Performance Tuning Guidelines for Windows Server 2012 R2

Copyright information

Contents

Performance Tuning Guidelines for Windows Server 2012 R2

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 topic provides a set of guidelines that you can use to tune the server settings in Windows Server 2012 R2 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 topic describes important tuning considerations and settings that can result in improved performance or energy efficiency. It also 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 R2, Windows Server 2012, and Windows Server 2008 R2 to maximize energy efficiency with minimal effects on performance. 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 and Windows Server 2012 R2. Be sure to use the latest tuning guidelines to avoid unexpected results.

This topic is split into the following sections:

Performance Tuning for Server Hardware

You should 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.

Note

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. For more info about these tradeoffs, see Server Hardware Power Considerations in this section.

For more performance tuning guidelines, see Performance Tuning Guidelines for Windows Server 2012 R2.

In this section:

- Server Hardware Performance Considerations
- Server Hardware Power Considerations

See Also

Performance Tuning Guidelines for Windows Server 2012 R2

Server Hardware Performance Considerations

The following table 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.

Component	Recommendation
Processors	Choose 64-bit processors for servers. 64-bit processors have significantly more address space, and are required for Windows Server 2012 R2. No 32-bit editions of the operating system will be provided, but 32-bit applications will run on the 64-bit Windows Server 2012 R2 operating system. To increase the computing resources in a server, you can use a processor with higher-frequency cores, or you can increase the

Component	Recommendation
	number of processor cores. If CPU is the limiting resource in the system, a core with 2x frequency typically provides a greater performance improvement than two cores with 1x frequency. Multiple cores are not expected to provide a perfect linear scaling, and the scaling factor can be even less if hyper-threading is enabled because hyper-threading relies on sharing resources of the same physical core. **Important** Make sure 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. For Hyper-V, make sure that the processor supports SLAT (Second Level Address Translation). It is implemented as Extended Page Tables (EPT) by Intel and Nested Page Tables (NPT) by AMD. You can verify this feature is present by using SystemInfo.exe on your server.
Cache	Choose large L2 or L3 processor caches. On newer architectures, such as Haswell, there is a unified Last Level Cache (LLC) or an L4. The larger caches generally provide better performance, and they often play a bigger role than raw CPU frequency.
Memory (RAM) and paging storage	Increase the RAM to match your memory needs. When your computer runs low on memory and it needs more immediately, Windows uses 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

Component	Recommendation
	guidelines for page file placement: • Isolate the page file on its own storage device, 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. 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 R2, the primary storage and network interfaces should be PCI Express (PCIe) so servers with PCIe buses are recommended. To avoid bus speed limitations, use PCIe x8 and higher slots for 10 GB 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. Consolidating small drives into fewer high-capacity drives can reduce overall storage

Component	Recommendation
	performance. Fewer spindles mean reduced request service concurrency; and therefore, potentially lower throughput and longer response times (depending on the workload intensity). The use of SSD and high speed flash disks is useful for read mostly disks with high I/O rates or latency sensitive I/O. Boot disks are good candidates for the use of SSD or high speed flash disks as they can improve boot times significantly.

The following table 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.

Recommendation	Description
A certified adapter	The adapter has passed the Windows Hardware 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.
Copper and fiber 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

Recommendation	Description
	adapters 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 enables packet receive-processing to scale with the number of available computer processors. This is particularly important with 10 GB Ethernet and faster.
Offload capability and other advanced features such as message-signaled interrupt (MSI)-X	Offload-capable adapters offer CPU savings that yield improved performance.
Dynamic interrupt and deferred procedure call (DPC) redirection	In Windows Server 2012 R2, Numa I/O enables PCIe storage adapters to dynamically redirect interrupts and DPCs and can help any multiprocessor system by improving workload partitioning, cache hit rates, and on-board hardware interconnect usage for I/O-intensive workloads.

See Also

Performance Tuning for Server Hardware

Server Hardware Power Considerations

In this topic:

- Power considerations and recommendations
- Processor terminology
- Power and performance tuning

Power considerations and recommendations

It is 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. The following table lists guidelines for power characteristics and capabilities of server hardware components.

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. For more info on cooperative power management techniques, see the section named Collaborative Processor Performance

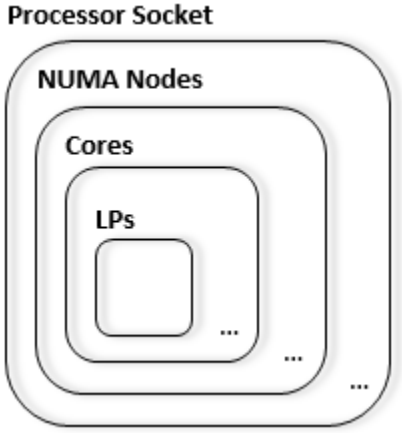
Component	Recommendation
	Control in the Advanced Configuration and Power Interface Specification.
Memory (RAM)	Memory accounts for an increasing fraction of the total system power. Many factors affect the energy consumption of a memory DIMM, such as memory technology, error correction code (ECC), bus 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. SSD drives are more power efficient than rotational drives. Also, 2.5-inch drives generally require less power than 3.5-inch drives. For more info about the energy costs for different RAID configurations, see Performance Tuning for Storage Subsystems.
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	Improving 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.

Component	Recommendation
USB devices	Windows Server 2012 R2 enables selective suspend for USB devices by default. However, a poorly written device driver can still disrupt system energy efficiency by a sizeable margin. To avoid potential issues, disconnect USB devices, disable them in the BIOS, or choose servers that do not require USB devices.
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 topic reflects the hierarchy of components available in the following figure. Terms used from largest to smallest granularity of components are the following:

- Processor socket
- NUMA node
- Core
- Logical processor



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 R2 is optimized for excellent energy efficiency with minimum performance impact across a wide range of customer workloads. **Processor Power Management (PPM) Tuning for the Windows Server Balanced Power Plan** describes the workloads used for tuning the default parameters in Windows Server 2012 R2, and provides suggestions for customized tunings. This section expands on 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 R2.

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.

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

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

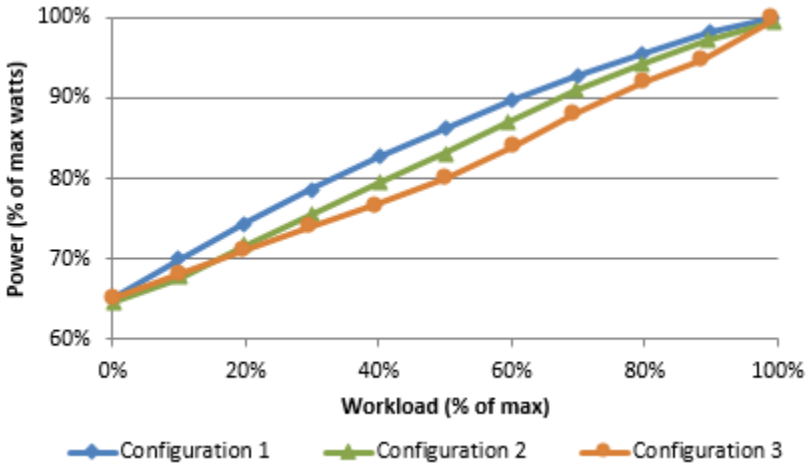
You should establish a baseline power measurement before you tune your server for energy efficiency.

If your server has the necessary support, you can use the power metering and budgeting features in Windows Server 2012 R2 to view system-level energy consumption by using Performance Monitor. 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 Hardware Certification 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 then 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) to generate a load line. The following figure shows load lines for three sample configurations:



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 the best configuration is. 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

PowerCfg.exe supports a command-line option that you can use to analyze the idle energy efficiency of your server. When you run PowerCfg.exe 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 apps are closed before you run PowerCfg.exe.

Note

PowerCfg.exe was introduced in 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 info about PowerCfg.exe, see Using PowerCfg to Evaluate System Energy Efficiency.

Using power plans in Windows Server

Windows Server 2012 R2 has three built-in power plans designed to meet different sets of business needs. These plans provide a simple way for you to customize a server to meet power or performance goals. The following table describes the plans, lists the common scenarios in which to use each plan, and gives some implementation details for each plan.

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 apps • App code that is sensitive to processor performance changes	Processors are always locked at the highest performance state (including "turbo" frequencies). All cores are unparked. Thermal output may be significant.
Power Saver	Limits performance to save energy and reduce operating cost. Not recommended without thorough testing to make sure performance is adequate.	• Deployments with limited power budgets • Thermal constraints	Caps processor frequency at a percentage of maximum (if supported), and enables other energy-saving features.

These power plans exist in Windows for alternating current (AC) and direct current (DC) powered systems, but we will assume that servers are always using an AC power source.

For more info on power plans and power policy configurations, see Power Policy Configuration and Deployment in Windows.

Note

Some server manufactures have their own power management options available through the BIOS settings. If the operating system does not have control over the power management, changing the power plans in Windows will not affect system power and performance.

Tuning processor power management parameters

Each power plan 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.

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 R2 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

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, the Processor Performance Boost Mode parameter must be configured by the administrator or by the default parameter settings for the chosen power plan. 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 R2, the default value for Boost Mode is 3.

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 enable Processor Performance Boost Mode 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 that Power Saver is 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 Processor Performance State** 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 (e.g., for repeatable testing), 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 Performance Monitor counter **% of maximum frequency** in the **Processor** group to see if any frequency caps were applied.

Processor performance increase and decrease of thresholds and policies

The speed at which a processor performance state increases or decreases is controlled by multiple parameters. The following four parameters have the most visible impact:

- **Processor Performance Increase Threshold** defines the utilization value above which a processor's performance state will increase. Larger values slow the rate of increase for the performance state in response to increased activities.
- **Processor Performance Decrease Threshold** defines the utilization value below which a processor's performance state will decrease. Larger values increase the rate of decrease for the performance state during idle periods.
- **Processor Performance Increase Policy and Processor Performance Decrease** Policy determine which performance state should be set when a change happens. "Single" policy means it chooses the next state. "Rocket" means the maximum or minimal power performance state. "Ideal" tries to find a balance between power and performance.

For example, if your server requires ultra-low latency while still wanting to benefit from low power during idle periods, you could quicken the performance state increase for any increase in load and slow the decrease when load goes down. The following commands set the increase policy to "Rocket" for a faster state increase, and set the decrease policy to "Single". The increase and decrease thresholds are set to 10 and 8 respectively.

```
Powercfg.exe -setacvalueindex scheme_current sub_processor PERFINCPOL 2
Powercfg.exe -setacvalueindex scheme_current sub_processor PERFDECPOL 1
Powercfg.exe -setacvalueindex scheme_current sub_processor PERFINCTHRESHOLD 10
Powercfg.exe -setacvalueindex scheme_current sub_processor PERFDECTHRESHOLD 8
Powercfg.exe /setactive scheme_current
```

Powercfg.exe -setacvalueindex scheme_current sub_processor PERFINCPOL 2
Powercfg.exe -setacvalueindex scheme_current sub_processor PERFDECPOL 1
Powercfg.exe -setacvalueindex scheme_current sub_processor PERFINCTHRESHOLD 10
Powercfg.exe -setacvalueindex scheme_current sub_processor PERFDECTHRESHOLD 8
Powercfg.exe /setactive scheme_current

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 parked 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 cores are responsible for the remainder of the workload. Core parking can potentially increase energy efficiency during lower usage

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 R2.

One scenario that core parking isn't always optimal for 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 may not always 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 Maximum 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 R2 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 plan 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
```

See Also

Performance Tuning for Server Hardware

Performance Tuning for Subsystems

This section describes performance tuning guidelines for the following subsystems:

- Performance Tuning for Network Subsystems
- Performance Tuning for Storage Subsystems
- Performance Tuning for Cache and Memory Manager Subsystems

For more performance tuning guidelines, see Performance Tuning Guidelines for Windows Server 2012 R2.

Server Performance Advisor 3.1

Microsoft Server Performance Advisor (SPA) 3.1 helps IT administrators collect metrics to identify, compare, and diagnose potential performance issues in a Windows Server 2012 R2, 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.

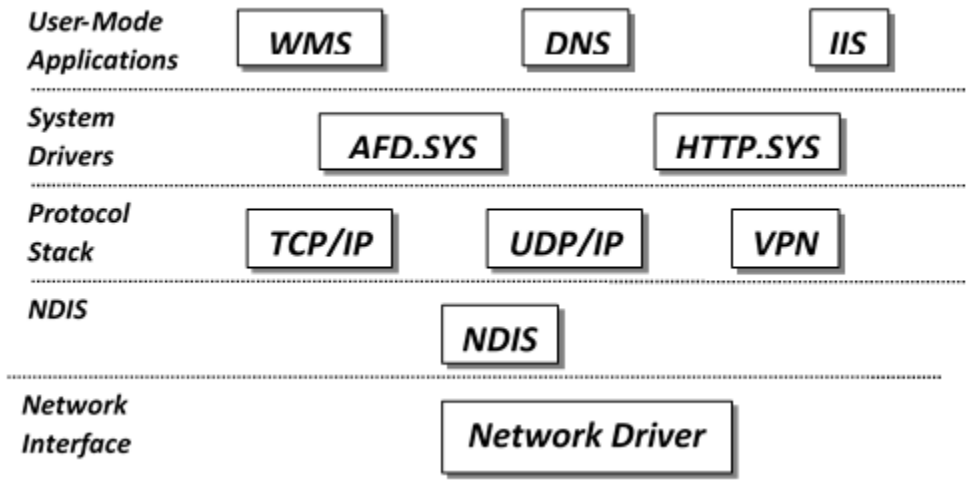
For more info about SPA, see **Microsoft Server Performance Advisor**.

See Also

Performance Tuning Guidelines for Windows Server 2012 R2

Performance Tuning for Network Subsystems

The following figure shows the network architecture, which includes many components, interfaces, and protocols.



In this section:

- Using NIC Teaming for Network Subsystem Performance
- Choosing a Network Adapter for Network Subsystem Performance
- Tuning a Network Adapter for Network Subsystem Performance

See Also

Performance Tuning for Subsystems

Using NIC Teaming for Network Subsystem Performance

NIC teaming, also known as load balancing and failover (LBFO), allows multiple network adapters on a computer to be placed into a team for the following purposes:

- Bandwidth aggregation
- Traffic failover to prevent connectivity loss in the event of a network component failure

This feature has been a requirement for independent hardware vendors (IHVs) to enter the server network adapter market. However, NIC teaming was not included in the operating system until Windows Server 2012.

In this topic:

NIC teaming configuration

There are two basic configurations for NIC teaming.

- **Switch-independent teaming** This configuration does not require the switch to participate in the teaming. Since in switch-independent mode the switch does not know that the network adapter is part of a team in the host, the adapters may be connected to different switches. Switch independent modes of operation do not require that the team members connect to different switches; they merely make it possible.
- **Switch-dependent teaming** This configuration requires the switch to participate in the teaming. Switch dependent teaming requires all the members of the team to be connected to the same physical switch.

Algorithms for load distribution

Outbound traffic can be distributed among the available links in many ways. One rule that guides any distribution algorithm is to try to keep all packets associated with a single flow (TCP-stream) on a single network adapter. This rule minimizes performance degradation caused by reassembling out-of-order TCP segments.

NIC teaming in Windows Server 2012 R2 supports the following traffic load distribution algorithms:

- **Hyper-V switch port** Since virtual machines have independent MAC addresses, the virtual machine's MAC address or the port it's connected to on the Hyper-V switch can be the basis for dividing traffic.
- **Address Hashing** This algorithm creates a hash based on address components of the packet and then assigns packets that have that hash value to one of the available adapters.

Usually this mechanism alone is sufficient to create a reasonable balance across the available adapters.

- **Dynamic** This algorithm takes the best aspects of each of the other two modes and combines them into a single mode. Outbound loads are distributed based on a hash of the TCP ports and IP addresses. Dynamic mode also rebalances loads in real time so that a given outbound flow may move back and forth between team members. Inbound loads are distributed as though the Hyper-V port mode was in use. Dynamic mode was added in Windows Server 2012 R2.

Compatibility

NIC teaming is compatible with all networking capabilities included in Windows Server 2012 R2 with five exceptions: SR-IOV, RDMA, Native host Quality of Service, TCP Chimney, and 802.1X Authentication.

NIC teaming configuration details and Windows PowerShell commands can be found in the Windows Server 2012 R2 NIC Teaming (LBFO) Deployment and Management guide.

See Also

Performance Tuning for Network Subsystems

Choosing a Network Adapter for Network Subsystem Performance

Network-intensive applications require high-performance network adapters. This topic explores some considerations for choosing network adapters.

In this topic:

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. The following table provides details about each offload capability.

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.
SR-IOV	Single root I/O virtualization (SR-IOV) allows direct assignment of network resources (virtual functions) to individual virtual machines when the network adapter supports this feature.

Receive-Side Scaling (RSS)

Windows Server 2012 R2, 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 R2, 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 R2 provides the following ways to tune RSS behavior:

- Windows PowerShell cmdlets: Get-NetAdapterRSS, Set-NetAdapterRSS, Enable-NetAdapterRss, and Disable-NetAdapterRss. For more info, see Network Adapter Cmdlets in Windows PowerShell.

  These cmdlets allow you to see and modify RSS parameters per network adapter. Pass the cmdlet name to **Get-Help** for more info.
- 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 things like 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 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. Use the **Get-NetAdapterRss** Windows PowerShell cmdlet to verify that the RSS settings have taken effect.

```
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 the indirection table. 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 if the SR-IOV NIC and virtual function driver supports VF RSS capability. In this case, the guest will have the benefit of RSS. Note that the host does not get RSS capability because the host is bypassed by SR-IOV.

NIC teaming 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 use RSS cmdlets on the teamed network adapter.

Receive Segment Coalescing (RSC)

Receive Segment Coalescing (RSC) helps performance in Windows Server 2012 R2 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 R2, 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

You can diagnose RSC by using the Get-NetAdapterRsc Windows PowerShell cmdlet. The following tables show some sample data from the output:

Name	IPv4Enabled	IPv4Operational	IPv4FailureReason
Ethernet	True	True	NoFailure

Name	IPv6Enabled	IPv6Operational	IPv6FailureReason
Ethernet	False	False	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 the 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 and 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 and Windows Server 2012 R2, configuration has been simplified by the use of the following Windows PowerShell cmdlets:

- Get-NetAdapterAdvancedProperty
- Set-NetAdapterAdvancedProperty
- Enable-NetAdapter
- Enable-NetAdapterBinding
- Enable-NetAdapterChecksumOffload
- Enable-NetAdapterLso
- Enable-NetAdapterIPSecOffload
- Enable-NetAdapterPowerManagement
- Enable-NetAdapterQos
- Enable-NetAdapterRDMA
- 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, 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

The following table lists the high-performance network adapter features that can improve performance in terms of throughput, latency, or scalability for some 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

Caution

The recommendations in the table above 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.

See Also

Performance Tuning for Network Subsystems

Tuning a Network Adapter for Network Subsystem Performance

You can optimize network throughput and resource usage by tuning the network adapter, if any tuning options are exposed by the adapter.

Note

The correct tuning settings depend on the network adapter, the workload, the host computer resources, and your performance goals.

In this topic:

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.

Important
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 by using the **Advanced Properties** tab of the network adapter. You can also see if a network adapter is RSS-capable by using the **Advanced Properties** tab.

RSS profiles and RSS queues

RSS Profiles was introduced 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 are 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.

RSS and VMQ

Most network adapters have queues that can be used for either RSS or VMQ, but not both at the same time. Therefore, some VMQ settings appear to be settings for RSS queues but are really settings on the generic queues that both RSS and VMQ use depending on which feature is presently in use. When a network adapter is connected to vSwitch, RSS is automatically disabled.

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

  Note
  This is system and BIOS dependent, and some systems will provide higher performance if the operating system controls power management.

- Set the operating system power management profile to **High Performance System**.

  Note
  This will not work properly if the system BIOS has been set to disable operating system control of power management.

- Enable Static Offloads, such as 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.

