt rates per NIC port scenarios, and the device supports interrupt affinity configuration, set network device interrupt affinity:

        - Using the IntPolicy tool, set interrupt affinity in a round-robin fashion starting from processor 0. If the SUT is a multinode system, determine on which nodes the NICs reside and set the affinity to processors that belong to the node on which each NIC resides. For detailed information on the IntPolicy tool, see "Resources" later in this guide.

    - For advanced network tuning information, see “Performance Tuning for the Networking Subsystem” earlier in this guide.

- Configure storage devices:

    - If the operating system is Windows Server 2008 R2, DPC redirection optimization is available on some storage drivers. If the storage device driver supports DPC redirection optimization, there is no need to set interrupt affinity on storage devices. If the storage device driver does not support DPC redirection, or if storage device driver interrupts are not distributed to processors on the same NUMA node where the device resides, set the interrupt affinity for each device by using IntPolicy as advised for networking devices.

    - For advanced storage tuning information, see “Performance Tuning for the Storage Subsystem” earlier in this guide.

- Configure disks for advanced performance:

    - From the Disk Management MMC snap-in (Diskmgmt.msc), select each disk in use, right-click to **Properties > Policies** and select **Advanced Performance** if it is enabled for the disk.

SQL Server Tunings for TPC-E Workload

The following SQL Server tunings improve performance and scalability in benchmark environments such as TPC-E; they are not intended for production environments:

- You can use the **-T834** start flag to enable SQL Server to use large pages, which improves performance in a benchmark environment. The use of large pages is not generally recommended outside of benchmark environments.
- If you disable SQL Server performance counters to avoid potential overhead, start SQL Server as a process instead of a service and use the **-x** flag:
  1. From the Services MMC snap-in (*Services.msc*), stop and disable SQL Services.
  2. Execute the following command from the SQL Server Binn directory:

     ```
     sqlservr.exe -c -x
     ```
- Enable the TCP/IP protocol to allow communication with client systems:
  - Navigate to **Start Menu > Programs > Microsoft SQL Server R2 > Configuration Tools > SQL Server Configuration Manager**. Then navigate to **SQL Server Network Configuration > Protocols** for MSSQL Server, right-click **TCP/IP**, and click **Enable**.
- Configure SQL Server according to the guidance in the following list. You can configure SQL Server by using the *sp_configure* stored procedure. Set the "show advanced options" value to 1 to display more available configuration options. Detailed information about the *sp_configure* stored procedure is available in "Resources" later in this guide:
  - You can set CPU affinity for the SQL process to isolate system resources for the SQL Server instance from other SQL Server instances or other applications running on the same system. You can also set CPU affinity for the SQL process to not use a set of logical processors that handle I/O interrupt traffic (network and disk).

    You can set CPU affinity for the SQL process in different ways, depending on processor count: Set **affinity mask** to partition the SQL process on specific cores up to 32 logical processors. To set affinity on more than 32 logical processors but fewer than 64 processors, use **affinity64 mask.** Starting with SQL Server 2008 R2, you can apply equivalent settings for configuring CPU affinity on as many as 256 logical processors using the ALTER SERVER CONFIGURATION SET PROCESS AFFINITY Data Definition Language (DDL) TSQL statement as the sp_configure affinity mask options are announced for deprecation. Use the **'alter server configuration set process affinity cpu ='** command to set affinity to the desired range or ranges of processors, separated by commas. For more information on best practices for installations with more than 64 logical processors, and for more information on DDL, see "Resources" later in this guide.
  - You can set a fixed amount of memory for the SQL Server process to use. About 3% of the total available memory is used for the system, and another 1% is used for memory management structures. SQL Server can use the rest of available memory, but not more.

The following equation is available to calculate total memory to be used by SQL Server:

TotalMemory – (1%memory * (numa_nodes)) – 3%memory – 1GB memory

- Leave the lightweight pooling value set to the default of 0. This enables SQL Server to run in threads mode. Threads mode performance is comparable to fibers mode.
- If it appears that the default settings do not allow sufficient concurrent transactions based on a throughput value lower than expected for the system and benchmark configuration, set the maximum worker threads value to approximately the number of connected users. Monitor the sys.dm_os_schedulers DMV to determine whether you need to increase the number of worker threads.
- In benchmark environments, set the default trace enabled value to 0. This is not recommended in production environments, because it reduces the ability to diagnose problems.
- Set the priority boost value to 1.

