E I G H T

Excerpts From:

Putting Theory into Practice: Sizing and Tuning Back Office Solution Scenarios

CHAPTER OBJECTIVES

Introduction

Have you ever wanted specific, practical, hands-on configuration recommendations that include guides for tuning and sizing Windows 2000 systems? Or have you ever wished for a performance checklist that can help you design and tune your new Windows 2000 solution? If so, you are in the right place. This chapter focuses on implementing specific Windows 2000 system solutions in the following application areas:

- Windows 2000 file servers and server consolidation
- Windows 2000 backup servers
- Messaging servers: Microsoft Exchange server
- Database servers: Microsoft SQL 7.0
- eCommerce: World Wide Web servers using Microsoft IIS 5.0

I worked with experts in all of the above fields to investigate strategies and tactics, perform stress testing (too many hours in the lab), and discuss real-world deployments. Following is the team that provided excellent input:

file servers and backups—John Pollen and Arnie Shimo; Microsoft Exchange—Jeff Lane, Mike Gillam, and Rick Patrick; Database Servers—Rob Cochran and Troy Landry; and Web Servers—Dan Matlick and Yemi Famogun. These colleagues are an outstanding mix of senior engineers and architects who develop and manage IT solutions for NASA's ODIN IT Outsourcing project for the Office of Space Flight, OAO Corporation, and MightyView Corporation.

Chapters 2 and 3 introduced tuning and sizing methodologies that serve as a foundation for developing Windows 2000–based solutions. Chapters 4 through 7 investigated tuning and sizing techniques for each of Windows 2000's major resources and provided examples. This chapter integrates the entire system, using the methodologies and techniques of previous chapters to work through various scenarios to implement Windows 2000 systems. These solution scenarios are designed to be realistic (and fun) and are based on feedback from my previous book, *Tuning and Sizing NT Server*, and associated web site: http://www.TuningAndSizing.com (formerly http://www.TuningAndSizingNT.com).

These case studies may or may not fit your environment exactly, which is okay. Why? The goal here is to help provide insight into tuning and sizing of Windows 2000 for maximum performance; to improve your understanding of the methodologies used; and to provide specific configuration recommendations via worksheet templates that you can reference (electronic copies of templates are available at http://www.TuningAndSizingWindows.com). Use these worksheet templates for reference; they may help your design process for Windows 2000 solutions. You may end up not using some areas while adding others that arise from your own experience. These configurations provide a solid starting point when developing a Windows 2000 solution for your unique environment. Remember, tuning and sizing systems is part science and part art; thus, based on knowledge you have acquired or enhanced with the previous chapters, you can take these configurations as a starting point and customize them for your environment.

It does not matter if the Windows 2000 solution you are developing is based on Microsoft Exchange Server, Oracle, Microsoft SQL Server, or a file server. Once you understand the concepts behind the methodology, you can use them to your advantage whenever you tune and size any type of Windows 2000 solution—even as technology changes. Do not get bogged down in the very small details. Even if your solutions are 12MB to the left or right, following a sound methodology will get you to a solution that is in the ball park, which can then be tuned to a very good fit.

For each of the scenarios, a consistent outline approach is followed so that you can quickly get the specific information for each application area.

Solution Scenario 1: Windows 2000 File Server Consolidation

Introduction

This scenario presents the most common services provided with the Windows 2000 platform, file servers. Many people today have older file servers that are ready to be retired. This scenario addresses that challenge.

Application Description

File servers are the most common Windows 2000 application function. A file server does just what the name implies—it acts as a data repository for storing and sharing files.

Application Performance Characterization

The key application that provides Windows 2000 file services is the native Windows 2000 server service. The server service provides RPC support and file, print, and named pipe sharing services.

Configuring the server service itself is completed in three places. Basic operations are controlled under Start / Programs / Administrative Tools / Computer Management / Services and Applications / Services / Server properties. Here you can control what is active on your server and what the system should do if it fails, such as try to restart the process. You also have the option of activating the File and Print sharing capabilities of the server service under Start / Settings / Network and Dial Up Connections / Properties for your LAN connection. You can then activate or disable File and Print Sharing for Microsoft Networks. When you select the properties of File and Print Sharing you access the controls for Windows 2000 memory management, specifically the file system cache. Memory management options are reviewed in chapter 5.

Windows 2000 file servers use the native file system to store files on the disk subsystem. For smaller file servers (< 4 disk drives), performance bottlenecks typically occur in the disk subsystem, memory subsystem, and then the other subsystems. On large file servers (> 5 disk drives), performance bottlenecks typically occur in the network subsystem, memory subsystem, disk subsystems, and then CPU subsystems.

When a client connects (Windows 2000 workstation) to the file server (Windows 2000 server service) over the network, the Windows 2000 file system cache is used heavily because it caches the network file services requests.

Sysmon Objects and Counters: File Server Bottleneck Detection and Sizing

The Sysmon objects and counters outlined in the previous chapters for CPU, memory, disk, and network subsystems are still applicable to Windows 2000 file servers. It is important to understand the big picture when tuning or sizing your system. In addition to the previously reviewed rules of thumb for solving bottlenecks and sizing objects and counters, specific objects and counters for file servers are outlined in Table 8.1.

In addition to the Sysmon counters in Table 8.1, it is helpful to understand which Windows 2000 process is running the server service. Locating the server service with Task Manager (launched when ctrl+alt+delete is depressed and Task Manager is selected) is not as obvious when using Windows 2000, because it operates under the Service Control Manager's services.exe executable. You can observe the LanManServer, LanManWorkstation, and some of the other services that the Service Control Manager (services.exe) is running by executing the tlist command from the support tools directory on the Windows 2000 distribution media (*tlist.exe –s | find "services.exe"*). With this information, you gain insight if other applications are using resources besides those associated with file services. If you are running a system management application, such as Concord's System Edge (http://www.Concord.com) or HP Open View's ManageX (http://www.hp.com), you will want to be alerted if for any reason the server service stops operating. If you do not have these types of system management software packages available, you can configure Windows 2000 to take action. This is accomplished by selecting Start / Programs / Administrative Tools / Component Services / Service / Server services / Recovery Tab; set Windows 2000 to take action on a service that has stopped. Actions include restart the service if it fails or run a file. Since the option to run a file is available, the options available to you with Windows 2000 are limited only by your imagination and scripting abilities.

TABLE 8.1 *Sysmon Counters for File Servers*

Object/Counter	Definition	Value to Monitor
Server: Sessions	The number of sessions currently active in the server. Indicates current server activity.	Influences sizing. Determines concurrent file server users.
Server: Bytes Total/sec Bytes Received/sec Bytes Sent/sec	The number of bytes the server has sent to and received from the network.	Influences tuning and sizing. Provides a good indicator of how much workload is contributed to file server services particularly if other applications are being hosted on the same system. Total Bytes/sec are helpful in sizing the server (network, disk subsystems). Bytes sent/received helps to characterize workload which is used when tuning network and disk subsystems.

TABLE 8.1 *Sysmon Counters for File Servers (Continued)*

Object/Counter	Definition	Value to Monitor
Server: Work Item Shortages	The number of times STATUS_DATA_NOT_ACCEPTED was returned at receive indication time. This occurs when no work item is available or can be allocated to service the incoming request.	Influences tuning. File servers must be healthy to run well. This counter indicates whether the InitWorkItems or MaxWorkItems parameters might need to be adjusted.
System: System Uptime	System Uptime is the elapsed time (in seconds) that the computer has been running since it was last started. This counter displays the difference between the start time and the current time.	Commonly referred to as CYA. File servers have become more important in the enterprise. This counter helps track the availability of your server. If you ever debate whether the physical server was down or a physical network was down, consider tracking this counter. If the network is down and the counter is greater than the detected down time, the network was the problem, not the server. This information can be obtained via SNMP requests also, if the proper security information is available such as the SNMP community string.

Scenario 1 Step by Step: Sizing and Tuning a Mid-Range Windows 2000 File Server

Sizing the Initial File Server Configuration

In chapter 3, the core sizing methodology was presented for sizing your Windows 2000 solution. That methodology is followed here for the development of the initial Windows 2000 server configuration.

1. DEFINE OBJECTIVE(S)

Due to almost constant hardware maintenance failures, multiple small servers have—after five years—reached the end of their usefulness. A new system is needed to share data for a mid-sized company. Since long-term costs—total cost of ownership (TCO)—are a concern, a decision is made to consolidate the current 20 file servers to 2 Windows 2000 file servers. This will meet the company's two objectives: (1) implement a new file server solution and (2) lower TCO by reducing the number of servers that must be managed. Using

two Windows 2000 file servers reduces concern about availability, even though a single file server could suffice.

2. UNDERSTAND BUSINESS AND TECHNICAL REQUIREMENTS NEEDED TO MEET YOUR OBJECTIVE(S)

- Which application(s) will meet your functional objective(s)?
  For security, all clients are Windows 2000 Professional or Windows NT. Thus, a desktop that supports international security standards is in place. Core file services provided with Windows 2000 will meet the company's requirements. It is important to understand that the servers will support multiple departments, and each department has its own specific needs.
- What are the availability requirements?
  These are very important servers; they will support the entire company. Backups must be completed every night (after 7:00 P.M. and before 5:00 A.M.) and the system must be restored to service within four hours.
- What are the ramifications if your system is down?
  It is important to keep a high availability Monday through Friday, but planned weekend maintenance is acceptable.
- What are the disk subsystem storage capacity requirements?
  The old system of 20 servers supports 200GB of data. It is estimated that this disk storage capacity will grow to 400GB in the next year.

3. DETERMINE LOADING CHARACTERISTICS

- *Number of total users per server*
  500
- *Number of concurrent users per server*
  60
- *Disk workload characteristics (read/write environment)*
  File server characteristics vary from department to department. Generally a mix of 70% read and 30% write based on a similar office environment.
- *Size of records/transactions*
  400KB documents on average.
- *Transaction rates*
  Based on historical information, the average document size is 400KB. Business customers expect a two-second response time. To determine the throughput needed to provide this response time, follow this basic formula: Number of concurrent users (60) multiplied by the size of average records (0.4MB) divided by response time (2 sec). For

this environment, a server that provides 12MB/sec (96Mbits/sec) is needed to meet the company's requirements. Another view of this is to consider that, in two seconds, 30MB of data must be transferred when 60 of the 500 users are concurrently accessing the file server.

- *Physical location of users (influences communications such as WAN, LAN, dial-up, terminal services, etc.)*
  All users reside on site and are connected via the local LAN.
- *Type of work the user will complete*
  Office automation to support sales activities and some engineering. Heavy on presentations and proposals.

4. DETERMINE PERFORMANCE REQUIREMENTS

Response time is the most important factor for the office. They would love to have their average response time less than two seconds on average.

5. UNDERSTAND FUTURE BUSINESS REQUIREMENTS

This is a public company, so growth is very important. The company expects to expand by at least 20% each year.

6. UNDERSTAND FUTURE SYSTEM ARCHITECTURES

The local team would like to quickly put in a system and run it without replacement for three years. Local support staff is small, which is just unheard of in the IT industry. Everyone always has the right size IT team so that long hours never occur. Rebuilding servers every few months to meet growing demands is undesirable, as is the associated downtime.

7. CONFIGURE THE SYSTEM

Finally, we get to the good stuff. At this point, you can configure the system. Choose the system resources required by your objective, with consideration for steps 1 through 6, and obtain the system. As you configure the system, look at each individual resource to meet the overall system's goals. Software and hardware vendors are good sources of information. They can suggest configurations that will get you started. Reviewing relevant industry benchmarks also can help provide insight into the initial configuration. Industry benchmarks are explored later in this chapter. The following steps are presented to help guide you through the actual system configuration process.

Having some sort of idea on how each system resource works and relates to the overall system performance is helpful when configuring your system. Chapters 4 through 7 reviewed each system resource in greater detail.

7.1 Application Characteristics Review • The application characteristics review was completed at the top of the Windows 2000 file server section.

7.2 Review Each System Resource

Disk I/O Requirements • This is the driving requirement and one where Windows 2000 gives you significant control, especially for file servers. First, we size the disk subsystems; then we can work backward from the disk subsystem configuration to size the rest of the file server. From the business requirements above, we will determine the disk storage and performance requirements.

Availability • Since this is a critical server, all data will reside on hot swappable RAID array–based disk subsystems that use some level of fault tolerance. Hot spare disk drives will also be incorporated into the solution to increase overall availability.

Performance Workload Characteristics • From the requirements above, the workload characteristics are typically read-intensive and will undoubtedly be random in nature. RAID levels 5, 10, and 1 are all valid options.

Performance Throughput and Transfers/sec • In this environment, the basic information provided is not Transactions/sec but throughput. There are two aspects to consider: operational performance and backup performance. Operationally, the file server must support 12MB/sec of continuous throughput based on the business's requirements. Although caching of data will aid in overall performance delivery, it is the disk subsystem that drives the performance of the file server in this random workload environment. Assuming that we use RAID level 5 due to its strong combination of availability, performance characteristics that match the expected workload, and good price/performance, we can use the disk performance data from chapter 6 (Table 6.9, Executive Summary of RAID Performance) to size the disk subsystem. Each 10,000rpm disk drive in a RAID 5 disk array provides approximately 1.11MB/sec of random read performance. Thus, to achieve 12MB/sec of continuous throughput, 11 disk drives are required. Yes, the math works out to 10.8 disk drives, but you always round up to the next integer. If we use 11 18GB disk drives in a single RAID 5 array, we will have only 180GB of available disk storage due to the parity overhead from RAID 5. (Usable disk space in a RAID 5 array = [(capacity of the N disk drives) × $(1/N)$]; thus, a twelfth disk drive is needed so that 198GB of usable storage is available.

Backup performance is reviewed in more detail in the Windows 2000 backup server scenario below. For now, there are two backup performance factors to consider: backup and restore. The current disk storage capacity is planned to start at 100GB and grow to 200GB within the year. Planning for 200GB to be backed up, a 12-disk RAID 5 array provides 32.04MB/sec (115GB/hour) when being read in a sequential fashion (2.67MB/sec per disk drive × 12 disk drives × 60sec/min × 60min/hour), which results in a full backup in less than four hours.

However (there is always a trade-off!), when you wish to restore the disk array, each disk in a RAID 5 array provides only 0.95MB/sec of sequential write performance. Thus, a 12-disk array provides 41GB/hour of restoration throughput (12 disk drives × 0.95MB/sec × 60sec/min × 60min/hour). Based on this data, it will take approximately 4.8 hours to restore the data. This does not meet the business goal of a full restore in four hours. To meet the business goal, three more disk drives must be added to the array to improve the total overall write performance throughput of the raid performance array from 41GB/hour to 51.3GB/hour. This results in a restore time that meets our target of a four-hour restore.

From the above analysis, we take the highest number of disk drives that meets our goals. Thus, 15 disk drives are needed to meet our business performance requirements.

Storage Capacity • Each file server must support 100GB today and 200GB by the end of the year. For maximum performance and to have the capability to run disk defragmentation tools, we do not want the disk drives' used capacity to exceed 60–80%. Cost is a concern to this company, so they plan on 74% used disk space. Factoring this tuning tactic into our total disk capacity needed, we will require 270GB of available disk space. The 270GB of disk capacity spread across 15 disk drives indicates each drive must support 18GB. Factoring in RAID 5 overhead, a sixteenth disk drive is needed to account for the parity overhead so that a full 270GB is actually available for use.

Reality Check • Industry benchmarks are always good for reference. Here we used internally run benchmarks on Windows 2000 disk subsystems for many of the measurements, calculations, and estimates for sizing. For file servers in general, the Ziff Davis NetBench benchmark is relevant. Keep in mind that real-world environments tend to cache activities slightly more than a well-designed benchmark. The NetBench benchmark results reviewed in chapter 3 were obtained from a Windows 2000 file server configured with a two-CPU Pentium III 650MHz Server, 1GB of RAM, 1 Gigabit Ethernet NIC, and a 14-disk RAID 5 array. This configuration resulted in providing 225Mbits/sec (28MB/sec) of throughput. This higher performance was achieved—with one less disk drive than is used in our scenario—because many of the operations were cached in the Windows 2000 file system cache. Our approach is slightly more conservative. Although the two examples are not an exact match, the comparison helps us to ensure that our current disk subsystem sizing is in the ball park and provides an excellent starting point to build our system.

Other Disk Requirements • Are there any other applications that require disk space? For our file server example, we still need to place Windows 2000 somewhere. Chapter 5 showed that you never want to place the operating system and pagefile on a RAID 5 array, for both operating system and perfor-

mance reasons. Due to the availability requirements, the operating system will be placed on its own RAID 1 mirror composed of two 9GB disk drives configured with one partition. The pagefile will be placed on this same RAID 1 Mirror in an effort to lower costs (vs. placing the pagefile on its own dedicated high-speed drive). As long as we do not page on a regular basis (something you never want to do!), this configuration will suffice but may need to be modified later.

Based on all of the above disk information, Table 8.2 outlines the overall disk subsystem requirements.

TABLE 8.2 *File Server Disk Subsystem Solution*

Application	Minimum Disk Storage Needed	Logical Partition	Recommended Configuration (All 10,000rpm, Ultra2 SCSI Disk Drives)
Windows 2000 operation system requirement	<500MB	C:	2 9GB Disk Drives in RAID 1 Array (9GB usable/3.15GB filled)
Pagefile	2GB	C:	
System management agent	5MB	C:	
Antivirus application	20MB	C:	
Security application (server-based firewall and IDS)	30MB	C:	
Data store 1	100GB	H:	9 18GB Disk Drives in a RAID 5 Array (162GB usable/100GB filled—61% used)
Data store 1	100GB	I:	9 18GB Disk Drives in a RAID 5 Array (162GB usable/100GB filled—61% used)

Note: More disk drives are used here than planned, 9 vs. 8. Why is that? The container that holds the RAID array will hold only 12 disk drives at a time. Thus, two containers were needed. This forced us to use two RAID 5 arrays, one for each container, and, of course, that introduced more overhead for parity. This drove the need for additional disk drives in the solution.

