Windows

Performance Tuning Guidelines for Windows Server 2012

April 12, 2013

Abstract

This guide describes important tuning parameters and settings that you can adjust to improve the performance and energy efficiency of the Windows Server 2012 operating system. It describes each setting and its potential effect to help you make an informed decision about its relevance to your system, workload, and performance goals.

The guide is for information technology (IT) professionals and system administrators who need to tune the performance of a server that is running Windows Server 2012.

For the most current version of this guide, see Performance Tuning Guidelines for Windows Server 2012.

Microsoft

Document History

Date	Change
April 12, 2013	Added note in the "Performance Tuning for TPC-E workload" section that the tunings are specifically for OLTP benchmarking and should not be perceived as general SQL tuning guidance.
October 12, 2012	Updated Server Core Installation Option, Correct Memory Sizing for Child Partitions, and Correct Memory Sizing for Root Partition

Contents

Introduction

When you run a server system in your organization, you might have business needs that are not met by using the default settings. For example, 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 tune the server settings in Windows Server® 2012 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 to develop and optimize the features, algorithms, and settings in Windows Server 2012 and Windows Server 2008 R2 to maximize energy efficiency with minimal effects on performance. This guide 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, Windows Server 2008, and Windows Server 2008 R2 to Windows Server 2012. 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 Virtualization Host
- Remote Desktop Gateway
- Virtualization Servers (Hyper-V)
- Performance Tools for Network Workloads
- SAP Sales and Distribution
- 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 hardware to provide 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 they 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 guide 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 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 are required for Windows Server 2012. No 32-bit editions of the operating system will be provided, but 32-bit applications will run on the 64-bit Windows Server 2012 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 hyperthreading is enabled because hyperthreading 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 of processors because the comparison can be a misleading indicator of speed.
Cache	Choose large L2 or L3 processor caches. The larger caches generally provide better performance, and they often play a bigger role than raw CPU frequency.

Component	Recommendation
Memory (RAM) and paging storage	Increase the RAM to match your memory needs. When your computer runs low on memory and it needs more immediately, modern operating systems use hard disk space to supplement system RAM through a procedure called paging. Too much paging degrades the overall system performance. You can optimize paging by using the following guidelines for page file placement: • Isolate the page file on its own storage device(s), or at least make sure it doesn't share the same storage devices as other frequently accessed files. For example, place the page file and operating system files on separate physical disk drives. • Place the page file 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 page file on a fault-tolerant drive, remember that fault-tolerant systems are often slower to write data 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 page files on different partitions of the same physical disk drive.
Peripheral bus	In Windows Server 2012, it is highly recommended that the primary storage and network interfaces are PCI Express (PCIe), and that servers with PCIe buses are chosen. Also, to avoid bus speed limitations, use PCIe x8 and higher slots for 10 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. 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 frequently accessed 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 a 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 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.
Receive Side Scaling (RSS) support	RSS is a technology that enables packet receive-processing to scale with the number of available computer processors. Particularly important with faster Ethernet (10 GB or more).
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 2012 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 much of this guide focuses on how to obtain the best performance from Windows Server 2012, it is also important to 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

Component	Recommendation
Processors	Frequency, operating voltage, cache size, and process technology 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 they may expose more power states for the Windows power management algorithms, which enables better power management at all levels of performance. Or they may use some of the new "cooperative" power management techniques that Microsoft has developed in partnership with hardware manufacturers.[1]
Memory (RAM)	Memory accounts for an increasing fraction of the 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, you should also factor in the energy cost of the paging disks.
Disks	Higher RPM means increased energy consumption. Also, 2.5-inch drives generally require less power than 3.5-inch drives. For more information about the energy costs 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 the system load decreases, eliminating otherwise unnecessary energy consumption.
USB devices	Windows Server 2012 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.
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.

Processor Terminology

The processor terminology used throughout this guide reflects the hierarchy of components available in Figure 1. Terms used from largest to smallest granularity of components are the following:

- Processor Socket
- NUMA node

[1] See Collaborative Processor Performance Control in the Advanced Configuration and Power Interface.

- Core
- Logical Processor

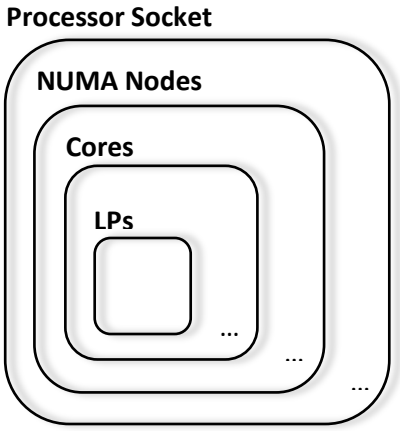

Figure 1. Processor terminology

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.

Windows Server 2012 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. However, the majority of server hardware and workloads should not require administrator power tuning when running Windows Server 2012.

Calculating Server Energy Efficiency

When you tune your server for energy savings, you must also consider performance. Tuning affects 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 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 data center 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 2012 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. If your server model qualifies for the new Enhanced Power Management 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 an instantaneous power draw or an 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 a “load line.” Figure 2 shows load lines for three sample configurations.

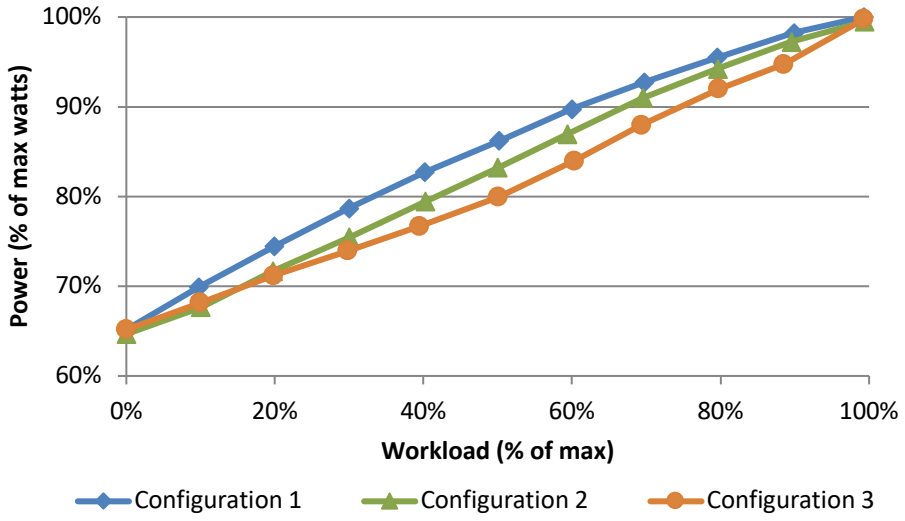

Figure 2. Sample load lines

You can use load lines to evaluate and compare the performance and energy consumption of configurations at all load points. In this particular example, it is easy to see what is the best configuration. However, there can easily be scenarios where one configuration works best for heavy workloads and one works best for light workloads. You need to thoroughly understand your workload requirements to choose an optimal configuration. Don’t assume that when you find a good configuration, it will always remain optimal. 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

The Windows PowerCfg tool supports a command-line option that you can use to analyze the idle energy efficiency of your server. When you run the **powercfg** command with the **/energy** option, the tool performs a 60-second test to detect potential energy efficiency issues. The tool 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.

Note Windows PowerCfg is not available in operating systems earlier than Windows 7 and Windows Server 2008 R2.

Shortened timer tick rates, drivers that lack power management support, and excessive CPU utilization are a few of the behavioral issues that are detected by the **powercfg /energy** command. This tool provides a simple way to identify and fix power management issues, 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 2012 has three built-in power plans designed to meet different sets 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. Targets good energy efficiency with minimal 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. Power and thermal limitations, operating expenses, and reliability considerations apply.	• Low latency applications • Application code that is sensitive to processor performance changes	Processors are always locked at the highest performance state (including “turbo” frequencies). All cores are unparked.
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 the Windows operating system for alternating current (AC) and direct current (DC) powered systems, but in this guide we assume that servers are using an AC power source.

For more information on power plans, power policies, and power policy configurations, 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, see Power Policy Configuration and Deployment in Windows. This document provides a detailed explanation of power plans and parameters, and it includes instructions for adjusting parameter values by using the PowerCfg tool.

Processor Performance Boost Mode

Intel Turbo Boost and AMD Turbo CORE technologies are features that allow 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 Windows Server 2012 configures Turbo technologies based on the power policy that is in use and the specific processor implementation.

Turbo is enabled for High Performance power plans on all Intel and AMD processors and it is disabled for Power Saver power plans. For Balanced power plans on systems that rely on traditional P-state-based frequency management, Turbo is enabled by default only if the platform supports the EPB register.

Note At the time of writing this guide, the EPB register is only supported in Intel Westmere and later processors.

For Intel Nehalem and AMD processors, Turbo is disabled by default on P-state-based platforms. However, if a system supports Collaborative Processor Performance Control (CPPC), which is a new alternative mode of performance communication between the operating system and the hardware (defined in ACPI 5.0), Turbo may be engaged if the Windows operating system dynamically requests the hardware to deliver the highest possible performance levels.

To enable or disable the Turbo Boost feature, you must configure the **Processor Performance Boost Mode** parameter. **Processor Performance Boost Mode** has five allowable values, as shown in Table 5. For P-state-based control, the choices are Disabled, Enabled (Turbo is available to the hardware whenever nominal performance is requested), and Efficient (Turbo is available only if the EPB register is implemented). For CPPC-based control, the choices are Disabled, Efficient Enabled (Windows specifies the exact amount of Turbo to provide), and Aggressive (Windows asks for "maximum performance" to enable Turbo). In Windows Server 2012, the default value for Boost Mode is 3.

Table 5. Processor Performance Boost Mode parameter values

Value (Name)	P-state-based Behavior	CPPC Behavior

Value (Name)	P-state-based Behavior	CPPC Behavior
0 (Disabled)	Disabled	Disabled
1 (Enabled)	Enabled	Efficient Enabled
2 (Aggressive)	Enabled	Aggressive
3 (Efficient Enabled)	Efficient	Efficient Enabled
4 (Efficient Aggressive)	Efficient	Aggressive

The following commands set **Processor Performance Boost Mode** to Enabled on the current power plan (specify the policy by using a GUID alias):

```
Powercfg -setacvalueindex scheme_current sub_processor
PERFBOOSTMODE 1
Powercfg -setactive scheme_current
```

Note 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 currently 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 previously shown with the desired alias to enable that power plan. For example, to adjust the Boost Mode in the Power Saver plan and make Power Saver the current plan, run the following commands:

```
Powercfg -setacvalueindex scheme_max sub_processor PERFBOOSTMODE 1
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 Processor Performance State** parameter or the **Maximum Processor Performance State** parameter.

The values for 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 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
PROCTHROTTLEMIN 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
PROCTHROTTLEMAX 75
Powercfg -setactive scheme_current
```

Note 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.

Processor Performance Core Parking Maximum and Minimum Cores

Core parking is a feature that was introduced in Windows Server 2008 R2. The processor power management (PPM) engine and the scheduler work together to dynamically adjust the number of cores that are available to run threads. The PPM engine chooses a minimum number of cores for the threads that will be scheduled. Cores that are chosen to "park" generally do not have any threads scheduled, and they will drop into very low power states when they are not processing interrupts, DPCs, or other strictly affinitized work. The remaining set of "unparked" cores are responsible for the remainder of the workload. Core parking can potentially 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 a reasonable balance of throughput and energy efficiency. On processors where core parking may not show as much benefit on generic workloads, it can be disabled by default. If your server has specific core parking requirements, you can control the number of cores that are available to park by using the **Processor Performance Core Parking Maximum Cores** parameter or the **Processor Performance Core Parking Minimum Cores** parameter in Windows Server 2012.

One scenario that core parking has difficulty with is when there are one or more active threads affinitized to a non-trivial subset of CPUs in a NUMA node (that is, more than 1 CPU, but less than the entire set of CPUs on the node). When the core parking algorithm is picking cores to unpark (assuming an increase in workload intensity occurs), it does not know to pick the cores within the active affinitized subset (or subsets) to unpark, and thus may end up unparking cores that won't actually be utilized.

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

```
Powercfg -setacvalueindex scheme_current sub_processor CPMINCORES
100
Powercfg -setactive scheme_current
```

To reduce the number of schedulable cores to 50 percent of the maximum count, set the **Processor Performance Core Parking Minimum Cores** parameter to 50 as follows:

```
Powercfg -setacvalueindex scheme_current sub_processor CPMAXCORES
50
Powercfg -setactive scheme_current
```

Processor Performance Core Parking Utility Distribution

Utility Distribution is an algorithmic optimization in Windows Server 2012 that is designed to improve power efficiency for some workloads. It tracks "unmovable" CPU activity (that is, DPCs, interrupts, or strictly affinitized threads), and it predicts the future work on each processor based on the assumption that any movable work can be distributed equally across all unparked cores. Utility Distribution is enabled by default for the Balanced power plans for some

processors. It can reduce processor power consumption by lowering the requested CPU frequencies of workloads that are in a reasonably steady state. However, Utility Distribution is not necessarily a good algorithmic choice for workloads that are subject to high activity bursts or for programs where the workload quickly and randomly shifts across processors. For such workloads, we recommend disabling Utility Distribution by using the following commands:

```
Powercfg -setacvalueindex scheme_current sub_processor
DISTRIBUTEUTIL 0
Powercfg -setactive scheme_current
```

Performance Tuning for the Networking Subsystem

Figure 3 shows the network architecture, which includes many components, interfaces, and protocols. The following sections discuss tuning guidelines for some of the components involved in server workloads.

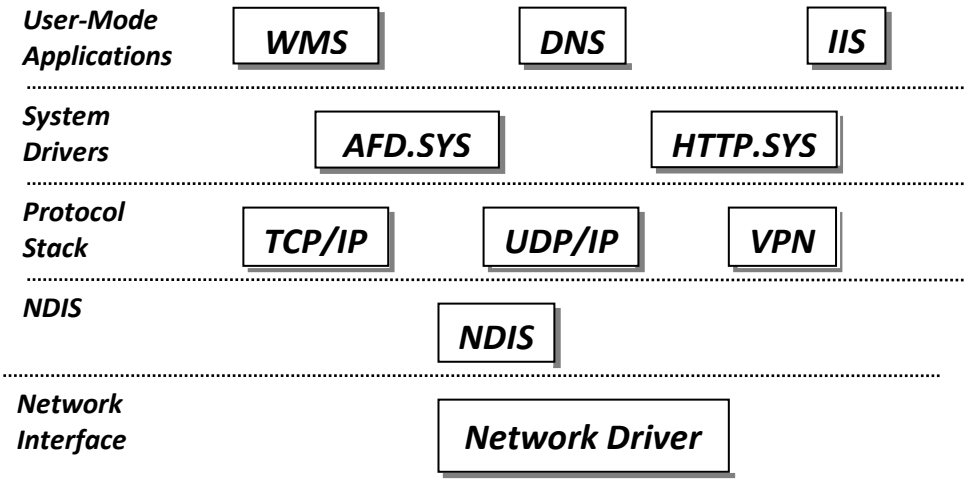

Figure 3. Network stack components

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 clients that use a transport data extension (TDX) or Winsock Kernel (WSK) interface to expose interfaces to user-mode applications. The WSK interface was introduced in Windows Server 2008 and Windows Vista, and it 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 explores some considerations for choosing network adapters.

Offload Capabilities

Offloading tasks can reduce CPU usage on the server, which improves the overall system performance. The network stack in Microsoft products can offload one or more tasks to a network adapter if you choose one that has the appropriate offload capabilities. Table 6 provides details about each offload capability.

Table 6. Offload Capabilities for Network Adapters

Offload type	Description
Checksum calculation	The network stack can offload the calculation and validation of Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) checksums on send and receive code paths. It can also offload the calculation and validation of IPv4 and IPv6 checksums on send and receive code paths.
IP security authentication and encryption	The TCP/IP transport layer can offload the calculation and validation of encrypted checksums for authentication headers and Encapsulating Security Payloads (ESPs). The TCP/IP transport layer can also offload the encryption and decryption of ESPs.
Segmentation of large TCP packets	The TCP/IP transport layer supports Large Send Offload v2 (LSOv2). With LSOv2, the TCP/IP transport layer can offload the segmentation of large TCP packets to the hardware.
Receive Segment Coalescing (RSC)	RSC is the ability to group packets together to minimize the header processing that is necessary for the host to perform. A maximum of 64 KB of received payload can be coalesced into a single larger packet for processing.
Receive-Side Scaling (RSS)	Receive-side scaling (RSS) is a network driver technology that enables the efficient distribution of network receive processing across multiple CPUs in multiprocessor systems.

Receive-Side Scaling (RSS)

Windows Server 2012, Windows Server 2008 R2, and Windows Server 2008 support Receive Side Scaling (RSS). A server may have multiple logical processors that share hardware resources (such as a physical core) and are treated as Simultaneous Multi-Threading (SMT) peers. Intel Hyper-Threading Technology is an example. RSS directs network processing to up to one logical processor per core. For example, given a server with Intel Hyper-Threading and 4 cores (8 logical processors), RSS will use no more than 4 logical processors for network processing.

RSS distributes incoming network I/O packets among logical processors so that packets that belong to the same TCP connection are processed on the same logical processor, which preserves ordering. RSS also load balances UDP unicast and multicast traffic from Windows Server 2012, and it routes related flows (as determined by hashing the source and destination addresses) to the same logical processor, thereby preserving the order of related arrivals. This helps improve scalability and performance for receive-intensive scenarios that have fewer network adapters than eligible logical processors.

Windows Server 2012 provides the following ways to tune RSS behavior:

- Windows PowerShell cmdlets: **Get-NetAdapterRSS**, **Set-NetAdapterRSS**, **Enable-NetAdapterRss**, **Disable-NetAdapterRss**

For more information, see Network Adapter Cmdlets in Windows PowerShell in the Windows Server Library.

These cmdlets allow you to see and modify RSS parameters per network adapter. Pass the cmdlet name to **Get-Help** for details.

- RSS Profiles: One of the parameters that is available is the RSS Profile, which is used to determine which logical processors are assigned to which network adapter. Possible profiles include:
    - **Closest**. Logical processor numbers near the network adapter's base RSS processor are preferred. Windows may rebalance logical processors dynamically based on load.
    - **ClosestStatic**. Logical processor numbers near the network adapter's base RSS processor are preferred. Windows will not rebalance logical processors dynamically based on load.
    - **NUMA**. Logical processor numbers will tend to be selected on different NUMA nodes to distribute the load. Windows may rebalance logical processors dynamically based on load.
    - **NUMAStatic**. This is the **default profile**. Logical processor numbers will tend to be selected on different NUMA nodes to distribute the load. Windows will not rebalance logical processors dynamically based on load.
    - **Conservative**: RSS uses as few processors as possible to sustain the load. This option helps reduce the number of interrupts.

Depending on the scenario and the workload characteristics, you can use the following Windows PowerShell cmdlet to choose how many logical processors can be used for RSS on a per-network adapter basis, the starting offset for the range of logical processors, and which node the network adapter allocates memory from:

- *** MaxProcessors**: Sets the maximum number of RSS processors to be used. This ensures that application traffic is bound to a maximum number of processors on a given interface.

```
set-netadapterRSS –Name "Ethernet" –MaxProcessors <value>
```

- *** BaseProcessorGroup**: Sets the base processor group of a NUMA node. This impacts the processor array that is used by RSS.

```
set-netadapterRSS –Name "Ethernet" –BaseProcessorGroup <value>
```

- *** MaxProcessorGroup**: Sets the Max processor group of a NUMA node. This impacts the processor array that is used by RSS. Setting this would restrict a maximum processor group so that load balancing is aligned within a k-group.

```
set-netadapterRSS –Name "Ethernet" –MaxProcessorGroup <value>
```

- *** BaseProcessorNumber**: Sets the base processor number of a NUMA node. This impacts the processor array that is used by RSS. This allows partitioning processors across network adapters. This is the first logical processor in the range of RSS processors that is assigned to each adapter.

```
set-netadapterRSS –Name "Ethernet" –BaseProcessorNumber <Byte
Value>
```

- *** NumaNode**: The NUMA node that each network adapter can allocate memory from. This can be within a k-group or from different k-groups.

```
set-netadapterRSS –Name "Ethernet" –NumaNodeID <value>
```

- *** NumberofReceiveQueues**: If your logical processors seem to be underutilized for receive traffic (for example, as viewed in Task Manager), you can try increasing the number of RSS queues from the default of 2 to the maximum that is supported by your network adapter. Your network adapter may have options to change the number of RSS queues as part of the driver.

```
set-netadapterRSS –Name “Ethernet” –NumberOfReceiveQueues <value>
```

For more information, see Scalable Networking: Eliminating the Receive Processing Bottleneck—Introducing RSS.

Understanding RSS Performance

Tuning RSS requires understanding the configuration and the load-balancing logic. To verify that the RSS settings have taken effect, **the Get-NetAdapterRss** Windows PowerShell cmdlet gives better insight.

```
PS C:\Users\Administrator> get-netadapterrss

    Name                            : testnic 2
    InterfaceDescription            : Broadcom BCM5708C NetXtreme II GigE (NDIS VBD Client)
    #66
    Enabled                         : True
    NumberOfReceiveQueues           : 2
    Profile                         : NUMAStatic
    BaseProcessor: [Group:Number]   : 0:0
    MaxProcessor: [Group:Number]    : 0:15
    MaxProcessors                   : 8

    IndirectionTable: [Group:Number]  :
                                      0:0   0:4   0:0   0:4   0:0   0:4   0:0   0:4
                                            …
                          (# indirection table entries are a power of 2 and based on # of
    processors)
                                            …
                                    0:0   0:4   0:0   0:4   0:0   0:4   0:0   0:4
```

In addition to echoing parameters that were set, the key aspect of the output is to understand indirection table output. The indirection table displays the hash table buckets that are used to distribute incoming traffic. In this example, the ***n:c*** notation designates the ***Numa K-Group*:*CPU index*** pair that is used to direct incoming traffic. We see exactly 2 unique entries (0:0 and 0:4), which represent k-group 0/cpu0 and k-group 0/cpu 4, respectively.