Deprecated TCP parameters

The following registry settings, under HKLM\System\CurrentControlSet\Services\Tcpip\Parameters, from Windows Server 2003 are no longer supported, and are ignored in later versions:

- TcpWindowSize
- NumTcbTablePartitions
- MaxHashTableSize

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
- 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

See Also

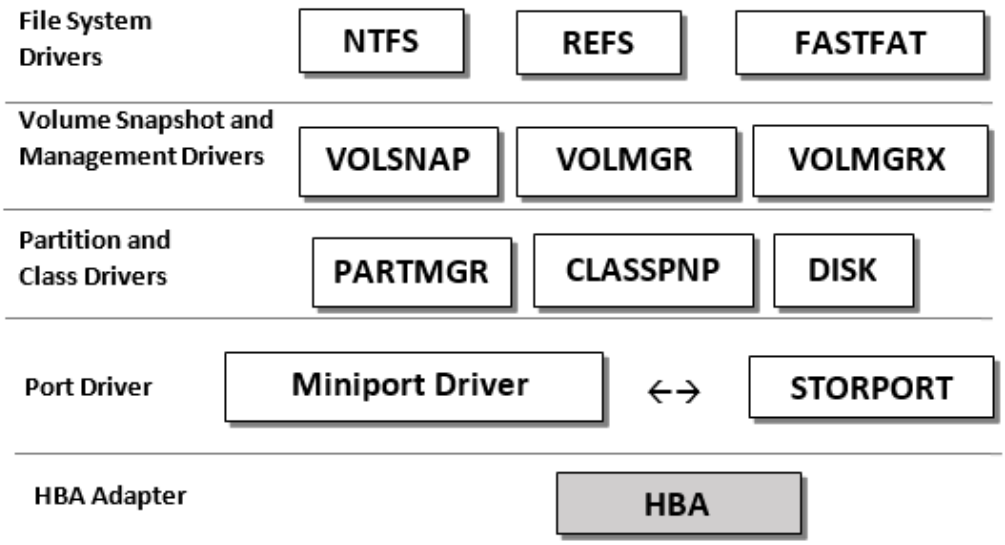
Performance Tuning for Network Subsystems

Performance Tuning for Storage Subsystems

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.

The following figure shows the storage architecture, which includes many components in the 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.

In this section:

- Choosing Storage for Storage Subsystem Performance
- Using Storage Spaces for Storage Subsystem Performance
- Using Storage-related Parameters and Performance Counters
- Using Storage Drivers for Storage Subsystem Performance
- Performance Tuning for Data Deduplication

See Also

Performance Tuning for Subsystems

Choosing Storage for Storage Subsystem Performance

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

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

In this topic:

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 any 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. The following describes some factors that affect server storage capacity:

Factor	Required storage capacity
Operating system files	At least 15GB of space. To provide space for optional components, future service packs, and other items, plan for an additional 3 GB – 5 GB for the operating system volume. A Windows Server installation can require even more space for temporary files.
Page file	A page file managed by the system is usually good enough. If you want to be sure that full memory dumps can be generated without configuring a dedicated dump file you can set both minimum and maximum size of the pagefile to the size of physical RAM + 50 MB and place the pagefile on any partition. Kernel memory dump requires less space and actual size varies and can be determine through trial and error. If size of the page file is too small, system generates a minidump and will log an event in the System event log during boot to let you know about this condition. For more on how to determine a page file size, see How to determine the appropriate page file size for 64-bit versions of Windows Server 2008 and Windows Server 2008 R2.
Memory dump	In Windows 7 and Windows Server 2008 R2 and later, the paging file does not have to be on the same partition as the partition on which the operating system is installed to get a memory dump. The requirement for a successful dump file creation is to have either a page file present on any partition or dedicated dump file configured if system runs with small page or without page file at all. For more info, see How to generate a kernel or a complete memory dump file in Windows Server 2008 and Windows Server 2008 R2. In Windows Server 2012 and later, you can enable a dump creation log file to troubleshoot issues with creating a dump file by using the following registry setting: HKLM\SYSTEM\CurrentControlSet\Control\CrashControl\EnableLogFile to 1 (DWORD)
Applications	Varies according to the application.

Factor	Required storage capacity
	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 apps 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 info, see **Hardware array capabilities**.
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. The following figure illustrates some storage deployment models that are available in Windows Server 2012 R2.

Alternatively, Windows Server 2012 introduced a new technology called Storage Spaces, which provides platform storage virtualization and is further enhanced in Windows Server 2012 R2. This enables customers to deploy storage solutions that are cost-efficient, highly-available, resilient, and performance by using commodity SAS/SATA hard drives and JBOD enclosures. For more info, see Using Storage Spaces for Storage Subsystem Performance.

The following table describes some of the options and considerations for a traditional storage array solution.

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, enterprise storage controllers should offer resiliency capabilities.
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. The mix of read and write operations as well as the IO size may also affect storage bandwidth. For example, a storage array may provide a higher I/O rate with a mix of 10% read operations compared to a mix of 50% read and 50% write operations. I/O size can also affect bandwidth with larger I/Os generally providing more bandwidth than smaller I/Os.

Understanding 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 uninterruptible power supplies or batteries is 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.

The following table lists some common storage array options.

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. Spanning 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.
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

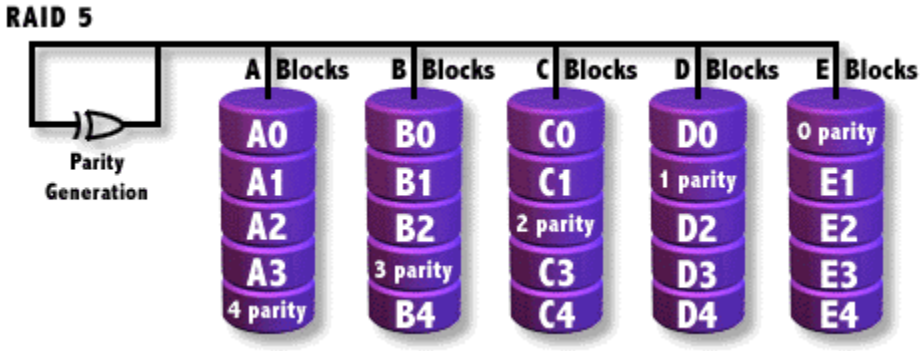
Option	Description
	reduced by write logging or battery-backed write caching, or by relaxing the requirement for dual write completions before returning the I/O completion notification. Mirroring has the potential to provide faster read operations than striping because it can (with a sufficiently intelligent 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.

Option	Description
	The cost and power characteristics are similar to those of mirroring.
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 the figure below. 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 blocks 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

Option	Description
	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. (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

Option	Description
	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. The following figure 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.



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. The following table describes common

configurations and their relative performance, reliability, availability, cost, capacity, and energy consumption.

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 unless workload has replication or resiliency mechanism • Single loss affects the entire array	**Cons** • Single loss prevents access to entire array unless workload has replication or resiliency mechanism	**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 implementation	**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
Rotated parity or parity disks (RAID 5) **Requirements** • One additional disk • Three-disk minimum	**Pros** • Balanced load • Potential for better read response times, throughput, and concurrency • Good performance when used with a write-back cache **Cons** • Up to 75% write	**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 reconstructio n, 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
	performance reduction compared to RAID 0 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 or battery used			
Multiple parity or double parity disks (RAID 6) **Requirements** • Two additional disks • Five-disk minimum	**Pros** • Balanced load • Potential for better read response times, throughput, and concurrency • Good performance when used with a write-back cache **Cons** • Up to 83%	**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	**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	**Pros** • Only two more disks to power **Cons** • Up to six times the power for write requests (excluding the idle power)

Configuration	Performance	Reliability	Availability	Cost, capacity, and power
	write performance reduction compared to RAID 0 because of multiple read-modify-write • 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 or battery used	until reconstructed	stopped	

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. Also helpful for other workloads with their own resiliency or replication mechanisms, such as some Microsoft Exchange Server configurations.
- Mirroring (RAID 1): It is always recommended to use striped mirroring (RAID 10) for workloads that involve more than 2 spindles. For only 2 spindles, RAID 1 is necessary, and can be used for 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, database logs and high-concurrency workloads.
- Rotated parity or parity disks (RAID 5): Web pages, data archives, semi-critical 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, data archives, 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).

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. You may need to add more HBAs in the system as throughput may exceed the capability of an individual storage or network HBA.
- 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 sequentially 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. 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 heterogeneous storage, but Storage Spaces provides a more robust implementation.

See Also

Performance Tuning for Storage Subsystems

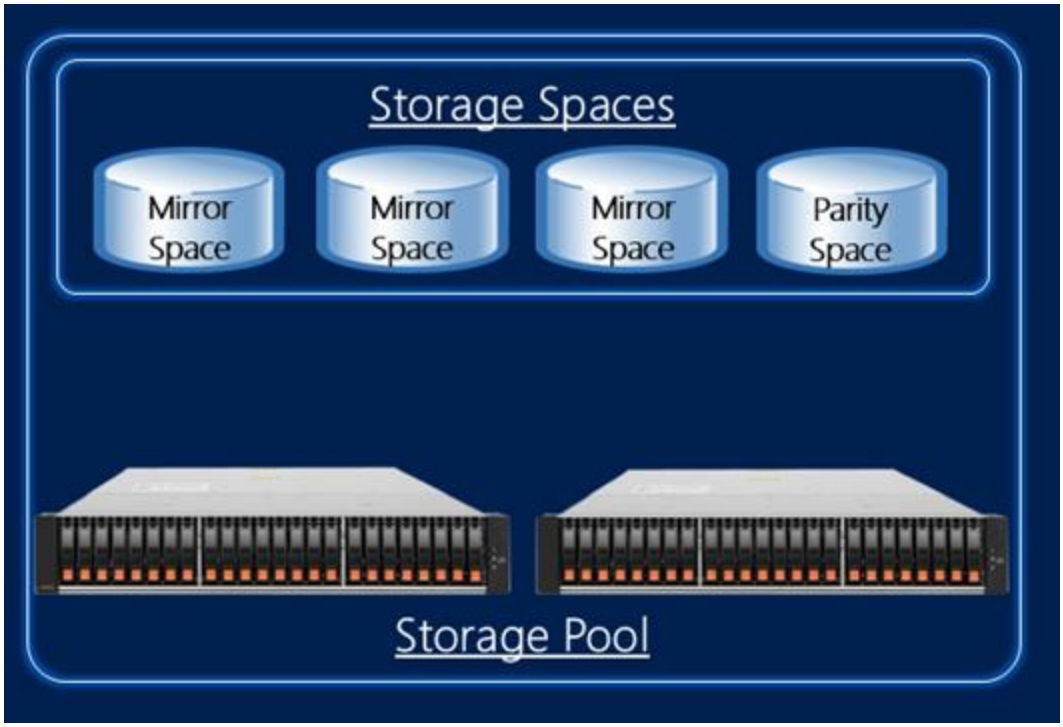
Using Storage Spaces for Storage Subsystem Performance

Windows Server 2012 introduced 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.

With Storage Spaces, there are two basic primitives which are used to manage the deployment:

- **Storage Pool** Aggregates and isolates any set of physical disks into a single point of management. A Storage Pool can contain up to 240 disks across number of enclosures, as long as the storage node has persistent communication with all the disks in the pool.
- **Storage Space** Carved out from the Storage Pool given a number of options, including most basically the desired resiliency setting and the desired capacity. The Storage Spaces driver determines the optimal physical data layout on the physical disks given these options to optimize for performance and resiliency (if resiliency was desired). Storage space is the unit of data access for the workload, and it is presented to Windows as a normal disk which can be initialized and formatted with a file system, such as NTFS or ReFS.

The following is a high-level diagram on Storage Spaces:



In this topic:

Storage Spaces resiliency options

Storage Spaces provides a number of resiliency options and mechanisms to protect against physical disk loss

Resiliency option	Description
Simple (no resiliency)	Data layout with this resiliency option is similar to striping as described in a prior section.
Mirror (resilient to one or two disk failures)	Data layout with this resiliency option is similar to striped mirroring as described in a prior section. Storage Spaces further has dirty region tracking for mirrored spaces to ensure that upon power failure, any in-flight updates to metadata are logged to ensure that the storage space can redo/undo operations to bring the storage space back into a resilient and consistent manner when power is restored and the system comes back up. This mechanism does not require battery backup.
Parity (resilient to one or two disk failures)	Data layout with single disk parity resiliency option is similar to parity as described in a prior section. With dual disk parity, conceptually the data layout is similar as described in a prior section, however the layout algorithm is different and is more optimal for single disk rebuild times than with many existing dual parity implementations. **Note** The dual parity resiliency option is new with Windows Server 2012 R2 and

Resiliency option	Description
	Windows 8.1 With either single or dual parity resiliency option, Storage Spaces maintains an on-disk journal which stages operations to the storage space which protects against power failures without the need for a battery. Larger sized journals, up to 16GB, can be configured in the presence of sufficient SSDs, and can be used to stage all writes which will improve performance of the virtual disk. Single-parity requires 2 SSDs, and dual-parity requires 3 SSDs.

Storage Spaces write-back cache (WBC)

Many enterprise workloads have bursts of random write I/O which can result in periodically high latencies to the underlying storage. For simple and mirrored spaces, a per-space persistent cache can be configured which uses available SSD capacity in the pool to stage these small write bursts, which are then destaged to the HDDs in a more optimal sequential manner.

By default, the system will configure a simple or mirrored storage space with 1GB of WBC (if there are sufficient SSDs and capacity in the storage pool). By using Windows PowerShell, you can choose an alternate WBC size at create time through the following option. It is recommended to keep the size of the WBC under 10GB in clustered deployments, and under 16GB in non-clustered deployments.

```
New-VirtualDisk <other settings> -WriteCacheSize 4GB
```

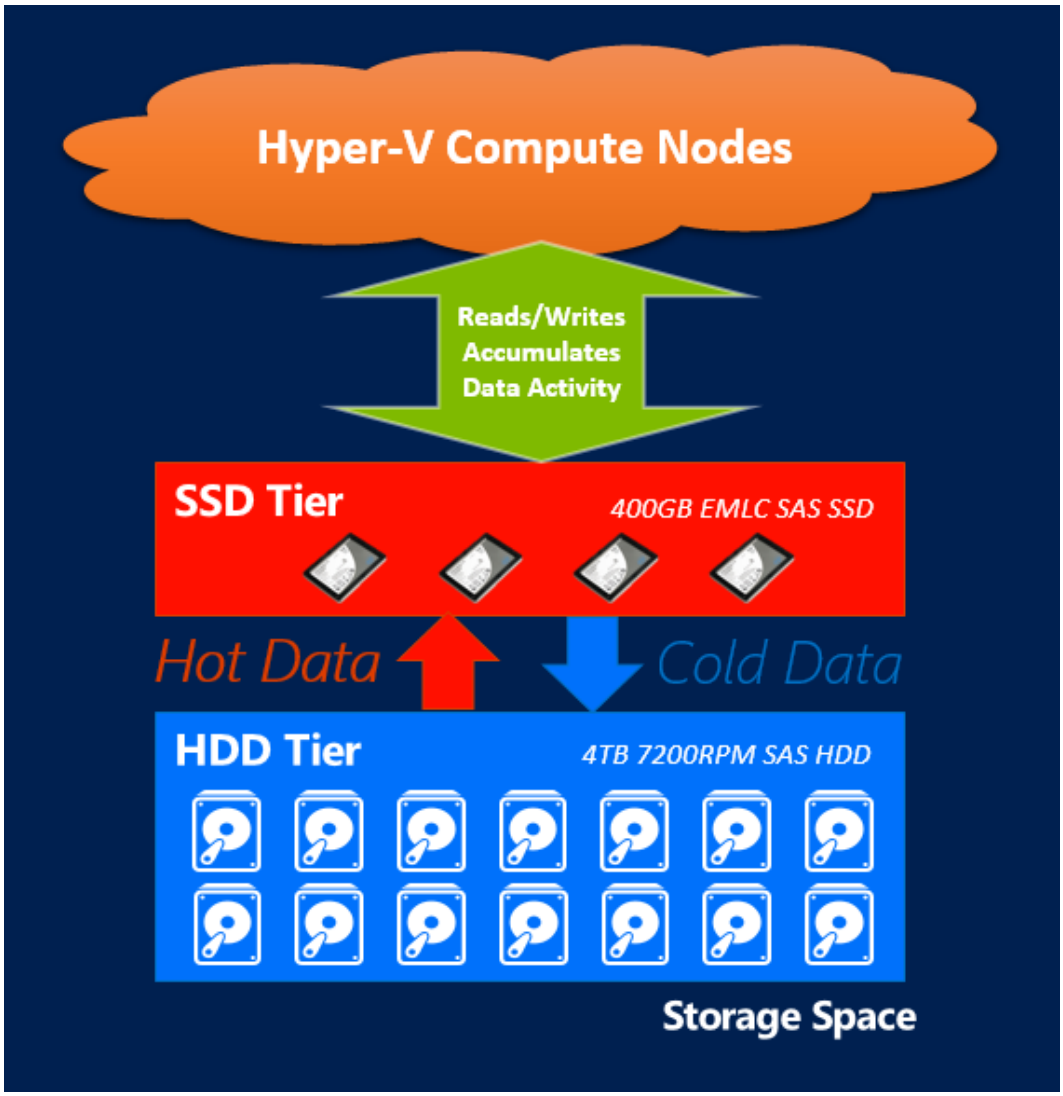
This capability is new with Windows Server 2012 R2 and Windows 8.1, and the size of the WBC cannot be changed after the Storage Space is created.

Storage Spaces automated data tiering (tiered storage)

Many enterprise workloads comprise of a large data set, where a majority of the data is not in active use, and only minority of the data is in active use (the working set). The working set changes over time. Given these characteristics, it is then natural to match the types of drives available in the market to the workload:

- Near-line drives with 7200 RPM rotational rate provide high capacity at low cost are natural to match with the majority cold data. Using these drives bring the $/TB of the deployment down.

- Solid state drives with flash memory provide high performance, and are natural to match a small number of these drives at higher cost to the minority working set. Using these types of drives increase the IOPS/$ of the deployment.



Having a mix of these types of drives will provide both capacity and performance for workloads, at far lower costs than can be achieved separately.

With Windows Server 2012 R2, Storage Spaces includes the capabilities to automatically place data onto the appropriate tier of storage according to its measured data activity, where frequently accessed data is placed on the SSD device tier, and less frequently accessed data is placed on the HDD device tier. The data activity is transparently measured by the file system (supported on NTFS or ReFS), and the data movement and tier processing is a configurable scheduled task which by default occurs at 1AM daily.

You can override the automatic placement based on heat, and assign whole files to be placed on the SSD or the HDD tier. Example workload scenarios for this would be a pooled virtual machine,

where the parent VHD is placed on the SSD tier which improves boot times. This override is done by using Windows PowerShell:

```
Set-FileStorageTier –DesiredStorageTierFriendlyName Mirror_VD1_SSD_Tier –FilePath
C:\ClusterStorage\Volume1\VHD\parent.vhdx
```

While processing of this request occurs automatically during the daily configurable 1AM scheduled task, users can immediately start processing of the tiered volume with the following Windows Powershell command:

```
Optimize-Volume –DriveLetter D –TierOptimize
```

Users can also determine the efficiency of tiering by looking at the Tiering Analysis report that is outputted when the command completes.

The task is configurable by using Task Scheduler:

```
Task scheduler -> \Microsoft\Windows\Storage Tiers Management\Storage Tiers Optimization
task
```

When purchasing storage for a tiered deployment, we recommend the following number of SSDs in a completely full disk enclosure of different bay capacities in order to achieve optimal performance for a diverse set of workloads:

	Minimum number of SSDs Recommended for Different Resiliency Settings		
Disk enclosure slot count	Simple space	2-way mirror space	3-way mirror space
12 bay	2	4	6
24 bay	2	4	6
60 bay	4	8	12
70 bay	4	8	12

For more information on using Windows PowerShell cmdlets to manage Storage Tiering, see Step-by-step for Storage Spaces Tiering in Windows Server 2012 R2.