CPU Requirements • The primary application to support is the server service that provides file services. The CPU's workload consists of supporting network operations, disk operations, and the associated logging and security checking of customers' rights. From historical information, the 20 Pentium 133MHz CPU-based servers (single CPU) that provided these file services ran at 70% usage on average. Twenty Pentium 133MHz CPUs running at 70% works out to 14 Penitium 133MHz CPUs running at 100%. Using the CPU consolidation chart from chapter 4, these 14 Pentium 133MHz CPUs equates to the CPU power of approximately two Pentium III class CPUs. What? An eight to one ratio? Yes,

this is a good place to start. As always, there are other factors that affect overall performance and these are discussed at length in earlier chapters. For this small configuration, however, we are reviewing only the CPU subsystem. Refer back to chapter 4 for a detailed discussion on CPU performance.

Industry standard benchmarks are also helpful when sizing systems. For a network-intensive file server environment, the Ziff Davis benchmark is a relative benchmark. Our solution needs to drive 96Mbits/sec (12MB/sec) during regular operations. From the benchmark information provided in chapter 3, a Windows 2000 file server with two Pentium III CPUs drove over 250Mbits/sec of overall throughput. Thus, a single Pentium III CPU should suffice. However, during backup operations, we will need to drive an even higher rate of disk and network operations to support just over 250Mbits/sec of throughput, similar to the Ziff Davis results. Based on this information, two Pentium III CPUs will be required, even though the bulk of the processing power will not be needed during normal operating hours. It is the backup and restore-to-service operations that drive the computing power of this specific file server solution.

Memory Requirements • Sizing RAM is an additive process. Our first step in sizing memory resources is to outline all applications that will use any memory resources.

TABLE 8.3 *Windows 2000 File Server Memory Sizing*

Application	Memory Usage (Working Set)
Windows 2000 operation system requirement	128MB
Windows 2000 file system cache (max for configurations less than 1GB in size)	432MB
Overhead for antivirus software running on server	10MB
Overhead for system management software running on server	8MB
Overhead for intrusion detection software	12MB
Memory needed per user connection	Included in file system cache overhead
Total Estimated Memory Usage	***592MB***

From historical data, each of the 20 former servers contained 128MB/sec of RAM. Since sizing RAM is an additive process, our analysis of the system's memory usage indicates that 1GB of RAM is needed in the new server. If we again review the Ziff Davis benchmark results, 1GB of memory was configured in the system.

Great data, but what amount of memory should be configured? Although memory usage is an additive process, there are some shared memory resources. Each of the 20 servers used in the earlier historical extrapolation had 64MB of memory used by the operating system, leaving approximately 640MB. In our memory sizing chart above—and what we have learned from our Windows 2000 memory usage investigation—adding any more than 592MB would not help this file server configuration. (Remember that Windows 2000 limits its file system cache usage to 432MB when less than 1GB of RAM is in the server.) Thus, from this information and the fact that server hardware supports memory in 512MB blocks, 512MB should handle the workload. Through proactive performance management, more RAM can be added as needed, but for now let's save a few dollars and not overconfigure the server.

Network I/O Requirements • This file server solution is projected to support the delivery of 96Mbits/sec in a local area network. The current LAN is planned to receive a switched Gigabit Ethernet in two years. For now, a full duplex 100BaseTX Ethernet-based connection should easily support the current bandwidth requirements. From chapter 7 we see that Windows 2000 can support 188Mbits/sec using a 100BaseTX full duplex Ethernet-based switched connection. Thus, we do have some room to grow until the Gigabit Ethernet network upgrade. Since this is an estimate of network usage, two Network Interface Cards (NIC) will be obtained that support Adaptive Load Balancing (ALB) so that two 100BaseTX channels can be teamed/trunked together if the need arises.

Leveraging two NICs also enables us to meet our customers' concerns over availability. Besides the increase in available bandwidth, NICs that support ALB also support fail-over operations. If one NIC fails, the second NIC will handle all the network traffic. One company that provides NICs with these features is Intel Corporation (http://www.intel.com). If even higher levels of availability were required, the next step would be to consider clustering technologies.

7.3 System Architecture Relationship Check • This is a final, common-sense check of a server. Since the business requirements mandate keeping this new file server for at least three years, leading-edge server technology will be obtained. Not cutting edge, as that technology tends to demand a premium price. From all of the sizing information above, we now have the following configuration: a one- to two-CPU-capable server, two Pentium III 800MHz CPUs, each with 256KB of full-speed (ATC) Level 2 cache, a 133MHz System Bus, 512MB of RAM, a single 64-bit 33MHz PCI I/O bus, a single hardware-based RAID controller, two disk drives for the operating system, 20 disk drives for the internal and external RAID array, and two 100BaseTX NICs. This configuration also meets our architecture checklist from chapter 4, which recommends a minimum of 256MB of RAM per Pentium III CPU.

An example server that would support this configuration is the Compaq DL-380 and standard Compaq external RAID array. Unfortunately, the external

RAID array only supports 12 disk drives per array. Due to this, two external SCSI RAID array units are obtained and the disk drives split between the two. Nine disk drives in one RAID 5 array, and nine disk drives in the second RAID 5 array. Why is another disk drive needed? Since one contiguous RAID array cannot be used, the second array must have an additional disk drive to support the disk capacity lost to RAID 5 overhead. Remember, if you have nine disks in a RAID 5 array, you lose the capacity of one disk drive to the parity information. All disk drives are active; you just lose the capacity of a single drive. In our example, 18GB (one disk drive) of logical data is lost to parity overhead per RAID 5 array, but 125GB is available for customer usage in each array.

Looking at the data path from the RAID arrays through the NICs, this file server configuration has no bottlenecks in place before we deploy. Potentially the CPU resources may be underpowered, but since we will proactively manage this file server, any shortcomings will be quickly detected and removed.

7.4 Future Business and System Architectures Check Review • This server has room to grow as needed—CPUs can be upgraded, disks can be dynamically added to the external disk arrays, RAM can be expanded to 4GB, and the NICs can be upgraded to Gigabit Ethernet.

7.5 Initial Hardware Configuration and Tuning • In this step, the initial configuration of the server hardware begins to take shape based on steps 7.1 and 7.2. See the Sizing and Tuning File Servers section below.

Table 8.4 *Windows 2000 File Server Hardware Configuration Data*

Mid-Range File Server Hardware Configuration
Compaq DL-380 (Why select the DL-380? We needed a solid system to use for a real-world reference.)

CPU Configuration	*CPUs Used in Solution*	*CPU Family*	*Level 2 Cache Size*	*Speed of CPU*	*Available CPU Slots*
	2	Intel Pentium III	256KB	733–1GHz	2 of 2 used
RAM Configuration	*Amount of RAM Used in Solution*	*System Bus Speed*	*Type of RAM*	*RAM Expansion*	
	512MB	133MHz	ECC SDRAM	4GB	
I/O Subsystem Configuration	*Number and Type of Adapters*	*Number of PCI I/O Buses*	*Placement of PCI Cards*	*Speed of PCI Bus*	
	2 Single Channel Full Duplex 100BaseTX	1	Second to top slot on PCI bus 0	64-bit 33MHz	
	I2O Enabled Ultra2 SCSI 2 Channel RAID Adapter		Top slot on PCI bus 0		

TABLE 8.4 *Windows 2000 File Server Hardware Configuration Data (Continued)*

Mid-Range File Server Hardware Configuration
Compaq DL-380 (Why select the DL-380? We needed a solid system to use for a real-world reference.)

Disk Subsystem Configuration	*Partition/ Mount Point*	*Files Located on partition*	*Placement of PCI Cards*	*Type of RAID Adapter*	*SCSI Channel*	*RAID Level*	*# Of 10K RPM UF Drives*
	C:	Windows 2000 installation, pagefile	N/A	Built-in RAID Adapter Compaq	1 of 1	1	2
	F:	Data Set 1	First Slot on PCI Bus	2-Channel External UF SCSI	1 of 2	5	9
	G:	Data Set 2	Same as adapter in row above	Same as adapter row above	2 of 2	5	9

7.6 INITIAL SOFTWARE CONFIGURATION AND TUNING • In this step, the initial software configuration of the server configuration begins to take shape based on steps 7.1 and 7.2. See the Tuning File Servers section below.

7.7 INITIAL GENERAL TUNING • See the Tuning File Servers section below.

8. STRESS TEST AND VALIDATE THE SERVER CONFIGURATION

For this scenario, historical information and industry standard benchmarks were relied upon to develop the file server configuration. File server behaviors are well known across the industry. Due to time constraints, a full stress test validation is not feasible.

9. PROACTIVELY FOLLOW THE CORE TUNING METHODOLOGY DURING STRESS AND VALIDATION TESTING

Not applicable as no stress tests were run.

10. DEPLOY THE SYSTEM INTO PRODUCTION

This was a real scenario and the file server system was deployed successfully. Expecting something else?

11. PROACTIVELY FOLLOW THE TUNING METHODOLOGY AFTER THE SYSTEM IS DEPLOYED

Environments do change. The information gathered following the core tuning methodology is helpful for staying one step ahead of user demand (see chapter 2). We can use that information to identify current load demands and solve potential system bottlenecks. This information is also useful when you are developing and sizing future system solutions and when you are adding additional capacity to your current system(s). If you have not quantifiably determined where and how to add additional capacity to your system, you are in great risk of wasting your initial investment.

General Windows 2000 File Server Sizing Configuration Chart Summary

Even though every environment is different, the following file server configuration chart provides excellent sizing guidelines for your initial server configurations. Follow the sizing template above, customize it to your environment and then proactively manage your Windows 2000 file server to an optimal fit!

TABLE 8.5 *Specific Configurations for Sizing File Servers*

Concurrent File Server Users (400KB average record size, 2-second response time)	System Bus Speed	Number of PCI I/O Buses	Pentium III 800MHz Class CPU	RAM	Number of UltraFast 10K rpm SCSI-3 Disk Drives (for data) in a RAID 5 array	Network Interface Card (NIC)
Small Scale <= 200 Users < 30 Concurrent	133MHz	1	1 (256KB Level 2 Cache)	256MB	5	100BaseTX
Mid-Level Scale <= 500 Users, <= 60 Concurrent (Scenario 1)	133MHz	1	1–2 (256KB Level 2 Cache)	512MB	11	100BaseTX Full Duplex
Enterprise Scale <= 1,000 Users, <= 110 Concurrent	100MHz	2	2–4 (512HB Level 2 Cache)	1–2GB	20	1–2 100BaseTX Full Duplex NICs (channels) teamed/ trunked together or Gigabit Ethernet
Extreme Scale <= 2,000 Users, <= 220 Concurrent	100MHz	2	4–8 (1–2MB Level 2 Cache)	2GB	40	2–3 100BaseTX Full Duplex NICs (channels) teamed/ trunked together or Gigabit Ethernet

Note: It is important to understand that, as we scale each system, a balanced approach is taken between all server resources so that no artificial bottlenecks are created before the file server is even deployed.

Windows 2000 File Server Tuning Summary

File Server–Specific Application Tuning

From an application perspective, the primary control of how well Windows 2000 file server services run is based on:

- Server hardware configuration optimized for your environment's workloads
- Windows 2000 memory management configuration, as it affects the behavior of the file system cache which file services relies on heavily
- Windows 2000 I/O subsystem optimization for delivery of file services

File server functions are handled natively to Windows 2000. Because of this, all tuning recommendations provided below are for general Windows 2000 file server tuning. As always, follow the core tuning methodology (chapter 2) when tuning your servers. However, the tuning recommendations provide a solid baseline to start with before your stress tests are run, baselines are developed, and your solution is deployed.

For the hands-on specifics of why, when, and how these tuning recommendations work and steps to implement them, refer back to the tuning sections in the previous chapters. Here, select information is provided to give you a jump-start into an optimized Windows 2000 solution.

SERVER HARDWARE TUNING

The server configuration section above reviewed specific recommendations for the file server configuration. However, every server and workstation implement the architecture and BIOS settings slightly differently; thus this section is more of a general reminder of what you should check at the hardware level. Refer to earlier chapters for specific information on each subsystem area that requires close attention, such as placement of I/O adapters, location of adapters on the PCI bus, and RAM configuration. Also, remember to update the BIOS and Windows 2000 drivers on all server hardware, which includes motherboards and the associated adapters. Once the BIOS levels and drivers are updated, go into their perspective setup and ensure that they are set correctly. For example, is the CPU set to run at its maximum speed? Are you sure

you received the correct hardware? Did you get the 900MHz instead of the 1GHz CPU? Are all disk drives the 15,000rpm version you ordered instead of 7,200? I've observed many instances when a Windows 2000 system did not meet expectations because a vendor sent the wrong hardware, the BIOS setting was incorrect, or a driver was not up-to-date.

WINDOWS 2000 FILE SERVER SPECIFIC APPLICATION TUNING

- Check the Windows 2000 memory management strategy. File servers rely heavily on the file system cache. By default, the Windows 2000 file system cache is set to maximize data throughput for file sharing. However, check this setting anyway at Start / Programs / Settings / Network and Dial Up Connections / Properties of your LAN adapter / File and Print Sharing for Microsoft Networks.
- Set application response to favor background processes (Start / Settings / Control Panel / System / Advanced / Performance Options / Set Background Services).
- Set the pagefile size to twice the amount of RAM in your server. Use a fixed size.
- Never place the pagefile on a RAID 5 array.
- If your system must page (which should be avoided), consider multiple pagefiles on physically separate disk drives, which are all configured to the exact same size and are on the same speed disks.

WINDOWS 2000 CPU SUBSYSTEM RESOURCE TUNING

Key CPU-related tuning you should consider:

- Off-load CPU operations to I2O enabled I/O adapters.
- Do not implement compression or encryption.
- If encryption is in use, additional resources (CPU, memory, disk) will need to be added depending upon workload patterns.
- If multiple CPUs are in use with Multiple NICs, consider using the Microsoft Interrupt-Affinity Filter Control Tool (IntFiltr; see chapter 4) to ensure interrupts are properly distributed among NICs and CPUs. (For step-by-step details on this, review chapter 7.)
- Do not use the server as your personal workstation (bad for your customers' response time).
- Do not use a 3GL or other fancy screen saver.
- Check all services running on your server. Operate only those you need and stop all others.
- Avoid running other applications on your file server.
- Run only those file services needed to provide file server services and manage the server.

WINDOWS 2000 MEMORY SUBSYSTEM RESOURCE TUNING

Key memory-related tuning you should consider:

- Check all services running on your server. Only operate those you need and stop all others.
- Force Windows 2000 to keep the Windows 2000 executive (kernel) from paging any of its executive system drivers to disk and ensure that they are available immediately if needed.
- Do not use a 3GL or other fancy screen saver.
- Do not use exotic wallpaper; it wastes memory resources.
- Ensure the pagefile is set to twice the size of physical RAM in your server to ensure physical memory can be committed to action.

WINDOWS 2000 DISK SUBSYSTEM RESOURCE TUNING

Key disk subsystem–related tuning you should consider:

- Defragment your disk subsystem on a regular basis; this will result in significant performance improvements!
- Perform a fresh, low-level format on all disk drives at the lowest level using the disk or RAID adapter in your system.
- Freshly format all disk drives with NTFS when installing Windows 2000.
- When formatting the disk drive with NTFS, select the ALU size that best matches your disk workload characteristics. For scenario 1, the recommended ALU is 64KB, as the workload is based on large file size. (Note: You must use a third-party defragmentation tool if the ALU is larger than 8KB.)
- If large NTFS partitions or volumes are in use, increase the size of the volume logs.
- Turn on write-back caching and read-ahead caching on hardware RAID adapters.
- Use only I2O enabled RAID adapters and those with their own CPUs.
- Keep storage capacity on the disk subsystem under 60–80% so that the fastest portion of the disk drive is used and so that there is room on the disk drives to perform defragmentation.
- Evenly distribute disk workload across the physical disk drives in the server.
- Defragment your disk subsystem on a regular basis; this will result in significant performance improvements!
- If DOS file names are not required (i.e., legacy DOS-based desktops are not in use), disable short name (8.3) file generation. (Note: To disable short name generation, use regedit32.exe to set the registry DWORD value of 1 in the following Registry location: `HKEY_LOCAL_MACHINE\SYSTEM\CurentControlSet\Control\Filesystem\`

`NtfsDisable8dot3nameCreation`. For more information on this technique, see chapter 6.

- Disable last access updates. `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\FileSystemNtfsDisableLastAccessUpdate`. Changing the default REG_DWORD value of this key from 0 to 1 will stop Windows 2000 from updating the last access time/date stamp on directories as directory trees are traversed.
- Understand the characteristics of all RAID levels and leverage different RAID levels as needed in the same file server solution.
- Use one Windows 2000 partition per physical disk device/array.
- Check all services running on your server. Operate only those you need and stop all others.
- Did I mention to defragment your disk subsystem on a regular basis; this will result in significant performance improvements!

WINDOWS 2000 NETWORK SUBSYSTEM RESOURCE TUNING

Key network subsystem related tuning you should consider:

- Teaming/trunking multiple NICs (Ethernet channels) together as needed to add network bandwidth to your Windows 2000 server seamlessly to your customers.
- Balance network activity across NICs if you cannot use NIC teaming/trunking technologies.
- If possible, use jumbo packets on your network.
- Remove all unnecessary protocols and Redirectors.
- Standardize on one network protocol: TCP/IP.
- If more than one network protocol is in use, set binding order such that the most commonly used protocol is first on the list.
- Tune the NIC driver to off-load IPSEC security operations and TCP operations on the NIC itself instead of the main system CPUs.
- Tune the NIC driver to increase its send and receive buffers.
- Use only I2O-enabled NICs.
- Use the fastest I/O bus enabled NIC possible (64-bit, 66MHz Peer PCI buses).
- Tune the Windows TCP/IP stack to your environment (TCP Hash Table, MTU Window Size). (Review the IIS 5.0 Solution Scenario for details on how to accomplish this.)
- Do not use autodetect for your NIC. Set it to the best performance level the internetwork devices will support. For example, 100BaseTX full duplex.