We further see only one k-group for this system (k-group 0) and a *n* (where n <= 128) indirection table entry. Because the number of receive queues is set to 2, only 2 processors (0:0, 0:4) are chosen even though maximum processors is set to 8. In effect, the indirection table is hashing incoming traffic to only use 2 CPUs out of the 8 that are available.

To fully utilize the CPUs, the number of RSS Receive Queues should be equal to or greater than Max Processors. For the previous example, the Receive Queue should be set to 8 or greater.

RSS and virtualization

RSS provides hashing and scalability to host interface only. RSS does not provide any interaction with virtual machines, instead users can configure VMQ in those scenarios.

RSS can be enabled for guest virtual machines in the case of SR-IOV because the virtual function driver supports RSS capability. In this case, the guest and the host will have the benefit of RSS. Note that the host does not get RSS capability because the virtual switch is enabled with SR-IOV.

LBFO and RSS

RSS can be enabled on a network adapter that is teamed. In this scenario, only the underlying physical network adapter can be configured to use RSS. A user cannot set RSS cmdlets on the teamed network adapter.

Receive-Segment Coalescing (RSC)

Receive Segment Coalescing (RSC) helps performance in Windows Server 2012 by reducing the number of IP headers that are processed for a given amount of received data. It should be used to help scale the performance of received data by grouping (or coalescing) the smaller packets into larger units. This approach can affect latency with benefits mostly seen in throughput gains. RSC is recommended to increase throughput for received heavy workloads. Consider deploying network adapters that support RSC. On these network adapters, ensure that RSC is on (this is the default setting), unless you have specific workloads (for example, low latency, low throughput networking) that show benefit from RSC being off.

In Windows Server 2012, the following Windows PowerShell cmdlets allow you to configure RSC capable network adapters: **Enable-NetAdapterRsc**, **Disable-NetRsc**, **Get-NetAdapterAdvancedProperty**, and **Set-NetAdapterAdvancedProperty**.

Understanding RSC diagnostics

RSC can be diagnosed through the following cmdlets.

```
PS C:\Users\Administrator> Get-NetAdapterRsc

Name                IPv4Enabled  IPv6Enabled  IPv4Operational IPv6Operational  IPv4FailureReason
IPv6Failure

Reason
----                -----------  -----------  --------------- ---------------  -----------------
----------
Ethernet                 True        False        True            False            NoFailure
NicProperties
```

The **Get** cmdlet shows whether RSC is enabled in the interface and if TCP enables RSC to be in operational state. The failure reason provides details about the failure to enable RSC on that interface.

In the previous scenario, IPv4 RSC is supported and operational in the interface. To understand diagnostic failures, one can see the coalesced bytes or exceptions caused. This gives an indication of the coalescing issues.

```
PS C:\Users\Administrator> $x = Get-NetAdapterStatistics “myAdapter”
PS C:\Users\Administrator> $x.rscstatistics

CoalescedBytes      : 0
CoalescedPackets    : 0
CoalescingEvents    : 0
CoalescingExceptions : 0
```

RSC and virtualization
RSC is only supported in the physical host when the host network adapter is not bound to the virtual switch. RSC is disabled by the operating system when host is bound to the virtual switch. Also, virtual machines do not get the benefit of RSC because virtual network adapters do not support RSC.

RSC can be enabled for a virtual machine when SR-IOV is enabled. In this case, virtual functions will support RSC capability; hence, virtual machines will also get the benefit of RSC.

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.

In Windows Server 2012, configuration has been simplified by the use of the following Windows PowerShell cmdlets:

- Get-NetAdapterAdvancedProperty
- SetNetAdapterAdvancedProperty
- Enable-NetAdapter
- Enable-NetAdapterBinding
- Enable-NetAdapterChecksumOffload
- Enable-NetAdapterLso
- Enable-NetAdapterIPSecOffload
- Enable-NetAdapterPowerManagemetn
- Enable-NetAdapterQos
- Enable-NetAdapterRDMA
- Enable-NetAdapter
- Enable-NetAdapterSriov

Message-Signaled Interrupts (MSI/MSI-X)

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

Interrupt Moderation

To control interrupt moderation, some network adapters 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 the 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 7 lists high-performance network adapter features that can improve performance in terms of throughput, latency, or scalability for some server roles.

Table 7. Benefits from Network Adapter Features for Different Server Roles

Server role	Checksum offload	Large Send Offload (LSO)	Receive-side scaling (RSS)	Receive Segment Coalescing (RSC)
File server	X	X	X	X
Web server	X	X	X	
Mail server (short-lived connections)	X		X	
Database server	X	X	X	
FTP server	X	X		X
Media server	X		X	X

Disclaimer The recommendations in Table 7 are intended to serve as guidance only for choosing the most suitable technology for specific server roles under a predetermined traffic pattern. The user's experience can be different, depending on workload characteristics and the hardware that is used.

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 with 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 Some network adapters require offload features to be independently enabled for send and receive paths.

Enabling RSS for Web Scenarios

RSS can improve web scalability and performance when there are fewer network adapters than logical processors on the server. When all the web traffic is going through the RSS-capable network adapters, incoming web requests from different connections can be simultaneously processed across different CPUs. It is important to note that due to the logic in RSS and HTTP for load distribution,

performance can be severely degraded if a non-RSS-capable network adapter accepts web traffic on a server that has one or more RSS-capable network adapters. We recommend that you use RSS-capable network adapters or disable RSS from the **Advanced Properties** tab. To determine whether a network adapter is RSS-capable, view the RSS information on the **Advanced Properties** tab for the device.

RSS Profiles and RSS Queues

RSS Profiles are new in Windows Server 2012. The default profile is **NUMA Static**, which changes the default behavior from previous versions of Windows. We suggest reviewing the available profiles and understanding when they are beneficial. If your logical processors seem to be underutilized for receive traffic, for example, as viewed in Task Manager, you can try increasing the number of RSS queues from the default of 2 to the maximum that is supported by your network adapter. Your network adapter may have options to change the number of RSS queues as part of the driver.

Increasing Network Adapter Resources

For network adapters that allow 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, it 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 it does expose buffer coalescing, increasing the number of coalesced buffers allows more buffers per send or receive, which improves performance.

Workload Specific Tuning

Tuning for Low Latency Packet Processing within the operating system

The network adapter has a number of options to optimize operating system-induced latency. This is the elapsed time between the network driver processing an incoming packet and the network driver sending the packet back. This time is usually measured in microseconds. For comparison, the transmission time for packet transmissions over long distances is usually measured in milliseconds (an order of magnitude larger). This tuning will not reduce the time a packet spends in transit.

Some tuning suggestions for microsecond-sensitive networks include:

- Set the computer BIOS to **High Performance**, with C-states disabled. However, note that this is system and BIOS dependent, and some systems will provide higher performance if the operating system controls

power management. You can check and adjust your power management settings from Control Panel or by using the **powercfg** command.

- Set the operating system power management profile to **High Performance System**. Note that this will not work properly if the system BIOS has been set to disable operating system control of power management.
- Enable Static Offloads, for example, UDP Checksums, TCP Checksums, and Send Large Offload (LSO)
- Enable RSS if the traffic is multi-streamed, such as high-volume multicast receive
- Disable the **Interrupt Moderation** setting for network card drivers that require the lowest possible latency. Remember, this can use more CPU time and it represents a tradeoff.
- Handle network adapter interrupts and DPCs on a core processor that shares CPU cache with the core that is being used by the program (user thread) that is handling the packet. CPU affinity tuning can be used to direct a process to certain logical processors in conjunction with RSS configuration to accomplish this. Using the same core for the interrupt, DPC, and user mode thread exhibits worse performance as load increases because the ISR, DPC, and thread contend for the use of the core.

System Management Interrupts

Many hardware systems use System Management Interrupts (SMI) for a variety of maintenance functions, including reporting of error correction code (ECC) memory errors, legacy USB compatibility, fan control, and BIOS controlled power management. The SMI is the highest priority interrupt on the system and places the CPU in a management mode, which preempts all other activity while it runs an interrupt service routine, typically contained in BIOS.

Unfortunately, this can result in latency spikes of 100 microseconds or more. If you need to achieve the lowest latency, you should request a BIOS version from your hardware provider that reduces SMIs to the lowest degree possible. These are frequently referred to as “low latency BIOS” or “SMI free BIOS.” In some cases, it is not possible for a hardware platform to eliminate SMI activity altogether because it is used to control essential functions (for example, cooling fans).

Note The operating system can exert no control over SMIs because the logical processor is running in a special maintenance mode, which prevents operating system intervention.

Tuning TCP

TCP Receive Window Auto-Tuning

Prior to Windows Server 2008, the network stack used a fixed-size receive-side window that limited the overall potential throughput for connections. One of the most significant changes to the TCP stack is TCP receive window auto-tuning. 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 it 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 can now fully use the network.

Windows Filtering Platform

The Windows Filtering Platform (WFP) that was introduced in Windows Vista and Windows Server 2008 provides APIs to non-Microsoft independent software vendors (ISVs) to create packet processing filters. Examples include firewall and antivirus software.

Note A poorly written WFP filter can significantly decrease a server's networking performance.

For more information, see Windows Filtering Platform in the Windows Dev Center.

TCP Parameters

The following registry keywords from Windows Server 2003 are no longer supported, and they are ignored in Windows Server 2012, Windows Server 2008 R2, and Windows Server 2008:

- **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.

Resource Utilization

- **IPv4, IPv6**
    - Datagrams Received/sec
    - Datagrams Sent/sec
- **TCPv4, TCPv6**
    - Segments Received/sec
    - Segments Sent/sec
    - Segments Retransmitted/sec
- **Network Interface(*), Network Adapter(*)**
    - Bytes Received/sec
    - Bytes Sent/sec

  - Packets Received/sec
  - Packets Sent/sec
  - 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.

- **Processor Information**
  - % Processor Time
  - Interrupts/sec
  - DPCs Queued/sec

    This counter is an average rate at which DPCs were added to the logical processor's DPC queue. Each logical 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.

Potential Network Problems

- **Network Interface(*), Network Adapter(*)**
  - Packets Received Discarded
  - Packets Received Errors
  - Packets Outbound Discarded
  - Packets Outbound Errors
- **WFPv4, WFPv6**
  - Packets Discarded/sec
- **UDPv4, UDPv6**
  - Datagrams Received Errors
- **TCPv4, TCPv6**
  - Connection Failures
  - Connections Reset
- **Network QoS Policy**
  - Packets dropped
  - Packets dropped/sec
- **Per Processor Network Interface Card Activity**
  - Low Resource Receive Indications/sec
  - Low Resource Received Packets/sec
- **Microsoft Winsock BSP**

- Dropped Datagrams
- Dropped Datagrams/sec
- Rejected Connections
- Rejected Connections/sec

Receive Side Coalescing (RSC) performance

- **Network Adapter(*)**
  - TCP Active RSC Connections
  - TCP RSC Average Packet Size
  - TCP RSC Coalesced Packets/sec
  - TCP RSC Exceptions/sec

Performance Tools for Network Workloads

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 with a multithreaded, asynchronous performance workload for measuring an achievable data transfer rate on an existing network setup.

For more information, see How to Use NTttcp to Test Network Performance in the Windows Dev Center.

When setting up NTttcp, consider 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 buffers that are posted would return before you need to use other buffers.
- It is best to bind each set of threads to a logical processor (the second delimited parameter in the **-m** option).
- Each thread creates a logical processor that connects to (listens) a different port.

Table 8. 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. Connects to a computer that uses IP 10.1.2.3. Posts two send-overlapped buffers. Default buffer size: 64 K. Default number of buffers to send: 20 K.

Syntax	Details
Example Syntax for a Receiver NTttcpr –m 1,0,10.1.2.3 –a 6 –fr	Single thread. Bound to CPU 0. Binds on local computer to IP 10.1.2.3. Posts six receive-overlapped buffers. Default buffer size: 64 KB. Default number of buffers to receive: 20 K. Posts full-length (64 K) receive buffers.

Note Make sure that you enable all offloading features on the network adapter.

TCP/IP Window Size

For 1 GB adapters, the settings shown in Table 8 should provide good throughput because NTttcp sets the default TCP window size to 64 K through a specific logical processor 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, the default TCP window size value for NTttcp 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 using it 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.

Server Performance Advisor 3.0

Microsoft Server Performance Advisor (SPA) 3.0 helps IT administrators collect metrics to identify, compare, and diagnose potential performance issues in a Windows Server 2012, Windows Server 2008 R2, or Windows Server 2008 deployment. SPA generates comprehensive diagnostic reports and charts, and it provides recommendations to help you quickly analyze issues and develop corrective actions.

For more information, see Server Performance Advisor 3.0 in the Windows Dev Center.

Performance Tuning for the Storage Subsystem

Decisions about how to design or configure storage software and hardware usually consider performance. Performance is improved or degraded as a result of trade-offs between multiple factors such as cost, reliability, availability, power, or ease-of-use. There are many components involved in handling storage requests as they work their way through the storage stack to the hardware, and trade-offs are made between such factors at each level. 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 4 shows the storage architecture, which includes many components in the driver stack.

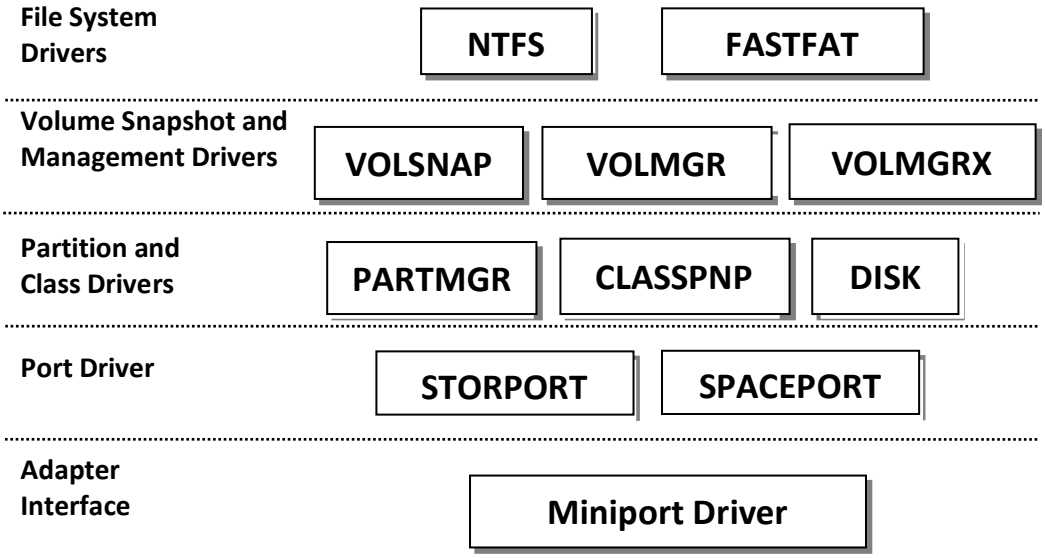

Figure 4. Storage driver stack

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:

- Understanding the characteristics of current and future storage workloads.
- Understanding that application behavior is essential for 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).

For example, 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.

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 vs. Write ratio
- Sequential vs. random access
- Typical request sizes
- Request concurrency, interarrival rates, and 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 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, redundancy, log files, and other factors. Table 9 describes some factors that affect server storage capacity.

Table 9. 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. A Windows Server installation can require even more space for temporary files.
Page 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 page file; otherwise, the page file might be limited because of space constraints (available disk capacity). The benefit of a page file of larger than 50 GB is unclear.
Memory dump	Depending on the memory dump file option that you have chosen, use an amount as large as the 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.
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 Solution

There are many considerations in choosing a storage solution that matches the expected workload. The range of storage solutions that are available to enterprises is immense.

Some administrators will choose to deploy a traditional storage array, backed by SAS or SATA hard drives and directly attached or accessed through a separately managed Fibre Channel or iSCSI fabric. The storage array typically manages the redundancy and performance characteristics internally. Figure 5 illustrates some storage deployment models that are available in Windows Server 2012.

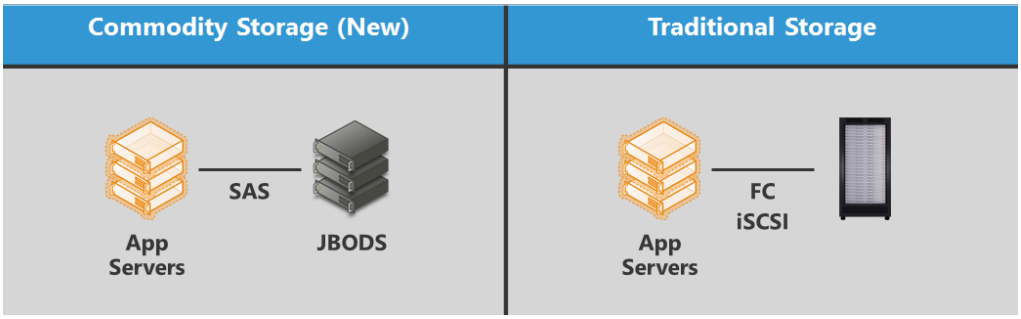

Figure 5. Storage deployment models

Alternatively, Windows Server 2012 introduces a new technology called Storage Spaces, which provides platform storage virtualization. This enables customers to deploy storage solutions that are cost-efficient, highly-available, resilient, and performant by using commodity SAS/SATA hard drives and JBOD enclosures. For more information, see Storage Spaces later in this guide.

Table 10 describes some of the options and considerations for a traditional storage array solution.

Table 10. Options for Storage Array Selection

Option	Description
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 benefit associated with SAS is typically attributed to performance.
Hardware RAID capabilities	For maximum performance and reliability, the enterprise storage controllers should offer resiliency capabilities. RAID levels 0, 1, 0+1, 5, and 6 are described in Table 11.
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. The more important values are the achievable bandwidths for the specific workloads to be run on servers that access the storage.

Hardware Array Capabilities

Most storage solutions provide some resiliency and performance-enhancing capabilities. In particular, storage arrays may contain varying types and capacities of caches that can serve to boost performance by servicing reads and writes at memory speeds rather than storage speeds. In some cases, the addition of noninterruptible power supplies or batteries are required to keep the additional performance from coming at a reliability cost.

A hardware-managed array is presented to the operating system as a single drive, which can be termed a logical unit number (LUN), virtual disk, or any number of other names for a single contiguously addressed block storage device.

Table 11 lists some common options for the storage arrays.

Table 11. Storage Array Performance and Resiliency Options (RAID levels)

Option	Description
Just a bunch of disks (JBOD)	This is not a RAID level. It provides a baseline for measuring the performance, reliability, availability, cost, capacity, and energy consumption of various resiliency and performance configurations. Individual disks are referenced separately, not as a combined entity. In some scenarios, a JBOD configuration 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 a JBOD configuration to a striped data layout. A JBOD configuration 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, a JBOD configuration provides a level of data isolation that can be interpreted as offering greater reliability than striped configurations.
Spanning	This is not a RAID level. It is the 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 a JBOD configuration.

Option	Description
Striping (RAID 0)	Striping is a data layout scheme in which sequential logical blocks of a specified size (the stripe unit) are distributed in a circular fashion across multiple disks. It presents a combined logical disk that stripes disk accesses over a set of physical disks. The overall storage load is balanced across all physical drives. For most workloads, a striped data layout provides better performance than a JBOD configuration 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, striping does not provide data recovery mechanisms such as those provided in the other resiliency schemes. Also, the loss of any disk results in data loss on a larger scale than a JBOD configuration 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
Mirroring (RAID 1)	Mirroring is a data layout scheme in which each logical block exists on multiple physical disks (typically two, but sometimes three in mission-critical environments). It presents a virtual disk that consists of a set of two or more mirrored disks. Mirroring often has worse bandwidth and latency for write operations when compared to striping or JBOD. This is because data from each write request must be written to a pair of 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. In more complex implementations, write latencies may be reduced by write logging or battery-backed write caching, or by relaxing the requirement for dual write completions before returning the I/O completion notification. Mirroring has the potential to provide faster read operations than striping because it can (with a sufficiently intelligent controller) read from the least busy physical disk of the mirrored pair, or the disk that will experience the shortest mechanical positioning delays. Mirroring is the most expensive resiliency scheme in terms of physical disks because half (or more) of the disk capacity stores redundant data copies. A mirrored array 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. Mirroring has greater power requirements than a non-mirrored storage configuration. It doubles the number of disks; therefore, it doubles the required amount of idle power. Also, mirroring performs duplicate write operations that require twice the power of non-mirrored write operations. In the simplest implementations, mirroring is the fastest of the resiliency schemes in terms of recovery time after a physical disk failure. Only a single disk (the other part of the broken mirror pair) must participate in bringing up the replacement drive. The second disk is typically still available to service data requests throughout the rebuilding process. In more complex implementations, multiple drives may participate in the recovery phase to help spread out the load for the duration of the rebuild.
Striped mirroring (RAID 0+1 or 10)	The combination of striping and mirroring is intended to provide the performance benefits of striping and the redundancy benefits of mirroring. The cost and power characteristics are similar to those of mirroring.