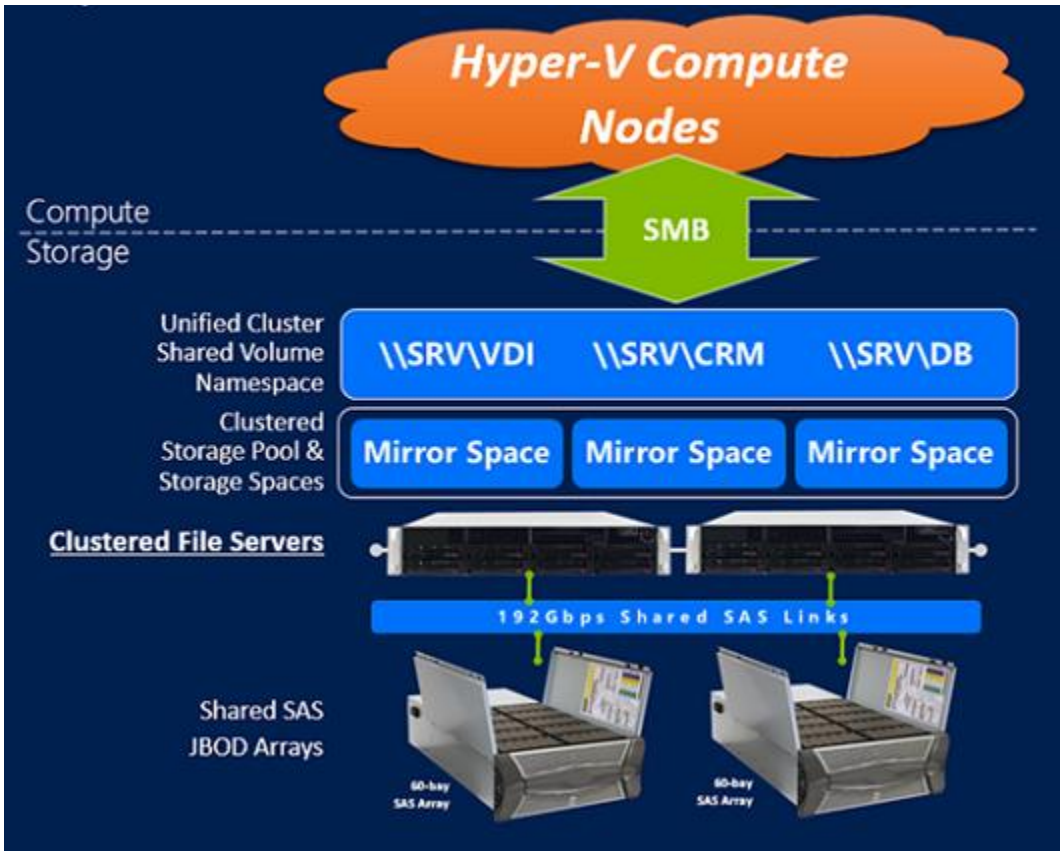
Storage Tiering is not supported with parity spaces or thinly provisioned volumes. Storage Tiering is only available with Windows Server 2012 R2, and is not available for Windows 8.1

Storage Spaces enclosure awareness

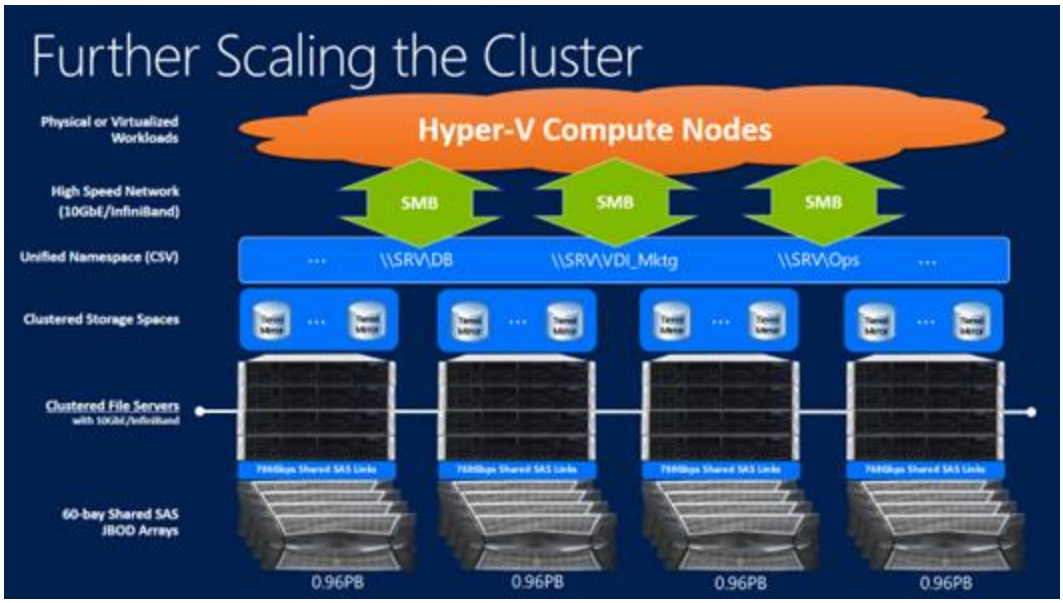
When creating a resilient virtual disk (mirrored or parity), Storage Spaces can further place separate data copies on separate enclosures to ensure resiliency to an entire enclosure failing. This capability is called Enclosure Awareness, and is available to Storage Spaces certified enclosures. For a list of Storage Space certified enclosures, see the Windows Server Catalog.

Storage Spaces clustering and continuous availability

By using Failover Clustering and Windows File and Storage Services, Storage Spaces provides continuously available storage deployments for clusters of virtualized workload.



As workload needs increase, you can expand the storage deployment by adding more storage nodes, or by adding more JBODs and then adding the physical disks to the large pool. The cluster shared volume namespace unifies data accesses across separate storage units in the cluster, where separate storage pools would have access to other pool's data through the high-speed cluster network.



Storage Spaces advanced configuration options

Storage Spaces provides multiple configuration options when creating a storage space which can be tuned according to your workload needs. For most workloads, it is recommended that users utilize the intelligent defaults. For more advanced guidance, see Storage Spaces - Designing for Performance.

See Also

Performance Tuning for Storage Subsystems

Using Storage-related Parameters and Performance Counters

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

In this topic:

- I/O priorities
- Logical disks and physical disks
- Processor information
- Power protection and advanced performance option
- Block alignment (DISKPART)
- Solid-state drives

- Trim and unmap capabilities
- Response times
- Queue lengths

I/O priorities

Windows Server 2012 R2, Windows Server 2012, Windows Server 2008 R2, and Windows Server 2008 can specify an internal priority level on individual I/Os. Windows Server 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 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 you span 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 average 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 average disk queue length is 0.37, then the % disk time will be 37. So if the average 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 average disk read queue length and average 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. Average 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**.

- **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 Performance Monitor if the requests are consistently interspersed. For guidelines, see Response times.

- **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 sub-workloads separately.

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

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

- **Split I/O / second**

  This counter 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 on the device
- Enable **Turn off Windows write-cache buffer flushing on the device** 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 caches 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 apps, 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 R2, NTFS exclusively uses cache flushes to protect its metadata.

The **Turn off Windows write-cache buffer flushing on the device** 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). Prior to Windows Server 2008, the partition boundary offset for a specific disk partition could be misaligned when it was compared to array disk stripe unit boundaries. This caused small requests to be unintentionally split across multiple disks. To force alignment, you were required to use diskpar.exe or DiskPart.exe at the time the partition was created.

In Windows Server 2012 R2, 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 R2, 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 have not tried to dynamically determine what data should optimally be stored on SSDs versus rotating media. It has changed with Windows Server 2012 R2 and Storage Spaces which allow data placement and migration to be performed without human intervention. This is called *tiering*. For more info about tiering, see Storage Spaces automated data tiering (tiered storage).

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 lower than write latency.
- Media lifetime is limited by the number of erase cycles 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. Besides overprovisioning, another method is to use higher quality flash, such as Single Level Cell (SLC) and Multiple Level Cell (eMLC), that can support higher write cycles but has a

higher cost per gigabyte. 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 and Windows Server 2012 R2 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 and Windows Server 2012 R2 will issue standards-based trims or unmaps for any Storage 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. You should investigate whether your storage devices support Trim or Unmap commands to ensure an efficient and performant deployment.

Response times

You can use tools such as Performance Monitor 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, Performance Monitor 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:

- As a baseline, on under-loaded storage subsystems, there is a floor of performance where read/write response times will match the manufacturer specifications for average seek times (incorporating command times, head movement, rotational latency, etc.). As per above, 5 to 15 ms. This model is also true for SSDs (with different values). When there is an increase over this floor that means that the volume of IO is exceeding the nominal specifications of the storage, or aggregate specifications in the case of an array. That means items are queuing. However, queuing, and in turn waiting on access to the media is not in and of itself not bad, it is just when the wait times become excessive. Excessive being a subjective concept to the

demands of the application. However, the following two bullets providing suggestions for consideration.

- For a high priority applications 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 low priority 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

This is a mathematically derived counter according to Little's Law, N = A * T (N = Queue Length, A = I/O per second, T = seconds per IO). As a result, the math works out such that this is relevant and easily translate nicely for scenarios where the number of physicals disks in the server are known. While still an accurate measure as storage becomes a more abstract concept, including the thin provisioning model and data-tiering, it becomes harder to effectively use this information. Essentially, the fact that all spindles are the same and exclusively available to the server/application is no longer valid.

As a result, several opinions exist about what constitutes excessive disk request queuing. These topics assume 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 you 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.

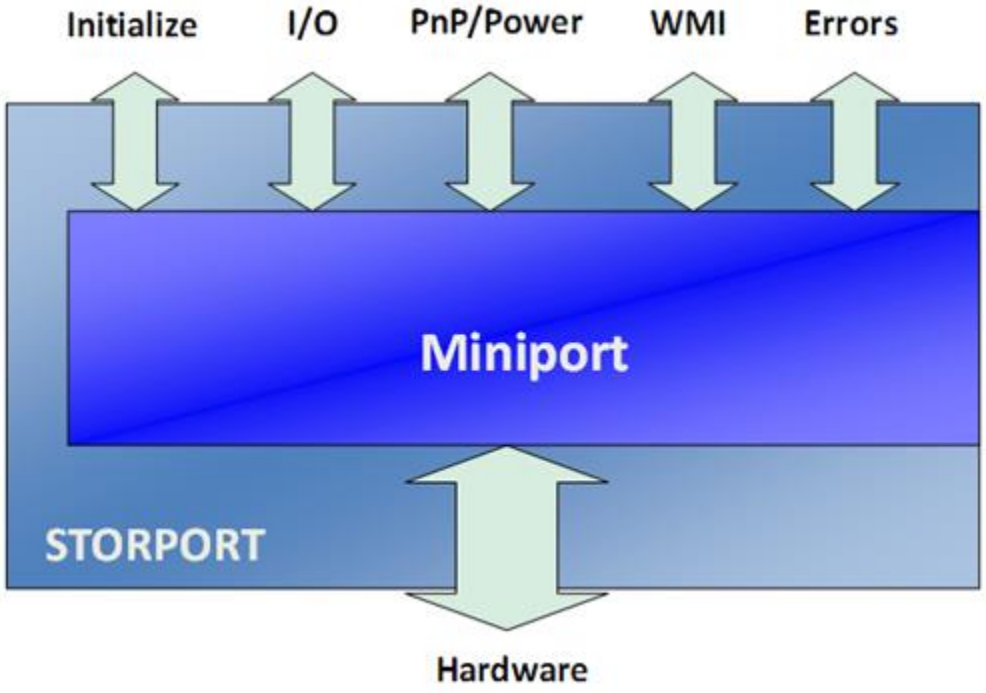
See Also

Performance Tuning for Storage Subsystems

Using Storage Drivers for Storage Subsystem Performance

Beginning with Windows Server 2003, storport.sys is the storage port driver model that is especially suitable for use with high-performance buses, such as Fibre Channel buses, and RAID adapters.

The Storport driver is utilized by mini-port drivers developed by various vendors and provides the following capabilities to mini-port drivers:



It is vital that the storport and mini-port drivers always be considered as a couple and when updating any of these, the vendor of the storage and/or the vendor of the HBA should be consulted for the latest supported combination of these two drivers.

If performance of the storage is not as expected, the following are some areas to investigate to determine whether the issue is in the operating system or lies outside of the operating system and needs to be investigated by HBA/storage/fabric vendors.

In this topic:

Storage latency (also known as slow I/O)

On configurations using Storport mini-port driver, the system has ETW events that helps track down poor hardware response time. For info on how to configure the ETW tracing, see Tracing with Storport in Windows 2012 and Windows 8 with KB2819476 hotfix.

When troubleshooting disk performance issues, Storport traces capture data from the last layer of software in Windows that an I/O Request Packet (IRP) will pass through before being handed off to hardware. It is an excellent tool for checking if slow disk performance is hardware related.

I/O completions

Storport mini-port drivers do batch I/O completions. This approach has a potential drawback in that it may cause the system to crash with a stop code of 0x133, DPC_WATCHDOG_VIOLATION, if Storport takes too long to process completed I/Os in a DPC routine. If this occurs, you can adjust the value of the registry entry HKLM\System\CurrentControlSet\Control\Storport\DpcCompletionLimit (REG_DWROD) to a lower value. The default value used with Windows Server 2012 R2 is 128 I/O requests. You can also adjust the registry entry to a larger value if you believe you will not run into 0x133 bugchecks in your configuration.

Storport miniports supporting MSI-X interrupts

For the best storage performance, make sure your configuration is using storage HBAs that support multiple MSI-X interrupts. Determine whether your storage HBA supports multiple MSI-X interrupts by using Device Manager. In Device Manager, find your storage HBA under the **Storage controllers** group. Right-click the storage HBA, and then select **Properties**. On the **Resource** tab, if you see multiple IRQ resource types, then the storage HBA supports multiple MSI-X interrupts.

Determining bottlenecks in storport queues

Storport supports both a LUN and adapter queue that is based on the hardware capabilities and system resources. Storport will not issue requests above the outstanding limits supported by the storage LUN, adapter, and Storport mini-port driver. To determine whether you may be hitting these limits, the Config keyword events in Storport's ETW provider can be enabled in the same way as described in Tracing with Storport in Windows 2012 and Windows 8 with KB2819476 hotfix except you should select the Config keyword instead of IO_Performance keyword.

For the Config events, look for the following:

- LUN queue depth
- Adapter queue depth

Data can be interpreted by using xperfview.

To interpret ETL data - LUN queue depth (Event ID = 0xf)

This is only logged if there are changes in LUN queue depth.

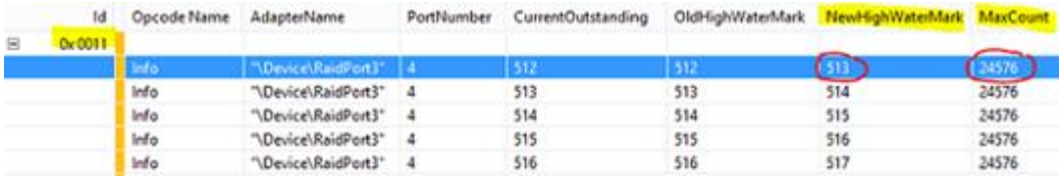
MaxQueueDepth is the outstanding commands to a LUN limit that Storport will enforce. To determine which miniport is used for AdapterName, look at HKLM\Hardware\Devicemap\Scsi\Scsi Port X\Driver. The X should match the "\Device\RaidPortX" for AdapterName.

Id	Opcode Name	AdapterName	PortNumber	PathID	TargetID	LUN	CurrentDepth	NewDepth	MaxDepth
0x 0011									
0x 000f									
	Info	"\Device\RaidPort3"	4	255	255	255	1024	1024	1024
	Info	"\Device\RaidPort3"	4	0	0	0	1024	1024	1024

To interpret ETL data - Adapter queue depth (Event ID = 0x11)

This is only logged if there are changes in the adapter queue depth.

MaxCount is the limit on outstanding IOs to an adapter. If NewHighWaterMark equals MaxCount, it means that the adapter is the bottleneck as we've hit the adapter outstanding I/O limit. If you don't see any of these events logged, it can mean one of 2 things: the system cannot generate I/Os fast enough or system has ran out of memory resource.



Id	Opcode Name	AdapterName	PortNumber	CurrentOutstanding	OldHighWaterMark	NewHighWaterMark	MaxCount
0x 0011							
	Info	"\Device\RaidPort3"	4	512	512	513	24576
	Info	"\Device\RaidPort3"	4	513	513	514	24576
	Info	"\Device\RaidPort3"	4	514	514	515	24576
	Info	"\Device\RaidPort3"	4	515	515	516	24576
	Info	"\Device\RaidPort3"	4	516	516	517	24576

See Also

Performance Tuning for Storage Subsystems

Performance Tuning for Data Deduplication

Data Deduplication involves finding and removing duplication within data without compromising its fidelity or integrity. The goal is to store more data in less space by segmenting files into small variable-sized chunks (32–128 KB), identifying duplicate chunks, and maintaining a single copy of each chunk. Redundant copies of the chunk are replaced by a reference to the single copy. The chunks are compressed and then organized into special container files in the System Volume Information folder.

In this topic:

Types of data on deduplication-enabled volumes

The data type suitable for Data Deduplication on general purpose file servers are the files that are written to rarely. Apps that frequently write data to files (like Microsoft SQL Server or Microsoft Exchange Server) are not recommended to be hosted on deduplication-enabled volumes. Frequently changing files may cause the deduplication job to do unnecessary optimization work which may impact performance.

For virtual machines hosted on deduplicated volumes, the only types of virtual machines that are officially supported are client operating systems supported by VDI.

Types of job schedules

There are three types of Data Deduplication jobs:

- **Optimization (daily)** Identify duplicate data and optimize duplicates away
- **Garbage Collection (weekly)** Remove unreferenced chunks of data that were part of deleted files
- **Scrubbing (weekly)** Identify and fix any corruptions

Storage and CPU

The Data Deduplication subsystem schedules one single threaded job per volume depending on system resources. To achieve optimal throughput, consider configuring multiple deduplication volumes, up to the number of CPU cores on the file server.

Memory

The amount of memory required by the deduplication optimization job is directly related to the number of optimization jobs that are running. During the optimization process, approximately 1 to 2 GB of RAM is necessary to process 1 TB of data per volume at maximum speed.

For example, a file server running concurrent optimization jobs on 3 volumes of 1 TB, 1 TB, and 2 TB of data respectively would need the following amount of memory, assuming a normal amount of file data changes:

Volume	Volume size	Memory used
Volume 1	1 TB	1-2 GB
Volume 2	1 TB	1-2 GB
Volume 3	2 TB	2-4 GB
Total for all volumes	1+1+2 * 1GB up to 2GB	4 – 8 GB RAM

By default, deduplication optimization will use up to 50% of a server's memory. In this example, having 8 to 16 GB of memory available on the file server would allow the deduplication to optimally allocate the expected amount of RAM during optimization. Allowing optimization to use more memory would speed optimization throughput. The amount of RAM given to a job can be adjusted by using the Windows PowerShell cmdlet.

```
Start-Dedupjob <volume> -Type Optmization  -Memory <50 to 80>
```

Machines where very large amount of data change between optimization job is expected may require even up to 3 GB of RAM per 1 TB of diskspace.

I/O throttling

All deduplication jobs are I/O intensive and may affect the performance of other apps when running. To alleviate potential problems, the default schedules can be modified by using Server Manager or by using the following Windows PowerShell command:

```
Set-DedupSchedule
```

To further alleviate I/O performance issues introduced by the most I/O intensive optimization jobs, I/O throttling may be manually enabled for the specific job to balance system performance. The following Windows PowerShell command to control I/O throttling:

```
Start-DedupJob <volume>  -Type Optimization -InputOutputThrottleLevel <Level>

where <Level> can be:   {None | Low | Medium | High | Maximum}
```

In the case of **Maximum**, deduplication jobs run with maximum throttling and deduplication will not make any progress in the presence of other I/Os resulting in very slow optimization. A throttle

level of **None**, is the most intrusive but will process deduplication jobs fastest at the expense of all other I/O activity on the system. By default, the optimization job runs with a throttle level of **Low**.

Garbage collection

For file servers that have large amounts of data that is frequently created and deleted, you might need to set the garbage collection job schedule to run more frequently to keep up with the changes and delete the stale data.

Custom Garbage Collection schedules can be set by using Server Manager or by using this Windows PowerShell command:

```
New-DedupSchedule
```

See Also

Performance Tuning for Storage Subsystems

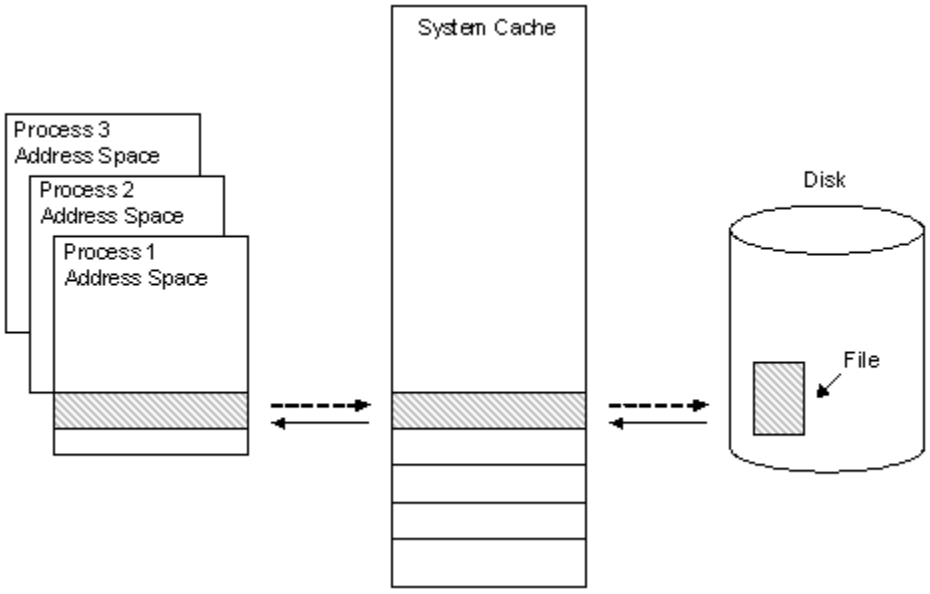
Performance Tuning for Cache and Memory Manager Subsystems

By default, Windows caches file data that is read from disks and written to disks. This implies that read operations read file data from an area in system memory, known as the system file cache, rather than from the physical disk. Correspondingly, write operations write file data to the system file cache rather than to the disk, and this type of cache is referred to as a write-back cache. Caching is managed per file object.Caching occurs under the direction of the Cache Manager, which operates continuously while Windows is running.