THIRD-PARTY TOOLS FOR TUNING WINDOWS 2000 FILE SERVERS

Key network subsystem–related tuning you should consider:

- Obtain a third-party disk defragmentation tool that provides enhanced defragmentation features such as scheduling and support of multiple NTFS ALU sizes. For example, Norton Speed Disk http://www.Norton.com.

Solution Scenario 3: Windows 2000 Exchange Servers

Introduction

Messaging servers have become the heart of many traditional and eBusinesses alike. As one of today's most commonly used applications, many have come to rely on it like they do the telephone. People expect messaging systems to always be available and to run fast.

Application Description

Microsoft Exchange is a scalable and robust messaging platform that has rapidly matured over the past several years. From a performance perspective, instead of thinking of this application as a mail and collaborative computing platform, consider it an advanced database system with a messaging application residing on top of it.

Application Performance Characterization

When we understand how an application operates, we can then correlate its operations with the performance concepts we have learned from the earlier chapters. The whole world is not quite "point and click" yet. There are several performance areas of an Exchange platform to consider at the most basic level. First, Exchange provides directory structure and directory information; e.g., e-mail address information for the people in your Exchange sites. This involves looking directory information up in Exchange itself (if in a Windows NT environment) or looking up addresses via Active Directory if you are in a Windows 2000 environment. Directory information is stored in the directory services (DS) database, which typically exhibits random performance characteristics.

Two more basic functions that Exchange provides are storage and delivery of e-mail. When e-mail arrives at the Exchange server, it is placed into

buffers in memory. Then it is placed into the transaction log (EDB**.**log), committed to the exchange message store by placing it into the Private Information Store's buffers, and finally into the Private Information Store's database. These steps are very important to remember, as they illustrate the importance of tuning and sizing memory and disk resources. We will need to configure enough memory to support all of the caching Exchange will need, a fast disk subsystem optimized for the sequential and write-intensive log files, and one to support the information store which is very random in nature and supports a good split of read and write operations. If you are using public folders (Public Information Store), consider it another database to manage with workload characteristics similar to that of the private information store.

It is highly recommended that the Exchange log files and information store databases be kept on different disk subsystems for improved performance and reliability.

Of course, you will want a fast, low-latency network for your customers to access the Exchange server and enough CPU horsepower to drive the overall solution.

The final performance consideration is the transfer of the e-mail messages either internally, for addresses inside the Exchange site, or externally, to the Internet via the Exchange Internet Mail connector. These messages are first placed into the delivery or send transactions logs to track their status.

Table 8.14 shows a summary of Exchange's key components.

TABLE 8.6 *Exchange Database Component Descriptions*

Exchange Database Components	Exchange Database Component Descriptions and Characterizations
Private Information Store	This component is a database, which maintains all messages in users' mailboxes, enforces storage limits, and delivers messages to users on the same server. Disk workload is characterized as random in nature.
Public Information Store	This component is a database that maintains information stored in public folders and replicates public folders as required. Disk workload is characterized as random in nature.
Information Store Logs	This component records all information store transactions. Disk workload is characterized as sequential and write intensive in nature.
Directory Service Logs	This component records all directory store transactions. Disk workload is characterized as sequential and write intensive in nature.
Directory Service	This component maintains information about an organization's recipients, servers, and messaging infrastructure. It is an implementation of the directory services standard that is roughly defined by the ISO X.500 specification except the entire directory is automatically replicated to all servers in the organization instead of portions being distributed among the servers as in X.500. The Directory Service is used by the other three core components to map addresses and route messages. Disk workload is characterized as random in nature.

TABLE 8.6 *Exchange Database Component Descriptions (Continued)*

Exchange Database Components	Exchange Database Component Descriptions and Characterizations
Message Transfer Agent	This component is responsible for moving messages between Information Stores, connectors, and third-party gateways. It is the service that performs the actual routing, submission and delivery of messages. Disk workload characterized as random in nature.

Sysmon Objects and Counters: Exchange Server Health, Bottleneck Detection, Tuning, and Sizing

The Sysmon objects and counters outlined in the previous chapters for CPU, memory, disk, and network subsystems are still applicable to Windows 2000 Exchange servers as it is important to understand the big picture when tuning or sizing messaging solutions. If one server resource becomes a bottleneck (i.e., queues begin to form in the CPU or disk subsystem), response time to the end user begins to suffer and message queues in the Exchange Server application begin to grow.

In addition to the previously reviewed rules of thumb for bottleneck and sizing objects and counters, it is recommended that you become familiar with the following Sysmon objects and counters that are added to Sysmon when Exchange is installed. These objects and counters—shown in Table 8.15—are helpful in determining the health of your Exchange server application and for tuning and sizing.

TABLE 8.7 *Sleuthing Out Microsoft Exchange Specific Health, Tuning, and Sizing Information*

Object: Counter	Definition	Value to Monitor
MSExchangeMTA: Work Queue Length	How many items are waiting for processing at the message transfer agent (MTA).	Health indicator and bottleneck detection The work queue length should be less than 1% of the users the Exchange server supports. It should also not grow over time. If it is > 1%, or growing for a significant amount of time, either a bottleneck is forming or another subsystem or external system is running slow or is not available.
MSExchange IS Private: Send Queue Size (also public if used)	The send queue size provides insight into how many requests to send messages are currently queued for delivery.	Health indicator and bottleneck detection The average send queue length should be less than 1% of the users the Exchange server supports. It should also not grow over time. If it is > 1%, or growing for a significant amount of time, either a bottleneck is forming or another subsystem is running slow. In the most optimally configured servers, this value is close to zero even under heavy loads!

TABLE 8.7 *Sleuthing Out Microsoft Exchange Specific Health, Tuning, and Sizing Information (Continued)*

Object: Counter	Definition	Value to Monitor
MSExchange IS Private:Average Time for Delivery	How long it takes for the IS to deliver a message.	Health indicator and bottleneck detection Baseline this value for your environment to determine what is acceptable. Once you baseline this value, if it begins to consistently grow to higher levels, a bottleneck is forming. Note, this counter collects information in the format of tens of milliseconds; hence a counter of 100 corresponds to 1 second.
MSExchange: IMC Internet Mail Connector in queues outbound	The send queue size provides insight into how many requests to send messages are currently queued.	Health indicator and bottleneck detection The average send queue length should be less than 1% of the users the Exchange server supports. It also should not grow over time. If it does, numerous problems could exist such as Domain Name Service (DNS) problems, issues with your Internet network connection, or a remote mail system that is down.
MSExchange Database Information Store (IS) and Directory Store (DS): Cache % Hit & Database Session hit ratio	This counter displays the percentage of message information that was found in the IS cache in memory that fulfilled the request vs. going to IS database on the physical disk subsystem.	Influences tuning and sizing The higher these values are, the better. The more you can cache in memory, the faster your Exchange server will be while lowering the workload on the disk subsystem housing the Information Store and Directory Store (DS). If you do not have a high hit rate (>90%), consider adding additional RAM and rerunning the Exchange Optimizer to improve the situation.
MSExchangeIS: Active User Count	How many users have done something in the last 10 minutes.	Influences sizing and tuning This counter comes closest to indicating the concurrent user rate on your Exchange server.
MSExchangeIS: User Count	How many users have logged on since the service has been restarted.	Influences sizing and tuning This counter takes some close tracking. Since it is a cumulative number, if your goal is to review usage over a week, the IS must be stopped and restarted at the start time (not realistic) or you must write down the start value and review it later in the week.
MSExchange IS Private: Messages Submitted/min and Messages Recipients/min	How many messages per minute are being sent and received.	Influences sizing and tuning This information is a good indicator of how much workload your Exchange server is experiencing.This can be helpful information for load balancing customers across multiple exchange servers in your site.

Microsoft provides a number of statistics through Sysmon for monitoring the performance of Exchange servers, but Sysmon counters do not provide every aspect of detailed instrumentation data. Critical data are also available in the Exchange Tracking Logs and the Windows 2000 Event Logs.

Scenario 3 Step by Step: Sizing a Mid-Range (3,000 User) Windows 2000 Exchange Server

Sizing the Initial Exchange Server Configuration

In chapter 3, the core sizing methodology was presented for sizing your Windows 2000 solution. That methodology is followed here for the development of the initial Windows 2000 Exchange server configuration.

1. DEFINE OBJECTIVE(S)

Ajax Rockets has decided to collapse the two different mail systems they have into one unified messaging system based on Microsoft Exchange. Next to payroll and their customer engineering application, e-mail is the most important application that they run and must be always available and very responsive.

2. UNDERSTAND BUSINESS AND TECHNICAL REQUIREMENTS NEEDED TO MEET YOUR OBJECTIVE(S)

- Which application(s) will meet your functional objective(s)?
  Microsoft Exchange 5.5 will be the messaging platform. However, it is also important to consider that Exchange 2000 will be considered once Windows 2000 Active Directory is in place.
- What are the availability requirements?
  These are critical servers, as they will support the entire company. The goal is to have a highly available solution. This is a high-profile, business-critical application. Backups must be completed nightly (after 9:00 P.M. and before 5:00 A.M., about eight hours) and the system must be restorable to service within four hours.
- What are the ramifications if your system is down?
  Similar to a denial of service attack, downtime will slow down many work projects and it is considered unacceptable.
- What are the disk subsystem storage capacity requirements?
  Currently there are 3,000 e-mail customers at Ajax Rockets who use approximately 20GB of storage in their mail server. This works out to approximately 7MB per user. The current plan is to have each customer keep all mail local on his or her desktop/laptop and use the mail server as a post office using Microsoft Outlook clients (with the

proper security patches installed). This approach will lower the size of the Information Store and consequently lower disk storage requirements. However, a 7MB mailbox on the server could be used up in just a few large attachments. 16GB used to be the largest Information Store Exchange could handle, but with better server technology, an updated Exchange 5.5, service packs, and better procedures, larger sized Information Stores are realistic. Ajax Rocket's goal is to provide each customer with 50MB of storage, which results in 150GB of required overall storage. Some customers will require much less than this, while others will require much more. Overall, this is a good starting point to meet their business needs.

Looking at the big picture, all disk storage requirements are outlined in Table 8.16.

TABLE 8.8 *Exchange Server Disk Storage Sizing Matrix*

Application	Disk Storage Capacity Needed
Windows 2000 operation system requirement	<500Mbytes
page file	2GB
Exchange server application	<600Mbytes
System management agent	5Mbytes
Antivirus application	20Mbytes
Security application (server-based firewall and IDS)	30Mbytes
Information store (private)	150GB
Information store logs	3GB (1Mbyte per user × 3000 users)
Information store (public)	Not used in this solution
Message transfer agent and directory services (DS) databases	2.4GB (0.8MB per user × 3,000 users)
Directory services (DS) logs	6GB (2MB per user × 3000 users)

3. DETERMINE LOADING CHARACTERISTICS

- *Number of total users per server*
  There will be 3,000 total customers. Due to availability concerns surrounding this project, having all of your eggs (e-mail customers) in one basket is not desirable. The 3,000 customers will be split across two servers, which results in 1,500 users per server. Now, in the

highly unlikely event one server were to fail, we only have half of the customers yelling and screaming!

- *Number of concurrent users per server*
  We expect all customers to use these servers each day, but from historical information we expect a 20% (on Friday night) to 100% (on Monday morning) concurrency rate, depending on the day of the week and if a large project is due.
- *Disk workload characteristics (read/write environment)*
  Understanding disk workload characteristics is paramount in an Exchange server environment and reviewed in detail in the above Application Performance Characterization section.
- *Size of records/transactions*
  Mail messages are expected to average approximately 2KB when no attachments are used and 400KB when attachments are used. We will investigate this closer during the stress test.
- *Physical location of users (Influences communications such as WAN, LAN, dial-up, terminal services, etc.)*
  All servers reside on site and are connected via the local switched Ethernet 100BaseTX LAN.
- *Type of work the user will perform*
  E-mail system operations, of course! Office automation and rocket scientist activities.

4. DETERMINE PERFORMANCE REQUIREMENTS

From a performance perspective, response time to the customer is the most important factor for a messaging solution. Ajax Rocket's business goal is to always have subsecond response time when interacting with the mail system. The second most important performance requirement is backup operations during each evening and being able to restore service within four hours. Of particular importance is the Information Store that houses recently received messages. Based on current business requirements, a worst-case scenario is that 75GB (per server) must be backed up in eight hours and restored in four hours. For backups, this translates to 9GB per hour for backups and 18GB per hour for restores. No problem, now that we mastered backup solutions from the previous scenario! Of course, this assumes that Exchange is offline during the restoration process. If Exchange is kept online, it generally takes twice as long to complete a restore.

5. UNDERSTAND FUTURE BUSINESS REQUIREMENTS

As this is a public company, growth is very important and the company does expect to expand by at least 30% each year, which is higher than the original 20% projection. This indicates that, in year two, there will be 3,900 customers to support and in year four there will be 6,240.

6. UNDERSTAND FUTURE SYSTEM ARCHITECTURES

The local team would like to put a system in quickly and would like it to be run without replacement for three years. Local support staff is still small, which is just unheard of in the IT industry. Additionally, Microsoft will continue to release new versions of Exchange and mail attachments are expected to continue to grow in size. Because of these facts, a truly leading-edge solution will need to be deployed.

7. CONFIGURE THE SYSTEM

Finally, to the good stuff. At this point, you can configure the Windows 2000 Exchange server. Choose the system resources required to meet your objective, with consideration for steps 1 through 6, and obtain the system. As you configure the system, look at each individual resource to meet the overall system configuration. Software and hardware vendors are good sources of information. They can suggest configurations that will get you started. Reviewing relevant industry benchmarks also can help provide insight into the initial configuration. Industry benchmarks are explored in depth in chapter 3. The following steps are presented to help guide you through the actual system configuration process.

Having some sort of idea of how each system resource works and relates to the overall system performance is helpful when configuring your system. Chapters 4 through 7 review the overall system architecture and each system resource in greater detail.

7.1 Application Characteristics Review • The general Exchange application characteristic description and backup technology review was completed at the top of the Windows 2000 Exchange Server scenario section.

7.2 Review Each System Resource for the Windows 2000 Exchange Server

Disk I/O Requirements • The disk subsystem drives the sizing of the rest of the Exchange solution. Why? The restore-to-service and response time requirements, and the database characteristics of Exchange, require a very well planned out disk subsystem. Sizing and tuning the disk subsystem will result in a highly available and high-performing solution. From the business requirements above, we will determine the disk storage and performance requirements.

Availability • Since this is a critical server, all data will reside on hot swappable RAID array–based disk subsystems that utilize some level of fault tolerance. Hot spare disk drives will also be incorporated into the solution to increase overall availability.

The database characteristics of Exchange require the physical separation of the log files and the databases they support for maximum performance and availability.

This is a critical requirement. If the database and log files were on the same RAID array and the array failed, you would lose all data entered since the last backup. If the databases (Information Store) and the logs (Information Store logs) are on separate RAID arrays, you can recover! If the IS database fails, you can restore the database from the backup system and then replay the logs (which would not have been affected since they were on another physical RAID set), and all transactions up to the point of the failure can be recovered. Any new messages that were sent to the Exchange server while it was temporarily off line would be queued at the sender's mail systems (until the retry intervals expire) and would be resent when your Exchange server was back on line.

Performance Workload Characteristics • From the workload characteristics of a general Exchange server above, we will use multiple fault-tolerant RAID levels to meet the performance requirements.

Performance: Throughput and Transfers/sec • In chapter 6 we saw that it is desirable to group applications with similar disk workload characteristics together on the same physical disk subsystem for maximum performance. Taking this approach with our current Exchange Server environment, each of the major databases (random read/write) and logs (sequential write) will be grouped together on the same physical disk subsystem. This also aids in sizing the overall disk subsystem.

In this environment we have two performance requirements: response time to the end user and throughput needed for backup operations. The largest portion of the disk subsystem requirement centers on the Information Store, so to begin we will focus our efforts there. Then we'll look at the Information Store logs. If the transactions/sec occurring on the disk subsystem become too slow or a disk queue begins to form, it will affect end user response time. To determine the transfers/sec required to support subsecond response time for a user, we can review industry standard benchmarks, sizing tools from vendors (Compaq & Dell provide these on their web sites), and historical information.

The most common industry benchmark for Exchange environments is LoadSim, which is a free benchmarking tool from Microsoft. LoadSim's operations were reviewed in detail in chapter 2. Recall that LoadSim can be installed on several clients (server class systems, really) to emulate an e-mail-based workload directed at a target Exchange server under test. Based on industry benchmark information collected from several vendor web sites (Microsoft, Compaq, Dell, etc.), my own tests, and those completed by my associates Jeff Lane, John Polen, and Dan Matlick, we determined that the following extrapolation characterizes the workload an Exchange Server would need to support to obtain a subsecond response time: 0.2 to 0.4 transactions/sec per concurrent user. Thus for our environment, 1500 users would require an Information Store that supported between 300 and 600 transactions/sec to meet our response-

time requirements. Because the Information Store must be placed on a fault-tolerant array, a RAID 5 array will be used. If we assume a cache hit ratio of 65%, the transactions/sec that need to be supported range from 75 to 150. From chapter 6 (Table 6.5), we found that on average each disk in a RAID 5 array experiencing a random read/write workload could support approximately 100 transfers/sec. Completing the math (150 transfers/sec / 100 transfers/sec/disk), we will need at least 2 disks to support the Information Store. Because we will store this data on a RAID 5 volume, however, we will need to calculate the write performance overhead incurred with RAID 5 arrays. From historical information at Ajax Rockets, we determine that there is a 60% read–40% write split of transactions. So the transfers/sec that must be supported are

[150 disk read transfers/sec × 0.6] + [4 × (150 disk write transfers/sec × 0.4) × 1.25 utilization factor]/100 transfers/sec/disk]

which works out to approximately 4.2 disk drives, which you always round up to the next nearest integer: 5. Now the utilization factor of 1.25, or 125%, is added in to ensure disk operations can be sustained at an 80% utilization rate. Each disk drive can support a random read/write workload of 100 transfers/sec, but the best response time is provided if the disk drives run at 80% of their maximum workload rate. For more detailed information on these disk concepts, review chapter 6.