Option	Description
Rotated parity or parity disks (RAID 5)	An array with rotated parity (denoted as RAID 5 for expediency) presents a logical disk that is composed of multiple physical disks that have data striped across the disks in sequential blocks (stripe units) in a manner similar to simple striping (RAID 0). However, the underlying physical disks have parity information spread throughout the disk array, as in the example shown in Figure 6. For read requests, RAID 5 has characteristics that resemble those of striping. However, small RAID 5 writes are much slower than those of other resiliency schemes because each parity block that corresponds to the modified data block(s) must also be updated. This process requires three additional disk requests in the simplest implementation, regardless of the size of the array. Each small write requires two reads (old data and old parity) and two writes (new data and new parity). Because multiple physical disk requests are generated for every logical write, bandwidth is reduced by up to 75 percent. RAID 5 arrays provide data recovery capabilities because data can be reconstructed from the parity. Such arrays can survive the loss of any one physical disk, as opposed to mirroring, which can survive the loss of multiple disks if the mirrored pair (or triplet) is not lost. RAID 5 requires additional time to recover from a lost physical disk compared to mirroring because the data and parity from the failed disk can be re-created only by reading all the other disks in their entirety. In a basic implementation, performance during the rebuilding period is severely reduced due to the rebuilding traffic and because the reads and writes that target the data that was stored on the failed disk must read all the disks (an entire “stripe”) to re-create the missing data. More complex implementations incorporating multiple arrays may take advantage of more parallelism from other disks to help speed up recovery time. RAID 5 is more cost efficient than mirroring 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 the basic implementation) 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 four times that of a JBOD configuration or two times that of a mirrored configuration.

Option	Description
Option	Description
(RAID 5 continued)	(Depending on the workload, there may be only two seeks, not four, that require moving the disk actuator.) Thus, although unlikely in most configurations, RAID 5 might have greater energy consumption. This might happen if a heavy workload is being serviced by a small array or an array of disks with idle power that is significantly lower than their active power.
Double rotated parity, or double parity disks (RAID 6)	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. More complex implementations may take advantage of algorithmic or hardware optimizations to reduce the overhead that is associated with maintaining the extra redundant data. As far as power and performance, the same general statements can be made for RAID 6 that were made for RAID 5, but to a larger magnitude.

Rotated redundancy schemes (such as RAID 5 and RAID 6) are the most difficult to understand and plan for. Figure 6 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.

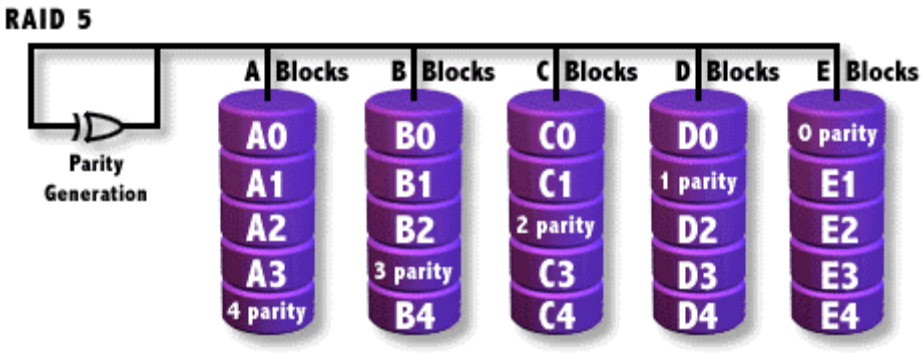

Figure 6. RAID 5 overview

Choosing the Right Resiliency Scheme

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

- Performance
- Reliability
- Availability
- Cost
- Capacity
- Power

To determine the best array configuration 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, availability, and reliability that your organization requires. Table 12 describes common configurations and their relative performance, reliability, availability, cost, capacity, and energy consumption.

Table 12. 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
Striping (RAID 0) **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
Mirroring (RAID 1) **Requirements:** • Two-disk minimum	**Pros:** • Two data sources for every read request (up to 100% performance improvement) **Cons:** • Writes must update all mirrors (simplest implementa-tion)	**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 twice the power

Configuration	Performance	Reliability	Availability	Cost, capacity, and power
Striped mirroring (RAID 0+1 or 10) **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 twice the power

Configuration	Performance	Reliability	Availability	Cost, capacity, and power
Rotated Parity or Parity Disks (RAID 5) **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; potential major slowdown • Danger of data in invalid state after power loss and recovery if not carefully implemented	**Pros:** • Single loss survivable; active 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 four times the power for write requests (excluding the idle power)

Configuration	Performance	Reliability	Availability	Cost, capacity, and power
Mutiple Rotated parity or double parity disks (RAID 6) **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; potential for major slowdown • Danger of data in invalid state after power loss and recovery if not carefully implemented	**Pros:** • Single loss survivable; active 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 six times the power for write requests (excluding the idle power)

The following are sample uses for various RAID levels:

- JBOD configuration: Concurrent video streaming
- Striping (RAID 0): Temporary or reconstructable data, workloads that can develop hot spots in the data, and workloads with high degrees of unrelated concurrency
- Mirroring (RAID 1): Database logs, critical data, and concurrent sequential streams
- Striped mirroring (RAID 0+1): A general purpose combination of performance and reliability for critical data, workloads with hot spots, and high-concurrency workloads
- Rotated parity or parity disks (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
- Multiple rotated parity or double parity disks (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, striped mirroring is usually a better solution than only mirroring.

To determine the number of physical disks that you should include in an array, 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 between data isolation (small arrays) and better load balancing (large arrays). For mirrored arrays, the trade-off is between better cost per capacity (for basic mirrors, which is a depth of two physical disks) and the ability to withstand multiple disk failures (for depths of three or four physical disks). Read and Write performance issues can also affect mirrored array size. For arrays with rotated parity (RAID 5), the trade-off is between better data isolation and mean time between failures (MTBF) for small arrays, versus better cost, capacity, and 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 3.5-inch storage enclosures. Smaller 2.5-inch disks can be racked more densely; therefore, they might require being divided 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

Hardware-managed arrays allow stripe unit sizes ranging from 4 KB to more than 1 MB. The 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 a JBOD configuration 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 a JBOD configuration. 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 five times or ten times, especially if the requests are aligned on some boundary (for example, 4 KB or 8 KB).
- If requests are large, and the average or 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). Note that this assumes some consistency in alignment between the request boundaries and the stripe unit boundaries.
- Optimal stripe unit size increases with concurrency 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 can be dedicated to handling the updates to the log files. Placing the page 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 that is utilizing hard disks usually uses the outermost tracks of the underlying disks, and therefore it provides better performance. Obviously, this guidance does not apply to solid-state storage.

Choosing and Designing Storage Tiers

With the cost of solid state devices dropping, it is important to consider including multiple tiers of devices into a storage deployment to achieve better balance between performance, cost, and energy consumption. Traditional storage arrays offer the ability to aggregate and tier heterogenous storage, but Storage Spaces provides a more robust implementation.

Storage Spaces

Windows Server 2012 introduces a new technology called Storage Spaces, which provides flexible configuration options and supports a range of hardware choices, while providing the user with similar sophisticated storage features that were previously available only with more traditional storage solutions.

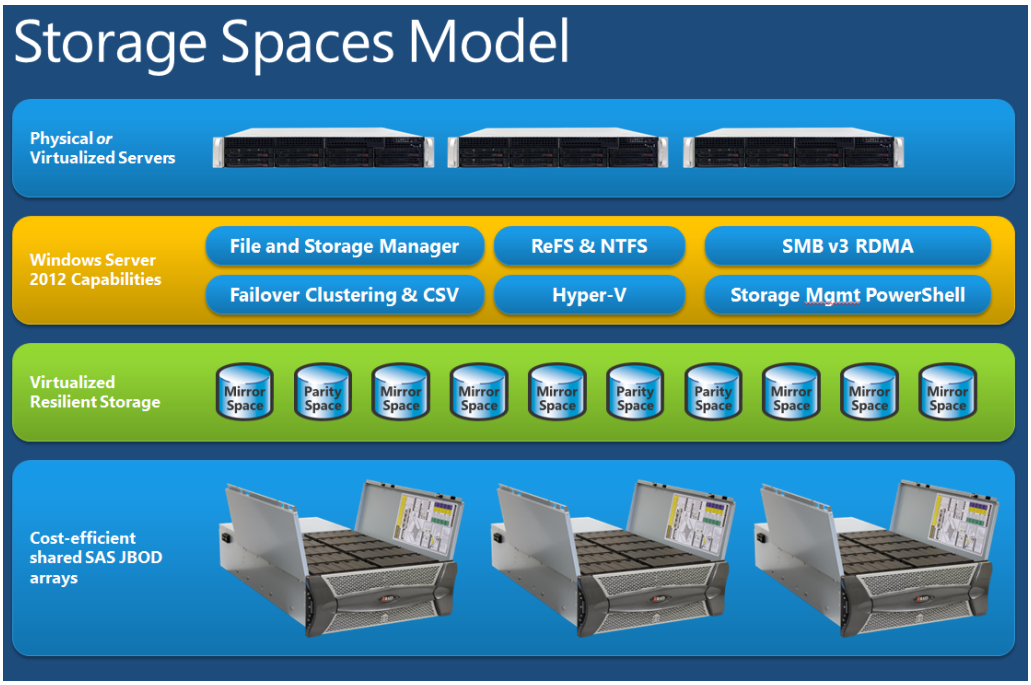

Figure 7. Storage Spaces deployment model

Storage Spaces Configuration Options

Storage Spaces provides multiple configuration options that can enhance performance when a storage space is created. These include the stripe unit size and number of columns to utilize in a storage space.

Storage Spaces allows a stripe unit size of 16 KB to 16 MB, with the default being 256 KB. Storage Spaces also provides tiering support. It gives administrators control over data placement or tiering at the Space granularity. If a workload requires high-performance storage, the space can be allocated from solid-state drives or 15 K RPM disks. Similarly, if the workload will have less intensive access (such as archiving), the space can be allocated from near-line disks. This level of tiering allows organizations to easily lower costs by only purchasing hardware that is necessary for the workload.

A storage space can also be configured at creation time with the number of columns that constitute the space. The number of columns corresponds to the number of physical disks among which to stripe data. For a mirrored storage space, the number of physical disks to use is equal to the number of copies (2 or 3) multiplied by the number of columns specified. For example, 4 columns on a two-way mirror will utilize 8 physical disks. By default, Storage Spaces will try to use 4 columns per copy in a two-way mirrored space if sufficient disks are available, but it will reduce that number as necessary. The number of columns can range from 1 column to 128 per copy. The following Windows PowerShell script can be used to configure stripe size and number of columns when creating a storage space:

```
New-Virtualdisk –Interleave (XKB) –NumberofColumns Y
```

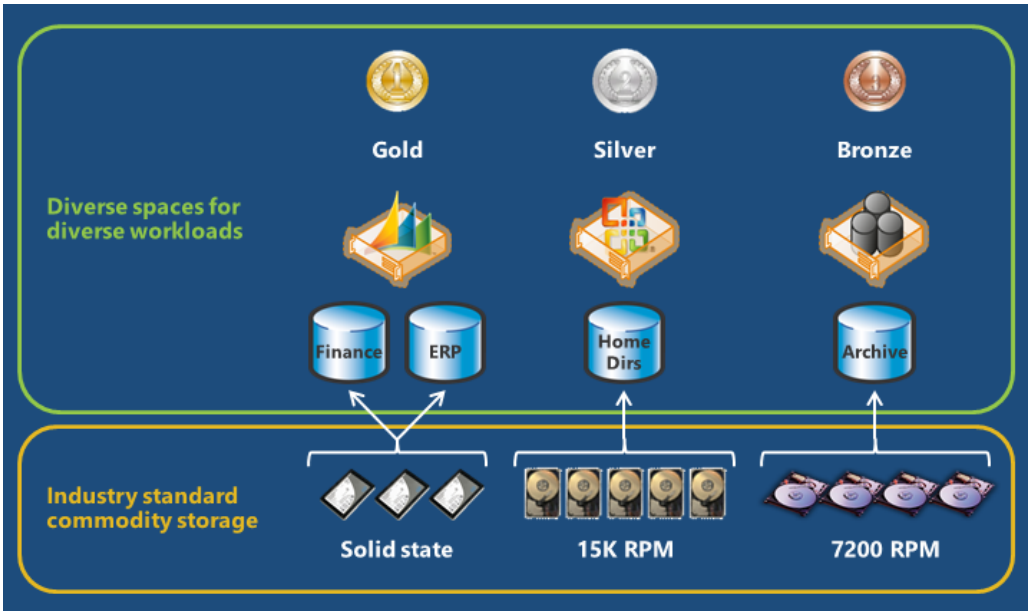

Figure 8. Tiered Storage Spaces on different types of media

Deployment Elements: A New Unit of Scale

Storage Spaces also enables a new unit of scale in Windows Server 2012, called Deployment Elements. Combining the virtualization capabilities of Storage Spaces, failover clustering, and cluster shared volumes (CSV), enterprises can deploy a resilient, performant, and elastic solution that can scale easily and quickly by using simple, cost-effective building blocks.

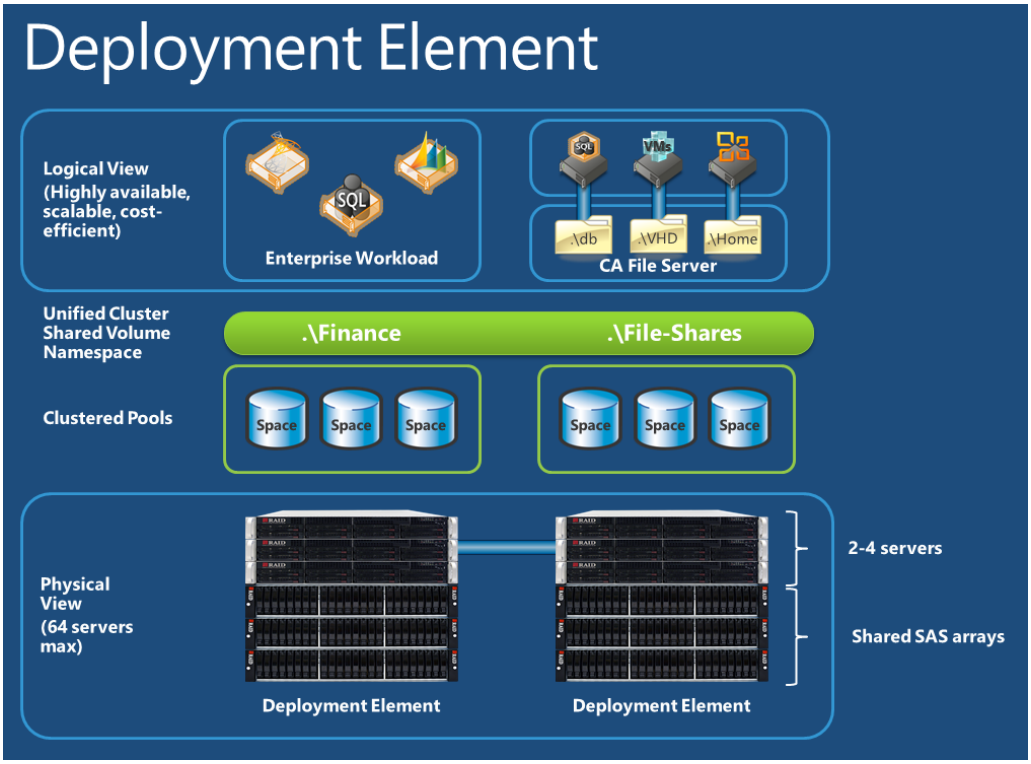

Figure 9. Deployment elements

Deployment Elements consist of 2 to 4 servers attached to shared SAS JBOD configurations. By connecting each deployment element together using a high-speed, low-latency network, Deployment Elements can be combined into a CSV cluster consisting of up to 64 servers, supporting up to 4,000 virtual machines.

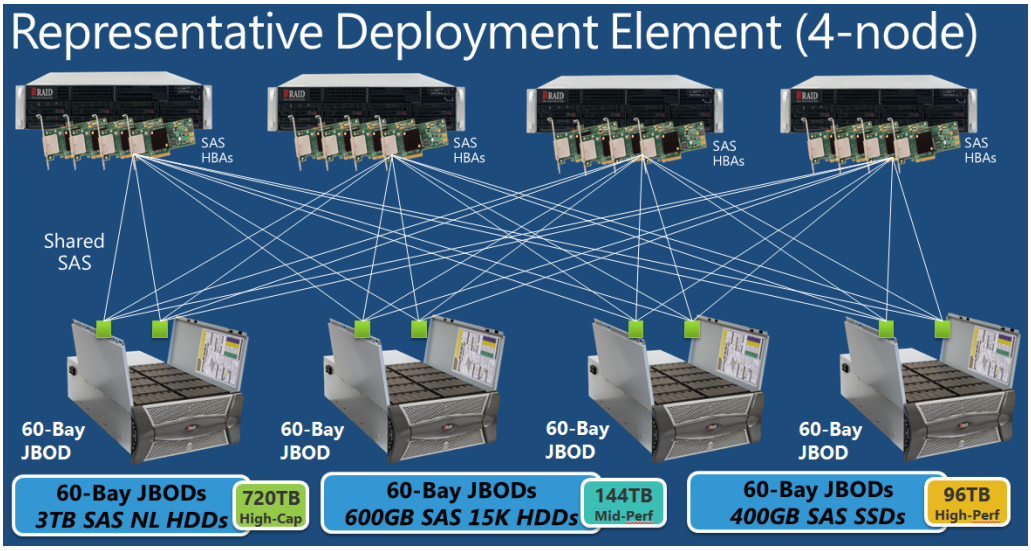

Figure 10. Components and configurations for deployment elements

Storage Spaces provides similar types of resiliency options as standalone array solutions, as described in the following table.

Option	Description
Striped spaces	Striping is a data layout scheme in which sequential logical blocks of a specified size (the stripe unit) are laid out in a circular fashion across multiple disks. It presents a combined logical disk that stripes access over a set of physical disks. The overall storage load is balanced across all physical drives.
Mirrored spaces	Mirroring is a data layout scheme in which each logical block exists on multiple physical disks. It presents a logical virtual disk that consists of a set of two or more mirrored disks. At a minimum, mirrored spaces can be configured to be resilient to at least one (two-way mirror) or two (three-way mirror) concurrent physical disk failures. A mirrored space may survive the loss of even more disks if all copies of a stripe unit are not simultaneously lost.
Parity spaces	Parity spaces present a logical disk that is composed of multiple physical disks that have data striped across the disks in stripe units. However, the underlying physical disks have parity information spread throughout the disk array. Parity spaces tends to have lower write performance than mirrored spaces, because each parity block that corresponds to the modified data block must also be updated. This process requires additional disk requests; however, parity spaces tend to be more space-efficient than mirrored spaces. Parity spaces provide data recovery capabilities because data can be reconstructed from the parity. Parity spaces can survive the loss of any one physical disk. Parity is more cost efficient than mirroring because it requires only one additional disk per virtual disk, instead of double (or triple) the total number of disks in an array as with mirroring.

Storage-Related Parameters and Performance Counters

This section describes performance counters that you can use for workload characterization and capacity planning and to identify potential bottlenecks.