File data in the system file cache is written to the disk at intervals determined by the operating system. Flushed pages stay either in system cache working set (when FILE_FLAG_RANDOM_ACCESS is set and file handle wasn't closed) or on the standby list where these become part of available memory.

The policy of delaying the writing of the data to the file and holding it in the cache until the cache is flushed is called lazy writing, and it is triggered by the Cache Manager at a determinate time interval. The time at which a block of file data is flushed is partially based on the amount of time it has been stored in the cache and the amount of time since the data was last accessed in a read operation. This ensures that file data that is frequently read will stay accessible in the system file cache for the maximum amount of time.

This file data caching process is illustrated in the following figure:



As depicted by the solid arrows in the preceding figure, a 256 KB region of data is read into a 256 KB cache slot in system address space when it is first requested by the Cache Manager during a file read operation. A user-mode process then copies the data in this slot to its own address space. When the process has completed its data access, it writes the altered data back to the same slot in the system cache, as shown by the dotted arrow between the process address space and the system cache. When the Cache Manager has determined that the data will no longer be needed for a certain amount of time, it writes the altered data back to the file on the disk, as shown by the dotted arrow between the system cache and the disk.

In this section:

- Cache and Memory Manager Potential Performance Issues
- Cache and Memory Manager Improvements in Windows Server 2012

See Also

Performance Tuning for Subsystems

Cache and Memory Manager Potential Performance Issues

Before Windows Server 2012, two primary potential issues caused system file cache to grow until available memory was almost depleted under certain workloads. When this situation results in the system being sluggish, you can determine whether the server is facing one of these issues.

In this topic:

- Counters to monitor

- System file cache contains NTFS metafile data structures
- System file cache contains memory mapped files

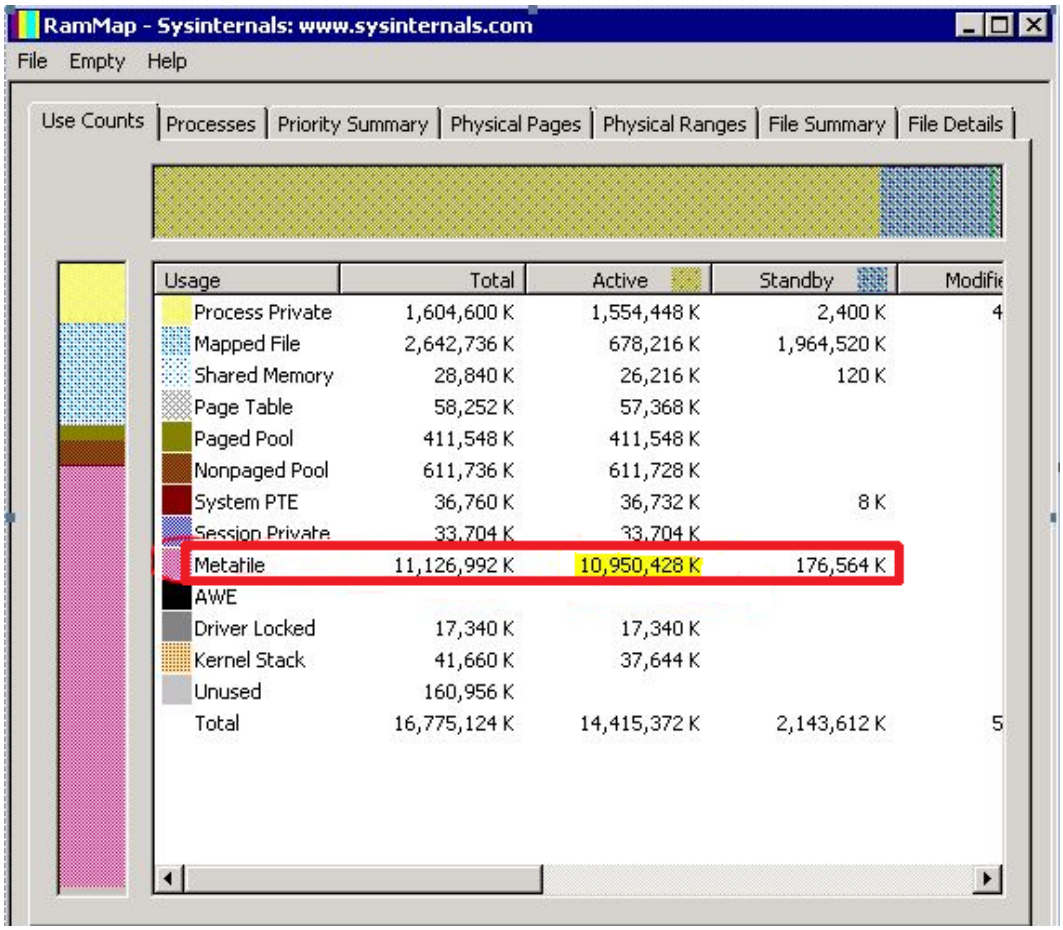
Counters to monitor

- Memory\Long-Term Average Standby Cache Lifetime (s) < 1800 seconds
- Memory\Available Mbytes is low
- Memory\System Cache Resident Bytes

If Memory\Available Mbytes is low and at the same time Memory\System Cache Resident Bytes is consuming significant part of the physical memory, you can use RAMMAP to find out what the cache is being used for.

System file cache contains NTFS metafile data structures

This problem is indicated by a very high number of active Metafile pages in RAMMAP output, as shown in the following figure. This problem might have been observed on busy servers with millions of files being accessed, thereby resulting in caching NTFS metafile data not being released from the cache.



The problem used to be mitigated by *DynCache* tool. In Windows Server 2012, the architecture has been redesigned and this problem should no longer exist.

System file cache contains memory mapped files

This problem is indicated by very high number of active Mapped file pages in RAMMAP output. This usually indicates that some application on the server is opening a lot of large files using CreateFile API with FILE_FLAG_RANDOM_ACCESS flag set.

This issue is described in detail in KB article 2549369. FILE_FLAG_RANDOM_ACCESS flag is a hint for Cache Manager to keep mapped views of the file in memory as long as possible (until Memory Manager doesn't signal low memory condition). At the same time, this flag instructs Cache Manager to disable prefetching of file data.

This situation has been mitigated to some extent by working set trimming improvements in Windows Server 2012 and Windows Server 2012 R2, but the issue itself needs to be primarily addressed by the application vendor by not using FILE_FLAG_RANDOM_ACCESS. An alternative solution for the app vendor might be to use low memory priority when accessing the

files. This can be achieved using the SetThreadInformation API. Pages that are accessed at low memory priority are removed from the working set more aggressively.

See Also

Performance Tuning for Cache and Memory Manager Subsystems

Cache and Memory Manager Improvements in Windows Server 2012

This topic describes Cache Manager and Memory Manager improvements in Windows Server 2012.

In this topic:

Cache Manager improvements

In addition to Cache Manager enhancements to read ahead logic for sequential workloads, a new API CcSetReadAheadGranularityEx was added to let file system drivers, such as SMB, change their read ahead parameters. It allows better throughput for remote file scenarios by sending multiple small-sized read ahead requests instead of sending a single large read ahead request. Only kernel components, such as file system drivers, can programmatically configure these values on a per-file basis. This API was added in Windows Server 2012.

Memory Manager improvements

Enabling page combining may reduce memory usage on servers which have a lot of private, pageable pages with identical contents. For example, servers running multiple instances of the same memory-intensive app, or a single app that works with highly repetitive data, might be good candidates to try page combining. The downside of enabling page combining is increased CPU usage.

Here are some examples of server roles where page combining is unlikely to give much benefit:

- File servers (most of the memory is consumed by file pages which are not private and therefore not combinable)
- Microsoft SQL Servers that are configured to use AWE or large pages (most of the memory is private but non-pageable)

Page combining is disabled by default but can be enabled by using the Enable-MMAgent Windows PowerShell cmdlet. Page combining was added in Windows Server 2012.

See Also

Performance Tuning for Cache and Memory Manager Subsystems

Performance Tuning for Server Roles

This section describes performance tuning guidelines for server roles in the following topics:

- Performance Tuning for Web Servers
- Performance Tuning for File Servers
- Performance Tuning for Active Directory Servers
- Performance Tuning for Remote Desktop Session Hosts
- Performance Tuning for Remote Desktop Virtualization Hosts
- Performance Tuning for Remote Desktop Gateways
- Performance Tuning for Hyper-V Servers

For more performance tuning guidelines, see Performance Tuning Guidelines for Windows Server 2012 R2.

See Also

Performance Tuning Guidelines for Windows Server 2012 R2

Performance Tuning for Web Servers

This topic describes performance tuning methods and recommendations for Windows Server 2012 R2 web servers.

In this topic:

- Selecting the proper hardware for performance
- Operating system best practices
- Tuning IIS 8.5
- NTFS file system setting
- Networking subsystem performance settings for IIS

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.

Performance Tuning for Server Hardware provides recommendations for hardware to avoid the following performance constraints:

- Slow CPUs offer limited processing power for CPU intensive workloads such as ASP, ASP.NET, and SSL scenarios.
- A small L2 or L3/LLC 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 best practices

If possible, start with a clean installation of the operating system. 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.

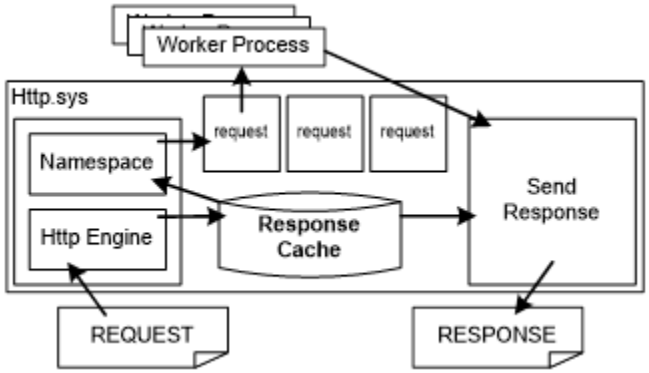
To reduce contention for system resources, install Microsoft SQL Server and IIS on different servers, if possible.

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

Tuning IIS 8.5

Internet Information Services (IIS) 8.5 is included with Windows Server 2012 R2. It uses a process model similar to that of IIS 7.0. A kernel-mode web driver (http.sys) receives and routes HTTP requests, and satisfies 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).

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. Multiple worker processes can be configured, which provides isolation at a reduced cost. For more info on how request handling works, see the following figure:



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. 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 8.5 automatically caches frequently requested files in http.sys.

Because a web server has kernel-mode and user-mode components, both components must be tuned for optimal performance. Therefore, tuning IIS 8.5 for a specific workload includes configuring the following:

- HTTP.sys 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 8.5.

Kernel-mode settings

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 8.5 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 non-zero 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 non-zero value that 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 maximum set size.

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

  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 R2, http.sys manages connections automatically. The following registry settings are no longer used:

- **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 8.5 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 8.5 Management Console to change them. Most settings are automatically detected, and they do not require a restart of the IIS 8.5 worker processes or web application server. For more info about the applicationHost.config file, see Introduction to ApplicationHost.config.

User-mode cache behavior settings

This section describes the settings that affect caching behavior in IIS 8.5. 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-	True

Attribute	Description	Default
	mode cache does not disable the kernel-mode cache.	
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.	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.	262144

Compression behavior settings

IIS starting from 7.0 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 8.5 lets compression be controlled independently for static and dynamic content. Static content typically refers to content that does not change, such as GIF or HTM files. Dynamic content is typically generated by scripts or code on the server, that is, ASP.NET pages. You can customize the classification of any particular extension as static or dynamic.

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

system.webServer/httpCompression

Attribute	Description	Default
staticCompression-EnableCpuUsage staticCompression-DisableCpuUsage dynamicCompression-EnableCpuUsage dynamicCompression-DisableCpuUsage	Enables or disables compression if the current percentage CPU usage goes above or below specified limits. Starting with IIS 7.0, compression is automatically disabled 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.	%SystemDrive%\inetpub\temp\IIS Temporary Compressed Files
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	100 MB

Attribute	Description	Default
	too large.	

system.webServer/urlCompression

Attribute	Description	Default
doStaticCompression	Specifies whether static content is compressed.	True
doDynamicCompression	Specifies whether dynamic content is compressed.	True

Note

For servers running IIS 8.5 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	True

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

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 drive 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.	%SystemDrive%\inetpub\logs\LogFiles

Application and site tunings

The following settings relate to application pool and site tunings.

system.applicationHost/applicationPools/applicationPoolDefaults

Attribute	Description	Default
queueLength	Indicates to 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,	True

Attribute	Description	Default
	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.	

Managing IIS 8.5 modules

IIS 8.5 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 removing 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 ASP.NET performance recommendations, see 10 Tips for Writing High-Performance Web Applications.

system.webServer/asp/cache

Attribute	Description	Default
diskTemplateCacheDirectory	The name of the directory that ASP uses to store compiled templates when the in-memory cache	%SystemDrive%\inetpub\temp\ASP Compiled Templates

Attribute	Description	Default
	overflows. Recommendation: 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.	
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	25

Attribute	Description	Default
	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.	

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 multi-threaded 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 setting:

```
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

You can use the IIS Administrator user interface to configure options for recycling IIS worker processes and 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 exceed a specified limit in kilobytes.	0
requests	Enable process recycling after a certain number of requests.	0
time	Enable process recycling after a specified time period.	29:00:00

Dynamic worker-process page-out tuning

Starting in Windows Server 2012 R2, IIS now offers the option of configuring sites to suspend after they have been idle for a while (in addition to the option of terminate, which existed since IIS 7).

Let's have a look at fine-tuning the settings for optimal performance.

The main purpose of both the WP Page-out and WP Termination features is to conserve memory utilization on the server, since a site can consume a lot of memory even if it's just sitting there, listening. Depending on the technology used on the site (static content vs ASP.NET vs other frameworks), the memory used can be anywhere from about 10 MB to hundreds of MBs, and this means that if your server is configured with many sites, figuring out the most effective settings for your sites can dramatically improve performance of both active and suspended sites.

Before we go into specifics, we must keep in mind that if there are no memory constraints, then it's probably best to simply set the sites to never suspend or terminate. After all, there's little value in terminating a worker process if it's the only one on the machine.

Note

In case the site runs unstable code, such as code with a memory leak, or otherwise unstable, setting the site to terminate on idle can be a quick-and-dirty alternative to fixing the code bug. This isn't something we would encourage, but in a crunch, it may be better to use this feature as a clean-up mechanism while a more permanent solution is in the works.]

Another factor to consider is that if the site does use a lot of memory, then the suspension process itself takes a toll, because the computer has to write the data used by the Worker Process to disk. If the WP is using a large chunk of memory, then suspending it might be more expensive than the cost of having to wait for it to start back up.

To make the best of the site suspension feature, you need to review your sites, and decide which should be suspended, which should be terminated, and which should be active indefinitely. For each action and each site, you need to figure out the ideal time-out period.

Ideally, the sites that you will configure for suspension or termination are those that have visitors every day, but not enough to warrant keeping it active all the time. These are usually sites with around 20 unique visitors a day or less. You can analyze the traffic patterns using the site's log files and calculate the average daily traffic.

Keep in mind that once a specific user connects to the site, they will typically stay on it for at least a while, making additional requests, and so just counting daily requests may not accurately reflect the real traffic patterns. To get a more accurate reading, you can also use a tool, such as Microsoft Excel, to calculate the average time between requests. For example:

	Request URL	Request time	Delta
1	/SourceSilverLight/Geosource.web/grosource.html	10:01	
2	/SourceSilverLight/Geosource.web/sliverlight.js	10:10	0:09

3	/SourceSilverLight/Geosource.web/clientbin/geo/1.aspx	10:11	0:01
4	/IClientAccessPolicy.xml	10:12	0:01
5	/ SourceSilverLight/GeosourcewebService/Service.asmx	10:23	0:11
6	/ SourceSilverLight/Geosource.web/GeoSearchServer….	11:50	1:27
7	/rest/Services/CachedServices/Silverlight_load_la…	12:50	1:00
8	/rest/Services/CachedServices/Silverlight_basemap….	12:51	0:01
9	/rest/Services/DynamicService/ Silverlight_basemap….	12:59	0:08
10	/rest/Services/CachedServices/Ortho_2004_cache.as…	13:40	0:41
11	/rest/Services/CachedServices/Ortho_2005_cache.js	13:40	0:00
12	/rest/Services/CachedServices/OrthoBaseEngine.aspx	13:41	0:01

The hard part, though, is figuring out what setting to apply to make sense. In our case, the site gets a bunch of requests from users, and the table above shows that a total of 4 unique sessions occurred in a period of 4 hours. With the default settings for site suspension, the site would be terminated after the default timeout of 20 minutes, which means each of these users would experience the site spin-up cycle. This makes it an ideal candidate for site suspension, because for most of the time, the site is idle, and so suspending it would conserve resources, and allow the users to reach the site almost instantly.

A final, and very important note about this is that disk performance is crucial for this feature. Because the suspension and wake-up process involve writing and reading large amount of data to the hard drive, we strongly recommend using a fast disk for this. Solid State Drives (SSDs) are ideal and highly recommended for this, and you should make sure that the Windows page file is stored on it (if the operating system itself is not installed on the SSD, configure the operating system to move the page file to it).

Whether you use an SSD or not, we also recommend fixing the size of the page file to accommodate writing the page-out data to it without file-resizing. Page-file resizing might happen when the operating system needs to store data in the page file, because by default, Windows is configured to automatically adjust its size based on need. By setting the size to a fixed one, you can prevent resizing and improve performance a lot.

To configure a pre-fixed page file size, you need to calculate its ideal size, which depends on how many sites you will be suspending, and how much memory they consume. If the average is 200 MB for an active worker process (W3WP) and you have 500 sites on the servers that will be suspending, then the page file should be at least (200 * 500) MB over the base size of the page file (so base + 100 GB in our example).

Note

When sites are suspended, they will consume approximately 6 MB each, so in our case, memory usage if all sites are suspended would be around 3 GB. In reality, though, you're probably never going to have them all suspended at the same time.

Secure Sockets Layer tuning parameters

The use of SSL imposes additional CPU cost. The most expensive component of SSL is the cost of establishing a session establishment because it 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.
- Selectively apply encryption only to pages or parts of the site that need it, rather to the entire site.

Note

- 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.

Internet Server Application Programming Interface (ISAPI)

No special tuning parameters are needed for ISAPI applications. If you write a private ISAPI extension, make sure that it is written for performance and resource use. For more info, see **Other issues that affect IIS performance**.

Managed code tuning guidelines

The integrated pipeline model in IIS 8.5 enables a high degree of flexibility and extensibility. Custom modules that are implemented in native or managed code can be inserted into the pipeline, or 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 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. After the compilation is complete, you should restart IIS. You should also 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 ASP.NET performance recommendations, see 10 Tips for Writing High-Performance Web Applications.

Other issues that affect IIS performance

The following issues can 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** and is set by default to 1. This switch reduces disk I/O load and latencies by disabling date and time stamp updating for the last file or directory access. Clean installations of Windows Server 2012 R2, Windows Server 2012, Windows Server 2008 R2, and Windows Server 2008 enable this setting by default, and you do not need to adjust it. Earlier versions of Windows did not set this key. If your server is running an earlier version of Windows, or it was upgraded to Windows Server 2012 R2, Windows Server 2012, Windows Server 2008 R2, or Windows Server 2008, you should enable this setting.

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

For networking subsystem performance settings, see Performance Tuning for Network Subsystems.

See Also

Performance Tuning for Server Roles

Performance Tuning for File Servers

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. For hardware recommendations, see Performance Tuning for Server Hardware. The following subsystem performance tuning topics also apply to file servers: Performance Tuning for Network Subsystems and Performance Tuning for Storage Subsystems.

In this topic:

SMB configuration considerations

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

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

Copying files is a common operation performed on a file server. Windows Server has several built-in file copy utilities that you can run by using a command prompt. Robocopy is recommended. Introduced in Windows Server 2008 R2, the **/mt** option of Robocopy 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. When you use Xcopy, we recommend adding the **/q** and **/k** options to your existing parameters. The former option reduces CPU overhead by reducing console output and the latter reduces network traffic.

SMB performance tuning

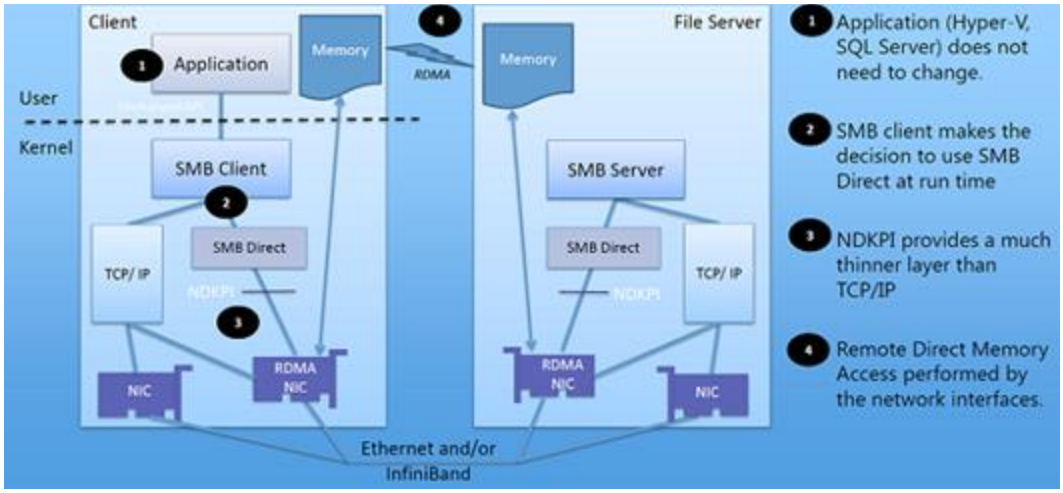
File server performance and available tunings depend on the SMB protocol that is negotiated between each client and the server, and on the deployed file server features. The highest protocol version currently available is SMB 3.02. You can check which version of SMB is in use

on your network by using Windows PowerShell, as described in Windows Server 2012: Which version of the SMB protocol (SMB 1.0, SMB 2.0, SMB 2.1, SMB 3.0, or SMB 3.02) are you using on your File Server?.

SMB 3.0 protocol

SMB 3.0 was introduced in Windows Server 2012 and further enhanced in Windows Server 2012 R2. This version introduced technologies that may significantly improve performance and availability of the file server. For more info on what's new in SMB 3.0 for Windows Server 2012 R2, see What's New for SMB in Windows Server 2012 R2.

The following figure shows a basic scenario with a single channel, single client, and single server. The scenario shows the optional paths of SMB packets that use either TCP/IP or RDMA.



The following sections describe what can be achieved by combining these techniques across multiple channels (multiple interfaces in the server) and multiple nodes joined in SMB file server cluster. For more info about SMB 3.0, see Updated Links on Windows Server 2012 File Server and SMB 3.0 and Updated Links on Windows Server 2012 R2 File Server and SMB 3.02.

SMB Direct

SMB Direct introduced the ability to use RDMA network interfaces for high throughput with low latency and low CPU utilization.

Whenever SMB detects an RDMA-capable network, it automatically tries to use the RDMA capability. However, if for any reason the SMB client fails to connect using the RDMA path, it will simply continue to use TCP/IP connections instead. All RDMA interfaces that are compatible with SMB Direct are required to also implement a TCP/IP stack, and SMB Multichannel is aware of that.

SMB Direct is not required in any SMB configuration, but it's always recommended for those who want lower latency and lower CPU utilization.

For more info about SMB Direct, see Improve Performance of a File Server with SMB Direct.

SMB Multichannel

SMB Multichannel allows file servers to use multiple network connections simultaneously and provides increased throughput.

For more info about SMB Multichannel, see The basics of SMB Multichannel, a feature of Windows Server 2012 and SMB 3.0.

SMB Transparent Failover

SMB Transparent Failover is a feature that allows an SMB client to continue to work uninterrupted when there's a failure in the SMB Scale-out file server cluster node that the client is using. This includes preserving information on the server side plus allowing the client to automatically reconnect to the same share and files on a surviving file server cluster node.

For more info about SMB Transparent Failover, see SMB Transparent Failover – making file shares continuously available.

SMB Scale-Out

SMB Scale-out allows SMB 3.0 in a cluster configuration to show a share in all nodes of a cluster. This active/active configuration makes it possible to scale file server clusters further, without a complex configuration with multiple volumes, shares and cluster resources. The maximum share bandwidth is the total bandwidth of all file server cluster nodes. The total bandwidth is no longer limited by the bandwidth of a single cluster node, but rather depends on the capability of the backing storage system. You can increase the total bandwidth by adding nodes.

For more info about SMB Scale-Out, see Scale-Out File Server for Application Data Overview and the blog post To scale out or not to scale out, that is the question.

Performance counters for SMB 3.0

The following SMB performance counters were introduced in Windows Server 2012, and they are considered a base set of counters when you monitor the resource usage of SMB 2 and higher versions. 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 during postprocessing by using Relog.exe.

- **SMB client shares**

  These counters display information about file shares on the server that are being accessed by a client that is using SMB 2.0 or higher versions.

  If you're familiar with the regular disk counters in Windows, you might notice a certain resemblance. That's not by accident. The SMB client shares performance counters were designed to exactly match the disk counters. This way you can easily reuse any guidance on application disk performance tuning you currently have. For more info about counter mapping, see Per share client performance counters blog.

- **SMB server shares**

  These counters display information about the SMB 2.0 or higher file shares on the server.

- **SMB server sessions**

  These counters display information about SMB server sessions that are using SMB 2.0 or higher.

  Turning on counters on server side (server shares or server sessions) may have significant performance impact for high IO workloads.
- **Resume key filter**

  These counters display information about the Resume Key Filter.
- **SMB direct connection**

  These counters measure different aspects of connection activity. A computer can have multiple SMB Direct connections. The SMB Direct Connection counters represent 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.
- **Physical disk, SMB, CSV File system performance counters relationship**

  For more info on how physical disk, SMB, and CSV file system counters are related, see the following blog post: Cluster Shared Volume Performance Counters.

Tuning parameters for SMB file servers

The following REG_DWORD registry settings can affect the performance of SMB file servers:

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