Disk Storage Tunings

Tune the disk storage:

- The TPC-E benchmark rules require disk storage redundancy. You can use RAID 1+0 if you have enough storage capacity. If you do not have enough capacity, you can use RAID 5 .
- If you use rotational disks, configure logical drives so that all spindles are used for database disks, if possible. Additional spindles improve overall disk subsystem performance.
- The TPC-E workload consists of two disk I/O workloads: random reads/writes in a 9:1 ratio on database tables, and sequential writes on the log. You can improve performance with proper write caching on the log disk *only in the case of battery backed up disk configurations* that are able to avoid data loss in case of power failure:
  - Enable 100% write caching for the log disk.

TPC-E Database Size and Layout

Tune the database size and layout:

- The TPC-E database consists of several file groups, and it can vary between different benchmark kits. Size is measured in number of customers, and for the database to be auditable, the ratio of database size (customers) to throughput (tpsE) should be approximately 500.
- You can perform more fine tuning on the database layout :
  - Database tables that have higher access frequency should be placed on the outer edge of the disk if rotational disks are used.

- The default TPC-E kit can be changed, and new file groups can be created. That way, file groups can consist of higher frequency access table(s) and they can be placed on the outer edge of the disk for better performance.

Client Systems Tunings

Tune the client systems:

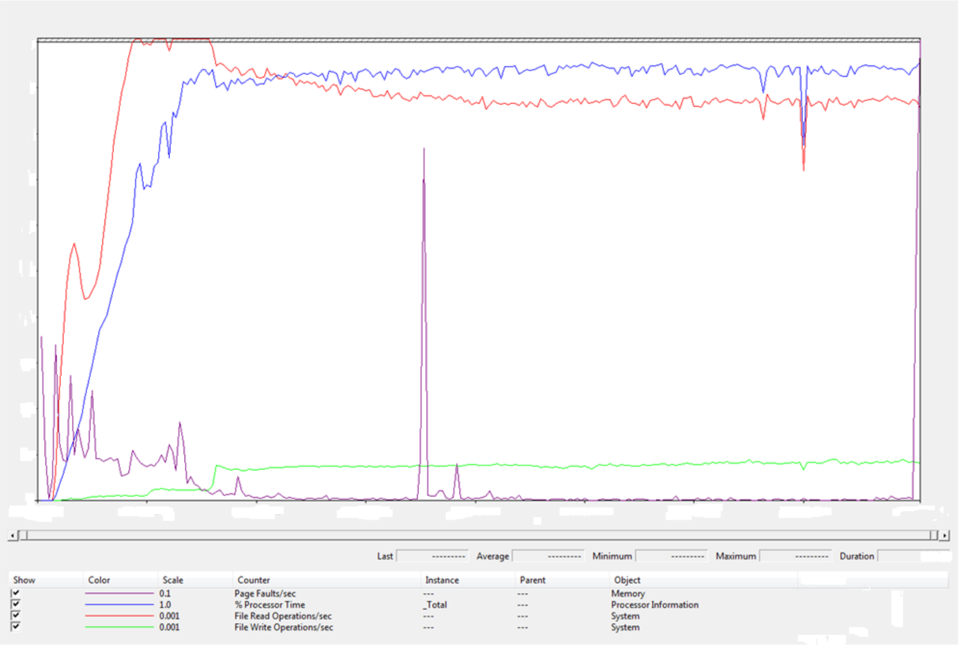
- Configure client systems the same way that the SUT is configured. See “Server Under Test (SUT) Tunings” earlier in this guide.
- In addition to tuning the client systems, you should monitor client performance and eliminate any bottlenecks. Follow these client performance guidelines:
    - CPU utilization on clients should not be higher than 80%, to accommodate activity bursts.
    - If any of the processors has high CPU utilization, consider using CPU affinity for benchmark processes to even out CPU utilization. If CPU utilization is still high, consider upgrading clients to the latest processors, or add more clients.
- Verify that time is synchronized between the master client and the SUT.