I/O Priorities

Windows Server 2012, Windows Server 2008 R2, and Windows Server 2008 can specify an internal priority level on individual I/Os. The Windows Server operating system primarily uses this ability to lower the priority of 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 SQL Server 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
```

Logical Disks and Physical Disks

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 few percent CPU overhead penalty.

The same counters are valuable in the logical disk and the physical disk counter objects. Logical disk statistics are tracked at the volume level, and physical disk statistics are tracked by the partition manager.

The following counters are exposed through volume and partition managers:

- **% Idle Time**

This counter is 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 0% idle, when in fact the 100-disk array could be up to 99% idle with only a single request outstanding.

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

The % Disk Time counter is nothing more than the Avg. Disk Queue Length counter multiplied by 100. It is the same value displayed in a different scale. If the Avg. Disk Queue Length is equal to 1, the % Disk Time will equal 100. If the Avg. Disk Queue Length is 0.37, then the % Disk Time will be 37. So if the Avg. Disk Queue length value is greater than 1, the % Disk Time will be greater than 100%. The same logic applies to the % Disk Read Time and % Disk Write Time counters. Their data comes from the Avg. Disk Read Queue Length and Avg. Disk Write Queue Length counters, respectively.

- **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 peak loads and workloads that contain significant bursts. These counters represent the number of requests that are active below the driver that takes the statistics. This means that the requests are not necessarily queued; they could actually be in service or completed and on the way back up the path. Possible active locations include the following:

- Waiting in an ATA port 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

It is important to note that these values are not strictly accurate; rather, they are derived values. Avg. Disk Queue Length is equal to (Disk Transfers/sec) * (Disk sec/Transfer). This is based on Little's Law from the mathematical theory of queues. The same derivation is performed for the Read and Write versions of this counter. The main concern for interpreting these values is that they make the assumption that the number of outstanding requests is the same at the start and the end of each sampling interval.

For guidelines, see Queue Lengths later in this guide.

- **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. If possible, you should observe individual or sub-workloads separately. You cannot differentiate multimodal distributions by using Perfmon if the requests are consistently interspersed.

For guidelines, see Response Times later in this guide.

- **Current Disk Queue Length**

This counter instantly measures the number of active requests; therefore, it is subject to extreme variance. This counter is of limited use except to check for the existence of many short bursts of activity or to validate specific instances of the Average Disk Queue Length counter values. (As described earlier, these values are derived rather than measured, and they rely on the number of outstanding requests being equal at the start and end of each sampling interval.)

- **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 subworkloads 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 subworkloads separately.

- **Split I/O / second**

This counter measures the rate of high-level I/Os split into multiple low-level I/Os due to file fragmentation. It is useful only if the value is not statistically significant in comparison to the disk I/O rate. 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 standard stacked drivers scheme in Windows is circumvented for a controller, "monolithic" drivers can assume the role of partition manager or volume manager. If so, the monolithic driver writer must supply the counters listed earlier through the Windows Management Instrumentation (WMI) interface, or the counters will not be available.

Processor Information

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

If the interrupt time and deferred procedure call (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 processes 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

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

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

Enable 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. 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, NTFS, 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 write-through request 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, SATA and 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 write caching or use SAS/SCSI/FC hardware to correctly maintain write-through semantics. In Windows Server 2012, NTFS exclusively uses cache flushes to protect its metadata.

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 (such as an uninterruptible power supply, or UPS) for all hardware write caches along the I/O path, you do not need to worry about write-through flags and cache flushes. By definition, any dirty data that resides in a power-protected write cache is safe, and it appears to have occurred “in-order” to the software. 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 the Windows Server operating system earlier than 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 the partition was created.

In Windows Server 2012, Windows Server 2008 R2, and Windows Server 2008, 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 an issue because small requests rarely cross large stripe unit boundaries.

Note Windows Server 2012, Windows Server 2008 R2, and Windows Server 2008 default to a 64 KB offset if the disk is smaller than 4 GB.

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

Solid-State Drives

Previously, the cost of large quantities of nonvolatile memory used as block storage was prohibitive for most server configurations. Exceptions included aerospace or military applications in which the high shock and vibration tolerance of solid-state storage is highly desirable.

As the cost of flash memory continues to decrease, new hierarchies of storage become feasible, where nonvolatile memory (NVM) is used to improve the storage subsystem response time on servers. The typical vehicle for incorporating NVM in a server is the solid-state drive (SSD). One cost-effective strategy is to place only the hottest data of a workload into nonvolatile memory. In Windows Server 2012, as in previous versions of Windows Server, partitioning can be performed only by applications that store data on an SSD; Windows operating systems do not try to dynamically determine what data should optimally be stored on SSDs versus rotating media. There are emerging non-Microsoft storage management products, in software and hardware+software combinations, which allow data placement and migration to be performed without human intervention. This is called *tiering*.

Choosing suitable SSDs to complement the hierarchy is a tradeoff between the cost of the additional storage layer, endurance of the media (with associated servicing costs), improved responsiveness of the system, and improved energy efficiency. Current server SSDs are designed around one or more types of flash memory. Some important flash memory characteristics include:

- Cost per capacity is orders of magnitude higher than for the rotational media, while SSD access times are 2-3 orders of magnitude better for random I/Os.
- Read latency is substantially higher than write latency.
- Media lifetime is limited by the number of erase and write operations.
- SSDs are inherently parallel devices, and they operate better with longer I/O queues.
- Power consumption of an SSD may spike with load, especially for random workloads. Contrary to many claims, an SSD's power efficiency needs to be evaluated relative to the load, and it is not always superior to that of a hard disk drive.

SSDs are commonly grouped by their designers along the axes of performance (especially latency), price per capacity, and endurance. Server grade SSDs are designed to withstand substantially higher I/O pressure by overprovisioning flash media, which increases cost per capacity. Therefore, server SSDs can support much higher random write IOPS rates, while being marginally better at read rates than desktop SSDs. Throughput characteristics of SSDs are improving with time, with the rapid rate of changes currently in the industry making it crucial to consult up-to-date performance, power, and reliability comparison data.

Due to superior media performance, the interconnect to the SSD plays an important role. While most of the client and entry-level server designs settled on using SATA interconnect (SATA version 3.0 becoming dominant recently) as the most compatible, better operation characteristics may be achieved by using

newer, PCIe-based interconnect schemes. PCIe SSDs bring substantial advantages, like more scalable throughput, lower latency, and, in some cases, improved energy efficiency relative to throughput. One of the drawbacks of this technology is its relative immaturity, creating vertically integrated silos based on proprietary driver stacks. To evaluate PCIe SSDs, it is important to test complete combinations of the storage device, server platform, and software layer, including the impact of reliability and maintainability on the total cost of ownership (TCO).

Trim and Unmap Capabilities

Windows Server 2012 provides storage allocation transparency to storage devices, including traditional storage arrays, hard disk drives, SSDs, and Storage Spaces. Although this transparency is critical for reducing capacity utilization in thinly provisioned environments, it can also have an important impact on performance and power consumption. Providing greater visibility into what's allocated for storage devices from a holistic view enables the devices to make better resource utilization decisions that result in higher performance. In addition, because the storage footprint for a deployment is reduced, power consumption can be reduced also.

The storage stack in Windows Server 2012 will issue standards-based "trims" or "unmaps" for any space that becomes unallocated, even within virtualized environments. Further, the new Storage Optimizer runs automatically to help further reduce the physical footprint of the data by consolidating data from sparsely populated "slabs" to more densely populated slabs.

Together, these technologies can help improve performance and power consumption. Administrators should investigate whether their storage devices support **Trim** or **Unmap** commands to ensure an efficient and performant deployment.

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 large or sequential requests, which 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 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 simple implementations of mirrored arrays, a write completion must 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. Although writes for mirrored arrays generally should not take twice the time to complete, they are typically slower than a JBOD configuration. Reads can experience a performance increase if the array controller is dynamically load balancing or factoring in spatial locality. More complex implementations may use logging or battery-backed write caching or other means to improve write latencies.

For RAID 5 arrays (rotated parity), small writes become four separate requests in the basic read-modify-write scenario. In the best case, this is approximately the equivalent of two mirrored reads plus a full rotation of the two disks that hold the data and corresponding parity, if you assume that the read/write pairs continue in parallel.

You must consider the performance effect of redundancy 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 virtual disk, 200 requests per second for a RAID 5 array or parity virtual disk, or even more than 200 requests per second if the requests are split across stripe units.

Use the following 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 subsystem 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. In this guideline, disk requests that 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 activity often occurs in bursts, 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 workloads that contain significant bursts. Conversely, a longer queue provides more opportunity for disk request schedulers to reduce positioning delays or to optimize for full stripe RAID 5 writes or mirrored read selection.

Because hardware has an increased capability to queue requests—either through multiple queuing agents along the path or through 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 array or virtual disk should be less than 0.6 per physical disk and the average queue length above a RAID 5 or RAID 6 array or a parity virtual disk 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 array or virtual disk should be less than 1.5 per physical disk, and the average queue length above a RAID 5 array or parity virtual disk 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 information about storage performance in Windows Server 2012, 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 network adapter.
- The file system becomes a bottleneck because of an inefficient disk subsystem or storage adapter.

Operating System Practices

If possible, perform a clean installation of the operating system software. Upgrading the software 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 you must keep it, 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 page file, operating system, web data, ASP template cache, and the Internet Information Services (IIS) log on separate physical disks if possible.

To reduce contention for system resources, install SQL Server 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 8.0

Internet Information Services (IIS) 8.0 is the version that ships as part of Windows Server 2012 . It uses a process model similar to that of IIS 6.0. A kernel-mode web driver (http.sys) receives and routes HTTP requests, and it can 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 8.0 process relies on the kernel-mode web driver, http.sys. Http.sys is responsible for connection management and request handling. The request can be served from the http.sys cache or passed to a worker process for further

handling (see Figure 11). 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 11 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-mode cache. The static file handler in IIS 8.0 automatically caches frequently requested files in http.sys.

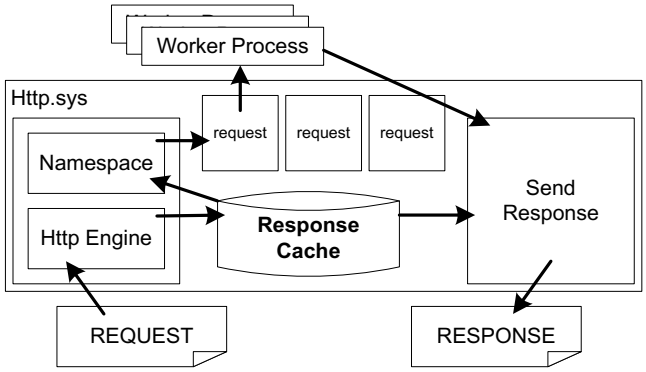

Figure 11. Request handling in IIS 8.0

Because a web server has kernel-mode and user-mode components, both components must be tuned for optimal performance. Therefore, tuning IIS 8.0 for a specific workload includes configuring the following:

- Http.sys (the kernel-mode web driver) and the associated kernel-mode cache
- Worker processes and user-mode IIS, including the 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 8.0.

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 registry entry:

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

Note 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 the kernel mode, which significantly lowers the CPU cost of handling the request. However, the kernel-mode cache of IIS 8.0 is based on physical memory, 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 or not the entry is being used. 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 caching. For most workloads, the cache should remain enabled. Consider disabling the cache if you expect a very low response and fragment caching.

- **UriMaxCacheMegabyteCount.** Default value: 0.

A nonzero value specifies the maximum memory that is available to the kernel-mode cache. The default value, 0, enables the system to automatically adjust how much memory is available to the cache.
Note 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 too low causes more frequent scavenger scans, and it can result in too many flushes and cache churn.

Request and Connection Management Settings

In Windows Server 2012, 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 2012:

- **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 8.0 worker process behavior. Most of these settings can be found in the following XML configuration file:

%SystemRoot%\system32\inetsrv\config\applicationHost.config

Use Appcmd.exe or the IIS 8.0 Management Console to change them. Most settings are automatically detected, and they do not require a restart of the IIS 8.0 worker processes or web application server.

User-Mode Cache Behavior Settings

This section describes the settings that affect caching behavior in IIS 8.0. The user-mode cache is implemented as a module that listens to the global caching events that are raised by the integrated pipeline. 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 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	Caches files up to the specified size. The actual value depends on the number and size of the largest files in the data set 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 2012, IIS 8.0 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 8.0 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 8.0 automatically disables compression if steady-state CPU increases above the Disable threshold. Compression is 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 can occupy. Compressed files are stored in the compression directory that is specified by the *directory* attribute.	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

Note For IIS 8.0 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 searches the list of allowed default documents for each name 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 websites 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.

You can selectively set the default document behavior on particular URLs by customizing the 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 enables default documents only where they are necessary and sets 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 kernel-mode web driver, http.sys, 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 computer 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). By imposing a simple limitation, which allows configuration only in virtual directories, IIS 8.0 can 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 of websites that have a very large set of randomly accessed static content.	True

Managing IIS 8.0 Modules

IIS 8.0 has been factored into multiple, user-extensible modules to support a modular structure. This factorization has a small cost. For each module the integrated pipeline must call 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 website.

A web server that is tuned 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 for ASP.NET, see 10 Tips for Writing High-Performance Web Applications.

system.webServer/asp/cache

Attribute	Description	Default
diskTemplateCacheDirectory	Contains the name of the directory that ASP uses to store compiled templates when the in-memory cache overflows. Recommendation: If possible, set to a directory that is not heavily used, for example, a drive that is not shared with the operating system, IIS log, or other frequently accessed content. The default directory is: %SystemDrive%\inetpub\temp \ASP Compiled Templates	See Description column
maxDiskTemplateCacheFiles	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	Specifies the maximum number of scripting engines that ASP pages will keep cached in memory. Recommendation: Set to as many script engines as the memory limit allows.	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, which can cause errors when it is serving requests or cause under-usage of CPU resources.	25

system.webServer/asp/comPlus

Attribute	Description	Default
executeInMta	Set to True if errors or failures are detected while IIS is serving 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\4.0.30319.0\Parameters
```

The following setting is useful to fully use resources on a system:

- **MaxConcurrentRequestPerCpu.** Default value: 5000.

  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 due to exceeding queue limits under a high load when the default setting is used.

Worker Process and Recycling Options

In the IIS Administrator user interface, the options for recycling IIS worker processes provide practical solutions to acute situations or events without requiring intervention or resetting a service or computer. 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 recycling 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 **recycling/periodicRestart** element is per-application, which means that each attribute in the following table is 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 computers that have a small, 2 GB address space. It can help avoid failed requests due to out-of-memory errors.	0
privateMemory	Enable process recycling if private memory allocations	0

Attribute	Description	Default
	exceed a specified limit in kilobytes.	
requests	Enable process recycling after a certain number of requests.	0
time	Enable process recycling after a specified time period.	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 (which involves a full handshake). Reconnection, encryption, and decryption also add to the cost. For better SSL performance, do the following:

- Enable HTTP keep-alives for SSL sessions. This eliminates the session establishment costs.
- Reuse sessions when appropriate, especially with non-keep-alive traffic.

Notes

- Larger keys provide more security, but they also use more CPU time.
- All components might not need to be encrypted. However, mixing plain HTTP and HTTPS might result in a pop-up warning that not all content on the page is secure.

ISAPI

No special tuning parameters are needed for the Internet Server Application Programming Interface (ISAPI) applications. If you write a private ISAPI extension, make sure that you code it efficiently for performance and resource use. For more information, see Other Issues that Affect IIS Performance later in this guide.

Managed Code Tuning Guidelines

The integrated pipeline model in IIS 8.0 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 they 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, custom modules add significant CPU cost due to marshaling data between native and managed code. If possible, you should implement global modules in native (C/C++) code.

Before you deploy an ASP.NET website, make sure that you compile all scripts. You can do this by calling one .NET script in each directory. Reset IIS after the compilation is complete. Recompile the scripts after you make changes to machine.config, web.config, or any .aspx scripts.

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 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 10 Tips for Writing High-Performance Web Applications.

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. 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 to serve requests is not recommended with IIS. Frequently creating and deleting CGI processes involves significant overhead. Better alternatives include using ISAPI application scripts and ASP or ASP.NET scripts. Isolation is available for each of these options.

NTFS File System Setting

The system-global switch **NtfsDisableLastAccessUpdate (REG_DWORD) 1** is located under HKLM\System\CurrentControlSet\Control\FileSystem\.

This switch reduces disk I/O load and latencies by disabling date and time stamp updating for the last file or directory access. This key is set to 1 by default. Clean installations of Windows Server 2012, Windows Server 2008 R2, and Windows Server 2008 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 it was upgraded to Windows Server 2012, Windows Server 2008 R2, or Windows Server 2008, 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 they 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 hardware recommendations. The sections on networking and storage subsystems also apply to file servers.

Server Message Block Model

This section provides information about the Server Message Block (SMB) model for client-server communication, including the SMB 1.0, SMB 2.0 and SMB 3.0 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 miniredirector (Mrxsmb.sys), which translate them into SMB protocol sessions and requests over TCP/IP. In Windows 8, the SMB 3.0 protocol is supported. The Mrxsmb10.sys driver handles legacy SMB traffic, and the Mrxsmb20.sys driver handles SMB 2.0 and SMB 3.0 traffic.

On the server, SMB connections are accepted and SMB requests are processed as local file system operations through the NT file system (NTFS), Resilient File System (ReFS) or the Cluster Shared Volume file system (CSVFS) and the local storage stack. The Srv.sys driver handles legacy SMB traffic, and the Srv2.sys driver handles SMB 2.0 and SMB 3.0 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) if the file is not opened with the write-through flag set.

Figure 12 summarizes the layers that a user request on a client computer must pass through to perform file operations over the network on a remote SMB file server.

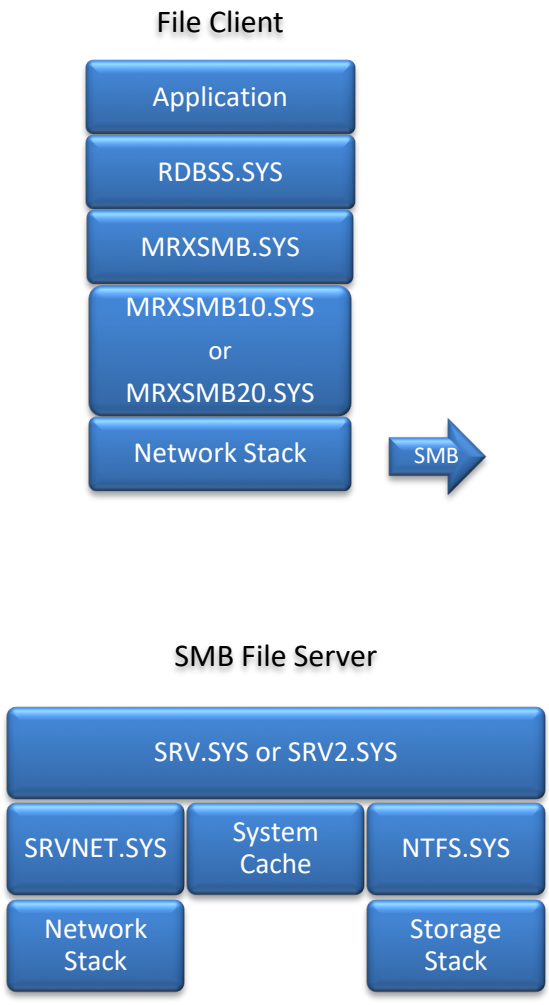

Figure 12. Layers on a remote SMB File Server

SMB Configuration Considerations

Do not enable any services or features that your 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, SMB encryption, and antivirus features.

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

Copying files is a common operations performed on a file server. The Windows Server operating system 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 (in Windows Server 2012 and 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 device or to a file.

Previous releases of the Windows Server operating system 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 2008 R2 and Windows Server 2012. 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 in Windows Server 2012 is 2, and in previous releases it is 0. This parameter determines whether NTFS generates a short name in the 8dot3 (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 8dot3 naming convention, NTFS does not generate a short name. A value of 2 means that this parameter can be configured per volume.
  Note The system volume will have 8dot3 enabled, whereas it is disabled by default in other volumes in Windows Server 2012.

  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 (disabled). For example, you would want to disable the setting if you have a clustered file server.

  In Windows Server 2012 and Windows Server 2008 R2, you can disable 8dot3name 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 8dot3 name creation on the volume D, run **fsutil 8dot3name set d: 1** from a Command Prompt window. You can view Help text by using the command **fsutil 8dot3name**. If you are disabling a new 8dot3 name creation on a volume that has existing data, consider stripping existing 8dot3 names from the volume. This can also be done with the **fsutil** tool. For example, to strip existing 8dot3 names on volume D and log the changes made, run **fsutil 8dot3name strip /l 8dot3_removal_log.log /s d:**. You can view Help text by typing the command **fsutil 8dot3name strip**.

- **TreatHostAsStableStorage**

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

  The default is 0. This parameter disables processing 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.

  Note If you have a clustered file server, it is possible that you may experience data loss if the server fails with this setting enabled. Therefore, evaluate it carefully prior to applying it.

- **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 cases (such as when there is a front-end server with a back-end IIS server) require a large amount of concurrency (for file change notification requests, in particular). The value of this entry can be increased to support these cases.

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

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

The defaults are 512 and 8192, 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, for example, copying files 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. 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 it can improve I/O performance, particularly on systems with many logical 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 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 computers 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.

- **NtfsDisableLastAccessUpdate**

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

The default is 1. In versions of Windows earlier than Windows Vista and Windows Server 2008, the default is 0. 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.

- **MaxMpxCt (SMB 1 clients only)**

```
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 it can improve performance for some client applications by enabling a deeper request pipeline. Increasing the value in conjunction with MaxCmds can also eliminate errors that are 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.

The following parameters are not required in Windows Server 2012:

- **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_DWO
RD)
```

- **TcpAckFrequency**

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

SMB Server Tuning Example

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

Parameter	Value	Default
NtfsDisable8dot3NameCreation	1	2
TreatHostAsStableStorage	1	0
AdditionalCriticalWorkerThreads	64	0
MaximumTunnelEntries	32	1024
MaxThreadsPerQueue	64	20
RequireSecuritySignature	0	0
MaxMpxCt (only applicable to SMB 1 clients)	32768	50

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 client computers and a server (see Figure 13). Applications on the client request files that are located on the server through the redirector (Rdbss.sys and NFS miniredirector Nfsrdr.sys). The miniredirector 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.

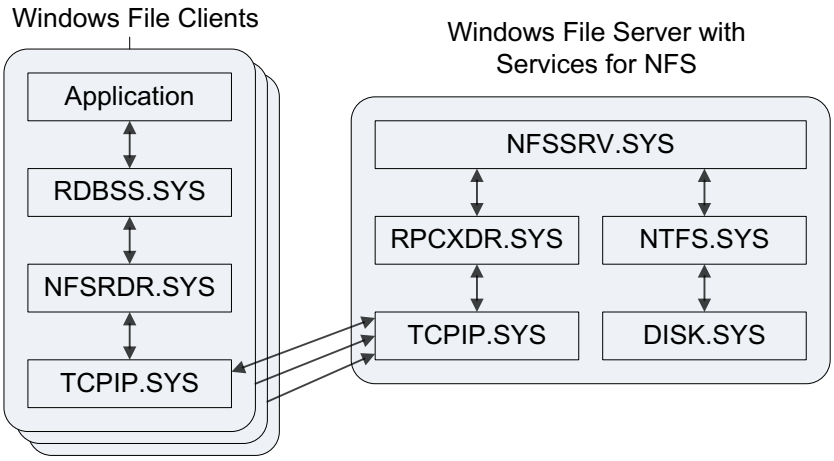

Figure 13. Microsoft services for NFS model for client-server communication

Tuning Parameters for NFS File Servers

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

- **OptimalReads**

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

Default is 0. Determines whether files are opened for FILE_RANDOM_ACCESS or for 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 prefetching.
For more information about File Access Services, see the File Servers section under 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. Services 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 it 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 that are 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 that is 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.