  ```
  HKLM\System\CurrentControlSet\Services\LanmanServer\Parameters\Smb2CreditsMin
  ```

  ```
  HKLM\System\CurrentControlSet\Services\LanmanServer\Parameters\Smb2CreditsMax
  ```

  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.

  Tip
  You can monitor SMB Client Shares\Credit Stalls /Sec to see if there are any issues with credits.
- **AdditionalCriticalWorkerThreads**

  ```
  HKLM\System\CurrentControlSet\Control\Session Manager\Executive\AdditionalCriticalWorkerThreads
  ```

  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.

Tip
The value may need to be increased if the amount of cache manager dirty data (performance counter Cache\Dirty Pages) is growing to consume a large portion (over ~25%) of memory or if the system is doing lots of synchronous read I/Os.

- **MaxThreadsPerQueue**

  `HKLM\System\CurrentControlSet\Services\LanmanServer\Parameters\MaxThreadsPerQueue`

  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.

  Tip
  An indication that the value may need to be increased is if the SMB2 work queues are growing very large (performance counter 'Server Work Queues\Queue Length\SMB2 NonBlocking *' is consistently above ~100).

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
AdditionalCriticalWorkerThreads	64	0
MaxThreadsPerQueue	64	20
MaxMpxCt (only applicable to SMB 1.0 clients)	32768	50

Services for NFS model

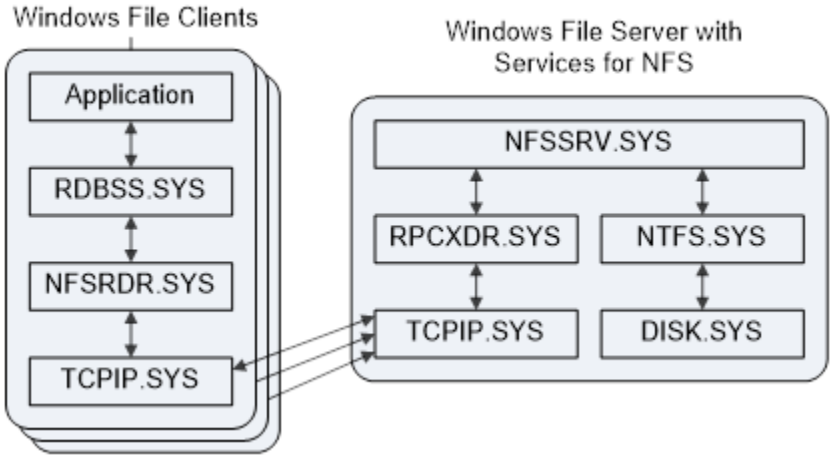
The following sections provide information about the Microsoft Services for Network File System (NFS) model for client-server communication. Since NFS v2 and NFS v3 are still the most widely deployed versions of the protocol, all of the registry keys except for MaxConcurrentConnectionsPerIp apply to NFS v2 and NFS v3 only.

No registry tuning is required for NFS v4.1 protocol.

Service 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. Applications on the client request files that are located on the server through the redirector (Rdbss.sys) and NFS mini-redirector (Nfsrdr.sys). The mini-redirector uses the NFS protocol to send its request through TCP/IP. The server receives multiple requests from the clients through TCP/IP and routes the requests to the local file system (Ntfs.sys), which accesses the storage stack.

The following figure shows the communication model for NFS.



Tuning parameters for NFS file servers

The following REG_DWORD registry settings can affect the performance of NFS file servers:

- **OptimalReads**

  `HKLM\System\CurrentControlSet\Services\NfsServer\Parameters\OptimalReads`

  The default is 0. This parameter 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.

  Note
  This setting must be carefully evaluated because it may have potential impact on system file cache grow.

- **RdWrHandleLifeTime**

  `HKLM\System\CurrentControlSet\Services\NfsServer\Parameters\RdWrHandleLifeTime`

  The default is 5. This parameter controls the lifetime of an NFS cache entry in the file handle cache. The parameter refers to cache entries that have an associated open NTFS file handle.

Actual lifetime is approximately equal to RdWrHandleLifeTime multiplied by RdWrThreadSleepTime. The minimum is 1 and the maximum is 60.

- **RdWrNfsHandleLifeTime**

  HKLM\System\CurrentControlSet\Services\NfsServer\Parameters\RdWrNfsHandleLifeTime

  The default is 5. This parameter controls the lifetime of an NFS cache entry in the file handle cache. The 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. The minimum is 1 and the maximum is 60.

- **RdWrNfsReadHandlesLifeTime**

  HKLM\System\CurrentControlSet\Services\NfsServer\Parameters\RdWrNfsReadHandlesLifeTime

  The default is 5. This parameter controls the lifetime of an NFS read cache entry in the file handle cache. Actual lifetime is approximately equal to RdWrNfsReadHandlesLifeTime multiplied by RdWrThreadSleepTime. The minimum is 1 and the maximum is 60.

- **RdWrThreadSleepTime**

  HKLM\System\CurrentControlSet\Services\NfsServer\Parameters\RdWrThreadSleepTime

  The default is 5. This parameter controls the wait interval before running the cleanup thread on the file handle cache. The value is in ticks, and it is non-deterministic. A tick is equivalent to approximately 100 nanoseconds. The minimum is 1 and the maximum is 60.

- **FileHandleCacheSizeinMB**

  HKLM\System\CurrentControlSet\Services\NfsServer\Parameters\FileHandleCacheSizeinMB

  The default is 4. This parameter specifies the maximum memory to be consumed by file handle cache entries. The minimum is 1 and the maximum is 1*1024*1024*1024 (1073741824).

- **LockFileHandleCacheInMemory**

  HKLM\System\CurrentControlSet\Services\NfsServer\Parameters\LockFileHandleCacheInMemory

  The default is 0. This parameter 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 (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\MaxIcbNfsReadHandlesCacheSize

The default is 64. This parameter 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. The minimum is 0 and the maximum is 0xFFFFFFFF.

- **HandleSigningEnabled**

  HKLM\System\CurrentControlSet\Services\NfsServer\Parameters\HandleSigningEnabled

  The default is 1. This parameter controls whether handles that are given out by NFS File Server are signed cryptographically. Setting it to 0 disables handle signing.

- **RdWrNfsDeferredWritesFlushDelay**

  HKLM\System\CurrentControlSet\Services\NfsServer\Parameters\RdWrNfsDeferredWritesFlushDelay

  The default is 60. This parameter is a soft timeout that controls the duration of NFS V3 UNSTABLE Write data caching. The minimum is 1, and the maximum is 600. Actual lifetime is approximately equal to RdWrNfsDeferredWritesFlushDelay multiplied by RdWrThreadSleepTime.

- **CacheAddFromCreateAndMkDir**

  HKLM\System\CurrentControlSet\Services\NfsServer\Parameters\CacheAddFromCreateAndMkDir

  The default is 1 (enabled). This parameter 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\AdditionalDelayedWorkerThreads

  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\NtfsDisable8dot3NameCreation

  The default in Windows Server 2012 and Windows Server 2012 R2 is 2. In releases prior to Windows Server 2012, the default is 0. This parameter determines whether NTFS generates a short name in the 8dot3 (MSDOS) 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 has 8dot3 enabled by default. All other volumes in Windows Server 2012 and Windows Server 2012 R2 have 8dot3 disabled by default. 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 file servers, the recommended setting is 1 (disabled).

- **NtfsDisableLastAccessUpdate**

  `HKLM\System\CurrentControlSet\Control\FileSystem\NtfsDisableLastAccessUpdate`

  The 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\Parameters\MaxConcurrentConnectionsPerIp`

  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 clients

The following REG_DWORD registry settings can affect the performance of client computers that interact with SMB or NFS file servers:

- **ConnectionCountPerNetworkInterface**

  `HKLM\System\CurrentControlSet\Services\LanmanWorkstation\Parameters\ConnectionCountPerNetworkInterface`

  Applies to Windows 8.1, Windows 8, Windows Server 2012, and Windows Server 2012 R2

  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**

  `HKLM\System\CurrentControlSet\Services\LanmanWorkstation\Parameters\ConnectionCountPerRssNetworkInterface`

  Applies to Windows 8.1, Windows 8, Windows Server 2012, and Windows Server 2012 R2

  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**

HKLM\System\CurrentControlSet\Services\LanmanWorkstation\Parameters\ConnectionCountPerRdmaNetworkInterface

Applies to Windows 8.1, Windows 8, Windows Server 2012, and Windows Server 2012 R2

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**

  HKLM\System\CurrentControlSet\Services\LanmanWorkstation\Parameters\MaximumConnectionCountPerServer

  Applies to Windows 8.1, Windows 8, Windows Server 2012, and Windows Server 2012 R2

  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**

  HKLM\System\CurrentControlSet\Services\LanmanWorkstation\Parameters\DormantDirectoryTimeout

  Applies to Windows 8.1, Windows 8, Windows Server 2012, and Windows Server 2012 R2

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

- **FileInfoCacheLifetime**

  HKLM\System\CurrentControlSet\Services\LanmanWorkstation\Parameters\FileInfoCacheLifetime

  Applies to Windows 8.1, Windows 8, Windows Server 2012, Windows Server 2012 R2, Windows 7, and Windows Vista

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

- **DirectoryCacheLifetime**

  HKLM\System\CurrentControlSet\Services\LanmanWorkstation\Parameters\DirectoryCacheLifetime

  Applies to Windows 7 and Windows Vista

  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**

  HKLM\System\CurrentControlSet\Services\LanmanWorkstation\Parameters\DirectoryCacheEntrySizeMax

  Applies to Windows 8.1, Windows 8, Windows Server 2012, Windows Server 2012 R2, Windows 7, and Windows Vista

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

- **FileNotFoundCacheLifetime**

  HKLM\System\CurrentControlSet\Services\LanmanWorkstation\Parameters\FileNotFoundCacheLifetime

  Applies to Windows 8.1, Windows 8, Windows Server 2012, Windows Server 2012 R2, Windows 7, and Windows Vista

  The default is 5 seconds. The file not found cache timeout period.

- **CacheFileTimeout**

  HKLM\System\CurrentControlSet\Services\LanmanWorkstation\Parameters\CacheFileTimeout

  Applies to Windows 8.1, Windows 8, Windows Server 2012, Windows Server 2012 R2, and Windows 7

  The default is 10 seconds. This setting controls the length of time (in seconds) that the redirector will hold on to cached data for a file after the last handle to the file is closed by an application.

- **DisableBandwidthThrottling**

  HKLM\System\CurrentControlSet\Services\LanmanWorkstation\Parameters\DisableBandwidthThrottling

  Applies to Windows 8.1, Windows 8, Windows Server 2012, Windows Server 2012 R2, Windows 7, and Windows Vista

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

- **DisableLargeMtu**

  HKLM\System\CurrentControlSet\Services\LanmanWorkstation\Parameters\DisableLargeMtu

  Applies to Windows 8.1, Windows 8, Windows Server 2012, Windows Server 2012 R2, Windows 7, and Windows Vista

  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**

  HKLM\System\CurrentControlSet\Services\LanmanWorkstation\Parameters\RequireSecuritySignature

  Applies to Windows 8.1, Windows 8, Windows Server 2012, Windows Server 2012 R2, Windows 7, and Windows Vista

  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**

  HKLM\System\CurrentControlSet\Services\LanmanWorkstation\Parameters\FileInfoCacheEntriesMax

  Applies to Windows 8.1, Windows 8, Windows Server 2012, Windows Server 2012 R2, Windows 7, and Windows Vista

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

- **DirectoryCacheEntriesMax**

  HKLM\System\CurrentControlSet\Services\LanmanWorkstation\Parameters\DirectoryCacheEntriesMax

  Applies to Windows 8.1, Windows 8, Windows Server 2012, Windows Server 2012 R2, Windows 7, and Windows Vista

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

- **FileNotFoundCacheEntriesMax**

  HKLM\System\CurrentControlSet\Services\LanmanWorkstation\Parameters\FileNotFoundCacheEntriesMax

  Applies to Windows 8.1, Windows 8, Windows Server 2012, Windows Server 2012 R2, Windows 7, and Windows Vista

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

- **MaxCmds**

  HKLM\System\CurrentControlSet\Services\LanmanWorkstation\Parameters\MaxCmds

  Applies to Windows 8.1, Windows 8, Windows Server 2012, Windows Server 2012 R2, Windows 7, and Windows Vista

  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 2.0 servers.

- **DormantFileLimit**

HKLM\System\CurrentControlSet\Services\LanmanWorkstation\Parameters\DormantFileLimit

Applies to Windows 8.1, Windows 8, Windows Server 2012, Windows Server 2012 R2, Windows 7, Windows Vista, and Windows XP

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.

Client tuning example

The general tuning parameters for client computers can optimize a computer for accessing remote file shares, 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

Starting in Windows 8, you can configure many of these file server settings by using the **Get-SmbServerConfiguration** and **Set-SmbServerConfiguration** Windows PowerShell cmdlets. Registry-only settings can be configured by using Windows PowerShell as well.