Based on current business requirements worst-case scenarios, 75GB must be backed up in eight hours and restored in four hours. For backups, this translates to 9GB per hour and 18GB per hour for restores! Again referring to chapter 6 (Table 6.9), each disk in a RAID 5 array experiencing a random read workload can support 1.11MB/sec of throughput and 0.95MB/sec for sequential write environments. To meet the backup requirements, 3 disk drives are needed to achieve 9GB/hour (2.5MB/sec / 1.11MB/sec/drive = 2.25 drives, rounded up to 3 drives). To meet the restore requirements, 6 disk drives are needed to provide 18GB/hour of restoration performance (5MB/sec / 0.95MB/sec/drive = 5.2 drives rounded up to 6 drives). Here we used random disk workload characterizations for backups (Exchange is still online) and sequential write workload characterizations since restores are complete on an idle system (Exchange is offline).

To meet our response time and throughput performance requirements, we take the greater number of disk drives that were calculated in the above two steps. Thus, 6 disk drives are required to meet our projected performance requirements for the Exchange Information Store. For more detail on overall backup performance, review the backup server scenario above.

The Information Store logs play a very important part in overall Exchange Server availability and performance. From these same extrapolations, we find that to support subsecond response times, the Information Store logs will need to support 0.03 to 0.05 transactions/sec per concurrent

user. Thus for our environment, 1500 users would require an Information Store log that supported between 45 and 75 transactions/sec to meet our response time requirements. If we assume a 50% cache hit ratio for the Information Store logs, this transaction rate lowers to 23 to 38 transactions/sec. Since we will want a fault-tolerant, low-latency environment for this sequential write-intensive environment, a RAID 1 or RAID 10 array would best meet our needs. A RAID 1 array would be desirable due to cost considerations. In chapter 6 we determined that in a sequential write-intensive workload, a single disk could support between 150 and 200 transfers/sec. Being conservative (all those safety features on Rockets!), Ajax Rockets prefers to us the 150 transfers/sec estimate for single-disk write-intensive sequential workloads (Table 6.9). However, since we are using a RAID 1 configuration, the actual number of I/Os we will need to support is

[0 disk reads/sec + (2 × 38 disk writes/sec) × 1.25 utilization factor] / 150 transfers/disk

which works out to approximately 0.6 disks. We round up to 1, but obviously 2 disks are required in a RAID 1 array. Thus a 2-disk RAID 1 array per database log will meet our projected needs. In future, if this workload begins to increase, we will consider using either a RAID 10 array or multiple transaction logs for each database log set.

Also note that we assume a high cache hit rate on the Information Store, which is the result of a well-tuned Exchange application and memory subsystem. A high Information Store cache hit ratio lowers the workload on the disk subsystem.

For the other Exchange databases and logs, we will determine the storage capacity requirements and follow the tuning recommendations above.

Storage Capacity • Even when focusing on disk storage capacity, continue to think performance. For Exchange Servers, it is not desirable to have the disk drive used storage capacity exceed 50–60%. There are several reasons for this. The first reason revolves around maintenance and recovery. If one of the major Exchange databases—such as the Information Store—becomes corrupt, a significant amount of free disk space is needed to run the recovery tools. The second reason revolves around performance. The fastest portion of the disk subsystem is the outer edge of the physical disk. If you keep the disk subsystem at less than 60% filled, it will just run faster! Disk defragmentation tools typically require at least 20% disk free space to operate, and 30%–40% to run optimally. The final reason involves the advent of self-replicating viruses that can quickly fill an Information Store; having additional disk space provides an increased buffer that gives the administrator more time to react before a store or log fills and the Exchange server becomes unstable.

Based on all of the above disk information, Table 8.17 outlines the overall disk subsystem requirements.

TABLE 8.9 *Exchange Server Disk Subsystem Solution (Server 1 of 2)*

Application	Minimum Disk Storage Needed	Logical Partition	Recommended Configuration (All 10,000rpm, Ultra2 SCSI Disk Drives)
Windows 2000 operating system requirement	<500Mbytes	C:	2 9GB disk drives in RAID 1 array (9GB usable/3.15GB filled)
Pagefile	2GB	C:	
Exchange server application	<600MB	C:	
System management agent	5MB	C:	
Antivirus application	20MB	C:	
Security application (server-based firewall and IDS)	30MB	C:	
Information Store (private)	75GB	H:	8 18GB disk drives in a RAID 5 Array (126GB usable/75 GB filled)
Information Store logs and DSA logs	3GB (1MB per user × 3,000 users) DSA—6GB (2MB per user × 3,000 users)	I:	2 18GB disk drives in RAID 1 array (18GB usable/9GB filled)
Information Store (public)	Not Used	N/A	N/A
MTA database and DS directory	MTA—2.4GB (0.8MB per user × 3,000 users) DS—2.4GB (0.8MB per user × 3,000 users)	J:	3 9GB disk drives in RAID 5 array (18GB usable/4.8GB filled)

CPU Requirements • The primary application to support from a CPU perspective is Exchange server application operations and the antivirus application. CPU workload involves supporting the messaging services, underlying message databases (stores), driving the disk subsystem, and network operations. Antivirus workloads can be significant, as we will learn later during the stress tests.

Searching for an industry standard benchmark for similar Exchange environments resulted in locating a published LoadSim Benchmark from Compaq. In this benchmark "Microsoft Exchange Server 5.5 on Compaq 1850R" found on the Compaq web site (http://www.compaq.com), a Compaq 1850R server supported 8,000 simultaneous medium workload LoadSim users with subsecond response times. The Compaq 1850R server was configured with Windows NT 4 (sorry, not enough Windows 2000 results in yet, but still relevant), two Pentium III 400MHz/512KB cache CPUs, 1GB of memory, two disk drives in a RAID 1 mirror for the logs, a single disk drive for the page file, a single disk drive for the operating system, and a 10-disk RAID 0 array for the Information Store. On the positive side, even while supporting

this workload, 25% of each CPU was still available. This would indicate that two 400MHz Pentium III CPUs would meet our CPU requirements. Note that this older Pentium III CPU technology did not use full-speed Level 2 cache even though it was 512KB in size. Newer, full-speed Level 2 (256KB) CPU cache—although smaller—provides equivalent and sometimes better performance in one- to two-CPU servers. Review chapter 4 for detailed information on CPU technology comparisons.

On the negative side, the Compaq 1850R was a benchmark special; that is, it was configured for speed only, not operations. Using a non–fault tolerant RAID 0 disk subsystem is not relevant to a production environment. It does provide insight into how many CPU cycles are needed to drive this size of disk array, but it does not provide enough performance workload insight for our Exchange server solution. Based on this, we will be completing a stress test of our total solution.

The sizing estimate in Table 8.18 does not provide much room for growth or surges in workload. Fortunately, our initial solution sizing recommendation will be to operate with dual 733MHz CPUs and faster system bus (133MHz vs. 100MHz). From a CPU-only perspective, a 733MHz CPU provides approximately 37% more performance per CPU than the older 400MHz CPU (see chapter 4).

TABLE 8.10 *Initial Exchange Server CPU Sizing*

Application	CPU Usage (Estimate) Based on Two Pentium III 400MHz CPUs
Windows 2000 and Exchange, Network operations	70% usage per CPU, CPU Queue of 2 (From Compaq LoadSim Benchmark)
Overhead for system management software running on server	1% (estimated)
Overhead for intrusion detection software	1% (estimated)
Overhead for antivirus software	10% (estimated)
Total Estimated CPU Usage	***82% per CPU***

Memory Requirements • Sizing RAM is an additive process. Our first step in sizing memory resources is to outline all applications that will use any memory resources. Microsoft Exchange manages its own cache and memory resources and does not rely heavily on the Windows 2000 dynamic file system cache. The amount of memory that Microsoft Exchange uses for caching operations is tunable using the Exchange Performance Optimizer application. It would be wonderful if all applications provided the ability to control their memory usage. The Compaq 1850R LoadSim results indicate that 1GB of RAM would meet the memory requirements for our solution. Table 8.19 is a breakdown of where the memory resources are being focused.

TABLE 8.11 *Exchange Server Memory Sizing Matrix*

Application	Memory Usage (Working Set)
Windows 2000 operation system requirement	128MB
Windows 2000 File System Cache	64MB
Overhead for system management software running on server	8MB
Overhead for intrusion detection software	12MB
Backup server application	10MB
Exchange server application and cache	802MB
Total Estimated Memory Usage	***1024MB***

Through proactive performance management, more RAM can be added as needed. The Exchange environment is one area where the old adage—more memory is better—is very true (true, that is, when the memory is tuned; see below). We will gain more insight once the stress test is run, but this estimate provides us a good starting point.

Network I/O Requirements • There are two network I/O requirements to consider—customer access to the services that Microsoft Exchange server provides and the bandwidth needed for backups and restores. The network (LAN) rarely hinders typical Exchange server environments, so for this configuration we will assume that Ajax Rocket's standard network policy of using 100BaseTX NICs in a switched network setting should suffice. Typically, it will be the backup and restore operations that drive the network requirements for this environment.

During the most intense portion of the backup process, the backup server will need to support 18.75GB/hour (75GB/4 hours to back up each server) of sustained throughput. A 100baseTX network can support 45GB/hour, so a single 100BaseTX NIC will provide the needed bandwidth. This takes care of the network connection into the Exchange server, but what about the network paths between the target Exchange servers that will be backed up and the backup server? As always, we will check the network data path for potential bottlenecks—follow the data! Refer to the backup server scenario above and chapter 7 for more discussion on that investigation.

7.3 System Architecture Relationship Check • This is a final, common-sense check of the Windows 2000 Exchange server configuration. Since the business requirements call for keeping this new Exchange server for at least three years, leading-edge server technology will be obtained. Not cutting edge, as that technology tends to demand a premium price.

From all of the sizing information above, we now have the following configuration: a one- to two-CPU capable server, two Pentium III 733MHz CPUs with 256KB each of full-speed (ATC) Level 2 cache, a 133MHz system

bus, 1024MB of RAM, a single 64-bit 33MHz PCI I/O bus (266MB/sec), one fibre channel adapter (100MB/sec), one 12-disk fibre channel RAID array system, one internal hardware–based RAID controller, two disk drives for the operating system, two disk drives for the log files, eight disk drives for the Information Store, and two 100BaseTX Ethernet NICs. This server configuration is balanced and there is nothing notable in the way of data path bottlenecks. One side note, notice that fibre channel is used versus SCSI for increased scalability and improved cable management.

There is one potential concern. First, only a single, fast Ethernet connection is planned for. From a network availability perspective, some of the folks at Ajax Rockets have encountered failed NIC cards in the past. With the mail server being such a critical item, they have elected to obtain two NICs and use adaptive fault tolerance. This is surprisingly easy to configure on NICs that support this feature and is found under the NICs' advanced tab.

7.4 Future Business and System Architectures Check Review • This server has room to grow in the disk and network areas but is limited in the CPU area. If the workload stays the same but more disk storage capacity is needed, that can easily be added to the external arrays. From a performance perspective, if the workload increases, some additions can be made but eventually another two-CPU class server would be required. We could go with a four-CPU class server, but this would significantly increase the cost of the current solution.

7.5 Initial Hardware Configuration and Tuning • Table 8.20 shows the initial configuration of the server hardware, based on steps 7.1 and 7.2. See the Sizing and Tuning Exchange Servers section below.

Note: The Public Information Store is not used for this solution. If it were actively used, it would not be recommended to be located on the same disk array as the Private Information Store.

TABLE 8.12 *Exchange Server Hardware Configuration Details*

Windows 2000 Exchange Server Hardware Configuration Server Selected:
Compaq ML-380 (Why select the ML-380? We needed a solid system to use for a real-world reference.)

CPU Configuration	*CPUs Used in Solution*	*CPU Family*	*Level 2 Cache Size*	*Speed of CPU*	*Available CPU Slots*
	2	Intel Pentium III	256KB	733MHz	2 of 2 Used

RAM Configuration	*Amount of RAM Used in Solution*	*System Bus Speed*	*Type of RAM*	*RAM Expansion*
	1024MB	133MHz	ECC SDRAM	4GB

TABLE 8.12 *Exchange Server Hardware Configuration Details (Continued)*

Windows 2000 Exchange Server Hardware Configuration Server Selected:
Compaq ML-380 (Why select the ML-380? We needed a solid system to use for a real-world reference.)

I/O Subsystem Configuration	*Number and Type of I/O Card*	*Number of PCI I/O Buses*	*Placement of PCI Cards*	*Speed of PCI Bus*
	1 Compaq Fibre Channel Adapter	1	Fibre Channel Card is placed into PCI bus 0, in the top slots	One (1) 64-bit 33MHz
	2 Single Channel Full Duplex 100BaseTX		The two 100BaseTX NICs in the next two slots in PCI bus 0.	

Disk Subsystem Configuration	*Partition/ Mount Point*	*Files Located There*	*Type of RAID Adapter*	*SCSI Channel*	*RAID Level*	*# of 10K RPM Ultra2 Disk Drives*
	C:	Windows 2000 installation, Exchange Application, pagefile, System Management Software, Exchange Antivirus Software	Built-in RAID Adapter Compaq	1 of 1	1	2 9GB Disk Drives
	H:	Private Information Store	Fibre Channel Attached RAID Unit	1 of 1	5	8 18GB Disk Drives
	I:	Private Information Store Logs and DSA Logs	Fibre Channel Attached RAID Unit	1 of 1	1	2 18GB Disk Drives
	J:	MTA and Directory Services (DSA)	Internal RAID Array	1 of 1	5	3 9GB Disk Drives

7.6 INITIAL SOFTWARE CONFIGURATION AND TUNING • In this step, the initial software configuration of the server configuration begins to take shape based on step 7.1 and 7.2. See the Tuning Exchange Servers section below.

7.7 INITIAL GENERAL TUNING • See the Tuning Exchange Servers section below. These recommendations were followed before the stress test was run in step 8.

8. STRESS TEST AND VALIDATE THE SERVER CONFIGURATION

Congratulations! We now have a solid configuration that will run Microsoft Exchange and should support the required number of users and provide subsecond performance. The key word here is "should." We have a solidly sized and tuned solution, which provides a great starting point for moving into production and proactive management. But, since this is such an important

application to Ajax Rocket's business, it was relatively easy to convince management that a stress test should be run to validate the solution.

To ensure that this configuration will support our objective, we will stress test the server before it enters production. We also wanted to see how well the Exchange server would react under the heavier workloads that we expect to experience as the company eventually grows to approximately 5,000 customers, which would mean that each server will need to eventually support 2,500 customers. One last concern—created by the recent waves of viruses being distributed through e-mail—was to see how our solution would react under load while running antivirus software directly on the Exchange server. This antivirus approach was taken because we could not rely on our customers not to turn off the antivirus software at their desktops. To accomplish this stress test, we used the Microsoft stress-testing tool LoadSim.

LoadSim is a tool used for simulating a client user load on an Exchange server. Its purpose is to enable a single Windows 2000 machine—called a LoadSim client—to simulate multiple Microsoft Exchange client users from a single Windows 2000 server. The operation of the LoadSim users is governed by a LoadSim profile. This profile controls factors such as how long a LoadSim "day" is; how many e-mail messages are sent in a day's time; how many times existing e-mail is open and read; whether to use distribution lists; whether to use public folders; etc. LoadSim emulates actual Microsoft Exchange/Outlook clients. Although customer profiles are configurable, the Heavy LoadSim Canonical Profile is a very close fit to the information obtained from Ajax Rockets in historical data. Table 8.21 outlines the various LoadSim Canonical Profiles.

TABLE 8.13 *LoadSim Canonical Profile*

LoadSim User Attribute	Attribute Detail	Light	Medium	Heavy
TEST DURATION	Length of a day (hours)	8	8	8
READING MAIL	New mail (times/day)	12	12	12
	Existing mail (times/day)	5	15	20
AFTER READING MAIL	% of reply	5%	7%	15%
	% of reply All	3%	5%	7%
	% of forward	5%	7%	7%
	% of move	20%	20%	20%
	% of copy	0%	0%	0%
	% of delete	40%	40%	40%
	% of do nothing	27%	21%	11%
DISTRIBUTION LISTS	Minimum size	2	2	2
	Maximum size	20	20	20
	Average size	10	10	10

TABLE 8.13 *LoadSim Canonical Profile (Continued)*

LoadSim User Attribute	Attribute Detail	Light	Medium	Heavy
	Distribution lists per site	30	30	30
	Cover 100% of users (no overlap)	Yes	Yes	Yes
ATTACHMENTS	% to run/load mail attachment (if one exists)	25%	25%	25%
INBOX SIZE	Inbox size limit (# messages)	20	125	250
SENDING MAIL	New mail (times/day)	2	4	6
	Save a copy in Sent Mail Folder?	Yes	Yes	Yes
	Number of random recipients	3	3	3
	% of time to add a Distribution List	30%	30%	30%
	Message priority	Normal	Normal	Normal
	Delivery receipt?	No	No	No
	Read receipt?	No	No	No
NEW MAIL MESSAGE CONTENT Text-only, no attachment	1K body (ups1K.msg)	Weight 90	Weight 60	Weight 50
	2K body (ups2K.msg)	0	16	10
	4K body (ups4K.msg)	0	4	5
NEW MAIL MESSAGE CONTENT 1K mail body, with attachment	10K attachment (ups10Kat.msg)	10	5	10
	Embedded bitmap object (upsBMobj.msg)	0	2	5
	Word attachment (upsWDatt.msg)	0	2	5
	Excel attachment (upsXLatt.msg)	0	4	5
	Embedded Excel object (upsXLobj.msg)	0	2	10
SCHEDULE+ CHANGES	Changes per day	1	5	10
	Update free/busy information?	No	No	No
	Average schedule file size	22K	22K	22K
PUBLIC FOLDERS	Folder activity	None	None	None
CALCULATED DAILY LOAD (based on these defaults)	TOTAL MAIL RECEIVED PER DAY	22.94	66.30	118.89
CALCULATED DAILY LOAD (based on these defaults)	TOTAL MAIL SENT PER DAY	4.70	14.18	30.67
	Mail sent as new mail	2.00	4.00	6.00
	Mail sent as a reply	1.05	3.76	13.03
	Mail sent as a reply to all	0.60	2.67	5.82
	Mail sent as a forward	1.05	3.76	5.82
CALCULATED DAILY LOAD	AVG. # RECIPIENTS FOR EACH MESSAGE	4.88	4.68	3.88

STRESS TEST CONFIGURATION • To generate the load on the Exchange server, five client systems were configured on the same 100BaseTX switched network as the Exchange server. The stress test configuration is shown in Figure 8-3.