Monitoring and Data Collection

The following list of performance counters is considered a base set of counters when you monitor the resource usage of the database server for the TPC-E workload. Log the performance counters to a local, raw (blg) performance counter log. It is less expensive to collect all instances (‘*’ wide character) and then extract particular instances while post-processing by using Relog.exe or Perfmon:

```
\IPv4\*
\Memory\*
\Network Interface(*)\*
\PhysicalDisk(*)\*
\Processor Information(*)\*
\Synchronization(*)\*
\System\*
\TCPv4\*
```

Note: If applicable, add the \IPv6* and \TCPv6* objects. To monitor overall performance, you can use the performance counter chart displayed in Figure 9 and the throughput chart displayed in Figure 10 to visualize run characteristics. The first part of the run in Figure 9 represents the warm-up stage where I/O consists of mostly reads. As the run progresses, the lazy writer starts flushing caches to the disks and as write I/O increases, read I/O decreases. The beginning of steady state for the run is when the read I/O and write I/O curves seem to be parallel to each other.



Last --------- Average --------- Minimum --------- Maximum --------- Duration

Show	Color	Scale	Counter	Instance	Parent	Object
☑	🟪	0.1	Page Faults/sec	---	---	Memory
☑	🟦	1.0	% Processor Time	_Total	---	Processor Information
☑	🟥	0.001	File Read Operations/sec	---	---	System
☑	🟩	0.001	File Write Operations/sec	---	---	System

Figure 9: TPC-E Perfmon Counters Chart

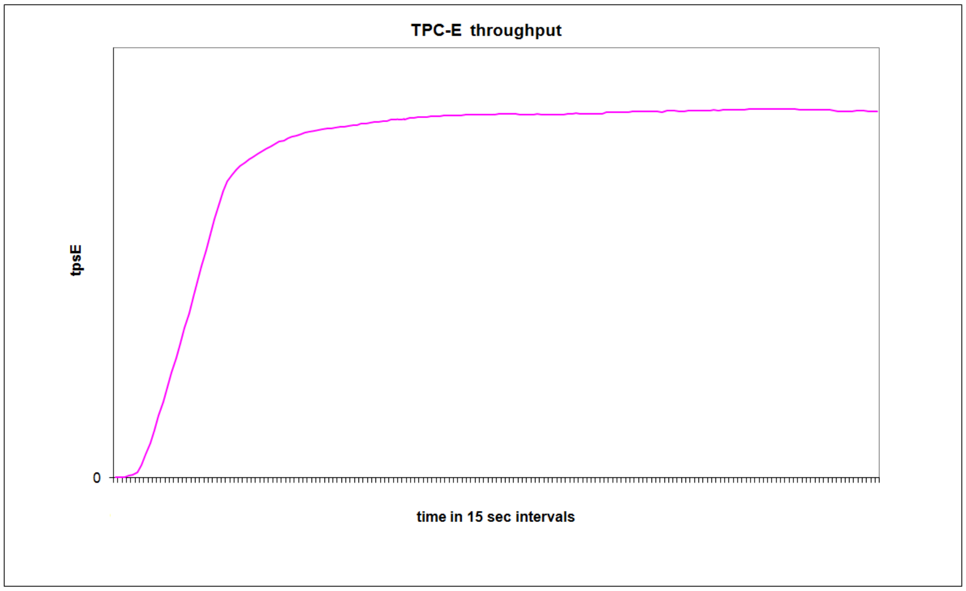


Figure 10. TPC-E Throughput Chart

You can use other tools such as Xperf to perform additional analysis.

Resources

Web Sites

Windows Server 2008 R2

http://www.microsoft.com/windowsserver2008/en/us/R2.aspx

Windows Server 2008

http://www.microsoft.com/windowsserver2008/

Windows Server Performance Team Blog

http://blogs.technet.com/winserverperformance/

Windows Server Catalog

http://www.windowsservercatalog.com/

SAP Global Benchmark: Sales and Distribution (SD)

http://www.sap.com/solutions/benchmark/sd.epx

Windows Sysinternals

http://technet.microsoft.com/sysinternals/default.aspx

Transaction Processing Performance Council

http://www.tpc.org/

IxChariot

http://www.ixiacom.com/support/ixchariot/

Power Management

Power Policy Configuration and Deployment in Windows

http://msdn.microsoft.com/windows/hardware/gg463243.aspx

Using PowerCfg to Evaluate System Energy Efficiency

http://msdn.microsoft.com/windows/hardware/gg463250.aspx

Interrupt-Affinity Policy Tool

http://msdn.microsoft.com/windows/hardware/gg463378.aspx

Networking Subsystem

Scalable Networking: Eliminating the Receive Processing Bottleneck—Introducing RSS

http://download.microsoft.com/download/5/D/6/5D6EAF2B-7DDF-476B-93DC-7CF0072878E6/NDIS_RSS.doc