- **HandleSigningEnabled**

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

Default is 1. Controls whether handles that are given out by NFS File Server are signed cryptographically. Setting it to the value 0 would cause handle signing to be disabled.

- **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 that are opened during NFS V2 and V3 CREATE and MKDIR RPC procedure handlers are retained in the file handle cache. Set this value to 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 that are created for the specified work queue. Delayed worker threads process work items that are not considered time-critical and that 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 8dot3 (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 8dot3 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 it 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.

- **MaxConcurrentConnectionsPerIp**

```
HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\Rpcxdr\Para
meters (REG_DWORD)
```

The default value of the MaxConcurrentConnectionsPerIp parameter is 16. You can increase this value up to a maximum of 8192 to increase the number of connections per IP address.

General Tuning Parameters for Client Computers

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

```
HKLM\System\CurrentControlSet\Services\LanmanWorkstation\Paramet
ers
```

- **ConnectionCountPerNetworkInterface**

  Windows 8 only.

  The default is 1, with a valid range from 1-16. The maximum number of connections per interface to be established with a server running Windows Server 2012 for non-RSS interfaces.

- **ConnectionCountPerRssNetworkInterface**

  Windows 8 only.

  The default is 4, with a valid range from 1-16. The maximum number of connections per interface to be established with a server running Windows Server 2012 for RSS interfaces.

- **ConnectionCountPerRdmaNetworkInterface**

  Windows 8 only.

  The default is 2, with a valid range from 1-16. The maximum number of connections per interface to be established with server running Windows Server 2012 for RDMA interfaces.

- **MaximumConnectionCountPerServer**

  Windows 8 only.

  The default is 32, with a valid range from 1-64. The maximum number of connections to be established with a single server running Windows Server 2012 across all interfaces.

- **DormantDirectoryTimeout**

  Windows 8 only.

  The default is 600 seconds. The maximum time server directory handles held open with directory leases.

- **FileInfoCacheLifetime**

  Windows Vista, Windows 7, or Windows 8 only.

  The default is 10 seconds. The file information cache timeout period.

- **DirectoryCacheLifetime**

  Windows Vista and Windows 7 only.

  The default is 10 seconds. This is the directory cache timeout.

  Note This parameter controls caching of directory metadata in the absence of directory leases.

- **DirectoryCacheEntrySizeMax**

  Windows Vista, Windows 7, or Windows 8 only.

  The default is 64 KB. This is maximum size of directory cache entries.

- **FileNotFoundCacheLifetime**

  Windows Vista, Windows 7, or Windows 8 only.
  The default is 5 seconds. The file not found cache timeout period.

- **CacheFileTimeout**

  Windows 7 or Windows 8 only.
  The default is 10 seconds. This setting controls the time (in seconds) for which the redirector will hold on to cached data for a file after the last handle to the file is closed by an application.

- **DisableBandwidthThrottling**

  Windows Vista, Windows 7, or Windows 8 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**

  Windows Vista, Windows 7, or Windows 8 only.

  The default is 0 for Windows 8 only. In Windows 8, the SMB redirector transfers payloads as large as 1 MB per request, which can improve file transfer speed. Setting this registry value to 1 limits the request size to 64 KB. You should evaluate the impact of this setting before applying it.

- **RequireSecuritySignature**

  Windows Vista, Windows 7, or Windows 8 only.

  The default is 0. Changing this value to 1 prevents SMB communication with computers 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**

  Windows Vista, Windows 7, or Windows 8 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**

  Windows Vista, Windows 7, or Windows 8 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**

  Windows Vista, Windows 7, or Windows 8 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**

  Windows Vista, Windows 7, or Windows 8 only.

  The default is 15. This parameter limits the number of outstanding requests on a session. Increasing the value can use more memory, but it can improve performance by enabling a deeper request pipeline. Increasing the value in conjunction with MaxMpxCt can also eliminate errors that are 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**

  Windows XP, Windows Vista, Windows 7, or Windows 8 only.

  The default is 1023. This parameter specifies the maximum number of files that should be left open on a shared resource after the application has closed the file.

- **ScavengerTimeLimit**

  Windows XP only.

  The default is 10. 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**

  Windows XP only.

  The default is 0. Some distributed applications lock parts of a Read-only file because synchronization across clients requires that file-handle caching and collapsing behavior is turned off for all Read-only files. This parameter can be set if such applications will not run on the system and collapsing behavior can be enabled on the client computer.

The following parameter is not required in Windows 8:

- **EnableWsd**

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

File Client Tuning Example

The following settings for the parameters that are described in the General Tuning Parameters for Client Computers section earlier can optimize a computer for accessing remote file shares in many cases, particularly over some high-latency networks (such as branch offices, cross-datacenter communication, home offices, and mobile broadband). The settings are not optimal or appropriate on all computers. You should evaluate the impact of individual settings before applying them.

Parameter	Value	Default
DisableBandwidthThrottling	1	0
RequireSecuritySignature	0	1
FileInfoCacheEntriesMax	32768	64
DirectoryCacheEntriesMax	4096	16
FileNotFoundCacheEntriesMax	32768	128
MaxCmds	32768	15
DormantFileLimit [Windows XP only]	32768	1023
ScavengerTimeLimit [Windows XP only]	60	10
DisableByteRangeLockingOnReadOnlyFiles [Windows XP only]	1	0

In Windows 8, you can configure many of these File Server settings with Windows PowerShell, for example, by using the **Get-SmbServerConfiguration** and **Set-SmbServerConfiguration** cmdlets.

Performance Tuning for a File Server Workload (FSCT)

File Server Capacity Tool (FSCT) is a file server capacity planning tool that measures how many users a file server can support. FSCT creates many users that connect to the server and perform typical operations such as downloading files, uploading files, browsing directories, and opening files. FSCT gives a throughput score for each user count and evaluates if the server is overloaded with that many users. The highest user count without overload is the maximum number of users that the server can support under this workload.

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 in Windows Server 2012 is 2, and in previous releases it is 0. This parameter determines whether NTFS generates a short name in the 8dot3 (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 8dot3 naming convention, NTFS does not generate a short name. A value of 2 means that this parameter can be configured per volume.

  Note The system volume will have 8dot3 enabled, whereas other volumes will have it disabled by default in Windows Server 2012.

  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 (disabled). For example, you would want to disable the setting if you have a clustered file server.

  In Windows Server 2012 and Windows Server 2008 R2, you can disable 8dot3 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 8dot3 name creation on volume D, run **fsutil 8dot3name set d: 1** from a Command Prompt window. You can view Help text by using the command **fsutil 8dot3name**. If you are disabling new 8dot3 name creation on a volume that has existing data, consider stripping existing 8dot3 names from the volume. This can also be done with the **fsutil** tool. For example, to strip existing 8dot3 names on volume D and log the changes made, run **fsutil 8dot3name strip /l 8dot3_removal_log.log /s d:**. You can view Help text by using the command **fsutil 8dot3name strip**.

- **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.

Note If you have a clustered file server, it is possible that you may experience data loss if the server fails with this setting enabled. Therefore, evaluate it carefully prior to applying it.

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, Windows Vista, Windows 7, or Windows 8 only.

The default is 1023. This parameter specifies the maximum number of files that should be left open on a shared resource after the application has closed the file.

- **ScavengerTimeLimit**

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

Windows XP only.

The default is 10. 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 only.

The default is 0. Some distributed applications lock parts of a Read-only file because synchronization across clients requires that file-handle caching and collapsing behavior is turned off for all Read-only files. This parameter can be set if such applications will not run on the system and collapsing behavior can be enabled on the client computer.

Performance Counters for SMB 3.0

The following SMB performance counters are new in Windows Server 2012, and they are considered a base set of counters when you monitor the resource usage of SMB. Log the performance counters to a local, raw (.blg) performance counter log. It is less expensive to collect all instances by using the wildcard character "*****", and then extract particular instances while postprocessing by using Relog.exe.

- **SMB Client Shares**

  This counter set displays information about file shares on the server that are being accessed by a client that is using SMB protocol version 2 or higher.

- **SMB Server Shares**

  This counter set displays information about the SMB2 file shares on the server.

- **SMB Server Sessions**

  This counter set displays information about SMB server sessions that are using SMB protocol version 2 or higher.

- **Resume Key Filter**

  Performance counters for the Resume Key Filter.

- **SMB Direct Connection**

  This performance counter set consists of counters that measure aspects of connection activity. A computer can have multiple SMB Direct connections. The SMB Direct Connection counter set represents each connection as a pair of IP addresses and ports, where the first IP address and port represent the connection's local endpoint and the second IP address and port represent the connection's remote endpoint.

- **CSV Cache**

  The following counters can be gathered to monitor aspects of the CSV Cache:

  - I/O satisfied from cache
    - Cache IO Read-Bytes
    - Cache IO Read-Bytes/Sec
    - Cache Read
    - Cache Read/Sec
  - I/O satisfied from disk
    - Disk IO Read-Bytes
    - Disk IO Read-Bytes/Sec
    - Disk Read
    - Disk Read/Sec
  - Total I/O
    - IO Read-Bytes
    - IO Read-Bytes/Sec
    - IO Read
    - IO Read/Sec

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 guide contain additional tuning information that might apply to specific hardware.

Registry-Tuning Parameters for NFS File Servers

You can tune the following registry parameters previously described in the Tuning Parameters for NFS File Servers section to enhance the performance of NFS servers:

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

Performance Tuning for Active Directory Servers

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

- Use an appropriate amount of RAM.

Active Directory uses RAM to cache as much of the directory database as possible. This reduces disk access and improves performance. The Active Directory database can grow, but the amount of data that can be cached is 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 the 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 when the computer is rebooted, so the initial hit rate is low.

- Use a good disk I/O subsystem.

Ideally, the server is equipped with sufficient RAM to cache the “hot” parts (that is, the most frequently parts) of the database entirely in memory. However, the on-disk database must 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, an appropriate selection for storage is 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. 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 cache the most frequently used part of the database in memory. Query performance on a recently rebooted server, or after the Active Directory service is restarted, might initially be low until the cache is populated. Active Directory automatically populates the cache as queries search data in 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 perform disk write caching It commits all write operations to the disk before it returns a successful completion status for an operation, unless there is an explicit request to not do this. Therefore, fast disk I/O is important to the performance of write operations 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 caching 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 attributes is useful when you search for objects that have the attribute name in a 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 then consider indexing some attributes that are used in the corresponding queries to improve the search performance. For more information about 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 sign in from a different grandchild domain in the same forest, the authentication request must travel trust path from the grandchild to the root domain and then take the path to the other grandchild domain.

To avoid this, you can create a shortcut trust directly between the two grandchild domains that avoids the long path. However, an administrator must manage trusts. Therefore, you must consider how frequently a given trust will be used before you create it. You can also create an external trust or a forest trust to reduce the trust path when authenticating between domains in different forests. For more information about how to create trusts, see Administering Domain and Forest Trusts.

Active Directory Performance Counters

You can use several resources to conduct a performance diagnosis for 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 if many queued disk operations exist:
- Avg. Disk Queue Length
- Avg. Disk Read Queue Length
- Avg. Disk Write Queue Length
- If lsass.exe uses excessive 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 excessive CPU, check Directory Services\ATQ Outstanding Queued Requests on the Directory Services tab 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 to see the activity inside the domain controller. On a server that is running Active Directory Domain Services or Active Directory Lightweight Directory Services, 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 Reliability and Performance Monitor 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 (LSA) and Security Account Manager (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 aspects of that workload to the overall CPU usage, and locates the source of that workload, such as an application that is 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 the 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 how to select Remote Desktop Session Host (RD Session Host) hardware, tune the host, and tune applications.

Selecting the Proper Hardware for Performance

For an RD Session Host deployment, 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, there was a discussion about server hardware guidelines. Those guidelines apply to Remote Desktop Services, and this section contains additional guidelines that are specific to RD Session Host servers, and this information is 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 logical processors can help reduce abnormal CPU congestion situations, which are usually caused by a few overactive threads that are contained by a similar number of logical processors.

Therefore, the more logical processors on a system, the lower the cushion margin that must be built in to 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 more suitable for systems that need large amounts of memory. Specifically, the x64 version of the 64-bit architecture is a good option for RD Session Host deployments because it provides 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

Memory configuration is dependent on the applications that users employ; however, the required amount of memory can be estimated by using the following formula:

TotalMem = OSMem + SessionMem * 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 code or data pages have little effect because only one copy is present on the system.

One interesting observation (assuming the disk system that is backing up the page file does not change) is that 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 these faults eventually overwhelm 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 most overlooked aspects when you configure an RD Session Host system, and it can be the most common limitation in systems that are deployed in the field.

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

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

Ideally, these areas should be backed up 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 write caching. Controllers with disk write caching offer improved support for synchronous write operations. Because all users have a separate hive, synchronous write operations 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 that are exhibited by the applications 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.

- 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 usage that is needed to complete an action to avoid adversely affecting other sessions.

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

- Minimize background idle loop processing.

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

- Minimize how often an application performs a state check or update.

  Disabling such behaviors or increasing the interval between polling iterations and timer firing significantly benefits CPU usage because the 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.

  Examples of such resources include registry keys and configuration files. Examples of 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 start with user sign-in or a session startup.

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

For memory consumption, consider the following suggestions:

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

  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 issues 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

Page file

Insufficient page file size can cause memory allocation failures in applications or in system components. A general guideline is that the combined size of the page files should be two to three times larger than the physical memory size. You can use the memory-to-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 page files, memory allocation begins to fail. Because of significant disk I/O activity that page file access generates, consider using a

dedicated storage device for the page file, ideally a high-performance storage device 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 sign-in, 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 removing the component that registers them from the startup list or by changing the configuration on applications and system components to disable them.

RemoteFX data compression

RemoteFX® compression can be configured under **Remote Session Environment** > **Configure compression for RemoteFX data**. Three values are possible:

- **Optimized to use less memory.** Consumes the least amount of memory per session but has the lowest compression ratio and therefore the highest bandwidth consumption.
- **Balances memory and network bandwidth.** 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. If you want to use this setting, you should assess the maximum number of sessions and test to that level with this setting before you place the server in production.

You can also choose to not use an RemoteFX compression algorithm. Choosing to not use an RemoteFX compression algorithm will use more network bandwidth, and it is only recommended if you are using a hardware device that is designed to optimize network traffic. Even if you choose to not use an RemoteFX compression algorithm, some graphics data will be compressed.

Device redirection

Device redirection can be configured under **Device and Resource Redirection**. Or, it can be configured through **Server Manager** -> **Remote Desktop Services** -> **Session Collection Properties**.

Generally, device redirection increases how much network bandwidth RD Session Host connections use because data is exchanged between devices on the client computers 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 CPU usage at sign-in. 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, when used,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

By default, Remote Desktop Connection (RDC) automatically chooses the right experience setting based on the suitability of the network connection between the server and client computers. We recommend that the RDC configuration remain at “Detect connection quality automatically.”

For advanced users, RDC provides control over a range of settings that influence network bandwidth performance for the Remote Desktop Services connection. You can access the following settings through the RDC user interface on the **Experience** tab or as settings in the RDP File:

Settings that apply when connecting to any computer:

- **Disable wallpaper** (RDP file setting: Disable wallpaper:i:0): Suppresses the display of desktop wallpaper on redirected connections. This setting can significantly reduce bandwidth usage if desktop wallpaper consists of an image or other content with significant costs for drawing.
- **Bitmap cache** (RDP file setting: Bitmapcachepersistenable:i:1): When this setting is enabled, it creates a client-side cache of bitmaps that are rendered in the session. It provides a significant improvement on bandwidth usage, and it should always be enabled (unless there are other security considerations).
- **Show contents of windows while dragging** (RDP file setting: Disable full window drag:i:1): When this setting is disabled, it reduces bandwidth by

displaying only the window frame instead of all the content when the window is dragged.

- **Menu and window animation** (represented by two RDP file settings: Disable menu anims:i:1 and Disable cursor setting:i:1): When this setting is disabled, it reduces bandwidth by disabling animation on menus (such as fading) and cursors.

  Font smoothing (RDPfile setting: Allow font smoothing:i:0): Controls ClearType font-rendering support. When connecting to computers running Windows 8 or Windows Server 2012, enabling or disabling this setting does not have a siginificant impact on bandwidth usage. However, for computers running versions earlier than Windows 7 and Windows 2008 R2, enabling this setting affects network bandwidth consumption significantly.

Settings that apply only when connecting to computers running Windows 7 and earlier operating system versions:

- **Desktop composition**: This setting is supported only for a remote session to a computer running Windows 7 or Windows Server 2008 R2.
- **Visual styles** (RDP file setting: disable themes:i:1): When this setting is disabled, it reduces bandwidth by simplifying theme drawings that use the classic theme.

By using the **Experience** tab within Remote Desktop Connection, 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 Remote Desktop Connection:

- **Detect connection quality automatically.** When this setting is enabled, Remote Desktop Connection automatically chooses settings that will result in optimal user experience based on connection quality. (This configuration is recommended when connecting to computers running Windows 8 or Windows Server 2012) .
- **Modem** (56 Kbps): When this setting is selected, it allows 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 name 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
- Detect connection quality automatically = 7

Desktop Size

Desktop size for remote sessions can be controlled through the RDC client user interface (on the **Display** tab under the **Remote desktop size** setting) or through 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 2012. 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 the user’s experience, and it 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 Virtualization Host

Remote Desktop Virtualization Host (RD Virtualization Host) is a role service that supports Virtual Desktop Infrastructure (VDI) scenarios and lets multiple concurrent users run Windows-based applications in virtual machines that are hosted on a server running Windows Server 2012 and Hyper-V.

Windows Server 2012 supports two types of virtual desktops, personal virtual desktops and pooled virtual desktops.

General Considerations

Storage

Storage is the most likely performance bottleneck, and it is important to size your storage to properly handle the I/O load that is generated by virtual machine state changes. If a pilot or simulation is not feasible, a good guideline is to provision one disk spindle for four active virtual machines. Use disk configurations that have good write performance (such as RAID 1+0).

When appropriate, use SAN-based disk deduplication and caching to reduce the disk read load and to enable your storage solution to speed up performance by caching a significant portion of the image.

Memory

The server memory usage is driven by three main factors:

- Operating system overhead
- Hyper-V service overhead per virtual machine
- Memory allocated to each virtual machine

For a typical knowledge worker workload, guest virtual machines running Window 8 should be given ~512 MB of memory as the baseline. However, dynamic Memory will likely increase the guest virtual machine’s memory to about 800 MB, depending on the workload.

Therefore, it is important to provide enough server memory to satisfy the memory that is required by the expected number of guest virtual machines, plus allow a sufficient amount of memory for the server.

CPU

When you plan server capacity for an RD Virtualization Host, the number of virtual machines per physical core will depend on the nature of the workload. As a starting point, it is reasonable to plan 12 virtual machines per physical core, and then run the appropriate scenarios to validate performance and density. Higher density may be achievable depending on the specifics of the workload.

We recommend enabling hyperthreading, but be sure to calculate the oversubscription ratio based on the number of physical cores and not the number of logical processors. This ensures the expected level of performance on a per CPU basis.

Note It is critical to use virtual machines that have Second Level Address Translation (SLAT) support. Most servers running Windows Server 2012 will have SLAT.

Virtual GPU

Microsoft RemoteFX for RD Virtualization Host delivers a rich graphics experience for Virtual Desktop Infrastructure (VDI) through host-side remoting, a render-capture-encode pipeline, a highly efficient GPU-based encode, throttling based on client activity, and a DirectX-enabled virtual GPU. RemoteFX for RD Virtualization Host upgrades the virtual GPU from DirectX9 to DirectX11. It also improves the user experience by supporting more monitors at higher resolutions.

The RemoteFX DirectX11 experience is available without a hardware GPU, through a software-emulated driver. Although this software GPU provides a good experience, the RemoteFX virtual graphics processing unit (VGPU) adds a hardware accelerated experience to virtual desktops.

To take advantage of the RemoteFX VGPU experience on a server running Windows Server 2012, you need a GPU driver (such as DirectX11.1 or WDDM 1.2) on the host server. For more information about GPU offerings to use with RemoteFX for RD Virtualization Host, please refer to your GPU provider.

If you use the RemoteFX VGPU in your VDI deployment, the deployment capacity will vary based on usage scenarios and hardware configuration. When you plan your deployment, consider the following:

- Number of GPUs on your system
- Video memory capacity on the GPUs
- Processor and hardware resources on your system

RemoteFX Server System Memory

For every virtual desktop that is enabled with a VGPU, RemoteFX uses system memory in the guest operating system and in the RemoteFX server. The hypervisor guarantees the availability of system memory for a guest operating system. On the server, each VGPU-enabled virtual desktop needs to advertise its system memory requirement to the hypervisor. When the VGPU-enabled virtual desktop is starting, the hypervisor reserves additional system memory in the RemoteFX server for the VGPU-enabled virtual desktop.