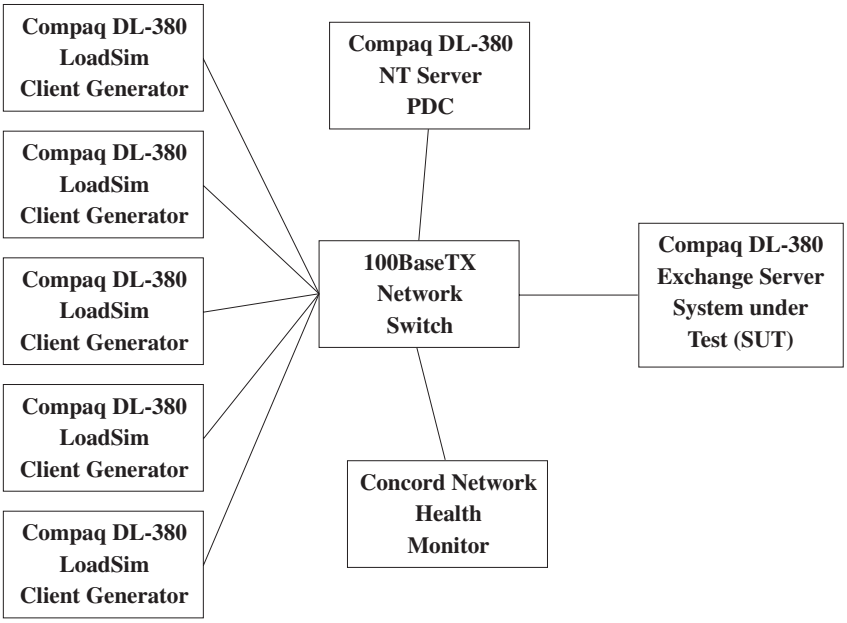

FIGURE 8–1 *Stress Test Environment*

The performance of the LoadSim client generators and Ethernet-based network was monitored at all times to ensure that they were not a source of any bottlenecks during the stress test. This step is critical during stress testing to ensure that the stress is correctly being placed on the system under test (SUT). Each client was configured with two disk drives, two Pentium III CPUs, 1024MB of RAM, and a 100BaseTX full duplex NIC. The Concord Network Health management system was used to ensure that there were not any significant numbers of network errors and that the network capacity was not exceeded.

STRESS TEST RESULTS • Once the stress test environment was in place, Exchange server configuration 1 was stress tested at the 1,000- and 2,500-user level with and without the antivirus software running (Table 8.22). Performance logging mode was enabled during all of the tests. The following are the LoadSim results. *Lower LoadSim scores indicate better Exchange server performance as perceived by the end user!*

TABLE 8.14 *LoadSim Results for Exchange Server under Test Configuration 1*

Number of Heavy LoadSim Users	Antivirus Software Running	95th Percentile Response Time Score (seconds)
1,000	No	0.115
1,000	Yes	0.138
2,500	No	0.152
2,500	Yes	0.161

The LoadSim Score represents a weighted average of the 95th percentile Exchange client response time (in milliseconds) for the various Exchange tasks defined in Table 8.21. The READ task is the task weighted highest, accounting for over half of the score.

9. PROACTIVELY FOLLOW THE CORE TUNING METHODOLOGY DURING STRESS AND VALIDATION TESTING

ANALYZING THE RESULTS: KEY OBSERVATIONS • This Exchange server under test configuration will indeed support 2,500 heavy LoadSim users while running the antivirus software, meeting Ajax Rocket's short- and long-term business objectives. We have now validated the initial configuration solution. Always thinking about a well-tuned solution, the performance logs were analyzed. The biggest surprise was the amount of processing power required. When the 2,500-user test was run without the antivirus, approximately 20% of each CPU was in use during the stress test. However, when the antivirus software was running and checking every message for viruses, CPU processing requirements jumped up to 60% per CPU! Forty percent more processing per CPU, 80% total overhead! This is quite a bit. Every antivirus package does work differently, but as we have learned from this test series, be very careful when planning new Exchange servers or adding antivirus software to current Exchange servers.

This is where proactive performance management comes into play and the importance of following the core tuning methodology (chapter 2) is stressed. If your current Exchange servers are running close to capacity, adding antivirus software and a slight increase in workload may cause harm to your Exchange servers! Always test! And stress test! Something may work functionally (installing it in a lab), but under load it is entirely different.

Reviewing the other server resources: all but 38MB of RAM was used during the test due to the way Exchange was tuned (we will be slightly more conservative before we enter production) and there was not a significant network load. The cache hit rate for the Information Store was in the range of 93–99%, which is very good and a significant factor in providing subsecond response time.

The disk subsystem performance data was interesting enough to share more detail on, and it's shown in Table 8.23.

TABLE 8.15 *Disk Subsystem Performance Data*

	OS/Page/App	DSA & MTA	Logs	Information Store (IS)
Object: LogicalDisk	**C:**	**J:**	**I:**	**H:**
% Disk Read Time	1.059	0.004	0.019	100
% Disk Write Time	32.68	0.002	7.679	100
Avg. Disk Bytes/Read	6,066.119	4,096	3,924.9	4,127.417
Avg. Disk Bytes/Transfer	6,470.954	4,096	2,178.7	4,367.2
Avg. Disk Bytes/Write	6,515.883	4,096	2,178.3	4,737.6
Avg. Disk Queue Length	0.337	0	0.077	3.292
Avg. Disk sec/Read	0.005	0.002	0.011	0.011
Avg. Disk sec/Transfer	0.016	0.003	0.001	0.014
Avg. Disk sec/Write	0.018	0.006	0.001	0.019
Disk Bytes/sec	132,528.891	85.065	157,772.469	1,003,611.125
Disk Transfers/sec	20.481	0.021	72.416	229.807

Reviewing the disk subsystem performance, we see that the workload was well spread out across the disk subsystem. Response time was good for the disk subsystem housing all subsystems, particularly the log files with an average of 0.001 seconds. We also gain some tuning insight into how to set the Allocation Unit (ALU) size of the various disk subsystems. All of the disk subsystems associated with Exchange should be set to an ALU size of 4KB (except for the Information Store, which should be set to an ALU size of 8KB), while the operating system disk subsystem should be set to 8KB for optimal performance. ALU size is set when formatting a disk drive or RAID array at the Windows 2000 disk administrator. Details on setting the ALU size are reviewed in chapter 6. This test also validates some of our sizing estimates, such as the Information Store disk subsystem configuration. Disk Transfers/sec average 229, with occasional spikes into the 400 range, which the eight-disk subsystem can support by noting the fast disk sec/Transfer response time and an average disk queue of only three (a queue greater than 16 would not have been acceptable).

10. DEPLOY THE SYSTEM INTO PRODUCTION

This was a real scenario, and the Exchange servers were deployed successfully. Expecting something else? Well, everything doesn't always work, but you reap the benefits of "best practices" as we share with you what worked and note some of the potential pitfalls along the way.

11. PROACTIVELY FOLLOW THE TUNING METHODOLOGY AFTER THE SYSTEM IS DEPLOYED

Environments do change. The information gathered following the core tuning methodology is helpful for staying one step in front of user demand by identifying current load demands and potential system bottlenecks. This information is also useful when developing and sizing future system solutions and when adding additional capacity to your current system(s). If you have not determined where and how to add additional capacity to your system, you are at great risk of wasting your time.

Exchange Server Sizing Configuration Chart Summary

Even though every environment is different, the Exchange server configuration shown in Table 8.24 provides sound sizing guidelines on which to base your initial server configurations. Follow the sizing template to customize the configuration to your environment and then proactively manage your Windows 2000 file server to an optimal fit!

TABLE 8.16 *Specific Configurations for Sizing Exchange Servers*

Concurrent Exchange Users	Pentium III Class CPU	RAM (MB)	Number of Ultra2 SCSI 10K RPM Disk Drives	Network Interface Card
<= 1000	1	512	7	100BaseTX
<= 2,500	1–2	1024	15	100BaseTX
<= 5,000	2–4	2GB	20+	(1–2) 100BaseTX or 1000BaseTX

It is possible to configure larger Exchange servers, and numerous vendors have achieved benchmark results in the 20,000 plus range. Even though it is possible to support more than 5,000 customers on a single server, very few will consider a single server due to the risks associated with a possible failure. Only you know the threshold of customers you are willing to accept on a single system!

Note: It is important to understand that as we scale each system, a balanced approach is taken between all server resources so that artificial bottlenecks are not created before the Exchange server is even deployed.

Windows 2000 Exchange Server Tuning Summary

Exchange Server Specific Application Tuning

The characteristics for Exchange's operation were reviewed at the beginning of this section. The two major areas in which Exchange must be tuned are in the memory and disk subsystems. Fortunately, Exchange provides a friendly and helpful tool—the Exchange Performance Optimizer—for most of the actual Exchange tuning. It is important to fully understand what your Exchange server will be running and the physical configuration of your server before running this tool! You will need this information to effectively leverage it to your advantage.

From a general tuning perspective, the first screen that shows up when running the Exchange Optimizer covers CPU and memory tuning of Exchange parameters (these parameters live in the registry and are adjusted after the Optimizer is complete). The first screen to review closely is shown in Figure 8-4.

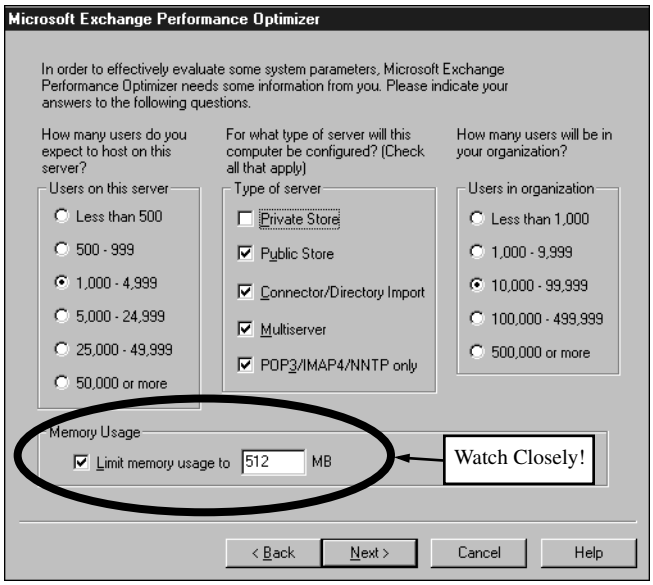

FIGURE 8–2 *The First Key Screen of the Exchange Optimizer*

Select options that are indicative to your environment. One key parameter that is commonly overlooked is the memory usage parameter. If you take the default, you are stating that "Exchange is the only application running on

my server, take ALL the memory." This is very important. If you are running another application on the server, such as a SQL database (not recommended), antivirus software, or even general management software, take the memory usage into account.

Earlier, when sizing the Exchange solution, we outlined memory for our system in Table 8.19. From this table, we estimated 802MB would be used for Exchange and the rest of the 1GB of total memory was for Windows 2000, the file system cache, and other management applications. If over time you observe memory not being used and your Exchange database cache hit ratio decreasing, rerun the optimizer and tune appropriately.

For the disk subsystem that we focused on so closely, the optimizer provides assistance in this area, too. Even though the optimizer tests the subsystem to determine which disks/arrays are the fastest and which are the largest, understand your disk subsystem design and place your key files where we planned. The optimizer is even nice enough to move the files for you. Figure 8-5 is a screen shot of the optimizer and disk subsystem tuning.

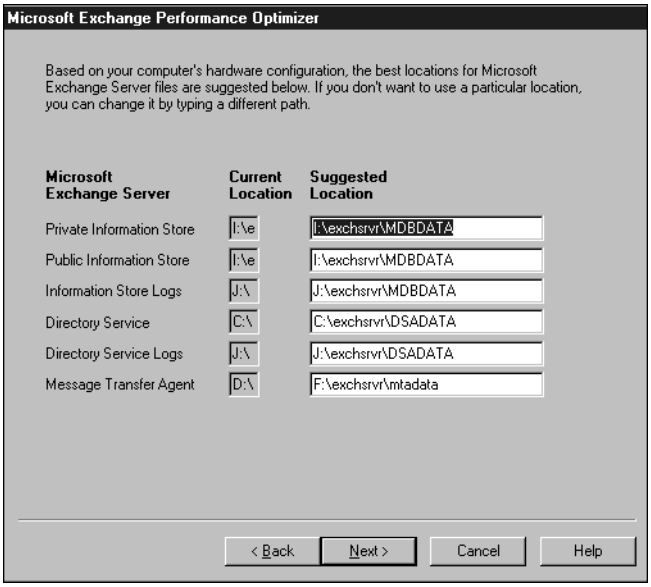

FIGURE 8–3 *The Exchange Optimizer and Tuning the Disk Subsystem*

Always rerun the optimizer any time server hardware components have been added or removed to ensure that the tuning settings take advantage of the resources available. The Optimizer does not take into consideration other applications that may reside on the server; thus manual intervention for those environments are required.

For additional hands-on specifics on why, when, and how these tuning recommendations work in detail and steps to implement them, refer back to the subsequent tuning sections of the previous chapters. Here select reformations are provided to give you a jump-start into an optimized Windows 2000 solution!

EXCHANGE SERVER HARDWARE TUNING

- Update the BIOS and Windows 2000 drivers on all server hardware (includes motherboards and the associated adapters).
- Ensure that the BIOS is configured for maximum performance.
- Ensure that the BIOS performance settings match the hardware you have installed.
- Ensure that you received the correct hardware.

EXCHANGE SERVER APPLICATION TUNING

- Configure the application correctly!
- Test and install the latest service packs for Exchange and Windows 2000.
- Configure each Exchange component in the best RAID array for its workload characteristics.
- Do not place the logs and stores on the same disk array.
- Run the Exchange Performance Optimizer after any server hardware or software changes.
- Review the Exchange Server Specific Application Tuning section above.

WINDOWS 2000 CPU SUBSYSTEM RESOURCE TUNING

Key disk CPU subsystem-related tuning you should consider:

- Set the application response optimization to background services (Control Panel / System / Advanced / Performance Options).
- With Microsoft Exchange V5.5, CPU utilization can be a bottleneck. On high-end systems (those with a large number of users), it may be necessary to add to the number of threads used by the STORE process that impact various queues in Exchange server in order to obtain the maximum amount of CPU utilization from the system. To accomplish this, edit the following registry key and add approximately 50% as a starting point: `HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\MSExchangeIS\ParameterSystem\Maxthreads` (Public and Private). If you run the Exchange Performance Optimizer in verbose mode, Perfwiz –v, you can see the current setting if the optimizer has changed this. Use this setting with caution. If CPU queues begin to form, throttle this value back. If you have plenty of CPU resources,

try increasing this value until you find the best value for your unique environment.

- Off-load CPU operations to I2O-enabled I/O adapters.
- Since multiple CPUs are in use with multiple NIC and SCSI adapters, consider using the Microsoft Interrupt-Affinity Filter Control Tool (IntFiltr; see chapter 4) to ensure interrupts are properly distributed between NICs and CPUs.
- Log off the server to ensure security and ensure no CPU cycles are wasted on your login session.
- If you do not log off, do not use a 3GL or other fancy screen savers, as they consume significant CPU resources.
- Check all services running on your server. Operate only those you need and stop all others.
- Avoid running another CPU-intensive applications on your Exchange server.

WINDOWS 2000 MEMORY SUBSYSTEM RESOURCE TUNING

Key memory-related tuning you should consider:

- Add the Heaps 8, Reg_dword parameter to `HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\MSExchangeIS\Parameter-System` to improve overall memory performance.
- Check the Windows 2000 memory management strategy. Exchange servers manage their own cache and do NOT rely on the file system cache. Ensure the Windows 2000 file system cache is set to optimize for Maximize Data Throughput for Network Applications. This tunable is set at Start / Programs / Settings / Network and Dial Up Connections / Properties of your LAN adapter / File and Print Sharing for Microsoft Networks.
- Check all services running on your server. Operate only those you need and stop all others.
- Set the pagefile size to twice the amount of RAM in your sever and to a fixed size.
- Never place the pagefile on a RAID 5 array.
- If your system must page (which should be avoided), consider multiple pagefiles on physically separate disk drives that are all configured to the exact same size and are housed on the same speed disk drives.
- Force Windows 2000 to keep the Windows 2000 executive (kernel) from paging any of its executive system drivers to disk and ensure that they are immediately available if needed (chapter 5).
- Do not use a 3GL or other fancy screen saver.
- Do not use exotic wallpaper; it wastes memory resources.

WINDOWS 2000 DISK SUBSYSTEM RESOURCE TUNING

Key disk subsystem–related tuning you should consider:

- Perform a fresh, low-level format on all disk drives at the lowest level using the disk or RAID adapter you are using.
- Freshly format all disk drives with NTFS when installing Windows 2000.
- When formatting the disk drive with NTFS, select the ALU size that best matches your disk workload characteristics. For Exchange servers, the recommended ALU is 4KB for all file systems except for the Information Store (IS). For the operating system and IS, use an 8KB ALU size.
- Match the stripe size of the low level RAID format with the ALU size used (see chapter 6 for details).
- Since we are using large NTFS volumes, increase the size of the NTFS log file size to 64MB, by using the *chkdsk* command: chkdsk drive-letter: /l:65536.
- Turn on write-back caching and read-ahead caching on hardware RAID adapters (ensure your RAID adapters have battery backed cache and are on the Microsoft certified hardware list).
- Use only I2O-enabled RAID adapters and those with their own CPUs.
- Keep the disk subsystem storage capacity usage under 50–60%, so that the fast portion of the disk drive is used, so that there is room on the disk drives to perform defragmentation, and operate exchange database recovery tools.
- Defragment your disk subsystem on a regular basis; this will result in significant performance improvements!
- Use one Windows 2000 partition per physical disk device/array.
- Group similar disk activities on the same physical disk subsystem and do not separate them via logical partitions on the same physical disk drive/array.
- Use more than one RAID level in your Exchange solution!
- Add disk drives as needed to keep the sec/transfers low and average queue size less than twice the number of disk drives in the disk array.
- Check all services running on your server. Operate only those you need and stop all others.