```
Set-ItemProperty -Path
"HKLM:\SYSTEM\CurrentControlSet\Services\LanmanWorkstation\Parameters"
RequireSecuritySignature -Value 0 -Force
```

SMB client performance monitor counters

For more info about SMB client counters, see Windows Server 2012 File Server Tip: New per-share SMB client performance counters provide great insight.

See Also

Performance Tuning for Server Roles

Using the File Server Capacity Tool (FSCT)

Using the SPECsfs2008 File Server

Performance Tuning for Active Directory Servers

Performance tuning Active Directory is focused on two goals:

- Active Directory is optimally configured to service the load in the most efficient manner possible
- Workloads submitted to Active Directory should be as efficient as possible

This requires proper attention to three separate areas:

- Proper capacity planning – ensuring sufficient hardware is in place to support existing load
- Server side tuning – configuring domain controllers to handle the load as efficiently as possible
- Active Directory client/application tuning – ensuring that clients and applications are using Active Directory in an optimal fashion

In this topic:

Capacity planning

Properly deploying a sufficient number of domain controllers, in the right domain, in the right locales, and to accommodate redundancy is critical to ensuring servicing client requests in a timely fashion. This is an in-depth topic and outside of the scope of this guide. To ensure proper configuration and sizing of Active Directory, see Capacity Planning for Active Directory Domain Services.

Updates and evolving recommendations

Massive improvements in both Active Directory and client performance optimizations have occurred over the last several generations of the operating system and these efforts continue. We

recommend that the most current versions of the platform be deployed to get the benefits, including:

- New capabilities
- Increased reliability
- Better performance
- Better logging and tools to troubleshoot

However, we realize that this takes time and many environments are running in a scenario where 100% adoption of the most current platform is impossible. Some improvements have been added to older versions of the platform and we'll continue to add more. We recommend keeping current with these updates.

Hardware basics

This is a summary of key points covered in much greater depth in the Capacity Planning for Active Directory Domain Services and is not a replacement for that content.

Read the following sections to optimize hardware for responsiveness of domain controllers to client requests.

Avoid going to disk

Active Directory caches as much of the database as memory allows. Fetching pages from memory are orders of magnitude faster than going to physical media, whether the media is spindle or SSD based. Add more memory to minimize disk I/O.

- For more info about storage subsystem tuning, see Performance Tuning for Storage Subsystems.
- Active Directory Best Practices recommend putting enough RAM to load the entire DIT into memory, plus accommodate the operating system and other installed applications, such as anti-virus, backup software, monitoring, and so on.
    - For limitations of the legacy platforms, see Memory usage by the Lsass.exe process on domain controllers that are running Windows Server 2003 or Windows 2000 Server.
    - Use the Memory\Long-Term Average Standby Cache Lifetime (s) > 30 minutes performance counter.
- Put the operating system, logs, and the database on separate volumes. If all or the majority of the DIT can be cached, once the cache is warmed and under a steady state, this becomes less relevant and offers a little more flexibility in storage layout. In scenarios where the entire DIT cannot be cached, the importance of splitting the operating system, logs, and database on separate volumes becomes more important.
- Normally, I/O ratios to the DIT are about 90% read and 10% write. Scenarios where write I/O volumes significantly exceed 10% - 20% are considered write-heavy. Write-heavy scenarios do not greatly benefit 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. Instead, it commits all write operations to the disk before it returns a successful completion status for an operation, unless there is an explicit request not to 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
- Increase the number of low-latency/high-RPM disks hosting the DIT and log files
- Write caching on the controller

- Review the disk subsystem performance individually for each volume. Most Active Directory scenarios are predominantly read-based, thus the statistics on the volume hosting the DIT are the most important to inspect. However, do not overlook monitoring the rest of the drives, including the operating system, and log files drives. To determine if the domain controller is properly configured to avoid storage being the bottleneck for performance, reference the section on Storage Subsystems for standards storage recommendations. Across many environments, the philosophy is to ensure that there is enough head room to accommodate surges or spikes in load. These thresholds are warning thresholds where the head room to accommodate surges or spikes in load becomes constrained and client responsiveness degrades. In short, exceeding these thresholds is not bad in the short term (5 to 15 minutes a few times a day), however a system running sustained with these sorts of statistics is not fully caching the database and may be over taxed and should be investigated.
  - Database ==> Instances(lsass/NTDSA)\I/O Database Reads Averaged Latency < 15ms
  - Database ==> Instances(lsass/NTDSA)\I/O Database Reads/sec < 10
  - Database ==> Instances(lsass/NTDSA)\I/O Log Writes Averaged Latency < 10ms
  - Database ==> Instances(lsass/NTDSA)\I/O Log Writes/sec – informational only.

    To maintain consistency of data, all changes must be written to the log. There is no good or bad number here, it is only a measure of how much the storage is supporting.
- Plan non-core disk I/O loads, such as backup and anti-virus scans, for non-peak load periods. Also, use backup and anti-virus solutions that support the low-priority I/O feature introduced in Windows Server 2008 to reduce competition with I/O needs of Active Directory.

Don’t over tax the processors

Processors that don’t have enough free cycles can cause long wait times on getting threads on to the processor for execution. Across many environments, the philosophy is to ensure that there is enough head room to accommodate surges or spikes in load to minimize impact on client responsiveness in these scenarios. In short, exceeding the below thresholds is not bad in the short term (5 to 15 minutes a few times a day), however a system running sustained with these sorts of statistics doesn’t provide any head room to accommodate abnormal loads and can easily be put into an over taxed scenario. Systems spending sustained periods above the thresholds should be investigated to how to reduce processor loads.

- For more info on how to select a processor, see Performance Tuning for Server Hardware.
- Add hardware, optimize load, direct clients elsewhere, or remove load from the environment to reduce CPU load.
- Use the Processor Information(_Total)\% Processor Utilization < 60% performance counter.

Avoid overloading the network adapter

Just like with processors, excessive network adapter utilization will cause long wait times for the outbound traffic to get on to the network. Active Directory tends to have small inbound requests and relatively to significantly larger amounts of data returned to the client systems. Sent data far exceeds received data. Across many environments, the philosophy is to ensure that there is enough head room to accommodate surges or spikes in load. This threshold is a warning threshold where the head room to accommodate surges or spikes in load becomes constrained and client responsiveness degrades. In short, exceeding these thresholds is not bad in the short term (5 to 15 minutes a few times a day), however a system running sustained with these sorts of statistics is over taxed and should be investigated.

- For more info on how to tune the network subsystem, see Performance Tuning for Network Subsystems.
- Use the Compare NetworkInterface(*)\Bytes Sent/Sec with NetworkInterface(*)\Current Bandwidth performance counter. The ratio should be less than 60% utilized.

Proper placement of domain controllers and site considerations

Proper site definition

This is critical to performance. Clients falling out of site can experience poor performance for authentications and queries. Furthermore, with the introduction of IPv6 on clients, the request can come from either the IPv4 or the IPv6 address and Active Directory needs to have sites properly defined for IPv6. The operating system prefers IPv6 to IPv4 when both are configured.

Starting in Windows Server 2008, the domain controller attempts to use name resolution to do a reverse lookup in order to determine the site the client should be in. This can cause exhaustion of the ATQ Thread Pool and cause the domain controller to become unresponsive. The appropriate resolution to this is to properly define the site topology for IPv6. As a workaround, you can optimize the name resolution infrastructure to respond quickly to domain controller requests. For more info see Windows Server 2008 or Windows Server 2008 R2 Domain Controller delayed response to LDAP or Kerberos requests.

Optimize for referrals

Referrals are how LDAP queries are redirected when the domain controller does not host a copy of the partition queried. When a referral is returned, it contains the distinguished name of the partition, a DNS name, and a port number. The client uses this information to continue the query on a server that hosts the partition. This is a DCLocator scenario and all of the recommendations site definitions and domain controller placement is maintained, but applications which depend on referrals are often overlooked. It is recommended to ensure AD Topology including site definitions and domain controller placement properly reflects the needs of the client. Also, this may include

having domain controllers from multiple domains in a single site, tuning DNS settings, or relocating the site of an application.

Optimization considerations for trusts

In an intra-forest scenario, trusts are processed according to the following domain hierarchy: Grand-Child Domain -> Child Domain -> Forest Root Domain -> Child Domain -> Grand-Child Domain. This means that secure channels at the forest root, and each parent, can become overloaded due to aggregation of authentication requests transiting the DCs in the trust hierarchy. This may also incur delays in Active Directories of large geographical dispersion when authentication also has to transit highly latent links to affect the above flow. Overloads can occur in inter-forest and down-level trust scenarios. The following recommendations apply to all scenarios:

- Properly tune the MaxConcurrentAPI to support the load across the secure channel. For more info, see How to do performance tuning for NTLM authentication by using the MaxConcurrentApi setting.
- Create shortcut trusts as appropriate based on load.
- Ensure that every domain controller in the domain is able to perform name resolution and communicate with the domain controllers in the trusted domain.
- Ensure locality considerations are taken into account for trusts.
- Enable Kerberos where possible and minimize use of the secure channel to reduce risk of running into MaxConcurrentAPI bottlenecks.

Cross domain trust scenarios are an area that has been consistently a pain point for many customers. Name resolution and connectivity issues, often due to firewalls, cause resource exhaustion on the trusting domain controller and impact all clients. Furthermore, an often overlooked scenario is optimizing access to trusted domain controllers. The key areas to ensure this works properly are as follows:

- Ensure the DNS and WINS name resolution that the trusting domain controllers are using can resolve an accurate list of domain controllers for the trusted domain.
    - Statically added records have a tendency to become stale and reintroduce connectivity problems over time. DNS forwards, Dynamic DNS, and merging WINS/DNS infrastructures are more maintainable in the long run.
    - Ensure proper configuration of forwarders, conditional forwards, and secondary copies for both forward and reverse lookup zones for every resource in the environment which a client may need to access. Again, this requires manual maintenance and has a tendency to become stale. Consolidation of infrastructures is ideal.
- Domain controllers in the trusting domain will attempt to locate domain controllers in the trusted domain that are in the same site first and then failback to the generic locators.
    - For more info on how DCLocator works, see Finding a Domain Controller in the Closest Site.
    - Converge site names between the trusted and trusting domains to reflect domain controller in the same location. Ensure subnet and IP address mappings are properly linked to sites in both forests. For more info, see Domain Locator Across a Forest Trust.

- Ensure ports are open, according to DCLocator needs, for domain controller location. If firewalls exist between the domains, ensure that the firewalls are properly configured for ALL trusts. If firewalls are not open, the trusting domain controller will still attempt to access the trusted domain. If communication fails for any reason, the trusting domain controller will eventually time out the request to the trusted domain controller. However, these time outs can take several seconds per request and can exhaust network ports on the trusting domain controller if the volume of incoming requests is high. The client may experience the waits to timeout at the domain controller as hung threads, which could translate to hung applications (if the application runs the request in the foreground thread). For more info, see How to configure a firewall for domains and trusts.
- Use DnsAvoidRegisterRecords to eliminate poorly performing or high-latency domain controllers, such as those in satellite sites, from advertising to the generic locators. For more info, see How to optimize the location of a domain controller or global catalog that resides outside of a client's site.

  Note
  There is a practical limit of about 50 to the number of domain controllers the client can consume. These should be the most site-optimal and highest capacity domain controllers.
- Consider placing domain controllers from trusted and trusting domains in the same physical location.

For all trust scenarios, credentials are routed according to the domain specified in the authentication requests. This is also true for queries to the LookupAccountName and LsaLookupNames (as well as others, these are just the most commonly used) APIs. When the domain parameters for these APIs are passed a NULL value, the domain controller will attempt to find the account name specified in every trusted domain available.

- Disable checking all available trusts when NULL domain is specified. How to restrict the lookup of isolated names in external trusted domains by using the LsaLookupRestrictIsolatedNameLevel registry entry
- Disable passing authentication requests with NULL domain specified across all available trusts. The Lsass.exe process may stop responding if you have many external trusts on an Active Directory domain controller

LDAP considerations

Verify LDAP queries

Verify that LDAP queries conform with the creating efficient queries recommendations.

There is extensive documentation on MSDN about how to properly write, structure, and analyze queries for use against Active Directory. For more info, see Creating More Efficient Microsoft Active Directory-Enabled Applications.

Optimize LDAP page sizes

When returning results with multiple objects in response to client requests, the domain controller has to temporarily store the result set in memory. Increasing page sizes will cause more memory usage and can age items out of cache unnecessarily. In this case, the default settings are optimal. There are several scenarios where recommendations were made to increase the page size settings. We recommend using the default values.

To tune these settings, see Windows Server 2008 and newer domain controller returns only 5000 values in a LDAP response.

Determine whether to add indices

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 the directory database. You can use logging to find the expensive and inefficient queries. Once you've identified them, you can index some attributes that are used in the corresponding queries to improve the search performance. For more info on how Active Directory Searches work, see How Active Directory Searches Work.

Some scenarios in which to add indices include:

- Client load in requesting the data is generating significant CPU usage and the client query behavior cannot be changed or optimized. By significant load, consider that it is showing itself in a Top 10 offender list in Server Performance Advisor or the built-in Active Directory Data Collector Set and is using more than 1% of CPU.
- The client load is generating significant disk I/O on a server due to an unindexed attribute and the client query behavior cannot be changed or optimized.
- A query is taking a long time and is not completing in an acceptable timeframe to the client due to lack of covering indices.
- Large volumes of queries with high durations are causing consumption and exhaustion of ATQ LDAP Threads. Monitor the following performance counters:
  - **NTDS\Request Latency** – This is subject to how long the request takes to process. Active Directory times out requests after 120 seconds (default), however, the majority should run much faster and extremely long running queries should get hidden in the overall numbers. Look for changes in this baseline, rather than absolute thresholds.

    Note
    High values here can also be indicators of delays in "proxying" requests to other domains and CRL checks.
  - **NTDS\Estimated Queue Delay** – This should ideally be near 0 for optimal performance as this means that requests spend no time waiting to be serviced.

These scenarios can be detected using one or more of the following approaches:

- Determining Query Timing with the Statistics Control

- Tracking Expensive and Inefficient Searches
- Active Directory Diagnostics Data Collector Set in Performance Monitor (Son of SPA: AD Data Collector Sets in Win2008 and beyond)
- **Microsoft Server Performance Advisor** Active Directory Advisor Pack
- Searches using any filter besides "(objectClass=*)" that use the Ancestors Index.

Considerations:

- Ensure that creating the index is the right solution to the problem after tuning the query has been exhausted as an option. Sizing hardware properly is very important. Indices should be added only when the right fix is to index the attribute, and not an attempt to obfuscate hardware problems.
- Indices increase the size of the database by a minimum of the total size of the attribute being indexed. An estimate of database growth can therefore be evaluated by taking the average size of the data in the attribute and multiplying by the number of objects that will have the attribute populated. Generally this is about a 1% increase in database size. For more info, see How the Data Store Works.
- If search behavior is predominantly done at the organization unit level, consider indexing for containerized searches.
- Tuple indices are larger than normal indices, but it is much harder to estimate the size. Use normal indices size estimates as the floor for growth, with a maximum of 20%. For more info, see How the Data Store Works.
- If search behavior is predominantly done at the organization unit level, consider indexing for containerized searches.
- Tuple Indices are needed to support medial search strings and final search strings. Tuple indices are not needed for initial search strings.
  - Initial Search String – (samAccountName=MYPC*)
  - Medial Search String - (samAccountName=*MYPC*)
  - Final Search String – (samAccountName=*MYPC$)
- Creating an index will generate disk I/O while the index is being built. This is done on a background thread with lower priority and incoming requests will be prioritized over the index build. If capacity planning for the environment has been done correctly, this should be transparent. However, write-heavy scenarios or an environment where the load on the domain controller storage is unknown could degrade client experience and should be done off-hours.
- Affects to replication traffic is minimal since building indices occurs locally.

For more info, see the following:

- Creating More Efficient Microsoft Active Directory-Enabled Applications
- Searching in Active Directory Domain Services
- Indexed Attributes

Troubleshooting

For info on troubleshooting, see Monitoring Your Branch Office Environment.

See Also

Performance Tuning for Server Roles

Performance Tuning for Remote Desktop Session Hosts

This topic discusses how to select Remote Desktop Session Host (RD Session Host) hardware, tune the host, and tune applications.

In this topic:

Selecting the proper hardware for performance

For an RD Session Host server deployment, the choice of hardware is governed by the application set and how users use them. The key factors that affect the number of users and their experience are CPU, memory, disk, and graphics. For info about server hardware guidelines, see Performance Tuning for Server Hardware. Those guidelines apply to Remote Desktop Services. This section contains additional guidelines that are specific to RD Session Host servers and is mostly related to the multi-user 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 Performance Tuning for Server Hardware.

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. 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 configuration

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

The disk activity that is generated on a typical RD Session Host server 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 server. 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 **Turn off Windows write-cache buffer flushing** on the device check boxes.

For more specific storage tunings, see Performance Tuning for Storage Subsystems.

Network configuration

Network usage for an RD Session Host server includes two main categories:

- RD Session Host connection traffic 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 server is driven by apps. Desktop apps 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 apps that are to be used on an RD Session Host server:

- 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 app 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 apps 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.

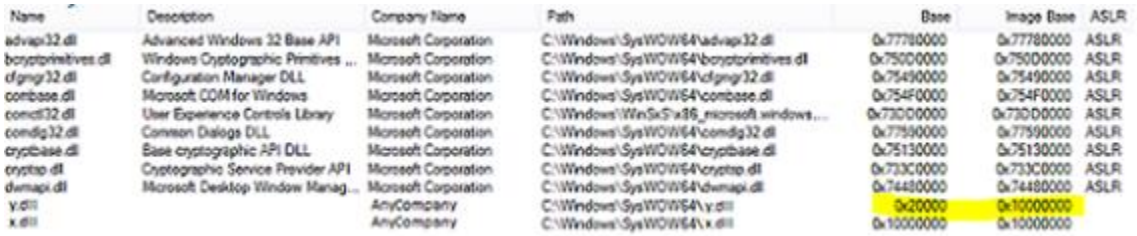
- Disable unnecessary processes that are registered to start with user sign-in or a session startup.

  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 more detailed info, you can use Autoruns for Windows.

For memory consumption, you should consider the following:

- Verify that DLLs loaded by an app are not relocated.
  - Relocated DLLs can be verified by selecting Process DLL view, as shown in the following figure, by using Process Explorer.

- Here we can see that y.dll was relocated because x.dll already occupied its default base address and ASLR was not enabled



Name	Description	Company Name	Path	Base	Image Base	ASLR
advapi32.dll	Advanced Windows 32 Base API	Microsoft Corporation	C:\Windows\SysWOW64\advapi32.dll	0x77780000	0x77780000	ASLR
bcryptprimitives.dll	Windows Cryptographic Primitives ...	Microsoft Corporation	C:\Windows\SysWOW64\bcryptprimitives.dll	0x75DD0000	0x75DD0000	ASLR
cfgmgr32.dll	Configuration Manager DLL	Microsoft Corporation	C:\Windows\SysWOW64\cfgmgr32.dll	0x75490000	0x75490000	ASLR
combase.dll	Microsoft COM for Windows	Microsoft Corporation	C:\Windows\SysWOW64\combase.dll	0x754F0000	0x754F0000	ASLR
comctl32.dll	User Experience Controls Library	Microsoft Corporation	C:\Windows\WinSxS\x86_microsoft.windows...	0x73DD0000	0x73DD0000	ASLR
comdlg32.dll	Common Dialogs DLL	Microsoft Corporation	C:\Windows\SysWOW64\comdlg32.dll	0x77590000	0x77590000	ASLR
cryptbase.dll	Base cryptographic API DLL	Microsoft Corporation	C:\Windows\SysWOW64\cryptbase.dll	0x75130000	0x75130000	ASLR
cryptsp.dll	Cryptographic Service Provider API	Microsoft Corporation	C:\Windows\SysWOW64\cryptsp.dll	0x733C0000	0x733C0000	ASLR
dwmapi.dll	Microsoft Desktop Window Manag...	Microsoft Corporation	C:\Windows\SysWOW64\dwmapi.dll	0x74480000	0x74480000	ASLR
y.dll		AnyCompany	C:\Windows\SysWOW64\y.dll	0x20000	0x10000000	
x.dll		AnyCompany	C:\Windows\SysWOW64\x.dll	0x10000000	0x10000000	

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 on an RD Session Host server.

- 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 apps or system components. You can use the memory-to-committed bytes performance counter to monitor how much committed virtual memory is on the system. For more info on page file sizing, see Performance Tuning for Storage Subsystems.

Antivirus

Installing antivirus 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 lets you examine the list of tasks that are scheduled for different events. For an RD Session Host server, 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 should disable any notifications by removing the component that registers them from the startup list or by changing the configuration on apps and system components to disable them. You can use **Customize Notifications Icons** to examine the list of notifications that are available on the server.

RemoteFX data compression

Microsoft RemoteFX compression can be configured by using Group Policy under **Computer Configuration > Administrative Templates > Windows Components > Remote Desktop Services > Remote Desktop Session Host > 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 a RemoteFX compression algorithm. Choosing to not use a 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 not to use a RemoteFX compression algorithm, some graphics data will be compressed.

Device redirection

Device redirection can be configured by using Group Policy under **Computer Configuration > Administrative Templates > Windows Components > Remote Desktop Services > Remote Desktop Session Host > Device and Resource Redirection** or by using the **Session Collection** properties box in Server Manager.

Generally, device redirection increases how much network bandwidth RD Session Host server 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 increases 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.
- Introduced in Windows Server 2008, Easy Print printer driver redirection 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, Easy Print causes slightly increased network bandwidth usage, but not significant enough to offset the other performance, manageability, and reliability benefits.

Audio redirection causes a steady stream of network traffic. Audio redirection also enables users to run multimedia apps 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 by using the **Experience** tab in Remote Desktop Connection or as settings in the RDP file.

The following settings apply when connecting to any computer:

- **Disable wallpaper** (Disable wallpaper:i:0) Does not show 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** (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** (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** (Disable menu anims:i:1 and Disable cursor setting:i:1): When these settings are disabled, it reduces bandwidth by disabling animation on menus (such as fading) and cursors.
- **Font smoothing** (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 significant 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.

The following settings only apply 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** (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 lists the options that are available to configure your connection speed:

- **Detect connection quality automatically** (Connection type:i:7) 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)** (Connection type:i:1) This setting enables persistent bitmap caching.
- **Low Speed Broadband (256 Kbps - 2 Mbps)** (Connection type:i:2) This setting enables persistent bitmap caching and visual styles.
- **Cellular/Satellite (2Mbps - 16 Mbps with high latency)** (Connection type:i:3) This setting enables desktop composition, persistent bitmap caching, visual styles, and desktop background.
- **High-speed broadband (2 Mbps – 10 Mbps)** (Connection type:i:4) This setting enables desktop composition, show contents of windows while dragging, menu and window animation, persistent bitmap caching, visual styles, and desktop background.
- **WAN (10 Mbps or higher with high latency)** (Connection type:i:5) This setting enables desktop composition, show contents of windows while dragging, menu and window animation, persistent bitmap caching, visual styles, and desktop background.
- **LAN (10 Mbps or higher)** (Connection type:i:6) This setting enables desktop composition, show contents of windows while dragging, menu and window animation, persistent bitmap caching, themes, and desktop background.

Desktop Size

Desktop size for remote sessions can be controlled by using the Display tab in Remote Desktop Connection or by using the RDP configuration 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 is 4096 x 2048.

See Also

Performance Tuning for Server Roles

Performance Tuning for Remote Desktop Virtualization Hosts

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 R2 and Hyper-V.

Windows Server 2012 R2 supports two types of virtual desktops, personal virtual desktops and pooled virtual desktops.

In this topic:

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 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.

Data Deduplication and VDI

Introduced in Windows Server 2012 R2, Data Deduplication supports optimization of open files. In order to use virtual machines running on a deduplicated volume, the virtual machine files need to be stored on a separate host from the Hyper-V host. If Hyper-V and deduplication are running on the same machine, the two features will contend for system resources and negatively impact overall performance.

The volume must also be configured to use the "Virtual Desktop Infrastructure (VDI)" deduplication optimization type. You can configure this by using Server Manager (**File and Storage Services** -> **Volumes** -> **Dedup Settings**) or by using the following Windows PowerShell command:

```
Enable-DedupVolume <volume> -UsageType HyperV
```

Note

Data Deduplication optimization of open files is supported only for VDI scenarios with Hyper-V using remote storage over SMB 3.0.

For more info on Data Deduplication, see Performance Tuning for Storage Subsystems.

Memory

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 x86 Window 8 or Windows 8.1 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. For x64, we see about 800 MB starting, increasing to 1024 MB.

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 server, 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 hyper-threading, 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.

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 R2, 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, contact your GPU provider.

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

The memory requirement for the RemoteFX-enabled server is dynamic because the amount of memory consumed on the RemoteFX-enabled 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 virtual GPU-enabled virtual desktop uses the video memory in the GPU hardware on 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-enabled server and the virtual GPU-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 virtual GPU-enabled virtual desktop is related to running the virtual GPU driver and a user-mode Remote Desktop Protocol stack.

On the RemoteFX-enabled server, the overhead is increased, because the system runs an additional process (rdvgm.exe) per virtual GPU-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 virtual GPU-enabled virtual desktop. We also recommend using the x64 architecture for virtual GPU-enabled virtual desktops because the performance on x64 virtual machines is better compared to x86 virtual machines.

RemoteFX GPU processing power

For every virtual GPU-enabled virtual desktop, there is a corresponding DirectX process running on the RemoteFX-enabled 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 machines 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

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-enabled server per second
- **Data: Send bytes/sec** Total number of bytes sent to the RemoteFX-enabled 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 virtual GPU performance counters, you can also measure the GPU utilization by using Process Explorer, 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 virtual desktop pools. 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

Failover Clustering in Windows Server 2012 and Windows Server 2012 R2 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 info on CSV cache, 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.

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 are some initial things to consider:

Service	Why?
Auto update	Pooled virtual desktops are updated by re-creating the virtual desktop template.
Offline files	Virtual desktops 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

Service	Why?
	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 WiFi 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

Note
This list is not meant to be a complete list, because any changes will affect the intended goals and scenarios. For more info, see Hot off the presses, get it now, the Windows 8 VDI optimization script, courtesy of PFE!.

Note
SuperFetch in Windows 8 is enabled by default. It is VDI-aware and should not be disabled. SuperFetch can further reduce memory consumption through memory page sharing, which is beneficial for VDI. Pooled virtual desktops running Windows 7, SuperFetch should be disabled, but for personal virtual desktops running Windows 7, it should be left on.

See Also

Performance Tuning for Server Roles

Performance Tuning for Remote Desktop Gateways

Note
In Windows 8 and Windows Server 2012 R2, Remote Desktop Gateway (RD Gateway) 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 topic 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, RD Gateway performs many packet forwarding operations between Remote Desktop Connection instances and the RD Session Host server instances within the customer's network.

Note

The following parameters apply to RPC transport only.

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

Thread tunings

- **Maxiothreads**

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

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

- **MaxPoolThreads**

  `HKLM\System\CurrentControlSet\Services\InetInfo\Parameters\MaxPoolThreads (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 Remote Desktop Connection 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\ServerReceiveWindow (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.

- **ServerReceiveWindow**

  `HKLM\Software\Microsoft\Rpc\ServerReceiveWindow (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.ReplaceThisText

See Also

Performance Tuning for Server Roles

Performance Tuning for Hyper-V Servers

Hyper-V is the virtualization server role in Windows Server 2012 R2. 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.

In this topic:

Hyper-V terminology

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

- **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 virtualized hardware resources (central processing unit or CPU, memory, and devices). The hypervisor 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 processor 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**

  The root partition that is created first and owns all the resources that the hypervisor does not, including most devices and system memory. The root partition 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 network switch**

  (also referred to as a virtual switch) A virtual version of a physical network switch. A virtual network can be configured to provide access to local or external network resources for one or more virtual machines.

- **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 (VSC)**

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

- **virtualization service provider (VSP)**

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

- **virtualization stack**

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

- **VMBus**

  Channel-based communication mechanism used for inter-partition communication and device enumeration on systems with multiple active virtualized partitions. The VMBus is installed with Hyper-V Integration Services.

- **Virtual machine queue (VMQ)**

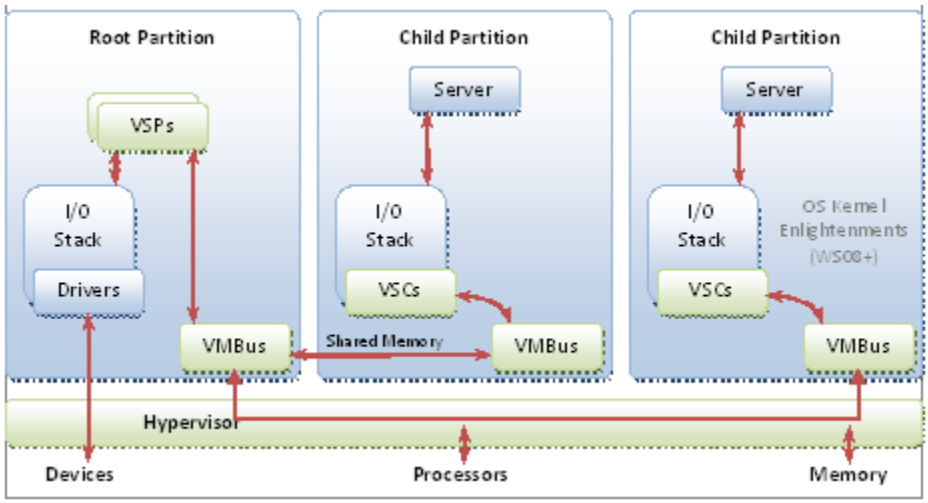
  Virtual machine queue (VMQ) is a feature available to computers running Windows Server 2008 R2 or later with the Hyper-V server role installed, that have VMQ-capable

network hardware. VMQ uses hardware packet filtering to deliver packet data from an external virtual machine network directly to virtual machines, which reduces the overhead of routing packets and copying them from the management operating system to the virtual machine.

Hyper-V architecture

Hyper-V features a Type 1 hypervisor-based architecture. 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 the integrated device electronics (IDE) disk controller and PS/2 input device port, and it supports Hyper-V-specific synthetic devices for increased performance and reduced overhead.



The Hyper-V-specific I/O architecture consists of virtualization service providers (VSPs) in the root partition and virtualization service clients (VSCs) in the child partition. Each service is exposed as a device over VMBus, which acts as an I/O bus and enables high-performance communication between virtual machines that use mechanisms such as shared memory. The guest operating system's Plug and Play manager 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.

Starting with Windows Server 2008, the operating system features enlightenments 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.

The following sections suggest best practices that yield increased performance on servers running Hyper-V role. 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 included with Server Performance Advisor.

Hyper-V server configuration

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 info, see Performance Tuning for Server Hardware.

- **Processors**

  Hyper-V in Windows Server 2012 R2 presents the logical processors as one or more virtual processors to each active virtual machine. You can achieve additional run-time efficiency by using processors that support Second Level Address Translation (SLAT) technologies such as 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 both the root and child partitions. The root partition requires memory to efficiently perform I/Os on behalf of the virtual machines and operations such as a virtual machine snapshot. Hyper-V ensures that sufficient memory is available to the root partition, and allows remaining memory to be assigned to child partitions. Child partitions should be sized based on the needs of the expected load for each virtual machine.

- **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. For the teamed NICs just one virtual switched is assigned. When you host multiple 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), virtual machine queue (VMQ) and SR-IOV.

  For more info about hardware considerations, see Performance Tuning for Server Hardware.

- **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 Network Subsystems.

Server Core installation option

Windows Server 2012 R2, 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 footprint for the host OS, and a smaller attack and servicing 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 by using Windows Powershell so administrators can manage it remotely.

Dedicated server role

The root partition should be dedicated to Hyper-V. Running additional server roles on a server running Hyper-V 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.

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

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.

You should consider carefully what options get added by using the **bcdedit** command in the guest operating system. Some options may adversely impact the performance of the guest operating system.

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 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 reflect the actual physical CPU usage. Use the following performance counters to monitor performance:

- **Hyper-V Hypervisor Logical Processor (*)\% Total Run Time** The total non-idle time of the logical processors
- **Hyper-V Hypervisor Logical Processor (*)\% Guest Run Time** The time spent running cycles within a guest or within the host
- **Hyper-V Hypervisor Logical Processor (*)\% Hypervisor Run Time** The time spent running within the hypervisor
- **Hyper-V Hypervisor Root Virtual Processor (*)*** Measures the CPU usage of the root partition
- **Hyper-V Hypervisor Virtual Processor (*)*** Measures the CPU usage of guest partitions

Hyper-V processor performance

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 virtual machine. 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 a list of supported guest operating systems, see Hyper-V Overview.

Use enlightened guest operating systems

The operating system kernels in Windows Server 2012 R2, Windows Server 2012, Windows Server 2008 R2, Windows Server 2008, Windows 8, and Windows 8.1, Windows 7, and Windows Vista with SP1 feature enlightenments that optimize their operation for virtual machines.

The enlightenments present in Windows Server 2012 R2, 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 Virtual Machine 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 virtual machine for improved performance.

Virtual processors

Hyper-V in Windows Server 2012 R2 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. For a list of supported guest operating systems, see Hyper-V Overview.

Enlightenments in Windows Server 2012 and Windows Server 2012 R2 improve scalability in multiprocessor virtual machines, like for example database workloads running in a virtual machine 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. 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 operating system on the sign-in screen when it is not being used and disable the screen saver.
- Review the scheduled tasks and services that are enabled by default.
- Review the ETW trace providers that are on by default by running **logman.exe query -ets**
- Improve server applications to reduce periodic activity (such as timers).
- Close Server Manager on both the host and guest operating systems.
- Don't leave Hyper-V Manager running since it constantly refreshes the virtual machine's thumbnail.

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.

Virtual NUMA

To enable virtualizing large scale-up workloads, Hyper-V in Windows Server 2012 introduced expanded virtual machine scale limits. A single virtual machine can be assigned up to 64 virtual

processors and 1 TB of memory. When creating such large virtual machines, memory from multiple NUMA nodes on the host system will likely be utilized. In such virtual machine configuration, if virtual processors and memory are not allocated from the same NUMA node, workloads may have bad performance due to the inability to take advantage of NUMA optimizations.

In Windows Server 2012 R2 and Windows Server 2012, Hyper-V presents a virtual NUMA topology to virtual machines. By default, this virtual NUMA topology is optimized to match the NUMA topology of the underlying host computer. Exposing a virtual NUMA topology into a virtual machine allows the guest operating system and any NUMA-aware applications running within it to take advantage of the NUMA performance optimizations, just as they would when running on a physical computer.

There is no distinction between a virtual and a physical NUMA from the workload's perspective. Inside a virtual machine, when a workload allocates local memory for data, and accesses that data in the same NUMA node, fast local memory access results on the underlying physical system. Performance penalties due to remote memory access are successfully avoided. Only NUMA-aware applications can benefit of vNUMA.

Microsoft SQL Server is an example of NUMA aware application. For more info, see Understanding Non-uniform Memory Access.

Virtual NUMA and Dynamic Memory features cannot be used at the same time. A virtual machine that has Dynamic Memory enabled effectively has only one virtual NUMA node, and no NUMA topology is presented to the virtual machine regardless of the virtual NUMA settings.

For more info on Virtual NUMA, see Hyper-V Virtual NUMA Overview.

Hyper-V 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, just as on non-virtualized systems. To ensure that you get maximum performance use SLAT-based hardware to minimize the performance cost of memory virtualization.

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 R2 guests is at least 512 MB. With Dynamic Memory, if applications in the virtual machine experience problems making large sudden memory allocations, you can increase the page file size for the virtual machine to ensure temporary backing while Dynamic Memory responds to the memory pressure.

For more info on Dynamic Memory, see Hyper-V Dynamic Memory Overview and 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 in Windows Server 2012 R2 monitors the runtime health of the root partition's management operating system to determine how much memory can safely be allocated to child partitions, while still ensuring high performance and reliability of the root partition.

Hyper-V 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 Virtual Machine Integration Services are not installed on the guest operating system. Disk I/O that is performed by using the IDE filter driver that is provided with the Virtual Machine 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 (SAS controller)

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. SCSI path is not emulated which makes it the preferred controller for any disk other than the operating system disk. In fact with Generation 2 VMs, it is the only type of controller possible. Introduced in Windows Server 2012 R2, this controller is reported as SAS to support shared VHDX.

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 R2 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. Introduce 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 R2 and Windows Server 2012 server 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.

To convert to new VHD format, run the following Windows PowerShell command:

```
Convert-VHD –Path E:\vms\testvhd\test.vhd –DestinationPath E:\vms\testvhd\test-converted.vhd
```

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.

To check for alignment by using Windows Powershell, examine the Alignment line, as shown below:

```
Get-VHD –Path E:\vms\testvhd\test.vhd


Path                    : E:\vms\testvhd\test.vhd
VhdFormat               : VHD
VhdType                 : Dynamic
FileSize                : 69245440
Size                    : 10737418240
MinimumSize             : 10735321088
LogicalSectorSize       : 512
PhysicalSectorSize      : 512
BlockSize               : 2097152
ParentPath              :
FragmentationPercentage : 10
Alignment               : 0
Attached                : False
DiskNumber              :
IsDeleted               : False
Number                  :
```

To verify alignment by using Windows PowerShell, examine the Alignment line, as shown below:

```
Get-VHD –Path E:\vms\testvhd\test-converted.vhd
```

```
Path                      : E:\vms\testvhd\test-converted.vhd
VhdFormat                 : VHD
VhdType                   : Dynamic
FileSize                  : 69369856
Size                      : 10737418240
MinimumSize               : 10735321088
LogicalSectorSize         : 512
PhysicalSectorSize        : 512
BlockSize                 : 2097152
ParentPath                :
FragmentationPercentage   : 0
Alignment                 : 1
Attached                  : False
DiskNumber                :
IsDeleted                 : False
Number                    :
```

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:

- 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 Hyper-V that does not support the VHDX format.

Types of virtual hard disk files

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

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.

Fixed virtual hard disk type

Space for the VHD is first allocated when the VHD file is created. This type of VHD file is less likely 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 virtual hard disk 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 virtual hard disk 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.

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. For example, if an application is allocating in chunks of 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 512byte 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 (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 write commands 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.

Each 4 KB write command 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 Server 2012 and Windows Server 2012 R2 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 or Windows Server 2012 R2.

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. To make full use of 4 KB sectors it's recommended to use VHDX format.

Support for native 4 KB disks

Hyper-V in Windows Server 2012 R2 supports 4 KB native disks. But it is still possible to store VHD disk on 4 KB native disk. 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-byte I/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.

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.

Pass-through disks should be avoided due to the limitations introduced with virtual machine migration scenarios.

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 entries 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:

Storage and networking have three registry entries 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 entries may be removed in future releases.

- **IOBalance_Enabled**

  ```
  HKLM\System\CurrentControlSet\Services\StorVsp\IOBalance_Enabled
  (REG_DWORD)
  ```

  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 virtual hard disks on local storage and disabled for virtual hard disks on remote SMB storage and networking. Enabling the balancing for networking can add significant CPU overhead in some scenarios.

- **IOBalance_KeepHwBusyLatencyTarget_Microseconds**

  ```
  HKLM\System\CurrentControlSet\Services\StorVsp\IOBalance_KeepHwB
  usyLatencyTarget_Microseconds (REG_DWORD)
  ```

  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**

  ```
  HKLM\System\CurrentControlSet\Services\StorVsp\IOBalance_Allowed
  PercentOverheadDueToFlowSwitching (REG_DWORD)
  ```

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.

Storage Quality of Service (QoS)

Starting in Windows Server 2012 R2, Hyper-V includes the ability to set certain quality-of-service (QoS) parameters for storage on the virtual machines. Storage QoS provides storage performance isolation in a multitenant environment and mechanisms to notify you when the storage I/O performance does not meet the defined threshold to efficiently run your virtual machine workloads.

Storage QoS provides the ability to specify a maximum input/output operations per second (IOPS) value for your virtual hard disk. An administrator can throttle the storage I/O to stop a tenant from consuming excessive storage resources that may impact another tenant.

You can also set a minimum IOPS value. They will be notified when the IOPS to a specified virtual hard disk is below a threshold that is needed for its optimal performance.

The virtual machine metrics infrastructure is also updated, with storage related parameters to allow the administrator to monitor the performance and chargeback related parameters.

Maximum and minimum values are specified in terms of normalized IOPS where every 8 KB of data is counted as an I/O.

Some of the limitations are as follows:

- Only for virtual disks
- Differencing disk cannot have parent virtual disk on a different volume
- Replica - QoS for replica site configured separately from primary site
- Shared VHDX is not supported

For more info on Storage Quality of Service, see Storage Quality of Service for Hyper-V.

NUMA I/O

Windows Server 2012 R2 and Windows Server 2012 supports large virtual machines, and any large virtual machine configuration (for example, a configuration with Microsoft 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 R2 and 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 expensive interprocessor interruptions.

Introduced in Windows Server 2012, there are a few registry entries, located at HKLM\System\CurrentControlSet\Enum\VMBUS\{device id}\{instance id}\StorChannel, 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 settings are configured on a per-adapter basis on the device's hardware key.

- **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 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.

Hyper-V in Windows Server 2012 R2 and 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. It's important to optimize the use of physical storage as much as possible.

Before Windows Server 2012, when applications delete content within a virtual hard disk, which effectively abandoned the content's storage space, the Windows storage stack in the guest operating system and the Hyper-V host had limitations that prevented this information from being communicated to the virtual hard disk and the physical storage device. This prevented the Hyper-V storage stack from optimizing the space usage by the VHD-based virtual disk files. It also prevented the underlying storage device from reclaiming the space that was previously occupied by the deleted data.

Starting from 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. 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 Virtual Fibre Channel disks.

Hyper-V 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 info on how to tune the network adapter in the root partition, including interrupt moderation, see Performance Tuning for Network Subsystems. 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 Virtual Machine Integration Services be installed.

Performance Monitors 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 (*) \ *.

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.

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.

Performance Monitor counters that represents 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 R2 and 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.

When NIC is connected to vSwitch, RSS is automatically disabled and VMQ mechanism has to be used to direct network processing to logical CPUs.

Some Intel multi-core processors may use Intel Hyper-Threading technology. When Hyper-Threading 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 hyper-threaded sibling cores. As an example, if the machine has an Intel processor with four physical cores, and Hyper-Threading 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 NIC teaming in Windows Server 2012 R2 with Hyper-V. For more info on NIC teaming, see **Network subsystem**. 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.

Teaming mode	Address hash modes	HyperVPort	Dynamic
Switch independent	Min-Queues	Sum-of-Queues	Sum-of-Queues
Switch dependent	Min-Queues	Min-Queues	Min-Queues

If the team is configured by using switch independent mode and the load distribution is set to Dynamic mode or 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-of-queues mode in the above table. For more info, see the Windows Server 2012 R2 NIC Teaming (LBFO) Deployment and Management Guide. Dynamic mode algorithm takes the best aspects of each of the other two modes and combines them into a single mode. Outbound loads are distributed based on a hash of the TCP ports and IP addresses. Dynamic mode also rebalances loads in real-time so that a given outbound flow may move back and forth between team members. Inbound loads are distributed as though the Hyper-V port mode was in use. Dynamic mode was added in Windows Server 2012 R2.

The following table compares 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

*RssBaseProcNumber, *MaxRssProcessors	8-core HT, 3 network adapters	16-core HT 4 network adapters
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 hyper-threading is in use on cores: MaxRssProcessors = min((Ct-2)/2/Nt, 16)

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

If the link speed is less than 10 GB, VMQ is disabled by default by the vmSwitch even though it will still show as enabled in the Windows PowerShell cmdlet **Get-NetAdapterVmq**. One way to verify that VMQ is disabled is to use the Windows PowerShell cmdlet **Get-NetAdapterVmqQueue**. This will show that there is not a QueueID assigned to the virtual machine or host vNIC. There shouldn't be a reason for VMQ usage for lower link speeds.

If you must enable the feature, you can do it by using the following registry entry:

```
HKLM\SYSTEM\CurrentControlSet\Services\VMSMP\Parameters\BelowTenGigVmqEnabled
```

For more info about VMQ, see the VQM Deep Dive series:

- VMQ Deep Dive, 1 of 3
- VMQ Deep Dive, 2 of 3
- VMQ Deep Dive, 3 of 3

MAC spoofing guidance

By default, each virtual machine has MAC address protection, so traffic on a MAC address (other than the one assigned in the host) is blocked. To allow a virtual machine to set its own MAC address, MAC Address Spoofing must be enabled by using Windows Powershell:

```
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 2012 introduced 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 by using the Windows PowerShell cmdlet, **Get-NetAdapterSriov**. Other IOV operations that are exposed through Windows PowerShell include:

- **Enable-NetAdapterSriov** Enable IOV on the physical network adapter
- **Disable-NetAdapterSriov** Disable IOV on the physical network adapter
- **New-VMSwitch** Use with the **–EnableIov** parameter 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.