The memory requirement for RemoteFX server is dynamic because the amount of memory consumed on the RemoteFX server is dependent on the number of monitors that are associated with the VGPU-enabled virtual desktops and the maximum resolution for those monitors.

RemoteFX Server GPU Video Memory

Every VGPU-enabled virtual desktop uses the video memory in the GPU hardwareon the host server to render the desktop. In addition to rendering, the video memory is used by a codec to compress the rendered screen. The amount of memory needed is directly based on the amount of monitors that are provisioned to the virtual machine.

The video memory that is reserved varies based on the number of monitors and the system screen resolution. Some users may require a higher screen resolution for specific tasks. There is greater scalability with lower resolution settings if all other settings remain constant.

RemoteFX Processor

The hypervisor schedules the RemoteFX server and the VGPU-enabled virtual desktops on the CPU. Unlike the system memory, there isn't information that is related to additional resources that RemoteFX needs to share with the hypervisor. The additional CPU overhead that RemoteFX brings into the VGPU-enabled virtual desktop is related to running the VGPU driver and a user-mode Remote Desktop Protocol stack.

In the RemoteFX server, the overhead is increased, because the system runs an additional process (rdvgm.exe) per VGPU-enabled virtual desktop. This process uses the graphics device driver to run commands on the GPU. The codec also uses the CPUs for compressing the screen data that needs to be sent back to the client.

More virtual processors mean a better user experience. We recommend allocating at least two virtual CPUs per VGPU-enabled virtual desktop. We also recommend using the x64 architecture for VGPU-enabled virtual desktops because the performance on x64 virtual machines is better compared to x86 virtual machines.

RemoteFX GPU Processing Power

For every VGPU-enabled virtual desktop, there is a corresponding DirectX process running on the RemoteFX server. This process replays all the graphics commands that it receives from the RemoteFX virtual desktop onto the physical GPU. For the physical GPU, it is equivalent to simultaneously running multiple DirectX applications.

Typically, graphics devices and drivers are tuned to run a few applications on the desktop. RemoteFX stretches the GPUs to be used in a unique manner. To measure how the GPU is performing on a RemoteFX server, performance counters have been added to measure the GPU response to RemoteFX requests.

Usually when a GPU resource is low on resources, Read and Write operations to the GPU take a long time to complete. By using performance counters, administrators can take preventative action, eliminating the possibility of any downtime for their end users.

The following performance counters are available on the RemoteFX server to measure the virtual GPU performance:

RemoteFX Graphics

- Frames Skipped/Second - Insufficient Client Resources:
  Number of frames skipped per second due to insufficient client resources
- Graphics Compression Ratio:
  Ratio of the number of bytes encoded to the number of bytes input

RemoteFX Root GPU Management

- Resources: TDRs in Server GPUs:
  Total number of times that the TDR times out in the GPU on the server
- Resources: Virtual machines running RemoteFX:
  Total number of virtual machines that have the RemoteFX 3D Video Adapter installed
- VRAM: Available MB per GPU:
  Amount of dedicated video memory that is not being used

- VRAM: Reserved % per GPU:
  Percent of dedicated video memory that has been reserved for RemoteFX

RemoteFX Software

- Capture Rate for monitor [1-4]:
  Displays the RemoteFX capture rate for monitors 1-4
- Compression Ratio:
  Deprecated in Windows 8 and replaced by Graphics Compression Ratio.
- Delayed Frames/sec:
  Number of frames per second where graphics data was not sent within a certain amount of time
- GPU response time from Capture:
  Latency measured within RemoteFX Capture (in microseconds) for GPU operations to complete
- GPU response time from Render:
  Latency measured within RemoteFX Render (in microseconds) for GPU operations to complete
- Output Bytes:
  Total number of RemoteFX output bytes
- Waiting for client count/sec:
  Deprecated in Windows 8 and replaced by Frames Skipped/Second - Insufficient Client Resources

RemoteFX VGPU Management

- Resources: TDRs local to virtual machines:
  Total number of TDRs that have occurred in this virtual machine (TDRs that the server propagated to the virtual machine are not included)
- Resources: TDRs propagated by Server:
  Total number of TDRs that occurred on the server and that have been propagated to the virtual machine

The following performance counters are present on the virtual desktop to measure the virtual GPU performance:

RemoteFX Virtual Machine VGPU Performance

- Data: Invoked presents/sec:
  Total number (in seconds) of present operations to be rendered to the desktop of the virtual machine per second
- Data: Outgoing presents/sec:
  Total number of present operations sent by the virtual machine to the server GPU per second
- Data: Read bytes/sec:
  Total number of read bytes from the RemoteFX server per second
- Data: Send bytes/sec:
  Total number of bytes sent to the RemoteFX server GPU per second

- DMA: Communication buffers average latency (sec):
  Average amount of time (in seconds) spent in the communication buffers
- DMA: DMA buffer latency (sec):
  Amount of time (in seconds) from when the DMA is submitted until completed
- DMA: Queue length:
  DMA Queue length for a RemoteFX 3D Video Adapter
- Resources: TDR timeouts per GPU:
  Count of TDR timeouts that have occurred per GPU on the virtual machine
- Resources: TDR timeouts per GPU engine:
  Count of TDR timeouts that have occurred per GPU engine on the virtual machine

In addition to the RemoteFX VGPU performance counters, users can measure the GPU utilization by using the new Process Explorer feature, which shows video memory usage and the GPU utilization.

Performance Optimizations

Dynamic Memory

Dynamic Memory enables more efficiently utilization of the memory resources of the server running Hyper-V by balancing how memory is distributed between running virtual machines. Memory can be dynamically reallocated between virtual machines in response to their changing workloads.

Dynamic Memory enables you to increase virtual machine density with the resources you already have without sacrificing performance or scalability. The result is more efficient use of expensive server hardware resources, which can translate into easier management and lower costs.

On guest operating systems running Windows 8 with virtual processors that span multiple logical processors, consider the tradeoff between running with Dynamic Memory to help minimize memory usage and disabling Dynamic Memory to improve the performance of an application that is computer-topology aware. Such an application can leverage the topology information to make scheduling and memory allocation decisions.

Tiered Storage

RD Virtualization Host supports tiered storage for a pooled virtual desktop collection. The physical computer that is shared by all pooled virtual desktops within a collection can use a small-size, high-performance storage solution, such as a mirrored solid-state drive (SSD). The pooled virtual desktops can be placed on less expensive, traditional storage such as RAID 1+0.

The physical computer should be placed on a SSD is because most of the read-I/Os from pooled virtual desktops go to the management operating system. Therefore, the storage that is used by the physical computer must sustain much higher read I/Os per second.

This deployment configuration assures cost effective performance where performance is needed. The SSD provides higher performance on a smaller size disk (~20 GB per collection, depending on the configuration). Traditional storage for pooled virtual desktops (RAID 1+0) uses about 3 GB per virtual machine.

CSV Cache

The Failover Clustering feature in Windows Server 2012 provides caching on Cluster Shared Volumes (CSV). This is extremely beneficial for pooled virtual desktop collections where the majority of the read I/Os come from the management operating system. The CSV cache provides higher performance by several orders of magnitude because it caches blocks that are read more than once and delivers them from system memory, which reduces the I/O.

For more information, see How to Enable CSV Cache.

Pooled Virtual Desktops

By default, pooled virtual desktops are rolled back to the pristine state after a user signs out, so any changes made to the Windows operating system since the last user sign-in are abandoned.

Although it's possible to disable the rollback, it is still a temporary condition because typically a pooled virtual desktop collection is re-created due to various updates to the virtual desktop template.

Hence, it makes sense to turn off Windows features and services that depend on persistent state. Additionally, it makes sense to turn off services that are primarily for non-enterprise scenarios.

Each specific service should be evaluated appropriately prior to any broad deployment. The following is a "starting list" to consider.

Service	Why?
Auto update	Pool virtual machines are updated by re-creating the virtual desktop template.
Offline files	VDI virtual machines are always online and connected from a networking point-of-view.
Background defrag	File-system changes are discarded after a user signs off (due to a rollback to the pristine state or re-creating the virtual desktop template, which results in re-creating all pooled virtual desktops).
Hibernate or sleep	No such concept for VDI
Bug check memory dump	No such concept for pooled virtual desktops. A bug-check pooled virtual desktop will start from the pristine state.
WLAN autoconfig	There is no WLAN device interface for VDI
Windows Media Player network sharing service	Consumer centric service
Home group provider	Consumer centric service
Internet connection sharing	Consumer centric service
Media Center extended services	Consumer centric service

This list is not meant to be a complete list, because any changes will affect the intended goals and scenarios.

Note SuperFetch in Windows 8 is enabled by default. It is VDI aware, and it should not be disabled. SuperFetch can further reduce memory consumption through memory page sharing, which is beneficial for VDI.

Performance Tuning for Remote Desktop Gateway

Note In Windows 8 and Windows Server 2012, RDGW supports TCP, UDP, and the legacy RPC transports. Most of the following data is regarding the legacy RPC transport. If the legacy RPC transport is not being used, this section is not applicable.

This section describes the performance-related parameters that help improve the performance of a customer deployment and the tunings that rely on the customer's network usage patterns.

At its core, the Remote Desktop Gateway (RD Gateway) performs many packet forwarding operations between Remote Desktop Connection instances and the RD Session Host server instances within the customer's network.

Internet Information Services (IIS) and RD Gateway export the following registry parameters to help improve system performance in the RD Gateway role service.

Note The following parameters apply to RPC transport only.

Thread tunings

- **MaxIoThreads**

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

  This application-specific thread pool specifies the number of threads that RD Gateway creates to handle incoming requests. If the registry is present, it takes effect. The number of threads equals the number of logical processes. If the number of logical processors is less then 5, the default is 5 threads.

- **MaxPoolThreads**

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

  This parameter specifies the number of IIS pool threads to create per logical processor. The IIS pool threads watch the network for requests and process all incoming requests. The **MaxPoolThreads** count does not include threads that RD Gateway consumes. The default value is 4.

Remote procedure call tunings for RD Gateway

The following parameters can help tune the remote procedure calls (RPC) that are received by RDC client and RD Gateway computers. 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 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 a value is not present, 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 window that the client uses for data that is received from the RPC proxy. The minimum value is 8 KB, and the maximum value is 1 GB. If a value is not present, 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*

The following performance counters are applicable only for legacy RPC transport:

\RPC/HTTP Proxy* RPC
\RPC/HTTP Proxy Per Server* RPC
\Web Service* RPC
\W3SVC_W3WP* RPC

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

Performance Tuning Remote Desktop Services Workload for Knowledge Workers

Remote Desktop Services capacity planning tools in Windows Server 2012 include automation framework and application scripting support that enable the simulation of user interaction with Remote Desktop Services. Be aware that the following tunings apply only to a synthetic Remote Desktop Services workload for knowledge workers, and they 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 Remote Desktop Services workload for knowledge workers uses Microsoft Office applications and Microsoft Internet Explorer. It operates in an isolated local network that has the following infrastructure:

- Domain controller with Active Directory, Domain Name System (DNS), and Dynamic Host Configuration Protocol (DHCP)
- Microsoft Exchange Server for hosting email
- IIS for hosting web services
- Load generator (a test controller) for creating a distributed workload
- A pool of Windows XP–based test systems to run the distributed workload, with no more than 60 simulated users for each physical test system
- Remote Desktop Services installed on an application server
- Microsoft Office

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

Table 13 provides guidelines for achieving the best performance for the Remote Desktop Services workload, and it suggests where bottlenecks might exist and how to avoid them.

Table 13. Hardware Recommendations for Remote Desktop Services 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 logical processors and dual-core or quad-core 64-bit CPUs).
Physical disks	• Separate the operating system files, page file, and user profiles (user data) into 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 non-uniform memory access (NUMA) 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 a 1 GB Ethernet.

Recommended Tunings on the Server

After you have installed the operating system and added the Remote Desktop Services role service, 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 page files (**Initial Size** equal to **Maximum Size**) with a total page file 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 page file is possible. Otherwise, the page file might be limited because of constraints in available disk space. There are no clear benefits of a page file larger than 100 GB. Make sure that no system-managed page files 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**.
- Click **Control Panel**, and 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 Remote Desktop Services 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**, and make the following changes:

- 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 Remote Desktop Services workload. Log the performance counters to a local performance counter log (.blg). It is less expensive to collect all instances by using the wild-card character "*****", and then extract particular instances 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 Event Tracing for Windows (ETW) events 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 Virtualization Servers

Hyper-V is the virtualization server role in Windows Server 2012. Virtualization servers can host multiple virtual machines 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, virtual machines 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 servers running Hyper-V. Tuning guidance that can yield increased performance on servers running Hyper-V, based on a live system state, is also available in the Hyper-V Advisor Pack that is distributed with the Server Performance Advisor tool described earlier in this guide.

For additional information about the Server Performance Advisor tool and the Hyper-V Advisor Pack, see Resources later in this guide.

Terminology

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

child partition
Any virtual machine 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 virtual machine environments and tune its behavior for virtual machines.

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 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 (central processing unit or CPU, memory, and devices). The hypervisor is responsible for controls and arbitrates access to the underlying hardware.

logical processor
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.

Hyper-V-specific device
A virtualized device with no physical hardware analog, so guests may need a driver (virtualization service client) to that Hyper-V-specific device. The driver can use virtual machine bus (VMBus) to communicate with the virtualized device software in the root partition.

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

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

virtualization service client
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
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 virtual machines. The virtualization stack works with and sits above the hypervisor. It also provides management capabilities.

Hyper-V Architecture

Hyper-V features the hypervisor-based architecture that is shown in Figure 14. The hypervisor virtualizes processors and memory and provides mechanisms for the virtualization stack in the root partition to manage child partitions (virtual machines) and expose services such as I/O devices to the virtual machines.

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 virtual machines, management APIs, and virtualized I/O devices. It also implements emulated devices such as integrated device electronics (IDE) and PS/2, but it supports Hyper-V-specific devices for increased performance and reduced overhead.

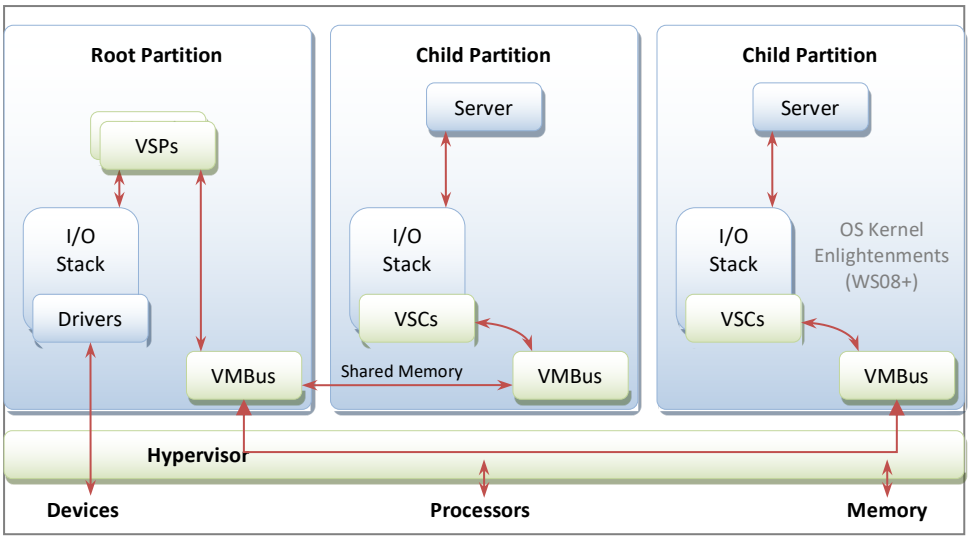

Figure 14. Hyper-V hypervisor-based architecture

The Hyper-V-specific I/O architecture consists of virtual service providers in the root partition and virtual service clients 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 virtual machines that use mechanisms such as shared memory. Plug and Play enumerates these devices, including VMBus, and loads the appropriate device drivers (virtual service clients). Services other than I/O are also exposed through this architecture.

Windows Server 2012, Windows Server 2008 R2, and Windows Server 2008 feature enlightenments to the operating system to optimize its behavior when it is running in virtual machines. The benefits include reducing the cost of memory virtualization, improving multicore 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 2012 for the Hyper-V server role.

Hardware Selection

The hardware considerations for servers running Hyper-V generally resemble those of non-virtualized servers, but servers running Hyper-V 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 2012 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 Extended Page Tables (EPT) or Nested Page Tables (NPT).

- **Cache**

Hyper-V can benefit from larger processor caches, especially for loads that have a large working set in memory and in virtual machine 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 virtual machine. Having additional memory available allows the root to efficiently perform I/Os on behalf of the virtual machines and operations such as a virtual machine 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, which enables each virtual switch to service a subset of virtual machines. When you hostmultiple virtual machines, using multiple network adapters enables distribution of the network traffic among the adapters for better overall performance.

To reduce the CPU usage of network I/Os from virtual machines, Hyper-V can use hardware offloads such as Large Send Offload (LSOv1, LSOv2), TCP checksum offload (TCPv4, TCPv6), and virtual machine queue (VMQ). No further configuration of the guest virtual machine is required. The network adapter provides coalesced packets to the host, and the host passes them along to the destination virtual machine.

For details about 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 the current and future needs of the virtual machines that the physical server hosts. Consider these requirements when you select storage controllers and disks and choose the RAID configuration. Placing virtual machines with highly disk-intensive workloads on different physical disks will likely improve overall performance. For example, if four virtual machines share a single disk and actively use it, each virtual machine can yield only 25 percent of the bandwidth of that disk. For details about 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 2012, Windows Server 2008 R2, and Windows Server 2008 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.

A Server Core installation 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. For a list of the supported guest operating systems, see Hyper-V Overview in the Windows Server Technical Library or the documentation that is provided with Hyper-V.

CPU Statistics

Hyper-V publishes performance counters to help characterize the behavior of the virtualization server and report 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 running cycles within a guest or within the host.
- \Hyper-V Hypervisor Logical Processor (*) \% Hypervisor Run Time

  The counter represents the time spent running 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 cores by assigning time slots 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 virtual machine increases the CPU usage, but this guide describes best practices for minimizing that overhead.

Virtual Machine Integration Services

The virtual machine Integration Services include enlightened drivers for the Hyper-V-specific I/O devices, which significantly reduces CPU overhead for I/O compared to emulated devices. You should install the latest version of the virtual machine 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 performance in a server running Hyper-V . 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 Server 2012, Windows Server 2008 R2, Windows Server 2008, Windows 8, Windows 7, and Windows Vista with SP1 feature enlightenments that optimize their operation for virtual machines. For best performance, we recommend that you use one of these Windows operating systems as a guest operating system.

The enlightenments present in Windows Server 2012, Windows Server 2008 R2, and Windows Server 2008 decrease the CPU overhead of the Windows operating system that is running in a virtual machine. 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 2012 supports a maximum of 64 virtual processors per virtual machine. Virtual machines 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.

Increase the number of virtual processors if the virtual machine 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. For more information, see see Hyper-V Overview in the Windows Server Technical Library.

Windows Server 2012 features enlightenments to the core operating system that improve scalability in multiprocessor virtual machines. Workloads can benefit from the scalability improvements in Windows Server 2012 if they run virtual machines with up to 64 virtual processors.

Background Activity

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

- Install the latest version of the virtual machine Integration Services.

- Remove the emulated network adapter through the virtual machine settings dialog box (use the Microsoft Hyper-V-specific adapter).
- Remove unused devices such as the CD-ROM and COM port, or disconnect their media.
- Keep the Windows guest on the sign-in screen when it is not being used.
- Use Windows Server 2012, Windows Server 2008 R2, or Windows Server 2008 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 default 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 virtual machine to reduce the overall CPU usage:

- Disable background services such as SuperFetch and Windows Search.
- Disable scheduled tasks such as Scheduled Defrag.
- Disable Aero glass 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 virtual machines, but we recommend the default values unless you have a compelling reason to alter them.

Weights and reserves prioritize or de-prioritize specific virtual machines if CPU resources are overcommitted. This ensures that those virtual machines 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 core.

Tuning NUMA Node Preference

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

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

disproportionate number of virtual machines being assigned to a single NUMA node.

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

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

Memory Performance

The hypervisor virtualizes the guest physical memory to isolate virtual machines from each other and to 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 2012, Windows Server 2008 R2, and Windows Server 2008 include kernel enlightenments and optimizations to the memory manager to reduce the CPU overhead from memory virtualization in Hyper-V. Workloads that have a large working set in memory can benefit from using Windows Server 2012, 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 scalability of Windows Server guests.

Correct Memory Sizing for Child Partitions

You should size virtual machine memory as you typically do for server applications on a physical computer. 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 virtual machine memory dynamically. The recommended initial memory size for Windows Server 2012 guests is at least 512 MB. With Dynamic Memory, if applications in the virtual machine experience launching problems, you can increase the page file size for the virtual machine.

To increase the virtual machine page file 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 virtual machine's Memory Demand when virtual machine reaches its steady state, and set the **Maximum Size** to three times the **Initial Size.**

For more information, see the Hyper-V Dynamic Memory Configuration 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 virtual machine 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, virtual machine snapshot, and management to support the child partitions. Hyper-V calculates an amount of memory known as the *root reserve*, which is guaranteed to be available to the root partition and never assigned to virtual machines. It is calculated automatically, based on the host's physical memory and system architecture. This logic applies for supported scenarios with no applications running in the root partition.