WINDOWS 2000 NETWORK SUBSYSTEM RESOURCE TUNING

Key network subsystem–related tuning you should consider:

- Teaming/trunking multiple NICs (Ethernet channels) together as needed to add network bandwidth and availability to your Exchange server (seamlessly to your customers), as shown in chapter 7.
- Remove all unnecessary protocols and Redirectors.
- Standardize on one network protocol: TCP/IP.

- If more than one network protocol is in use, set binding order such that the most commonly used protocol is first on the list.
- Tune the NIC driver to off-load IPSEC security operations and TCP operations on the NIC itself instead of the main system CPUs.
- Tune the NIC driver to increase its send and receive buffers.
- Use only I2O-enabled NICs.
- Use the fastest I/O bus enabled NIC possible (64-bit, 66MHz PCI slot)
- Tune the Windows TCP/IP stack to your environment (TCP Hash Table, MTU Window Size). See chapter 7 for the details on this or the web server scenario in this chapter for step-by-step details on how to implement.
- Do not use autodetect for your NIC. Set it to the speed that is the best performance level the internetwork devices will support. For example: 100BaseTX full duplex.

THIRD-PARTY TOOLS FOR TUNING WINDOWS 2000 EXCHANGE SERVERS

There are key third-party tools you should consider for enhancing the performance of your Exchange environment.

- Obtain a third-party disk defragmentation tool that provides enhanced defragmentation features such as scheduling and support of multiple NTFS ALU sizes. Recommended vendor to consider: Norton Speed Disk http://www.Norton.com.
- For heavily loaded Exchange servers, consider running a Windows 2000 scheduler and kernel subsystem enhancer. One such product is Autopilot from SunBelt software. Autopilot analyzes performance data to assist the operating system in making scheduling decisions. In my LoadSim performance testing, adding Autopilot to an Exchange server under load improved response times to the end user from 10–40%. Autopilot is available at: http://www.SunBeltSoftware.com.

Solution Scenario 5: World Wide Web Server Implemented with Microsoft IIS 5.0

Introduction

Web servers are the front-line warriors of the Internet, providing everything from information portals to robust eCommerce sites to a new generation of application services. With higher reliance on web servers, their performance has become increasingly important; in fact, from end users' perspective, web server performance is paramount. A web site that makes a customer wait for more than a few seconds has just lost that customer! Even worse than that, whenever your web site is running slow, the customer's perception of your company's reputation sags too!

In this scenario, the focus is on sizing and tuning web servers. Here, I do not harp on such obvious information as it is better to use compiled web applications native for your web server software engine (such as ASP or ISAPI for IIS, NAPI for Netscape, etc.) versus using a scripting language (such as Perl under CGI). Advising folks what language to use for their web applications is not new information and, besides that, everyone has their own development skill sets and may not be in a big hurry to change. Here the focus is on what you can do to tune your web server at the system level, which encompasses areas such as Windows 2000's use of server resources (CPU, network, disk, memory) and Microsoft Internet Information Services 5.0's software engine itself.

Application Description

Microsoft IIS 5 is a scalable and robust web (httpd) platform that has matured over the past several years. From a performance perspective, consider it an advanced communication system that delivers static and dynamically generated content.

Application Performance Characterization

When we understand how an application operates and the workload it is expected to support, we can then correlate its operations with the performance concepts we have learned throughout the earlier chapters. The whole world is not quite point and click yet. The primary services provided by IIS are web and ftp services that reside in the Inetinfo.exe process. Understanding this process is important for overall optimization of your IIS 5.0 web server. The Inetinfo.exe process not only includes these key services, but internally also handles a shared thread pool, object cache, and logging services. This is one very important process and it will be the focus of much of our IIS specific tuning.

People are creative. Web servers have now morphed into online file services, front end GUIs for back-end applications, as well as a secure portal into a company's Intranet or for collecting private information over the Internet such as credit card information. Heck, you can even run terminal services through a web browser (no terminal service client required!). Because of this, it becomes just as important to really understand the workload that IIS must complete because that affects sizing and tuning. The most common bottlenecks for IIS 5.0 are, in order, network, CPU, memory, and the disk subsystem.

Table 8.33 shows a summary of the key components of IIS 5 and those it relies on from Windows 2000.

TABLE 8.17 *IIS Component Descriptions*

IIS 5.0 Components	IIS 5.0 Component Descriptions and Characterizations
Inetinfo.exe	This is the primary IIS 5.0 process that drives web, ftp, and SMTP services. It also includes the shared thread pool, cache, and logging services. Special performance note: *never* let memory get so low that parts of this process are paged out! If this occurs, your web server will run very slowly.
Windows 2000 File System Cache	IIS 5 uses the Windows 2000 dynamic file system cache to cache web pages and files that are requested from the web server or pushed to it. Performance note: Review chapter 5 to ensure you understand the detailed performance characteristics of the file system cache so you can take the best possible advantage of it. IIS 5 relies heavily on the file system cache.
IIS 5.0 Logs	IIS 5 supports one memory mapped log file for each web site it supports that has logging enabled. These files are part of the working set of the InetInfo.exe process and are mapped in 64KB chunks. Performance note: Besides using memory from the Inetinfo.exe process itself, it also uses memory resources from the Windows 2000 file system cache when data is written to the log files residing on disk subsystem.
IIS Object Cache	Contrary to popular opinion, general files are not cached in the object cache which resides inside of the working set of Inetinfo.exe. Objects such as file handles, file directory information listings, and other objects that are used frequently and are expensive to retrieve get stored in the object cache. Performance note: Configure and tune sufficient memory such that the object cache provides > 90% hit rate.
ASP Template and Scripting Engine Caches	Active server pages (ASP) have their own support structure inside of IIS, customized to ASP environments. These two cache structures work hand in hand. The IIS template cache holds pointers to script engines while the script engine cache holds precompiled scripts of ASPs that are ready to run. As more cache hits that are achieved in these two caches, the more CPU overhead is decreased, as fewer pages will be required to be dynamically generated. Performance note: Since these caches do support ASP's dynamic page generation nature, it is a challenge to ensure that they are effective.

TABLE 8.17 *IIS Component Descriptions (Continued)*

IIS 5.0 Components	IIS 5.0 Component Descriptions and Characterizations
Shared Pooled Thread Resources	This component resides inside of Inetinfo.exe and provides the worker bee threads that actually execute the applications and services that reside inside of InetInfo.exe. These threads are stored in the nonpaged pool of memory resources. Performance note: Track nonpaged memory usage; you may need to tune what Windows 2000 makes available (chapter 5). Also, track the number of threads as they can be optimized to your environment (later in this chapter).
Http Connection Data Structures	This component is responsible for tracking active connections. Each connection consumes a small amount of system resources (CPU and Memory). This component is also part of the Inetinfo.exe working set. Performance note: Were you already reminded to never let Windows 2000 page portions of Inetinfo.exe page to disk?

Unleash Your Web Server's True Power!

You might be surprised just how much web power is available via a two-CPU-based Windows 2000 IIS 5.0 web server. No single industry benchmark is perfect, but for a frame of reference, let's review a published web performance result based on the SPECweb99 industry standard benchmark. This benchmark measures the throughput of simultaneous web connections that the web server can support. A quick check of the http://www.spec.org web site found a Compaq DL-360 with two 800MHz Pentium III CPUs supporting 1,020 concurrent customers. For this example above, the amount of data delivered when 1,020 customers are all simultaneously pulling data at the benchmark minimum allowable throughput level of 320,000 bits/sec translates into 326Mbits/sec. If you were serving this data over the Internet, you would require seven T3 (45Mbits/sec each) lines to the Internet.

At this performance level, the Compaq DL-360 server provided 2,831 operations (hits)/sec. This works out to roughly 10,191,600 web operations per hour! Even with multiple operations required to provide a single web page, this is enough power to handle some serious web sites. Most sites will rarely require this level of performance, but it is nice to know what an optimized Windows 2000 web solution running on some basic server iron can provide.

By the end of this scenario, you should be able to unleash the power of your web server, but how large a network connection will you need for your customers? Consider the common network connectivity options shown in Table 8.34 when thinking about where your web server bottleneck may be

occurring. Note that we do not take network protocol overhead into account in this example, just a relative comparison of how many 10KB web pages each medium can support.

TABLE 8.18 *Relative Theoretical Network Throughput and Media Access Characterization*

LAN Network Technology	Throughput Megabits/sec	Throughput MB/sec	Number of 10KB Web Pages that Media Can Simultaneously Download
Ethernet (100BaseT)	100	12.5	1,250
Ethernet (Gigabit Ethernet/ 1000BaseTX/FL)	1,000	125	12,500
WAN Network Technology	**Throughput Megabits/sec**	**Throughput MB/sec**	**Number of 10KB Web Pages That Media Can Download**
Dedicated PPP/SLIP	Modem (up to 56Kbit/s)	0.007	0.7
T1	1.544	0.193	19.3
Digital Subscriber Line (DSL)	7	0.875	87.5
T3	45	5.5	550

Sysmon Objects and Counters: IIS 5.0 Server Health, Bottleneck Detection, Tuning, and Sizing

The Sysmon objects and counters outlined in the previous chapters for CPU, memory, disk, and network subsystems are still applicable to Windows 2000 web servers as it is important to understand the big picture when tuning or sizing web server solutions. If one server resource becomes a bottleneck (queues begin to form in the CPU or disk subsystem), response time to the end user begins to suffer.

In addition to the previously reviewed rules of thumb for bottleneck and sizing objects and counters, it is recommended that you become familiar with the Sysmon objects and counters that are added to Sysmon when IIS 5.0 is installed (Table 3.35). These objects and counters are helpful in determining the health of your IIS 5 server application and for tuning and sizing.

Microsoft provides a number of statistics through Sysmon for monitoring the performance of IIS 5 servers, but Sysmon counters do not provide every aspect of detailed instrumentation data. Critical data is also available in the IIS Tracking Logs and the Windows 2000 Event Logs.

TABLE 8.19 *Sleuthing Out Microsoft IIS Specific Health, Tuning, and Sizing Information*

Object: Counter	Definition	Value to Monitor
Web Service:Total Not Found Errors	Total Not Found Errors is the number of requests that couldn't be satisfied by the server because the requested document could not be found.These are generally reported as an HTTP 404 error code to the client.The count is the total since service startup.	Health indicator and bottleneck detection Why waste resources looking for pages that are not there? This counter is a good indicator of questionable web site maintenance (cleaning up of old links).
Process: Inetinfo.exe All parameters	Obtain and track information on the InetInfo.exe process to determine its overall performance health.	Health indicator and bottleneck detection Tracking all of the Inetinfo.exe counters provides insight into the health and performance of your web server.
Active Server Pages: Errors/sec	The numbers of errors per second, including connection errors, compile errors, and run-time errors.	Health indicator Why waste resources looking for pages that are not there? This counter is a good indicator of how well the ASPs are written. If this number is greater than 0, something is wrong with the test scripts, server configuration, or scripts in ASPs
Active Server Pages: Requests Queued	The number of requests waiting for service from the queue.	Bottleneck detection If you are running ASPs on your web site, track this value. If the ASP queue grows at all, customer requests are failing and they will receive server busy errors in their browsers (if a customer message was not defined).To resolve this condition, ensure that CPU cycles are available, then increase the IIS queue. This is reviewed later in this scenario.
Internet Information Services Global: Cache Hits %	Instantaneous value of how well the IIS object cache is doing.	Tuning and sizing If this value is < 90% and you have additional memory resources available, increase its available cache size through the registry. (See IIS 5.0 tuning section.)
Internet Information Services Global: Cache Flushed	How often the cache is emptied due to time-out or because object changed.	Tuning and sizing If objects are moving through the object cache too swiftly, consider upping the time to live values in the registry.

TABLE 8.19 *Sleuthing Out Microsoft IIS Specific Health, Tuning, and Sizing Information (Continued)*

Object: Counter	Definition	Value to Monitor
Internet Information Services Global: File Cache Hits % and Cache: Data Map Hits %	How well the Windows 2000 file system cache is performing for IIS related tasks.	Tuning and sizing Is the percentage of data maps in the file system cache that could be resolved without having to retrieve a page from the disk, because the page was already in physical memory. The higher this percentage, the better performance the customer will receive.
Web Service: Total Connection Attempts	Total Connection Attempts is the number of connections that have been attempted using the Web service (counted since service startup). This counter is for all web servers hosted on this server.	Influences tuning and sizing This information is helpful in sizing the overall solution.
Web Service: Maximum Connections	Maximum Connections is the maximum number of simultaneous connections established with the web service.	Health of IIS If the Web Services: Total connection attempts is much greater than the maximum connections, then there may be a performance or configuration problem that is stopping customers from connecting to the web site.
Web Service: Current Connection	Current Connections is the current number of connections established with the web service.	Influences sizing and tuning How much simultaneous activity is your web site experiencing. Excellent for tuning and sizing.

Scenario 5 Step by Step: Sizing and Tuning a Mid-Range Windows 2000 Web Server

Sizing the Initial Web Server Configuration

In chapter 3, the core sizing methodology was presented for sizing your Windows 2000 solution. That methodology is followed here for the development of the initial Windows 2000 web server configuration.

1. DEFINE OBJECTIVE(S)

Before the merger of the two companies, both companies ran into regular problems with their web and eCommerce presence regularly being over-

whelmed. Now that the merger is complete, Ajax Rockets has decided that its Internet presence is too important to continually run into performance issues. The web server will primarily be the front end for eCommerce operations. It will also be used to host internal documents accessible by internal associates only. Although the technical team recommended separate systems for intranet and eCommerce web solutions, they were overruled by management. Go figure. The technical team will still provide an outstanding solution.

2. UNDERSTAND BUSINESS AND TECHNICAL REQUIREMENTS NEEDED TO MEET YOUR OBJECTIVE(S)

- Which application(s) will meet your functional objective(s)?
  Microsoft IIS 5.0 will be the web server engine.
- What are the availability requirements?
  These are critical servers, as they will support the entire company for eCommerce and intranet activities. The goal is to have a highly available solution. This is a high-profile, business-critical application. No downtime is considered acceptable.
- What are the ramifications if your system is down?
  If the system does not provide service, loss of revenue and loss of associate productivity will occur.
- What are the disk subsystem storage capacity requirements?
  There are approximately 1GB of web server application space required and 10GB of various marketing material required. This web server will front end the eCommerce database and intranet document database repository.

Looking at the big picture, all disk storage requirements are outlined in Table 8.36.

TABLE 8.20 *IIS 5.0 Server Disk Storage Sizing Matrix*

Application	Disk Storage Capacity Needed
Windows 2000 operation system requirement and IIS 5	<500MB
Pagefile	2GB
Web server application	3GB
System management agent	5MB
Security application (server-based firewall and IDS)	30MB
Web Content eCommerce and Intranet	10GB

3. DETERMINE LOADING CHARACTERISTICS

- *Number of total users per server*
  Marketing's best estimate is that 240 customers/sec will visit and interact with the web site. This works out to 14,400 hits/minute or 864,000

hits per hour in a worst-case—or best-case—scenario, depending on how you look at it. On average, the customer visitation rate is expected to be much lower. However, for a critical eCommerce solution, Ajax Rockets has decided to design a solution for a worst-case scenario to handle the holiday rush.

Also, due to availability requirements, they intend to spread the web workload across two IIS 5 servers using Network Load Balancing (NLB), which is included in the Windows 2000 Advanced Server. Using this technology, a shared-nothing cluster is created that will respond to the same IP or web site address in a load balanced manner. Thus, the workload will be split between the two servers. For more information on NLB, check out the following web site: http://www.microsoft.com/TechNet/win2000/nlbovw.asp

- *Number of concurrent users per server*
  240. See discussion under total users.
- *Disk workload characteristics (read/write environment)*
  The goal is to cache as much as possible in memory to improve overall performance. Except for logging, all data are considered random in nature with most of the requests being read intensive, as all write-intensive information is stored in the back-end databases.
- *Size of average records/transactions*
  Average size of each web page is 20KB.
- *Physical location of users (Influences communications such as WAN, LAN, dial-up, terminal services, etc.)*
  All servers reside on site and are connected internally via the local switched Ethernet 100BaseTX LAN for Internet access. Currently, two redundant T3 lines are in place.
- *Type of work the user will complete*
  This web site is part of the overall eCommerce and intranet solution for Ajax Rockets. Primary work that will be completed includes eCommerce transactions centered on sales and marketing information, an Intranet Knowledge management services, and search engine services for the web site.

4. DETERMINE PERFORMANCE REQUIREMENTS

From a performance perspective, response time to the customer is the most important factor for a web server solution. Ajax Rocket's business goal is to provide all web responses to customers within three seconds. The second most important performance requirement is backup operations during each evening and being able to restore the backup data within four hours. For this environment, using Windows 2000 NLB, only one of the servers must be backed up, since all data is replicated to each of the web servers (using a third-party tool named Double Take, available at http://www.SunBeltSoftware.com) to ensure each web server provides the same consistent data to

customers. Data such as ASP updates are placed onto the primary web server and Double Take replicates this data to the other servers in the cluster over the network.