Windows Filtering Platform

http://msdn.microsoft.com/windows/hardware/gg463267.aspx

Networking Deployment Guide: Deploying High-Speed Networking Features

http://download.microsoft.com/download/8/E/D/8EDE21BC-0E3B-4E14-AAEA-9E2B03917A09/HSN_Deployment_Guide.doc

Storage Subsystem

Disk Subsystem Performance Analysis for Windows

(Parts of this document are out of date, but many of the general observations and guidelines are still accurate.)

http://msdn.microsoft.com/windows/hardware/gg463405.aspx

Web Servers

10 Tips for Writing High-Performance Web Applications
http://go.microsoft.com/fwlink/?LinkId=98290

File Servers

Performance Tuning Guidelines for Microsoft Services for Network File System
http://technet.microsoft.com/library/bb463205.aspx

[MS-FSSO]: File Access Services System Overview
http://msdn.microsoft.com/library/ee392367(v=PROT.10).aspx

How to disable the TCP autotuning diagnostic tool
http://support.microsoft.com/kb/967475

Active Directory Servers

Active Directory Performance for 64-bit Versions of Windows Server 2003
http://www.microsoft.com/downloads/details.aspx?FamilyID=52e7c3bd-570a-475c-96e0-316dc821e3e7

How to configure Active Directory diagnostic event logging in Windows Server 2003 and in Windows 2000 Server
http://support.microsoft.com/kb/314980

Remote Desktop Session Host Capacity Planning

RD Session Host Capacity Planning in Windows Server 2008 R2
http://www.microsoft.com/downloads/details.aspx?displaylang=en&FamilyID=ca837962-4128-4680-b1c0-ad0985939063

RD Virtualization Host Capacity Planning in Windows Server 2008 R2
http://www.microsoft.com/downloads/details.aspx?displaylang=en&FamilyID=bd24503e-b8b7-4b5b-9a86-af03ac5332c8

Virtualization Servers

Hyper-V Dynamic Memory Configuration Guide
http://technet.microsoft.com/library/ff817651(WS.10).aspx

NUMA Node Balancing
http://blogs.technet.com/b/winserverperformance/archive/2009/12/10/numa-node-balancing.aspx

Hyper-V WMI Provider
http://msdn2.microsoft.com/library/cc136992(VS.85).aspx

Hyper-V WMI Classes
http://msdn.microsoft.com/library/cc136986(VS.85).aspx

Requirements and Limits for Virtual Machines and Hyper-V in Windows Server 2008 R2
http://technet.microsoft.com/library/ee405267(WS.10).aspx

Network Workload

Ttcp

http://en.wikipedia.org/wiki/Ttcp

How to Use NTttcp to Test Network Performance

http://msdn.microsoft.com/windows/hardware/gg463264.aspx

Sales and Distribution Two-Tier Workload and TPC-E Workload

Setting Server Configuration Options

http://go.microsoft.com/fwlink/?LinkId=98291

How to: Configure SQL Server to Use Soft-NUMA

http://go.microsoft.com/fwlink/?LinkId=98292

How to: Map TCP/IP Ports to NUMA Nodes

http://go.microsoft.com/fwlink/?LinkId=98293

ALTER SERVER CONFIGURATION SET PROCESS AFFINITY (Transact-SQL) (How to Set Process Affinity using DDL)

http://msdn.microsoft.com/library/ee210585.aspx

Best Practices for Running SQL Server on Computers That Have More Than 64 CPUs

http://msdn.microsoft.com/library/ee210547.aspx

SAP with Microsoft SQL Server 2008 and SQL Server 2005: Best Practices for High Availability, Maximum Performance, and Scalability

http://www.sdn.sap.com/irj/sdn/sqlserver?rid=/library/uuid/4ab89e84-0d01-0010-cda2-82ddc3548c65