Storage I/O Performance

This section describes the different options and considerations for tuning storage I/O performance in a virtual machine. The storage I/O path extends from the guest storage stack, through the host virtualization layer, to the host storage stack, and then to the physical disk. Following are explanations about how optimizations are possible at each of these stages.

Virtual Controllers

Hyper-V offers three types of virtual controllers : IDE, SCSI, and Virtual host bus adapters (HBAs).

IDE

IDE controllers expose IDE disks to the virtual machine. The IDE controller is emulated, and it is the only controller that is available when the Integration Services are not installed on the guest. Disk I/O that is performed by using the IDE filter driver that is provided with the Integration Services is significantly better than the disk I/O performance that is provided with the emulated IDE controller. We recommend that IDE disks be used only for the operating system disks because they have performance limitations due to the maximum I/O size that can be issued to these devices.

SCSI

SCSI controllers expose SCSI disks to the virtual machine, and each virtual SCSI controller can support up to 64 devices. For optimal performance, we recommend that you attach multiple disks to a single virtual SCSI controller and create additional controllers only as they are required to scale the number of disks connected to the virtual machine.

Virtual HBAs

Virtual HBAs can be configured to allow direct access for virtual machines to Fibre Channel and Fibre Channel over Ethernet (FCoE) LUNs. Virtual Fibre Channel disks

bypass the NTFS file system in the root partition, which reduces the CPU usage of storage I/O.

Large data drives and drives that are shared between multiple virtual machines (for guest clustering scenarios) are prime candidates for virtual Fibre Channel disks.

Virtual Fibre Channel disks require one or more Fibre Channel host bus adapters (HBAs) to be installed on the host. Each host HBA is required to use an HBA driver that supports the Windows Server 2012 Virtual Fibre Channel/NPIV capabilities. The SAN fabric should support NPIV, and the HBA port(s) that are used for the virtual Fibre Channel should be set up in a Fibre Channel topology that supports NPIV.

To maximize throughput on hosts that are installed with more than one HBA, we recommend that you configure multiple virtual HBAs inside the Hyper-V virtual machine (up to four HBAs can be configured for each virtual machine). Hyper-V will automatically make a best effort to balance virtual HBAs to host HBAs that access the same virtual SAN.

Virtual Disks

Disks can be exposed to the virtual machines through the virtual controllers. These disks could be virtual hard disks that are file abstractions of a disk or a pass-through disk on the host.

Virtual Hard Disks

There are two virtual hard disk formats, VHD and VHDX. Each of these formats supports three types of virtual hard disk files.

VHD Format

The VHD format was the only virtual hard disk format that was supported by Hyper-V in past releases. In Windows Server 2012, the VHD format has been modified to allow better alignment, which results in significantly better performance on new large sector disks.

Any new VHD that is created on a Windows Server 2012 operating system has the optimal 4 KB alignment. This aligned format is completely compatible with previous Windows Server operating systems. However, the alignment property will be broken for new allocations from parsers that are not 4 KB alignment-aware (such as a VHD parser from a previous version of Windows Server or a non-Microsoft parser).

Any VHD that is moved from a previous release does not automatically get converted to this new improved VHD format.

You can check the alignment property for all the VHDs on the system, and it should be converted to the optimal 4 KB alignment. You create a new VHD with the data from the original VHD by using the **Create-from-Source** option.

VHDX Format

VHDX is a new virtual hard disk format introduced in Windows Server 2012, which allows you to create resilient high-performance virtual disks up to 64 terabytes. Benefits of this format include:

- Support for virtual hard disk storage capacity of up to 64 terabytes.

- Protection against data corruption during power failures by logging updates to the VHDX metadata structures.
- Ability to store custom metadata about a file, which a user might want to record, such as operating system version or patches applied.

The VHDX format also provides the following performance benefits (each of these is detailed later in this guide):

- Improved alignment of the virtual hard disk format to work well on large sector disks.
- Larger block sizes for dynamic and differential disks, which allows these disks to attune to the needs of the workload.
- 4 KB logical sector virtual disk that allows for increased performance when used by applications and workloads that are designed for 4 KB sectors.
- Efficiency in representing data, which results in smaller file size and allows the underlying physical storage device to reclaim unused space. (Trim requires pass-through or SCSI disks and trim-compatible hardware.)

When you upgrade to Windows Server 2012, we recommend that you convert all VHD files to the VHDX format due to these benefits. The only scenario where it would make sense to keep the files in the VHD format is when a virtual machine has the potential to be moved to a previous release of the Windows Server operating system that supports Hyper-V.

Types of Virtual Hard Disk Files

There are three types of VHD files. Following are the performance characteristics and trade-offs between the three VHD types.

Fixed type

Space for the VHD is first allocated when the VHD file is created. This type of VHD file 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 file types because Reads and Writes do not need to look up the mapping of the block.

Dynamic type

Space for the VHD is allocated on demand. The blocks in the disk start as zeroed blocks, but they 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 disk access and CPU overhead when looking up the blocks' mapping in the metadata.

Differencing type

The VHD points to a parent VHD file. Any Writes to blocks not written to 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 VHD file.

The following recommendations should be taken into consideration with regards to selecting a VHD file type:

- When using the VHD format, we recommend that you use the fixed type because it has better resiliency and performance characteristics compared to the other VHD file types.
- When using the VHDX format, we recommend that you use the dynamic type because it offers resiliency guarantees in addition to space savings that are associated with allocating space only when there is a need to do so.
- The fixed type is also recommended, irrespective of the format, when the storage on the hosting volume is not actively monitored to ensure that sufficient disk space is present when expanding the VHD file at run time.
- Snapshots of a virtual machine create a differencing VHD to store Writes to the disks. 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.

Block Size Considerations

Block size can significantly impact performance**.** It is optimal to match the block size to the allocation patterns of the workload that is using the disk. If an application is allocating in chunks of, for example, 16 MB, it would be optimal to have a virtual hard disk block size of 16 MB. A block size of >2 MB is possible only on virtual hard disks with the VHDX format. Having a larger block size than the allocation pattern for a random I/O workload will significantly increase the space usage on the host.

Sector Size Implications

Most of the software industry has depended on disk sectors of 512 bytes, but the standard is moving to 4 KB disk sectors. To reduce compatibility issues that might arise from a change in sector size, hard drive vendors are introducing a transitional size referred to as 512 emulation drives (512e).

These emulation drives offer some of the advantages that are offered by 4 KB disk sector native drives, such as improved format efficiency and an improved scheme for error correction codes (ECC). They come with fewer compatibility issues that would occur by exposing a 4 KB sector size at the disk interface.

Support for 512e Disks

A 512e disk can perform a Write only in terms of a physical sector—that is, it cannot directly write a 512-byte sector that is issued to it. The internal process in the disk that makes these Writes possible follows these steps:

- The disk reads the 4 KB physical sector to its internal cache, which contains the 512-byte logical sector referred to in the Write.
- Data in the 4 KB buffer is modified to include the updated 512-byte sector.

- The disk performs a Write of the updated 4 KB buffer back to its physical sector on the disk.

This process is called “Read-Modify-Write” or RMW. The overall performance impact of the RMW process depends on the workloads. The RMW process causes performance degradation in virtual hard disks for the following reasons:

- Dynamic and differencing virtual hard disks have a 512-byte sector bitmap in front of their data payload. In addition, footer, header, and parent locators align to a 512-byte sector. It is common for the virtual hard disk driver to issue 512-byte Writes to update these structures, resulting in the RMW process described earlier.
- Applications commonly issue Reads and Writes in multiples of 4 KB sizes (the default cluster size of NTFS). Because there is a 512-byte sector bitmap in front of the data payload block of dynamic and differencing virtual hard disks, the 4 KB blocks are not aligned to the physical 4 KB boundary. The following figure shows a VHD 4 KB block (highlighted) that is not aligned with physical 4 KB boundary.

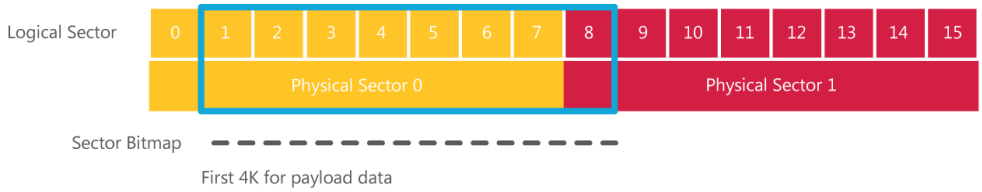

Figure 15: VHD 4 KB block

Each 4 KB Write that is issued by the current parser to update the payload data results in two Reads for two blocks on the disk, which are then updated and subsequently written back to the two disk blocks. Hyper-V in Windows Server2012 mitigates some of the performance effects on 512e disks on the VHD stack by preparing the previously mentioned structures for alignment to 4 KB boundaries in the VHD format. This avoids the RMW effect when accessing the data within the virtual hard disk file and when updating the virtual hard disk metadata structures.

As mentioned earlier, VHDs that are copied from previous versions of Windows Server will not automatically be aligned to 4 KB. You can manually convert them to optimally align by using the **Copy from Source** disk option that is available in the VHD interfaces.

By default, VHDs are exposed with a physical sector size of 512 bytes. This is done to ensure that physical sector size dependent applications are not impacted when the application and VHDs are moved from a previous version of Windows Server to Windows Server 2012.

After you confirm that all VHDs on a host are not impacted by changing the physical sector size to 4 KB, you can set the following registry key to modify all the physical sector sizes of all the VHDs on that host to 4 KB. This helps systems and applications that are 4 KB-aware to issue 4 KB sized I/Os to the VHDs.

```
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\vhdmp\Parameters\ Vhd1PhysicalSectorSize4KB = (REG_DWORD)
```

A non-zero value will change all VHDs to report 4k physical sector size.

By default, disks with the VHDX format are created with the 4 KB physical sector size to optimize their performance profile regular disks and large sector disks.

Support for Native 4 K Disks

Hyper-V in Windows Server 2012 makes it possible to store virtual hard disks. This is done by implementing a software RMW algorithm in the virtual storage stack layer that converts 512-byte access and update requests to corresponding 4 KB accesses and updates.

Because VHD file can only expose themselves as 512-byte logical sector size disks, it is very likely that there will be applications that issue 512-byteI/O requests. In these cases, the RMW layer will satisfy these requests and cause performance degradation. This is also true for a disk that is formatted with VHDX that has a logical sector size of 512 bytes.

It is possible to configure a VHDX file to be exposed as a 4 KB logical sector size disk, and this would be an optimal configuration for performance when the disk is hosted on a 4 KB native physical device. Care should be taken to ensure that the guest and the application that is using the virtual disk are backed by the 4 KB logical sector size. The VHDX formatting will work correctly on a 4 KB logical sector size device.

Block Fragmentation

Just as the allocations on a physical disk can be fragmented, the allocation of the blocks on a virtual disk can be fragmented when two virtually adjacent blocks are not allocated together on a virtual disk file.

The fragmentation percentage is reported for disks. If a performance issue noticed on a virtual disk, you should check the fragmentation percentage. When applicable, defragment the virtual disk by creating a new virtual disk with the data from the fragmented disk by using the **Create from Source** option.

Pass-through Disks

The VHD in a virtual machine can be mapped directly to a physical disk or logical unit number (LUN), instead of to a VHD file. The benefit is that this configuration bypasses the NTFS file system 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 pass-through disks, especially if they are I/O intensive. Virtual machines that can be migrated between virtualization servers (such as in a quick migration) must also use drives that reside on a LUN of a shared storage device.

Advanced Storage Features

I/O Balancer Controls

The virtualization stack balances storage I/O streams from different virtual machines so that each virtual machine 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)
```

Storage and networking have three registry keys at the **StorVsp** and **VmSwitch** paths, respectively. Each value is a DWORD and operates as explained in the following list.

Note We do not recommend this advanced tuning option unless you have a specific reason to use it. These registry keys may be removed in future releases.

- **IOBalance_Enabled**

The balancer is enabled when it is set to a nonzero value, and it is 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 it 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 virtual machine before switching to another virtual machine. This setting is primarily for storage where finely interleaving I/Os from different virtual machines can increase the number of disk seeks. The default is 8 percent for both storage and networking.

NUMA I/O

Windows Server 2012 supports large virtual machines, and any large virtual machine configuration (for example, a configuration with SQL Server running with 64 virtual processors) will also need scalability in terms of I/O throughput.

The following key improvements in the Windows Server 2012 storage stack and Hyper-V provide the I/O scalability needs of large virtual machines:

- An increase in the number of communication channels created between the guest devices and host storage stack
- A more efficient I/O completion mechanism involving interrupt distribution amongst the virtual processors to avoid interprocessor interruptions, which are expensive

There are a few registry keys that allow the number of channels to be adjusted. They also align the virtual processors that handle the I/O completions to the virtual CPUs that are assigned by the application to be the I/O processors. The registry values are set on a per-adapter basis on the device's hardware key. The following two values are located in HKLM\System\CurrentControlSet\Enum\VMBUS\{device id}\{instance id}\Device Parameters\StorChannel

- **ChannelCount (DWORD)**: The total number of channels to use, with a maximum of 16. It defaults to a ceiling, which is the number of virtual processors/16.
- **ChannelMask (QWORD)**: The processor affinity for the channels. If it is not set or is set to 0, it defaults to the existing channel distribution algorithm that you use for normal storage or for networking channels. This ensures that your storage channels won't conflict with your network channels.

Offloaded Data Transfer Integration

Crucial maintenance tasks for VHDs, such as merge, move, and compact, depend on copying large amounts of data. The current method of copying data requires data to be read in and written to different locations, which can be a time-consuming process. It also uses CPU and memory resources on the host, which could have been used to service virtual machines.

Storage area network (SAN) vendors are working to provide near-instantaneous copy operations of large amounts of data. This storage is designed to allow the system above the disks to specify the move of a specific data set from one location to another. This hardware feature is known as an offloaded data transfer.

The storage stack for Hyper-V in Windows Server 2012 supports Offload Data Transfer (ODX) operations so that these operations can be passed from the guest operating system to the host hardware. This ensures that the workload can use ODX-enabled storage as it would if it were running in a non-virtualized environment. The Hyper-V storage stack also issues ODX operations during maintenance operations for VHDs such as merging disks and storage migration meta-operations where large amounts of data are moved.

Unmap Integration

Virtual hard disk files exist as files on a storage volume, and they share available space with other files. Because the size of these files tends to be large, the space that they consume can grow quickly. Demand for more physical storage affects the IT hardware budget, so it's important to optimize the use of physical storage as much as possible.

Currently, when applications delete content within a virtual hard disk, which effectively abandons the content's storage space, the Windows storage stack in the guest operating system and the Hyper-V host have limitations that prevent this information from being communicated to the virtual hard disk and the physical storage device. This prevents the Hyper-V storage stack from optimizing the space usage by the VHDX-based virtual disk files. It also prevents the underlying storage device from reclaiming the space that was previously occupied by the deleted data.

In Windows Server 2012, Hyper-V supports unmap notifications, which allow VHDX files to be more efficient in representing that data within it. This results in smaller files size, and it allows the underlying physical storage device to reclaim unused space.

Only Hyper-V-specific SCSI, Enlightened IDE and virtual Fibre Channel controllers allow the unmap command from the guest to reach the host virtual storage stack.

Hyper-V-specific SCSI is also used for pass-through disks. On the virtual hard disks, only virtual disks formatted as VHDX support unmap commands from the guest.

For these reasons, we recommend that you use VHDX files attached to a SCSI controller when not using regular pass through or virtual Fibre Channel disks.

Network I/O Performance

Hyper-V supports Hyper-V-specific and emulated network adapters in the virtual machines, but the Hyper-V-specific 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, see 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.

Hyper-V-specific Network Adapter

Hyper-V features a Hyper-V-specific network adapter that is designed specifically for virtual machines to achieve significantly reduced CPU overhead on network I/O when it is compared to the emulated network adapter that mimics existing hardware. The Hyper-V-specific 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 settings dialog box in the virtual machine and replaced with a Hyper-V-specific network adapter. The guest requires that the virtual machine Integration Services be installed.

Perfmon counters that represent the network statistics for the installed Hyper-V-specific network adapters are available under the following counter set: \Hyper-V Virtual Network Adapter (*) \ *.

Install Multiple Hyper-V-specific Network Adapters on Multiprocessor virtual machines

Virtual machines with more than one virtual processor might benefit from having more than one Hyper-V-specific network adaptor installed in the virtual machine. Workloads that are network intensive, such as a web server, can make use of greater parallelism in the virtual network stack if a second Hyper-V-specific network adapter is installed in a virtual machine.

Offload Hardware

As with the native scenario, offload capabilities in the physical network adapter reduce the CPU usage of network I/Os in virtual machine scenarios. Hyper-V currently supports LargeSend Offload (LSOv1, LSOv2) and TCP checksum offload (TCPv4, TCPv6). The offload capabilities must be enabled in the driver for the physical network adapter in the root partition. For details about offload capabilities in network adapters, refer to Choosing a Network Adapter.

Drivers for certain network adapters disable LSOv1, and they enable LSOv2 by default. System administrators must explicitly enable LSOv1 by using the **Properties** dialog box for the driver 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 virtual machine can be connected to a virtual network switch. If the physical server has multiple network adapters, virtual machines with network-intensive loads can benefit from being connected to different virtual switches to better use the physical network adapters.

Perfmon counters that represent the network statistics for the installed Hyper-V-specific switches are available under the following counter set: \Hyper-V Virtual Switch (*) \ *.

VLAN Performance

The Hyper-V-specific 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 Large Send Offload and checksum offload. Without this support, Hyper-V cannot use hardware offload for packets that require VLAN tagging, and network performance can be decreased.

Dynamic VMQ

Dynamic virtual machine queue (VMQ or dVMQ) is a performance optimization of VMQ in Windows Server 2012 that automatically scales the number of processors that are in use for VMQ, based on the traffic volume. This section provides guidance about how to configure the system to take full advantage of dVMQ.

The configuration of dVMQ is controlled by the same settings as RSS. It uses the following standard keywords in the registry for the base CPU number and the maximum number of CPUs to use: *RssBaseProcNumber and *MaxRssProcessors.

Some Intel multi-core processors may use Intel Hyper-Threading Technology. When Hyper-Threading Technology is enabled, the actual number of cores that are used by dVMQ should be half the total number of logical processors that are available in the system. This is because dVMQ spreads the processing across individual physical cores only, and it will not use hyperthreaded sibling cores. As an example, if the machine has an Intel processor with four physical cores, and Hyper-Threading Technology is enabled, it will show a total of eight logical processors. Only four logical processors are available to VMQ. VMQ will use cores 0, 2, 4, and 6.

There may be situations where starting with logical processor 0 (which corresponds to core 0) as the *RssBaseProcNumber is acceptable. However, general guidance is to avoid using core 0 as the base CPU because it is generally used for default network queues and workload processing in the root partition.

Based on the root virtual processor utilization, the RSS base and maximum logical processors for a physical network adapter can be configured to define the set of root virtual processors that are available for VMQ processing. Selecting the set of VMQ processors can better isolate latency-centric or cache-sensitive virtual machines or workloads from root network processing. Use the Windows PowerShell cmdlet, **Set-NetAdapterVmq**, to set the correct base processor number and the maximum processor number.

There are limited hardware queues available, so you can use the Hyper-V WMI API to ensure that the virtual machines using the network bandwidth are assigned a hardware queue. To disable VMQ for specific virtual network adapters, open Hyper-V Manager, and then open the settings for a virtual machine. For the virtual network adapter on which you want to disable VMQ, expand the port settings, and in the **Hardware Acceleration** section, clear the **Enable Virtual Machine Queue** check box.

When a host has multiple network adapters, each for a different virtual switch instance, and it is using dVMQ, do not configure dVMQ to use overlapping processor sets. Ensure that each network adapter that is configured for dVMQ has its own set of cores. Otherwise performance results may be impaired and unpredictable.

There are two separate dVMQ configuration recommendations, based on the type of network adapter teaming (also known as load balancing and failover or LBFO) in Windows Server 2012 with Hyper-V. If the team is configured by using switch dependent-mode or address hashing, each network adapter in the team should have identical values for *RssBaseProcNumber and *MaxRssProcessors. This mode is referred to as Min mode in the following table.

If the team is configured by using switch independent mode and Hyper-V switch port addressing, each network adapter in the team should be configured by using non-overlapped processor sets as earlier described for multiple network adapters. This is referred to as the Sum mode in the following table. For more information, see the Windows Server 2012 NIC Teaming (LBFO) Deployment and Management Guide.

Teaming Modes

	Address hash (2-tuple, 4-tuple, or MAC)	**Hyper-V switch port**
Switch dependent mode	Min	Min
Switch independent mode	Min	Sum

The following table has example comparisons of the dVMQ settings (RssBaseProcNumber, MaxRssProcNumber) for two virtual machines that use the Min and Sum configurations.

***RssBaseProcNumber, *MaxRssProcessors**	**8-core HT, 3 network adapters**	**16-core HT, 4 network adapters**
Min example configuration	Each NIC=2, 3	Each NIC=2,7
Sum example configuration	NIC1=2,1 NIC2=4,1 NIC3=6,1	NIC1=2,2 NIC2=6,2 NIC3=10,2 NIC4=14,1

The following formula can be used to compute even distribution of cores among network adapters for a team. It is assumed that the first core is skipped; that is, RssBaseProcNumber starts at 2.
The maximum number of cores per virtual machine that dVMQ will use is 16.
Ct = total number of cores
Nt = total number of network adapters in the team

The general case assumes hyperthreading is in use on cores:
MaxRssProcessors = min((Ct-2)/2/Nt, 16)

In the event that hyperthreading is disabled or unavailable:
MaxRssProcessors=min((Ct-1)/Nt, 16)

MAC Spoofing Guidance

By default each virtual machine has MAC address protection, so traffic on a MAC address (other that the one assigned in the host) is blocked. To allow a virtual machine to set its own MAC address, the **Enable MAC Spoofing** check box must be selected with the Windows PowerShell cmdlet **Set-VMNetworkAdapter** as follows:

```
PS C:> get-vm “MyVM” | set-vmNetworkAdapter
–MacAddressSpoofing On
```

Note If MAC spoofing is enabled on a virtual machine that is connected to an SR-IOV mode virtual switch, traffic will still be blocked. MAC spoofing should only be enabled for a virtual machine on a non-SR-IOV mode virtual switch. If **Enable MAC Spoofing** is selected, the SR-IOV virtual function is removed and traffic is routed through the virtual switch.

Single Root I/O Virtualization

Windows Server 8 introduces support for single root I/O virtualization (SR-IOV), which allows the direct assignment of network resources (virtual functions) to individual virtual machines when specialized hardware is available. SR-IOV can be enabled for networking and CPU intensive workloads to reduce virtualization overhead for networking I/O.

The number of virtual functions available for assignment is defined by the network adapter, and they can be queried with the Windows Powershell cmdlet, **Get-NetAdapterSrio**v. Other IOV operations that are exposed through Windows Powershell include:

- **Enable-NetAdapterSriov**: Enables IOV on the physical network adapter
- **Disable-NetAdapterSriov**: Disables IOV
- **New-VMSwitch**: Use withthe **–EnableIov** flag to create an IOV switch

When networking virtual machine-to-virtual machine, the virtual function bandwidth is limited by the bandwidth of the physical adapter. In cases where the virtual function bandwidth is saturated, switching to the Hyper-V-specific network adapter for inter-virtual machine communication can improve throughput by decoupling virtual machine-to-virtual machine network I/O traffic from the physical adapter bandwidth.

Live Migration

Live migration enables you to transparently move running virtual machines from one node of a 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 divided into two major phases. The first phase is the copying the memory contents on the virtual machine from the current host to the new host. The second phase is transfering the virtual machine 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 minimize the time that is required to complete a live migration, and it ensures consistent migration times.

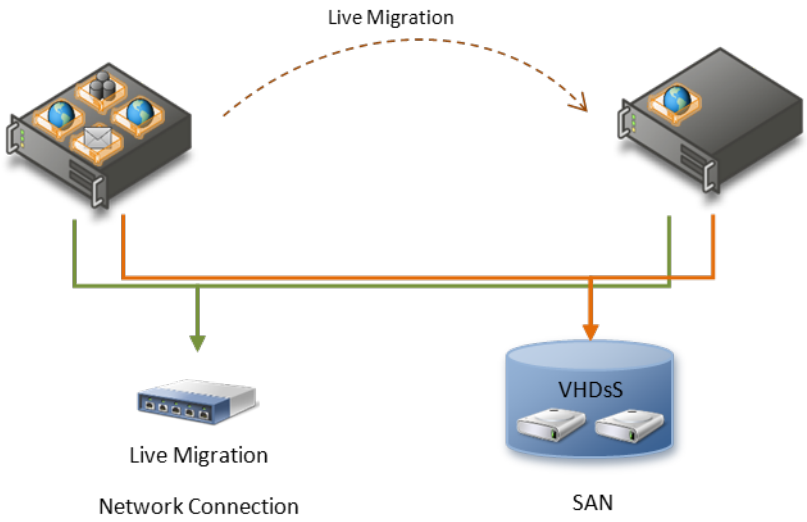

Figure 16. Example Hyper-V live migration configuration

In addition, increasing the number of receive and send buffers on each network adapter that is 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 SAP Sales and Distribution

SAP AG has developed several standard application benchmarks. The Sales and Distribution workload represents one of the important classes of workloads that are used to benchmark SAP enterprise resource planning installations. The updates to SAP include added requirements, such as subsecond response time and a Unicode code page. For more information, see SAP with Microsoft SQL Server 2008 and SQL Server 2005: Best Practices for High Availability, Maximum Performance, and Scalability.

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 sales and distribution users increases before performance levels off because of resource limitations.

The following sections provide guidelines that can benefit two and three tier configurations for sales and distribution benchmarks for SAP enterprise resource planning in Windows Server2012. 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 then set one or more fixed-size page files (set the **Initial Size** equal to **Maximum Size**). The page file size should meet the total virtual memory requirements of the workload. Make sure that no system-managed page files are in the virtual memory on the Application Server.
- Navigate to Performance Settings > Visual Effects, and then select the Adjust for best performance check box.
- To enable SQL to use large pages, configure 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.
- Navigate to **Start > All Programs > Administrative Tools > System Configuration > Tools** tab, select **Disable UAC**, and then reboot the system.
  Note The UAC setting can be used for benchmark environments, but enabling UAC might be a security compliance requirement in production environments.

For a virtualized configuration, these settings apply across the host and virtual machine operating systems. For latency sensitive virtualized configurations, to partition network I/O processing from SAP computing resources, consider limiting the number of logical processors that are available for the VMQ interrupt

processing. This can be accomplished by configuring the base and maximum RSS logical processors.

Tunings on the Database Server

When the database server is running SQL Server, consider setting the following SQL Server configuration options by using *sp_configure*. For detailed information about the *sp_configure* stored procedure, see Server Configuration Options.

- **Apply CPU affinity for the SQL Server process.**
  Set an affinity mask to partition the SQL Server process on specific cores. If required, use the affinity64 mask server configuration option to set the affinity on more than 32 cores. In SQL Server 2012 and SQL Server 2008 R2, you can apply equivalent settings for configuring CPU affinity on as many as 640 logical processors by using the ALTER SERVER CONFIGURATION Transact-SQL statement because the sp_configure affinity mask options are announced for deprecation.

  Note For the current two-tier SAP Sales and Distribution 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:

- For information about how to subdivide the CPUs in a hardware NUMA node into more CPU nodes (known as Soft-NUMA), see Configure SQL Server to Use Soft-NUMA.
- To provide NUMA node locality for SQL Server, set preferred NUMA node hints (applies to Windows Server 2012 and Windows Server 2008 R2). For the following commands, 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.
    Run **%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.
    Run **%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 native enterprise resource planning 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.

If SQL Server is running in a dedicated system or virtual machine, the recommended protocol is TPC/IP.

Tunings on SAP Application Server

- The ratio between the number of Dialog (D) processes versus Update (U) processes in the SAP enterprise resource planning installation might vary, but usually a ratio of 1D:1U or 2D:1U per logical processor is a good start for the Sales and Distribution 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, might be improved based on the analysis of previous experiments.
  For large virtual machines that span multiple NUMA nodes, virtual NUMA is available by default and can be used to partition dialog instances. For more information, see Virtualizing SAP Applications on Windows.

- For smaller virtual machines without virtual NUMA, the preferred NUMA node can be configured for each virtual machine for better load balancing and performance.

- To further partition computing resources 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. This provides better CPU and memory locality. The affinity setting in the SAP instance profiles is supported for 64 logical processors. The affinity setting is not recommended when running SAP worker processes inside a virtual machine because experiments have shown that SAP latency can be improved by allowing the guest and hypervisor schedulers to better allocate CPU resources.

- For large scale virtualized deployments, the selection of VMQ processors can be altered to better partition SAP virtual machines from root network I/O processing. Consider restricting the VMQ processors to logical processors (and NUMA nodes) on which contention with SAP virtual machines is minimized. This may improve latency for heavily loaded configurations.

- Use the flat memory model that SAP AG released in November 2006, with the SAP Note No. 1002587 (Flat Memory Model on Windows) for SAP kernel 7.00 patch level 87.

- In Windows Server 2012 and Windows Server 2008 R2, the operating system 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 following commands, 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. Run **%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. Run **%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 then set the "NodeOptions"=dword:00000100 value as follows:

```
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 in November 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
```

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 Map TCP/IP Ports to NUMA Nodes.

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 a non-virtualized SAP enterprise resource planning Sales and Distribution workload. Log the performance counters to a local (.blg) performance counter log. It is less expensive to collect all instances by using the wildcard character (*), and then extract particular instances while post-processing by using Relog.exe as follows:

\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.

For virtualized configurations of the SAP enterprise resource planning Sales and Distribution workload the following list of performance counters can be collected for performance monitoring of the Hyper-V host:

\Hyper-V Hypervisor Partition(*)*
\Hyper-V Hypervisor Root Partition(*)*
\Hyper-V Hypervisor Logical Processor(*)*
\Hyper-V Hypervisor Root Virtual Processor(*)*
\Hyper-V Hypervisor Virtual Processor(*)*
\Hyper-V Dynamic Memory Balancer(*)*
\Hyper-V Dynamic Memory VM(*)*
\Hyper-V VM Vid Numa Node(*)*
\Hyper-V VM Vid Partition(*)*
\PhysicalDisk(*)*
\Hyper-V Virtual Storage Device(*)*
\Network Interface(*)*
\Hyper-V Virtual Network Adapter(*)*
\Hyper-V Virtual Switch(*)*
\Hyper-V Virtual Switch Processor(*)*

Performance Tuning for OLTP Workloads

The TPC-E benchmark is one of the database workloads used to evaluate online transaction processing (OLTP) performance in SQL Server and Windows Server. TPC-E uses a central database that runs transactions simulating the activities of a brokerage firm. The primary metric for TPC-E is Trade-Result transactions per second (tpsE).

A non-clustered TPC-E benchmark setup consists of two parts: a set of client systems and the server under test. To achieve maximum system utilization and throughput, you can tune the operating system, SQL Server, storage, memory, processors, and network.

For more information about the TPC-E benchmark, see the Transaction Processing Performance Council website.

Server Under Test Tunings

The following are general tunings that are applicable across native and virtualized server root configurations for the database server:

- Set the power scheme to High 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.

The following are settings that are recommended for the **native**, non-virtualized configurations only:

- Configure page files for best performance:
    - Navigate to **Performance Settings > Advanced > Virtual memory** and configure one or more fixed-size page files with Initial Size equal to Maximum Size. The page file size should be equal to the total virtual memory requirement of the workload. Make sure that no system-managed page files are in the virtual memory on the application server.
    - Navigate to **Performance Settings > Visual Effects**, and then select **Adjust** for best performance.
- Configure network devices:
  For virtualized setups, consider utilizing VMQ and dynamic VMQ where applicable. The following tunings are applicable for native TPC-E only.
    - The number of network devices is determined from previous runs. Network device utilization should not be higher than 65%-75% of total network adapter bandwidth. Use 1 Gbps network adapters at minimum.
    - From the Device Manager MMC snap-in (Devmgmt.msc), navigate to **Network Adapters**, and then determine the network devices to be used. Disable devices that are not being used.

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

- Configure storage devices as follows:
  - Disable low priority I/O. For each logical volume in HKLM\SYSTEM\CurrentControlSet\Enum\SCSI under Device Parameters\ClassPnp, create the registry entry IdlePrioritySupported (REG_DWORD) and set the value to 0.
  - For advanced storage tuning information, see Performance Tuning for the Storage Subsystem earlier in this guide.
- Configure disks for advanced performance as follows:
  - From the Disk Management MMC snap-in (Diskmgmt.msc), select each disk in use, right-click **Properties,** click **Policies,** and then select **Advanced Performance** if it is enabled for the disk.

    Note This setting is for native configurations only (for direct attached storage). For virtualized setups, consider the recommendations that are provided in the **Storage I/O Performance** topic under Performance Tuning for Virtualization Servers earlier in this guide.

SQL Server Tunings for OLTP Workloads

The following SQL Server tunings can improve performance and scalability for workloads characterized by large memory usage, high transaction rates, and high CPU utilization.

Important: The tunings in this section are specifically for OLTP benchmarking and should not be perceived as general SQL tuning guidance.

- Use the **-T834** start flag to enable SQL Server to use large pages.
- Start SQL Server as a process instead of a service and use the **-x** flag for native operating system configurations to disable SQL Server performance counters and avoid potential overhead:
  - From the Services MMC snap-in (Services.msc), stop and disable SQL Server services.
  - Run the following command from the SQL Server binn directory:

```
sqlservr.exe -c -x -T661 -
T834
```

For virtualized setups, leave SQL Server to run as a service with the **Manual** startup type as follows:

net start MSSQLSERVER /x /T661 /T834

The purpose of each parameter is as follows:

-x : Disable SQL Server perfmon counters

-T661: Disable the ghost record removal process

-T834: Use Microsoft Windows large-page allocations for the buffer pool

Note Other parameters that are sometimes used are:

-T652: Disable page prefetching scans

-T8744: Disallow prefetch for ranges

- Enable the TCP/IP protocol to allow communication with client systems:

  Navigate to **Start Menu > Programs > Microsoft SQL Server 2012 > 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. For detailed information about the *sp_configure* stored procedure, see Server Configuration Options in the MSDN Library.

- Set the SQL processor affinity mask option 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 the SQL processor affinity mask option to not use a set of logical processors that handle I/O interrupt traffic for the network and the disk.

  You can set the SQL processor affinity mask option as follows, depending on processor count:

  - Partition SQL process to run 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.
  - In SQL Server 2008 R2 and SQL Server 2012, you can apply equivalent settings to configure CPU affinity on as many as 640 logical processors by using the ALTER SERVER CONFIGURATION Transact-SQL statement because 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 about best practices for installations on native operating system configurations with more than 64 logical processors, see ALTER SERVER CONFIGURATION (Transact-SQL).

  Note Windows Server 2012 supports guests with no more than 64 virtual processors, thus any SQL affinity masks for a virtualized SQL Server instance should be set accordingly.

- 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 remaining available memory, but not more.

  Use the following equation to calculate the 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 thread mode. Thread mode performance is comparable to fiber mode.

- Set the maximum worker threads value to approximately the number of connected users 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. Monitor the sys.dm_os_schedulers Dynamic Management Views to determine whether you need to increase the number of worker threads.
- Set the priority boost value to 1.
- Consider partitioning the entire system so that dedicated NUMA nodes perform storage and/or network I/O processings and the remaining the NUMA nodes run SQL Server process for the largest scale systems. (This is for native systems only, and it does not apply in a virtualized environment.)
    - For network I/O, choose the number of network adapters to use and number of RSS processors for each network adapter. Find the smallest number of NUMA nodes that can satisfy the RSS requirement. Assign RSS processors to these nodes. For more information, see Performance Tuning for the Networking Subsystem earlier in this guide.
    - For storage I/O, choose the number of processors to handle the interrupt for each storage host bus adapters (HBA). Each storage HBA will have its own dedicated interrupt target processor. Find the smallest number of NUMA nodes that can act as the interrupt targets for all storage HBAs. Assign the interrupt affinity of each HBA to processors in the chosen NUMA nodes.
    - For each network adapter or storage HBA, ideally, the chosen NUMA node to handle its I/O processing is local to the device.
    - For the rest of the NUMA node in the system, assign the SQL affinity.

Disk Storage Tunings

Tune the disk storage as follows:

- For 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.
- You can improve performance with proper write caching in the case of battery-backed write caching, which is able to avoid data loss in the case of power failure. Enable 100% write caching for the log disk.

TPC-E Database Size and Layout

The database size and layout can also be tuned for performance as follows:

- You can perform the following fine tuning on the database layout for rotational media:
    - 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.
- For solid-state drives, consider the following configuration: For each logical disk use 80% of available disk space for partitions and leave 20% unused.

Selecting a storage configuration for the virtualized TPC-E environment can present trade-offs between performance and factors such as portability or cost saving. Pass-through disks can be considered as a high performance option for virtualized workloads with high SQL I/O disk rates such as TPC-E.

When you consider VHD or VHDX disks, fixed VHDs can deliver better throughtput over differencing or dynamic VHDs. Based on the TPC-E database size, consider placing the entire TPC-E database in a single VHDX (for small TPC-E DB) and store the VHDX on high-performance storage such as solid-state drives.

Another option for larger TPC-E databases is to partition the TPC-E database into more than one VHD or VHDX file placed in different logical or physical storage units when high physical disk response times are observed.

Client Systems Tunings

Tune the client systems as follows:

- Configure client systems the same way that the server under test is configured. See Server Under Test 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 logical 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 the time is synchronized between the master client and the server under test.

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 (.blg) performance counter log. It is less expensive to collect all instances by using the wildcard character (*), and then extract particular instances while post-processing by using Relog.exe or Perfmon:

\IPv4*
\Memory*
\Network Interface(*)*
\PhysicalDisk(*)*
\Processor Information(*)*
\Synchronization(*)* for Windows Server 2012 and Windows Server 2008 R2

\SynchronizationNUMA(*)* for Windows Server 2012
\System*
\TCPv4*

\NUMA Node Memory(*)* for Windows Server 2012

Note If applicable, add the \IPv6* and \TCPv6* objects. To monitor overall performance, you can use the performance counter chart that is displayed in Figure 17 and the throughput chart displayed in Figure 18 to visualize run characteristics.

The first part of the run in Figure 17 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.

Figure 17: TPC-E Perfmon counters chart

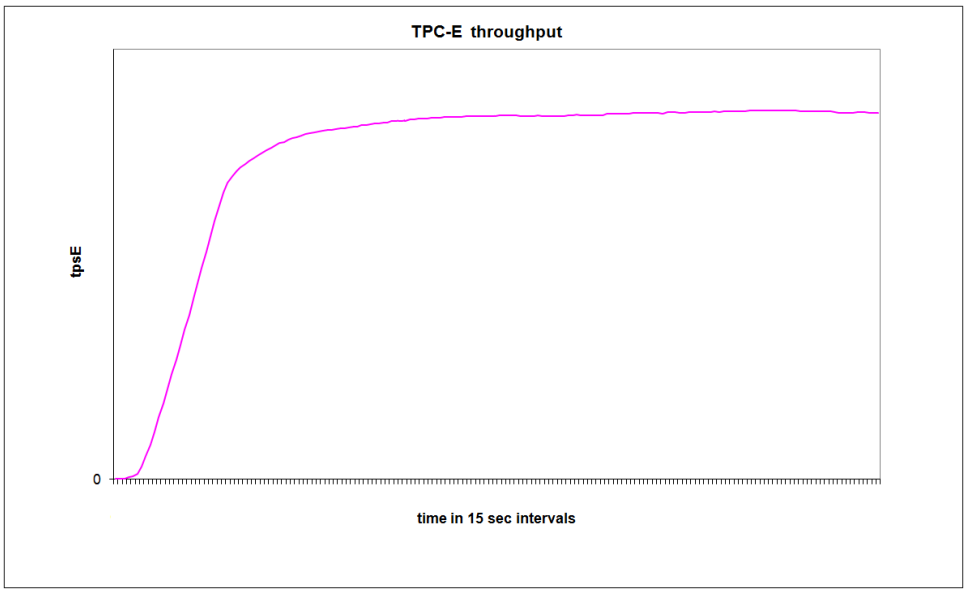

Figure 18. TPC-E throughput chart

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

Root Counters

The following list of performance counters is considered a base set of counters when you monitor the resource usage from the root operating system:

\Cache*
\Hyper-V Hypervisor Logical Processor(*)*
\Hyper-V Hypervisor Partition(*)*
\Hyper-V Hypervisor Root Partition(*)*
\Hyper-V Hypervisor Root Virtual Processor(*)*
\Hyper-V Hypervisor Virtual Processor(*)*
\Hyper-V Hypervisor*
\Hyper-V Virtual IDE Controller (Emulated)(*)*
\Hyper-V Virtual Machine Bus*
\Hyper-V Virtual Storage Device(*)*
\Hyper-V Virtual Switch Port(*)*
\Hyper-V VM Vid Numa Node(*)*
\Hyper-V VM Vid Partition(*)*
\NUMA Node Memory(*)*
\IPv4*
\Memory*
\Network Interface(*)*
\Per Processor Network Activity Cycles(*)*
\Per Processor Network Interface Card Activity(*)*
\PhysicalDisk(*)*
\Processor Information(*)*
\SynchronizationNuma(*)*
\System*
\TCPv4*

If needed, you can collect (from the context of a guest) information such as:

- SQL Server performance counters
- Memory utilization
- Physical disk size

Resources

Websites

Windows Server 2012
http://www.microsoft.com/en-us/server-cloud/windows-server/2012-default.aspx

Windows Server 2008 R2
http://www.microsoft.com/windowsserver2008/R2.aspx

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/

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

NT Testing TCP Tool (NTTTCP) 3.0
http://msdn.microsoft.com/en-us/windows/hardware/gg463264.aspx

Web Capacity Analysis Tool (WCAT)
http://www.iis.net/community/default.aspx?tabid=34&g=6&i=1466

File Server Capacity Tool (FSCT)
http://www.microsoft.com/download/details.aspx?id=27284

Windows Server 2012 NIC Teaming (LBFO) Deployment and Management
http://go.microsoft.com/fwlink/?LinkId=215654

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

Storage Subsystem

Disk Subsystem Performance Analysis for Windows
(Note: 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

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

What's New in Hyper-V
http://technet.microsoft.com/library/hh831410.aspx

About Virtual Machines and Guest Operating Systems
http://technet.microsoft.com/library/cc794868(v=ws.10)

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 (Transact-SQL)
http://msdn.microsoft.com/library/ee210585.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

Server Tuning Tools

Server Performance Advisor 3.0
http://msdn.microsoft.com/windows/hardware/hh367834.aspx