Based on current business requirements worst-case scenarios, 13GB must be backed up in eight hours and restored in four hours. If a failure occurs, Ajax Rockets relies on the other web servers in the cluster to provide web services while the down server is restored. To restore this amount of data in four hours (which is more disk intensive than backing up the data), the disk and network subsystem will need to support a restoration rate of 3.25GB/hour, which works out to 0.9MB/sec. This is relatively easy to accommodate. Follow the approach outlined in the backup server scenario in this chapter to meet the backup goals.

5. UNDERSTAND FUTURE BUSINESS REQUIREMENTS

Ajax Rockets is an optimistic company and expects that orders will grow by at least 30% per year. To accommodate this, they could build larger servers or add additional web servers to the NLB Cluster, which supports up to 32 servers in a shared-nothing cluster. Ajax Rockets has decided to take the cluster approach to scale their solution. As needed, they plan to add identical servers to meet the projected demand. This concept is referred to as Redundant Array of Inexpensive Servers (RAIS).

6. UNDERSTAND FUTURE SYSTEM ARCHITECTURES

By taking and adopting the RAIS strategy, Ajax Rockets is looking to implement cutting-edge but smaller servers to meet its web services needs.

7. CONFIGURE THE SYSTEM

Finally, to the good stuff. At this point, you can configure the Windows 2000 IIS 5 server. Choose the system resources required in meeting your objective, with consideration for steps 1 through 6, and obtain the system. As you configure the system, look at each individual resource to meet the overall system configuration. Software and hardware vendors are good sources of information and can suggest configurations that will get you started. Reviewing relevant industry benchmarks also can help provide insight into the initial configuration. Industry benchmarks are explored in depth in chapter 3. The following steps are presented to help guide you through the actual system configuration process.

Having some sort of idea on how each system resource works and relates to the overall system performance is helpful when configuring your system. Chapters 4 through 7 review the overall system architecture of each system resource in greater detail.

7.1 Application Characteristics Review • The general IIS 5 application characteristics and backup technology review was completed at the top of the web server scenario section.

7.2 Review Each System Resource for the Windows 2000 Web Server

Disk I/O Requirements • The disk subsystem houses much of the information provided via the web server while the back-end databases provide the dynamic content data. The goal is to cache much of the data needed for web transactions, but at 10GB of potential data, we will need an optimized disk subsystem for low latency, as this is an interactive solution. From the business requirements above, we will determine the disk storage and performance requirements.

Availability • Even though clustering is used in this solution, all data will reside on hot swappable RAID array–based disk subsystems that use some level of fault tolerance. Transactional data is kept on the back-end database servers.

Performance Workload Characteristics • From the workload characteristics of a general IIS server above, we will need to support only two RAID levels to meet the performance requirements: the random nature of the eCommerce and intranet data and the sequential write-intensive data from the IIS logs.

Performance: Throughput and Transfers/sec • Based on what we learned about tuning disk subsystems in chapter 6, it is desirable to group applications with similar disk workload characteristics together on the same physical disk subsystem for maximum performance. Taking this approach with our current web server environment, the eCommerce and intranet data will be grouped together on the same physical disk subsystem while the write-intensive log file data will be placed on a separate disk subsystem.

In this environment, we have two performance requirements—response time and throughput to the end user and throughput needed for backup operations. If the transactions/sec occurring on the disk subsystem becomes too high and disk queues begin to form, it will affect end-user response time. To determine the transfers/sec required to support three-second response time for a user, we can review industry standard benchmarks and historical information.

The most common industry benchmark for web server environments is SpecWeb99. SpecWeb99 results are available at http://www.spec.org. However, for this environment, historical information is more helpful.

Backup performance was discussed in the backup scenario and is not a very large workload component in this scenario, so we will not cover it here. In a worst-case scenario—assuming that the file system cache that supports IIS 5 is not effective—the disk subsystem will need to simultaneously support 240 eCommerce customers, each requesting 20KB documents, and 20 internal

associates, each requesting 20 400KB documents. This would require that the disk subsystem provide 4.8MB for eCommerce requests in three seconds (1.6MB per second) and 8MB for intranet operations in three seconds (2.6MB/sec) of overall throughput. If each disk transaction provided 32KB of data, this indicates that we need to support 50 transactions/sec for eCommerce applications and 81 transactions/sec for intranet operations. Since this environment is random in nature, read intensive, and fault tolerance is required, a RAID 5 array would be perfect. If we placed the eCommerce and intranet data on the same array, we will need to support 131 transfers/sec and 4.12 MB/sec of throughput. From chapter 6, we found that each disk drive in a RAID 5 array can support approximately 1.11MB/sec of throughput and 100 transfers/sec. Reviewing the worst case of what we need to support, two disks would provide the transactions/sec [131 disk read transfers/sec] + [4 × (0 disk write transfers/sec)] × [1.25 utilization factor/100 transfers/sec/disk] needed, but we will need four disk drives to support the throughput requirements (4.12MB/sec / 1.11MB/sec per drive).

To meet our response time and throughput performance requirements, we take the greater number of disk drives that were calculated in the above steps. Thus, four disk drives are required (the higher number between the response time and throughput time calculations above) to meet our projected performance requirements. For more details on disk subsystem performance, review chapter 6.

Storage Capacity • Even when focusing on disk storage capacity, continue to think performance. The fastest portion of the disk subsystem is the outer edge of the physical disk. If you keep the disk subsystem at less than 60% filled, it will just run faster! Disk defragmentation tools typically require at least 20% disk free space to operate and 30–40% to run optimally. Based on all of the above disk information, Table 8.37 outlines the overall disk subsystem requirements.

TABLE 8.21 *Web Server Disk Subsystem Solution (Server 1 of 2)*

Application	Minimum Disk Storage Needed	Logical Partition	Recommended Configuration (All 10,000rpm, Ultra2 SCSI Disk Drives)
Windows 2000 operation system and IIS requirements	<500MB	C:	2 9GB disk drives in RAID 1 array
System management agent	5MB	C:	
Antivirus application	20MB	C:	
Security application (server-based firewall and IDS)	30MB	C:	
Web applications	3GB	C:	

TABLE 8.21 *Web Server Disk Subsystem Solution (Server 1 of 2) (Continued)*

Application	Minimum Disk Storage Needed	Logical Partition	Recommended Configuration (All 10,000rpm, Ultra2 SCSI Disk Drives)
Pagefile and IIS log file (historical data on who visited the web site and their patterns)	Pagefile: 2GB IIS Logs: ~2GB	L:	2 9GB disk drives in RAID 1 array (9GB usable/4GB filled)
eCommerce and intranet data	10GB	H:	4 9GB disk drives in a RAID 5 array (27GB usable/10GB filled)

CPU Requirements • The primary operations to support from a CPU perspective are the IIS web server, the static and dynamic web application operations, delivering content over the network, and security related operations associated with Secure Socket Layer encryption (SSL). SSL operations introduce a CPU overhead of three to six times more than non-SSL operations. SSL operations will be used in this site, but are leveraged only when absolutely necessary. From a CPU perspective, the initial creation of an SSL session is the most expensive portion of the SSL operation. Later we will look into tuning techniques of how to lessen the effect that SSL has on our site.

Table 8.38 shows historical information for two current web servers that are online.

TABLE 8.22 *Initial Web Server CPU Sizing*

Application	CPU Usage (Estimate) Based on One Pentium III 800MHz CPU	Total CPU Usage
Windows 2000 and General operations		10%
Overhead for system management software		1% (estimated)
Overhead for intrusion detection software		1% (estimated)
Per eCommerce user connection CPU requirements	0.4% of CPU resource/user × 240 users	96%
Per intranet user connection CPU requirements	0.8% of CPU resources/user × 20 users	16%
Search engine requirements		6%
Total Estimated CPU Usage		***130% Total CPU usage or 65% per CPU if 2 CPUs are used***

For a general reality check we should ask: Can two CPUs provide this level of web server workload? To answer that question, we obtained a SpecWeb99 benchmark from http://www.spec.org to see how much performance two CPUs in an intense web server environment could provide. The benchmark was configured using a Compaq DL-360 with two Pentium III 800MHz CPUs. This benchmark configuration was able to sustain 2,831 operations (hits)/sec. This level of performance is higher than our currently projected workload.

With the above information, two 800MHz/256K Pentium III CPUs will be configured. We are estimating that only 63% of each CPU will be used during peak hours, which provides room to grow. However, since Ajax Rockets is concerned with performance availability, there will be two servers configured identically with Windows 2000 NLB software, which will balance the workload between the two servers. This should provide more than adequate performance support for the projected workload in a worst-case scenario.

Memory Requirements • Sizing RAM is an additive process. Our first step in sizing memory resources is to outline all applications that will use any memory resources. Microsoft IIS manages only some of its own cache and memory requirements and relies heavily on the Windows 2000 dynamic file system cache.

The amount of memory that IIS 5 uses for caching operations is tunable by editing the IIS management GUI in conjunction with direct registry tuning. Table 8.39 shows a breakdown of where the memory resources are being focused. This information is based on historical data and our own targets.

Through proactive performance management, more RAM can be added (or removed) as needed. Web server environments love properly tuned RAM! This estimate is based on historical information collected (memory working set was monitored on the two precious web servers) and insight into how Windows 2000 takes advantage of memory resources. It provides us a good place to start. For a more detailed discussion of Windows 2000 and memory resources, refer to chapter 5.

TABLE 8.23 *Web Server Memory Sizing Matrix*

Application	Memory Usage (Working Set)
Windows 2000 operating system requirement	128MB
Windows 2000 file system cache (maximum for configurations less than 1GB in size)	432MB (This is the maximum amount of memory that Windows 2000 will commit to the file system cache when less than 1GB of memory is configured)
Overhead for system management software running on server	8MB

TABLE 8.23 *Web Server Memory Sizing Matrix (Continued)*

Application	Memory Usage (Working Set)
Overhead for intrusion detection software	12MB
Backup server application	10MB
Memory used for each user concurrent user connection (300 × ~20KB)	6MB
IIS 5 (program code, object cache, template cache, script engine cache, HTTP structures, pooled threads, etc.)	428MB
Total Estimated Memory Usage	***1024MB***

Network I/O Requirements • This is the critical component in the web server solution and where most web servers run into problems. How much bandwidth is required? Let's review Ajax Rocket's expected workload for the web server. Note that for network calculations, we convert to Megabits per second, as that is the common notation used with networking technologies.

First, we have the eCommerce requirement to simultaneously support 240 customers each requesting a 20KB document with a three-second response time, which works out to 12.8Mbits/sec (1.6MB/sec) in a worst-case scenario; remember, this is before web and network overhead. If we add a 20% overhead ratio, we will need 15.36Mbits/sec of bandwidth. For our simultaneous intranet customers, each requesting a 400KB document, we require 20.8Mbits/sec (2.6MB/sec). Adding 20% network and web server overhead, this works out to 24.96Mbits/sec. From a backup and restore perspective, less than 7.2Mbits/sec (0.9MB/sec) is needed. With this information, we compose the network bandwidth summary chart shown in Table 8.40.

TABLE 8.24 *Web Server Network Sizing*

Service	Network Throughput Required Internally	Network Throughput Required to Internet
eCommerce Operations	N/A	15.36Mbits/sec
Intranet Operations	24.96Mbits/sec	8.32Mbits/sec
Backup Operations	7.2Mbits/sec	N/A

Sizing network bandwidth is not all science; there is some art involved. It is also important to know your environment well. A partial T3 connection to the Internet would meet the immediate eCommerce requirements, and a single 100BaseTX NIC (100Mbits/sec) would meet the network requirements of the web server itself. However, one third of the company is still located at another location and the network connection between them is based on a

Virtual Private Network (VPN) connection that uses the Internet. Thus, at a minimum, the network bandwidth required to support both the eCommerce and intranet applications is estimated to be 24Mbits/sec.

7.3 System Architecture Relationship Check • This is a final, common-sense check of the Windows 2000 web server configuration. Since the business requirements require keeping this new web server for at least three years, leading-edge server technology will be obtained. Not cutting edge, as that technology tends to demand a premium price.

From all of the sizing information above, we now have the following configuration: a one- to two-CPU capable server, two Pentium III 800MHz CPUs with 256KB each of full-speed (ATC) Level 2 cache, a 133MHz system bus, 1024MB of RAM, a single 64-bit 33MHz PCI I/O bus (266MB/sec), one internal hardware based RAID controller, two disk drives for the operating system and log files, four disk drives for the static web content, and one 100BaseTX Ethernet NIC. This server configuration is balanced and there is nothing notable in the way of data path bottlenecks.

There is one problem with this solution and a potential concern. First, only a single fast Ethernet connection is planned for. From a network availability perspective, some of the folks at Ajax Rockets have encountered failed NICs in the past. With the web server being such an important item, they have elected to obtain two NICs and use Adaptive Fault Tolerance. This is surprisingly easy to configure on NICs that support this feature and is found under the NIC's advanced tab.

7.4 Future Business and System Architectures Check Review • This server has room to grow in the disk and network areas but is limited in the CPU area. If the workload stays the same but more disk storage capacity is needed, that can easily be added to the external arrays. From a performance perspective, if the workload does increase additional servers can be added to the Windows 2000 Network Load Balanced cluster to meet the increased demands.

7.5 Initial Hardware Configuration and Tuning • In this step, the initial configuration of the server hardware begins to take shape based on steps 7.1 and 7.2 (Table 8.41). See the Sizing and Tuning Web Servers section below as needed.

Table 8.25 *IIS 5.0 Web Server Hardware Configuration Specifications*

Windows 2000 Web Server Hardware Configuration
Compaq ML-370 (Why select the ML-370? We needed a solid system to use for a real-world reference.)

CPU Configuration	*CPUs Used in Solution*	*CPU Family*	*Level 2 Cache Size*	*Speed of CPU*	*Available CPU Slots*
	2	Intel Pentium III	256KB	800MHz	2 of 2 used

TABLE 8.25 *IIS 5.0 Web Server Hardware Configuration Specifications (Continued)*

Windows 2000 Web Server Hardware Configuration
Compaq ML-370 (Why select the ML-370? We needed a solid system to use for a real-world reference.)

RAM Configuration	*Amount of RAM Used in Solution*	*System Bus Speed*	*Type of RAM*	*RAM Expansion*
	1024MB	133MHz	ECC SDRAM	4GB

NIC Card Configuration	*Number and Type of I/O Card*	*Number of PCI I/O Buses*	*Placement of PCI Cards*	*Speed of PCI Bus*
	1 Internal RAID Adapter	2	Not applicable. Built into motherboard.	One (1) 64-bit 33MHz; one 32-bit 33MHz
	2 Single Channel Full Duplex 100Base TX		The two 100BaseTX NICs are placed into the first two available PCI slots 0 and 1 in the 64-bit PCI bus.	

Disk Subsystem Configuration	*Partition/ Mount Point*	*Files Located There*	*Type of RAID Adapter*	*SCSI Channel*	*RAID Level*	*# of 10K rpm Ultra2 SCSI Disk Drives*
	C:	Windows 2000 installation, IIS application and logs, system management software, intrusion detection software	Built-in RAID adapter Compaq	1 of 1	1	2 9GB disk drives
	L:	IIS logs and page file	Built-in RAID adapter Compaq	1 of 1	1	2 9GB disk drives
	H:	eCommerce and Intranet Data	2 Channel Ultra2 SCSI	1 of 2	5	4 9GB disk Drives

7.6 INITIAL SOFTWARE CONFIGURATION AND TUNING • In this step, the initial software configuration of the server configuration begins to take shape based on steps 7.1 and 7.2. See the Tuning Web Servers section below.

7.7 INITIAL GENERAL TUNING • See the Tuning Web Servers section below.

8. STRESS TEST AND VALIDATE THE SERVER CONFIGURATION

Although not recommended, Ajax Rockets has decided that they do not have the time to stress test their solution, due to time to market pressures. However, Ajax Rockets feels strongly that with the current design and through proactive management, they can quickly scale the solution as needed to meet their needs.

9. PROACTIVELY FOLLOW THE CORE TUNING METHODOLOGY DURING STRESS AND VALIDATION TESTING ANALYZING THE RESULTS: KEY OBSERVATIONS

Not applicable since no stress testing was completed.

10. DEPLOY THE SYSTEM INTO PRODUCTION

A very similar scenario to this was deployed successfully after some Internet connectivity problems were resolved.

11. PROACTIVELY FOLLOW THE TUNING METHODOLOGY AFTER THE SYSTEM IS DEPLOYED

Environments do change. The information gathered following the core tuning methodology is helpful for staying one step in front of user demand by identifying current load demands and potential system bottlenecks. Also, use this information when developing and sizing future system solutions and when adding additional capacity to your current system(s). If you have not quantifiably determined where and how to add additional capacity to your system, you are in great risk of wasting your time.

IIS 5.0 Web Server Sizing and Configuration Chart Summary

Even though every environment is different, the IIS 5 web server configuration chart (Table 8.42) provides sound sizing guidelines on which to base your initial server configurations. Follow the sizing template above to customize to your environment and then proactively manage your Windows 2000 Web Server to an optimal fit!

TABLE 8.26 *Specific Configurations for Sizing Web Servers*

***Concurrent* Web Users (Hits/sec – 10KB page)**	**Pentium III Class CPUs**	**RAM**	**Number of Ultra2 SCSI 10K rpm Disk Drives**	**Network Interface Card**
<= 100 (8,640,000 hits/day)	1	512MB	4	100BaseTX
<= 250 (20,736,000 hits/day)	1–2	1024MB–2GB	7	100BaseTX
<= 500 (43,200,000 hits/day)	2–4	2–4GB	14	(1–2) 100BaseTX or 1000BaseTX

Wow, more than 8 million hits per day! These numbers can be deceiving if you look fast. Few eCommerce solutions or basic web sites have their workload well balanced throughout the day. Reality is sink or swim! You may have an enormous storm of 7 million hits in one hour, and only a few hundred the rest of the day; that works out to 7.2 million per day. Interesting, isn't it? The

real challenge for eCommerce environment is to plan for and be able to stay alive during the peak, or worst-case, workload scenarios!