SR-IOV is ideal for high I/O workloads that do not require port policies, QoS, or network virtualization enforced at the host virtual switch. These features are not supported for SR-IOV because traffic doesn't go through virtual switch.

SR-IOV is not compatible with LBFO on the host but can be used in conjunction with LBFO inside a virtual machine. If the NIC doesn't support VF RSS, workloads that are network intensive, such as a Web server might benefit of greater parallelism in the virtual network stack if a second Hyper-V-specific network adapter is installed in a virtual machine.

When SR-IOV is enabled, any policy that is added to the switch will turn SR-IOV off for that virtual machine, reverting the network path back through the vSwitch. This guarantees that policies are enforced.

To be able to use RSS with SR-IOV, VF RSS has to be supported by NIC, and the following conditions must be true:

- The virtual machine must be created with multiple cores.
- The SR-IOV network adapter must support VF RSS. This means that the network adapter has multiple RSS tables; one for each VF that uses RSS.
- The network adapter driver must also advertise that the network adapter supports VF RSS.
- RSS must be enabled in the virtual machine.

Virtual Receive Side Scaling

Virtual Receive Side Scaling (vRSS), introduced in Windows Server 2012 R2, enables the software processing of inbound (received) networking traffic to be shared across multiple

processors inside the host and virtual machine. vRSS can dynamically balance the inbound network traffic load.

vRSS works by scaling a virtual machine's receive side traffic to multiple virtual processors, so that the incoming traffic spreads over the total number of cores available to that virtual machine. vRSS also spreads send side traffic from a virtual machine onto multiple processors, so that virtual switch processing does not generate bottlenecks on a single physical processor when sending large amounts of outgoing traffic.

Configuring vRSS

Before you enable vRSS in a virtual machine, you must do the following:

- Verify that the network adapter is VMQ-capable and has a link speed of at least 10 GB. Also, verify that VMQ is enabled on the host machine. vRSS will not work if the host does not support VMQ.
- Verify that an SR-IOV Virtual Function (VF) is not attached to the virtual machine's network adapter. This can be done by using the **Get-NetAdapterSriov** Windows PowerShell cmdlet. If a VF driver is loaded, vRSS will use the vRSS settings exposed from this driver. If the VF driver does not expose RSS, vRSS is disabled.
- If you are using NIC Teaming, you must properly configure VMQ to work with the NIC Teaming settings. For more info on deploying and managing NIC Teaming, see Windows Server 2012 R2 NIC Teaming (LBFO) Deployment and Management Guide.

You can enable vRSS a few different ways:

1. Run the following Windows PowerShell as an administrator inside the virtual machine: **AdapterName** should be replaced with the name of the vmNIC adapter that RSS should be enabled on.

   ```
   Enable-NetAdapterRSS -Name "AdapterName"
   ```

2. Run the following Windows PowerShell as an administrator inside the virtual machine: **AdapterName** should be replaced with the name of the vmNIC adapter that RSS should be enabled on.

   ```
   Set-NetAdapterRSS -Name "AdapterName" -Enabled $True
   ```

3. From the **Advanced** tab in the settings of the network adapter.

The vRSS settings in the virtual machine are configured by using the same Windows PowerShell cmdlets as native RSS.

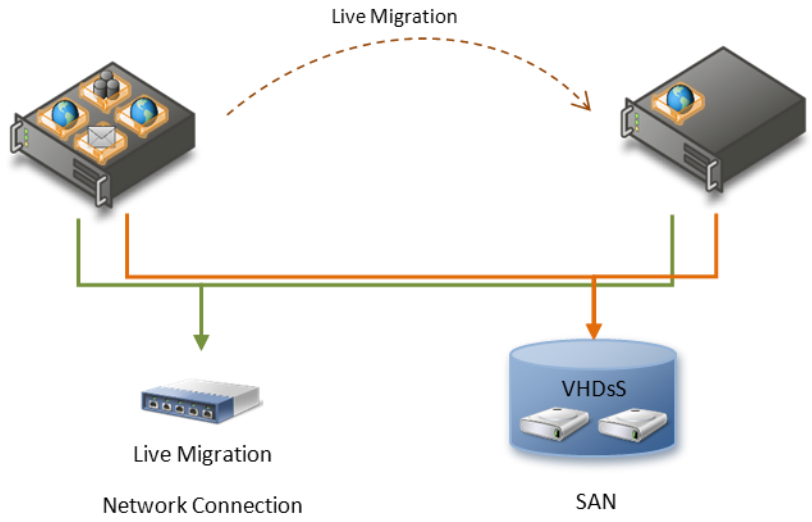
Live Migration

Live Migration lets 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.

Note
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 copies the memory of the virtual machine from the current host to the new host. The second phase transfers 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.



Additionally, increasing the number of send and receive buffers on each network adapter that is involved in the migration can improve migration performance. For more info, see **Network subsystem**.

Windows Server 2012 R2 introduced an option to speed up Live Migration by compressing memory before transferring over the network or use Remote Direct Memory Access (RDMA), if your hardware supports it.

Detecting bottlenecks in a virtualized environment

This section should give you some hints on what to monitor by using Performance Monitor and how to identify where the problem might be when either the host or some of the virtual machines do not perform as you would have expected.

Processor bottlenecks

Here are some common scenarios that could cause processor bottlenecks:

- One or more logical processors are loaded
- One or more virtual processors are loaded

You can use the following performance counters from the host:

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

- Virtual Processor Utilization - \Hyper-V Hypervisor Virtual Processor(*)\% Total Run Time
- Root Virtual Processor Utilization - \Hyper-V Hypervisor Root Virtual Processor(*)\% Total Run Time

If the **Hyper-V Hypervisor Logical Processor(_Total)\% Total Runtime** counter is over 90%, the host is overloaded. You should add more processing power or move some virtual machines to a different host.

If the **Hyper-V Hypervisor Virtual Processor(VM Name:VP x)\% Total Runtime** counter is over 90% for all virtual processors, you should do the following:

- Verify that the host is not overloaded
- Find out if the workload can leverage more virtual processors
- Assign more virtual processors to the virtual machine

If **Hyper-V Hypervisor Virtual Processor(VM Name:VP x)\% Total Runtime** counter is over 90% for some, but not all, of the virtual processors, you should do the following:

- If your workload is receive network-intensive, you should consider using vRSS.
- If the virtual machines are not running Windows Server 2012 R2, you should add more network adapters.
- If your workload is storage-intensive, you should enable virtual NUMA and add more virtual disks.

If the **Hyper-V Hypervisor Root Virtual Processor (Root VP x)\% Total Runtime** counter is over 90% for some, but not all, virtual processors and the **Processor (x)\% Interrupt Time and Processor (x)\% DPC Time** counter approximately adds up to the value for the **Root Virtual Processor(Root VP x)\% Total Runtime** counter, you should ensure enable VMQ on the network adapters.

Memory bottlenecks

Here are some common scenarios that could cause memory bottlenecks:

- The host is not responsive.
- Virtual machines cannot be started.
- Virtual machines run out of memory.

You can use the following performance counters from the host:

- Memory\Available Mbytes
- Hyper-V Dynamic Memory Balancer (*)\Available Memory

You can use the following performance counters from the virtual machine:

- Memory\Available Mbytes

If the **Memory\Available Mbytes** and **Hyper-V Dynamic Memory Balancer (*)\Available Memory** counters are low on the host, you should stop non-essential services and migrate one or more virtual machines to another host.

If the **Memory\Available Mbytes** counter is low in the virtual machine, you should assign more memory to the virtual machine. If you are using Dynamic Memory, you should increase the maximum memory setting.

Network bottlenecks

Here are some common scenarios that could cause network bottlenecks:

- The host is network bound.
- The virtual machine is network bound.

You can use the following performance counters from the host:

- Network Interface(*network adapter name*)\Bytes/sec

You can use the following performance counters from the virtual machine:

- Hyper-V Virtual Network Adapter (*virtual machine name name*<*GUID*>)\Bytes/sec

If the **Physical NIC Bytes/sec** counter is greater than or equal to 90% of capacity, you should add additional network adapters, migrate virtual machines to another host, and configure Network QoS.

If the **Hyper-V Virtual Network Adapter Bytes/sec** counter is greater than or equal to 250 MBps, you should add additional teamed network adapters in the virtual machine, enable vRSS, and use SR-IOV.

If your workloads can't meet their network latency, enable SR-IOV to present physical network adapter resources to the virtual machine.

Storage bottlenecks

Here are some common scenarios that could cause storage bottlenecks:

- The host and virtual machine operations are slow or time out.
- The virtual machine is sluggish.

You can use the following performance counters from the host:

- Physical Disk(*disk letter*)\Avg. disk sec/Read
- Physical Disk(*disk letter*)\Avg. disk sec/Write
- Physical Disk(*disk letter*)\Avg. disk read queue length
- Physical Disk(*disk letter*)\Avg. disk write queue length

If latencies are consistently greater than 50ms, you should do the following:

- Spread virtual machines across additional storage
- Consider purchasing faster storage
- Consider Tiered Storage Spaces, which was introduced in Windows Server 2012 R2
- Consider using Storage QoS, which was introduced in Windows Server 2012 R2
- Use VHDX

See Also

Performance Tuning for Server Roles

Performance Tuning for Workloads

This section describes performance tuning guidelines for workloads in the following topics:

- Performance Tuning for NTttcp
- Using the File Server Capacity Tool (FSCT)
- Using the SPECsfs2008 File Server
- Performance Tuning for the Sales and Distribution Workload
- Performance Tuning for Online Transaction Processing (OLTP)

See Also

Performance Tuning Guidelines for Windows Server 2012 R2

Performance 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 info on NTttcp, see How to Use NTttcp to Test Network Performance. You can download the latest version from TechNet.

When setting up NTttcp, you should 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.

The following table lists some examples:

Syntax	Details
NTttcp.exe -s -m 1,0,10.1.2.3 -a 2 -t 10	This is an example of a sender with the following characteristics: • 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 KB. • Default number of buffers to send: 20 KB. • Test runs for 10 seconds.
NTttcp.exe -r -m 1,0,10.1.2.3 -a 6 -t 10	This is an example of a receiver with the following characteristics: • 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 KB. • Posts full-length (64 KB) receive buffers. • Test runs for 10 seconds.

Important
Make sure to enable all offloading features on the network adapter.

TCP/IP Window Size

For 1 GB adapters, the settings shown in the table above should provide good throughput because NTttcp sets the default TCP window size to 64 KB 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.

See Also

Performance Tuning for Workloads

Using the File Server Capacity Tool (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.

In this topic:

Tuning for servers

The following REG_DWORD registry settings can affect the performance of file servers:

- **NtfsDisable8dot3NameCreation**

  ```
  HKLM\System\CurrentControlSet\Control\FileSystem\NtfsDisable8dot3NameCreation
  ```

  The default in Windows Server 2012 R2 and Windows Server 2012 is 2, and in previous releases it is 0. This parameter determines whether NTFS generates a short name in the 8dot3 (MSDOS) 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 and Windows Server 2012 R2.

  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 R2, 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 by using fsutil. For example, to disable 8dot3 name creation on volume D, run **fsutil 8dot3name set d: 1** from an elevated

command prompt. If you are disabling new 8dot3 name creation on a volume that has existing data, consider stripping existing 8dot3 names from the volume by using fsutil. 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:**.

- **TreatHostAsStableStorage**

  ```
  HKLM\System\CurrentControlSet\Services\LanmanServer\Parameters\TreatHostAsStableStorage
  ```

  The default is 0. This parameter disables processing write flush commands from clients. If the registry entry is 1, the server performance and client latency can improve but there's potential risk of data loss. Workloads that resemble the NetBench file server benchmark benefit from this behavior.

Tuning for clients

The following REG_DWORD registry settings can affect the performance of client computers:

- **DormantFileLimit**

  ```
  HKLM\system\CurrentControlSet\Services\LanmanWorkstation\Parameters\DormantFileLimit
  ```

  This settings applies to 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.

See Also

Performance Tuning for Workloads

Performance Tuning for File Servers

Using the SPECsfs2008 File Server

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.

Tuning parameters for NFS file servers

You can tune the following registry parameters 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

See Also

Performance Tuning for Workloads

Performance Tuning for File Servers

Performance Tuning for the Sales and Distribution Workload

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 sub-second 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 Server 2012. Some of these recommendations might not apply to the same degree for production systems.

In this topic:

Operating system tunings on the server

Configure the following in advanced system settings:

- Under the **Performance** heading, click **Settings**, click **Advanced**, and then click **Change**. Set one or more fixed-size page files where the **Initial Size** equals **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 virtual memory on the application server.
- Under the **Performance** heading, click **Settings**, click **Visual Effects**, and then click the **Adjust for best performance** option.
- 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 by using Group Policy.
- Disable User Account Control (UAC), and then restart the computer

  Note
  Disabling UAC can be used for benchmark environments, but enabling UAC might be a security compliance requirement in production environments.

For virtualized environments, 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. You can do this by configuring the base and maximum RSS logical processors.

Database server tunings

When the database server is running Microsoft SQL Server, you should set the following SQL Server configuration options by using the sp_configure stored procedure. For more info about sp_configure, 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** 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 hardware that supports NUMA, 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 the preferred NUMA node hints (applies to Windows Server 2012 R2, Windows Server 2012, and Windows Server 2008 R2).
  - Set the NUMA preferred node from an elevated command prompt:

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

  - Query the setting:

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

  - 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, you should use TCP/IP.

SAP application server tunings

- 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 hardware that supports NUMA, 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 virtual benchmark deployments, one needs to use at least D-VMQ and should enable RSS in the Microsoft Virtual Network Adapter properties of the virtual machine when using a 3-Tier configuration. For 2-Tier, configuration usage of RSS within the virtual machine is not necessary. On the host consider restricting the VMQ processors to logical processors (and NUMA nodes) on which contention with SAP virtual machines is minimized. For 2-Tier benchmark workloads, you should use SR-IOV-enabled network adapters. For 3-Tier configurations, the SR-IOV network adapters need to support RSS over SR-IOV since RSS within the virtual machine is required.
- Running SAP SD benchmarks in virtual machines, the SAP memory should use large pages. For this purpose, the user context starting SAP requires the **Lock Pages in memory** user right assignment. The SAP profile parameter to enable large pages is em/largepages=yes
- In Windows Server 2012 R2, Windows Server 2012, and Windows Server 2008 R2, the operating system supports more than 64 logical processors. On hardware that supports NUMA, you should set the preferred NUMA nodes in addition to setting hard affinities by using the following steps:
  - Set the NUMA preferred node from an elevated command prompt:

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

  - Query the setting:

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

  - Remove the setting:

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

- To allow each SAP worker process in a dialog instance to inherit the ideal NUMA node from its Win32 service, create the following REG_DWORD registry entries 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\
  ```

You can use Coreinfo to provide topology details about logical and physical processors, processor sockets, NUMA nodes, and processor cache. For more info, 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, use the following list of performance counters to monitor 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(*)*

See Also

Performance Tuning for Workloads

Performance Tuning for Online Transaction Processing (OLTP)

This topic describes performance tuning methods and recommendations for Online Transaction Processing (OLTP).

In this topic:

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 plan to **High Performance**.
- Enable SQL Server to use large pages by enabling the **Lock pages in memory** user right assignment for the account that will run the SQL Server in Group Policy.

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

- Configure the page files
  - Under the **Performance** heading, click **Settings**, click **Advanced**, and then click **Change**. Set one or more fixed-size page files where the **Initial Size** equals **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 virtual memory on the application server.

    - Under the **Performance** heading, click **Settings**, click **Visual Effects**, and then click the **Adjust for best performance** option.
- Configure network adapters.
    - Enable VMQ and dynamic VMQ for virtualized environments.
    - Disable unused network devices by using Device Manager.
    - For advanced network tuning information, see Performance Tuning for Network Subsystems.
- Configure storage devices.
    - Disable low priority I/O. For each logical volume in HKLM\SYSTEM\CurrentControlSet\Enum\SCSI under Device Parameters\ClassPnp, create a REG_DWROD registry entry named **IdlePrioritySupported** and set the value to 0.
    - For advanced storage tuning information, see Performance Tuning for Storage Subsystems.
- Configure the disks for advanced performance by using Disk Management. Right-click each disk, click **Properties**, click **Policies**, and then click **Advanced Performance**.

Note

This setting is for direct attached storage only.

SQL Server tunings

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 performance monitor counters
- **-T661** Disable the ghost record removal process
- **-T834** Use Microsoft Windows large-page allocations for the buffer pool

Other parameters that are sometimes used are as follows:

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

  In Server Configuration Manager, 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 Network Subsystems.
    - 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, if you have enough storage capacity, you should use RAID 1+0. If you do not have enough capacity, you should 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.

Client tunings

Tune the clients as follows:

- Configure client systems the same way that the server under test is configured.
- 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 OLTP 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 Performance Monitor.

- IPv4*
- Memory*
- Network Interface(*)*
- PhysicalDisk(*)*
- Processor Information(*)*
- Synchronization(*)* for Windows Server 2008 R2
- SynchronizationNUMA(*)* for Windows Server 2012 and WS12 R2
- 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 the figure below. The first part of the run in that figure 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.

You can use other tools, such as Windows Performance Analyzer (WPA), to perform additional analysis.

Root counters

For virtualized configurations of the OLTP workload, use the following list of performance counters to monitor the Hyper-V root:

- 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(*)*
- IPv4*
- Hyper-V VM Vid Partition(*)*

- NUMA Node Memory(*)*
- 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 the following info from the guest operating system:

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

See Also

Performance Tuning for Workloads

Additional Resources for Performance Tuning Guidelines

Use the links in this topic to learn more about the concepts that were discussed in Performance Tuning Guidelines for Windows Server 2012 R2.

Websites

- Windows Server 2012 R2
- Windows Server Performance Team Blog
- Windows Server Catalog
- SAP Global Benchmark: Sales and Distribution (SD)
- Windows Sysinternals
- Transaction Processing Performance Council
- Windows Assessment and Deployment Kit

Power Management

- Power Policy Configuration and Deployment in Windows
- Using PowerCfg to Evaluate System Energy Efficiency
- Interrupt-Affinity Policy Tool
- Processor Power Management (PPM) Tuning for the Windows Server Balanced Power Plan

Networking Subsystem

- Scalable Networking: Eliminating the Receive Processing Bottleneck—Introducing RSS
- Windows Filtering Platform
- Networking Deployment Guide: Deploying High-Speed Networking Features
- NT Testing TCP Tool (NTTTCP) 3.0
- Web Capacity Analysis Tool (WCAT)
- File Server Capacity Tool (FSCT)
- Windows Server 2012 R2 NIC Teaming (LBFO) Deployment and Management

Network Workload

- Ttcp
- How to Use NTttcp to Test Network Performance

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.

Web Servers

- 10 Tips for Writing High-Performance Web Applications

File Servers

- Performance Tuning Guidelines for Microsoft Services for Network File System
- [MS-FSSO]: File Access Services System Overview
- How to disable the TCP autotuning diagnostic tool

Active Directory Servers

- Active Directory Performance for 64-bit Versions of Windows Server 2003
- How to configure Active Directory diagnostic event logging in Windows Server 2003 and in Windows 2000 Server

Virtualization Servers

- Hyper-V Dynamic Memory Configuration Guide
- NUMA Node Balancing
- Hyper-V WMI Provider
- Hyper-V WMI Classes
- What's New in Hyper-V in Windows Server 2012 R2
- About Virtual Machines and Guest Operating Systems
- Optimizing and Troubleshooting Hyper-V Storage
- Optimizing and Troubleshooting Hyper-V Networking

Print Servers

- Print Server Scalability and Capacity Planning

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

- Setting Server Configuration Options
- How to: Configure SQL Server to Use Soft-NUMA
- How to: Map TCP/IP Ports to NUMA Nodes
- ALTER SERVER CONFIGURATION (Transact-SQL)
- SAP with Microsoft SQL Server 2008 and SQL Server 2005: Best Practices for High Availability, Maximum Performance, and Scalability
- SAP Hyper-V benchmark

Server Tuning Tools

- Microsoft Server Performance Advisor