Windows 2000 IIS 5.0 Web Server Tuning

IIS 5.0 Web Server Specific Application Tuning

The basic characteristics for IIS 5.0 web server operations were reviewed at the beginning of this section. Here we look at the high-level holistic system level approach of tuning an IIS 5.0 web server. Before we jump into the hands-on tactics, let's cover some web optimization strategies first. Remember that to get the most out of your web server it is important to understand what type of content is really being provided. Follow the guidelines presented in the sizing section of this scenario so that you can be sure of starting with a sound, properly sized platform. This is important, since each content type taxes your web server resources in a slightly different manner.

When tuning your IIS 5.0 web server, remember to follow the core Tuning Methodology Review in chapter 2.

From a holistic system perspective, the primary parts of our Windows 2000 IIS 5.0 web server tuning strategy are:

- Size the disk subsystem layout for low latency (completed in the earlier sizing section).
- Size the CPU, memory, and network subsystems for the workload so that you can meet the business requirements (completed in the earlier sizing section).
- Optimize the Windows 2000 network subsystem for web-based workloads.
- Tune the web server software engine to use all available server resources (CPU, memory, network, disk, etc.).
- Optimize Windows 2000 subsystems.

So, how can you accomplish these strategies? Read on. We will now review the step-by-step tactics that you can quickly employ to maximize your web server's performance.

Caution 1: There are numerous tuning options outlined below that require editing the Windows 2000 registry and the IIS metabase. Ensure that you properly back up the registry and metabase and understand how to recover them. Test your backups before tuning any registry entries or metabase entries.

Caution 2: Many of the tuning options presented increase either the CPU or memory resource usage. Ensure that you have followed the tuning methodology from chapter 2 and have a solid baseline of the performance of your web server before beginning any tuning. With this information and continued proactive management of your web server, you can then quantitatively determine if your efforts are successful. At a minimum you should track the following key system monitor objects and counters throughout the web server tuning process so that you do not overwhelm your server (Object: Counter): Processor: %Processor Time, Process: All Counters for Inetinfo.exe process, System: Processor Queue Length, Server: Pooled & NonPooled Memory: System Cache Resident Bytes, Memory: Available Mbytes, and Memory: Pages/sec. Also consider running the customer perspective scripts from chapter 2 that measure the http response time of your web server over the network.

Strategy: Optimize the Windows 2000 Network Subsystem for Web-Based Workloads

OPTIMIZING WINDOWS 2000 TCP SUBSYSTEM FOR WEB SERVER APPLICATIONS

Assume that our sizing for the network above, in conjunction with proactive tuning, is successful and we have no physical network bottlenecks between us and our customers. So, now that the superhighway leading to your web server is not a bottleneck, consider optimizing the Windows 2000 TCP stack for web server operations. You will want your IIS 5 web server to respond to as many connections as your resources will allow, so don't let an ill-tuned TCP stack stand in your way and slow you down—tune it!

Before we begin optimizing the TCP stack, remember that, as with all tuning, track all other system resources so that you ensure your tuning operations result in no adverse effects. Specifically when tuning TCP parameters, monitor the nonpaged and paged pool resources closely with Sysmon, since increasing these TCP parameter settings increases the amount of pooled resources that are used.

The transmission control blocks store data for each TCP connection. A control block is attached to the TCB hash table for each active connection. If there are not enough control blocks available when a web request (http) arrives at your server via TCP/IP, there is added delay while it waits for additional control blocks to be created. By increasing the TCB timewait table size, latency overhead is reduced by allowing more web connections to be serviced in a faster fashion. To adjust this operation, add the following registry value: `HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\Tcpip\Parameters\MaxFreeTcbs=0x9c4`.

This example increases the TCB timewait table to 40,000 entries from the default of 2,000.

Now that overhead time introduced by TCP is lowered for the web server, you must adjust the corresponding hash table, which is where the TCBs are stored. Accomplish this by adding the following registry value `HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\Tcpip\Parameters\MaxHashTableSize=0x2000`.

This increases the TCB hash table size from a default of 512 to 8,192, allowing more room for connection information.

In addition to increasing the size of the TCP hash table, we want also to improve the efficiency of how TCBs are located by increasing the number of TCB table partitions available; that helps to mitigate search contention. To accomplish this, add the following registry value: `HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\Tcpip\Parameters\NumTcbTablePartitions` to 8 from its default value of 4. Note, this value must be a power of 2 and is typically optimized when you have between two to four times the number of CPUs in your server.

When one user stops communicating with the web server, perhaps moving off to another web site (which you hope does not belong to a competitor), the original web server keeps the TCP socket pair in an open state for a default time of 240 seconds. However, on very busy web sites, this behavior may use all of the available ports before they are freed. You can adjust this behavior by adding the registry value: `HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\Tcpip\Parameters\TcpTimedWaitDelay=0x3c`, sets TIME_WAIT parameter to 60 seconds.

This is a non-RFC 1122 value, but should be acceptable for most large web sites. What if all of the TCP ports are being used? Add more. The default maximum port range is 1024 to 50,000. You can expand this range by adding the key `HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\Tcpip\Parameters\MaxUserPort` to 0xfffe. This would set the maximum open ports to 65,534.

If you can lower the amount of overhead associated with each web transaction, more data can be passed in every network exchange, thus improving the effective bandwidth throughput (web pages load faster). Windows 2000 has improved its TCP stack performance, but you can further enhance its performance by tuning the TCP window size, which is the amount of data received at one time on a TCP connection. Windows 2000 does dynamically tune this value and supports TCP windows scaling as needed; however, by default, the TCP window size is only 8KB. It is desirable to increase this value for 10BaseTX or greater performing networks to 17,520 by adding the following registry key: `HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\Tcpip\Parameters\TcpWindowSize`. To improve performance on high delay but reliable networks (such as satellite links or long-haul, intercontinent fiber runs), consider increasing the TCP WindowSize to 64KB. When a value of this

size is used, Windows 2000 negotiates the windows size using the TCP three-way handshake to find an optimal size. Having a larger TCP window enables more data to be sent before an acknowledgment is needed, again improving the effective bandwidth of the media in use.

It is important to note that TCB information is stored in the nonpaged memory pool. If the web server is experiencing memory bottlenecks and more memory cannot be allotted to the server, lower the above values shown in this section.

TUNING NIC USAGE FOR IMPROVED WEB SERVER RESPONSE TIME

Another way you can improve your web server response time is by optimizing the NIC and CPU relationship for web server workloads. Each web request received over the network generates an interrupt to the processor requesting service. If the processor does not find the request urgent enough (a high enough interrupt level), it will defer the request. This deferred interrupt request becomes a Deferred Procedure Call (DPC). As more and more requests come into the web server, the number of interrupts and DPCs increase. When an interrupt is sent to a particular CPU and it gets deferred, additional server overhead is incurred if this DPC is shipped off to another CPU in the server. This is Windows 2000's default behavior and can be costly from a performance perspective in high load situations. To stop this from happening, set the following registry value: `HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\NDIS\Parameters\ProcessorAffinityMask` to 0.

This forces the CPU that handled the interrupt to also handle any associated DPCs. This also ensures that the network interface card or cards are not associated with a specific CPU. This improves the CPU's servicing of interrupts and DPCs generated by the network interface card(s).

TUNING WINDOWS 2000 AND IIS 5.0 TO KEEP INETINFO.EXE WHERE YOU WANT IT: IN RAM

Regardless of whether you're providing an Internet- (WAN) or intranet- (LAN) based solution, networks introduce enough latency to slow down web requests to your users, so avoid adding any more in your web server solution. Accessing memory is tremendously faster than going to the disk drive to get what your web server might need. It's the difference between running 50 yards (memory access) to your refrigerator for your favorite cold beverage compared with running a 25-mile marathon (disk drive access) for the same item. Which would you prefer? With this in mind, there are several tuning activities we can complete to improve the chances that what we want is found in RAM.

IIS is a multithreaded application that runs as a single process instance, referred to as Inetinfo.exe under Windows 2000. Included in this process, IIS provides an object cache, which is a cache of objects associated with Active Server Pages (ASP), web services (http), FTP, and SMTP services. The object

cache is part of IIS's working set; i.e., the area in physical RAM that Inetinfo.exe occupies. So our goal is to keep Inetinfo's working set, and consequently as much of its frequently used objects in RAM as possible. Unfortunately, there are a few other activities occurring on your web server that may want memory space too. Every web connection provided by your web server takes up some RAM, and if the request references data residing on the web server's file system, then Windows 2000's file system is called into action.

HOW TO TUNE THE WINDOWS 2000 FILE SYSTEM CACHE FOR IIS 5

So what can one do to keep this important process in memory? It depends on how much memory you have in your server. If you have planned for the Windows 2000 file system cache to grow to its maximum size (432MB if you have 1GB of memory, 960MB if you have 2GB of memory), then select the "Maximize Data throughput for file sharing" Windows 2000 memory management strategy. How to implement this setting was covered in chapter 5, but as a reminder you can set the File System Cache strategy by right-clicking My Network Places, then selecting properties for your local area connection, then properties for File and Print Sharing for Microsoft Networks.

For example, in this web server solution scenario, 1GB of RAM was configured. Using this tactic, Windows 2000 will allow the file system cache to grow to its full size of 432MB, leaving the rest for other applications, specifically IIS 5.0, that will be tuned to leverage the available memory resources. This is where the information from sizing the memory subsystem reviewed above becomes invaluable, as you must know what is running on your web server and how much memory is taken before tuning the file system cache or IIS directly.

If, however, you are running a web server with less than 1GB of memory and plan to let the file system cache grow as needed and contend with other processes for memory resources, set the Windows 2000 memory strategy to "Maximize data throughput for network applications." Using this tactic, the file system cache will most likely not grow to its maximum size, as the working set of other applications will take priority over file system cache requests. This is a good approach to use if you do not have enough memory resources configured and your web server has other applications residing on it.

If you ever observe the working set of Inetinfo.exe get smaller, other applications' working sets get larger, the pages/sec increasing, and the pagefile growing (using the System Monitor—Sysmon), it is time to tune the system and possibly add more memory. When this situation occurs, parts of Inetinfo.exe are most likely being paged to disk and your web server will begin to run slowly. You can observe the working set of any processes under Sysmon's Process Object, select the process to track, and watch its working set counter.

OPTIMIZING IIS LARGE FILE, FILE CACHE USAGE

We have optimized the Windows 2000 memory subsystem, but what about IIS interaction with the file system cache? By default, IIS does not like to cache files that are larger than MaxCachedFileSize. If you have a web site that is working with larger files (greater than the default value of 256KB) and you do not find the file system cache effective, you will want to increase the size of files IIS will cache. This is accomplished by editing the following registry entry: `HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\InetInfo\Parameters\MaxCachedFileSize`. This will ensure IIS will cache the files that may be indicative of your larger, file-based web server environment.

OPTIMIZING IIS GENERAL FILE CACHE USAGE

How much of Windows 2000 file system cache will Inetinfo.exe use? By default, 50% of available memory. You can adjust this higher for dedicated web servers or lower it for servers that host multiple applications. Adjust this behavior by adding the MemCacheSize (MB) REG_DWORD to `HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\InetInfo\Parameters\MaxCachedFileSize`. If for some reason the file system cache is not working well at all, ensure that `HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\InetInfo\Parameters\MaxCachedFileSize\DisableMemoryCache` is set to 0. If it is set to 1, IIS's use of the file system cache will be disabled. You can observe the effectiveness of the file system cache from IIS's perspective by reviewing the Sysmon object Internet Information Systems Global File Cache Hits % counter.

Strategy: Tune the Web Server Software Engine to Use All Available Server Resources

IIS 5.0 has its own application-specific tunable areas. Our primary goal in tuning IIS is to ensure that it takes advantage of all available resources and does not limit performance from a web server engine perspective. For the IIS 5 application itself, we look at how we can tune it to optimally leverage each resource in the server. IIS 5 tuning is not relegated to the IIS snap in GUI interfaces found in the Microsoft Management Console (MMC) and registry (accessed with regedt32.exe and regedit.exe tools) settings alone. IIS 5 also includes a metabase of configuration information that is leveraged for its operations and optimization also. The IIS metabase can be accessed from the command line using the adsutil.vbs administrative utility that is located in the c:\Inetpub\AdminScripts directory structure. There is also a metabase editor named MetaEdit 2.1 that can be downloaded from Microsoft that is located at: http://support.microsoft.com/support/kb/articles/Q232/0/68.ASP?LN=EN-US&SD=gn&FR=0. This is a helpful tool for backing up your metabase and reviewing all of the different metabase entries and associated values.

GENERAL IIS 5 TUNING

Now that you have a handle on how Windows 2000's memory operations can be tuned for IIS 5 environments, the easiest memory tuning setting for IIS 5 itself is found under the IIS MMC snap-in by taking the following steps: Start / Programs / Administrative Tools / Computer Management / Services and Applications / Right Click and Select Properties for Internet Information Services (not a specific web site, the top level IIS bar / Performance tab).

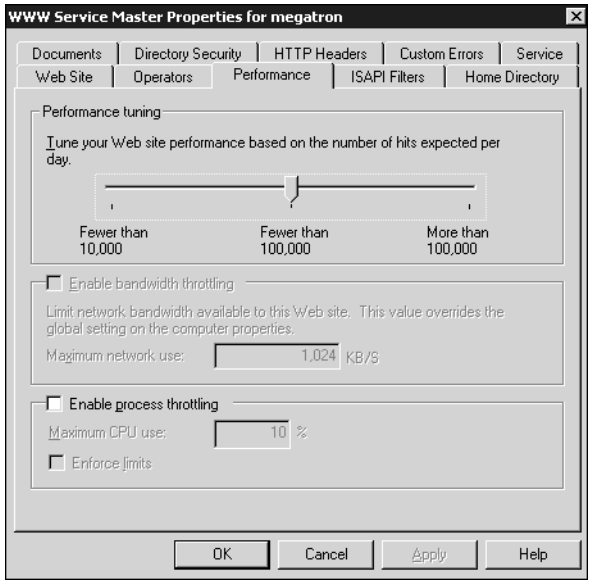

FIGURE 8–4 *IIS 5 General Performance Property Settings*

The settings in Figure 8-9 can have a significant impact on IIS 5.0 performance. The slider is used to set the expected number of hits in a day—fewer than 10,000, fewer than 100,000, or more than 100,000. The setting directly affects how much memory is set aside for connections. If you have enough memory resources, adjust it to its maximum value. If you begin to run low on memory resources, add more RAM or scale back its resource usage. This GUI-based tuning option is helpful and nice for the normal Windows administrator and will work well for most sites. If you are interested in more advanced topics, continue on!

So, now that you have all of this general web server tuning knowledge, let's look deeper into some additional specific IIS 5.0 optimization settings.

TUNING THE INETINFO.EXE OBJECT CACHES

The object cache resides inside of the Inetinfo.exe process and is used to store objects such as file handles, file directory information listings, and other frequently used objects that are expensive to retrieve. We can tune the size of Inetinfo.exe's object cache larger as needed so that we achieve a greater than 90% hit ratio.

The default value of the IIS object cache is 10% of memory, but you can set it to any value you want. If you have a large number of static web pages (which many people are beginning to use again, since serving them induces less overhead on your server), perhaps 320MB worth, you will want to set the IIS object cache to be larger. With a larger cache, you create more room in cache to support other services that reside in Inetinfo.exe, too. For example, perhaps we planned to provide 100MB of memory for the object cache. We adjust the object cache size by adding following the registry entry: `HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\InetInfo\Parameters\MemoryCacheSize=0x 5F5E100`.

Note that this value was added as a REG_DWORD using hexadecimal notation.

To determine if the IIS cache is large enough to be effective, check the Perfmon Counter Internet Information Services Global: Cache Hits %. How effective the cache is depends on the workload being placed on the web server. Typically, if the cache hit rate percentage is less than 90%, then the IIS object cache size should be increased. Remember, before increasing the IIS object cache size, ensure that there is sufficient available memory. You may encounter a greater challenge when trying to improve the cache hit rates for web environments that generate high amounts of dynamic web page content, unless they are generating the same dynamic content over and over. There are third-party tools we mention later in this scenario that focus on caching dynamic content when they are based on Active Server Pages.

If you run a large web site that serves many files, you can improve the performance of delivering these files by keeping their associated file handles with these files in the object cache. You can control how many cached file handle objects IIS 5 keeps in memory for file-related web service by adjusting (you may need to add this registry key) the OpenFileInCache registry entry located at: `HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\InetInfo\Parameters\OpenFileInCache`. Interactively increase this value 20% until you meet your objective. For a static environment or one in which the dynamic generated pages are low, this is a good tactic. As always, closely watch memory resource usage when implementing.

Any cache is more effective if, when the request is placed, the relevant data is already in the cache. Based on your web server customers' habits, you can tune the IIS 5 object cache to keep related web page information in its cache for longer periods of time. Note the converse of this can also happen, in which case you will want to increase the rate in which the IIS object cache

is flushed by lowering the ObjectCacheTTL value. To accomplish this, edit the following registry entry: `HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\InetInfo\Parameters\ObjectCacheTTL=xFFFFFFF`. This example keeps the IIS open descriptors active forever. Once information is in the cache, it stays until more information is needed and the old information is flushed out.

Track your successes with this tuning technique by monitoring Sysmon's Internet Information Services Global object's File Cache Hits %. & File Cache Flushes counters. If the rate of cache flushes is a high value and is associated with a high value of cache misses and page faults (Memory: Page Faults/sec), the cache may be flushing too quickly. When the cache is flushed too quickly, repeated requests for the same data do not result in a cache hit when it should. If this occurs, increase the ObjectCacheTTL (if you have not already set it to no time-out), MemoryCacheSize, and OpenFileInCache even more.

Tune Your Web Application to Use Available CPU Resources

CONTROLLING WHERE WEB SERVER APPLICATIONS (PROCESSES) EXECUTE

One of the most important CPU-related tuning you can complete is determining where your web applications run. Whenever possible, develop and implement your web applications with applications native to IIS; e.g., ASP and ISAPI as they are the most efficient in the IIS 5 environment.

As we investigated in chapter 4, less overhead is introduced into a system when context switches occur from within the same process, vs. two threads changing context, each from a different process. IIS 5 has three modes of web application process operations, which are shown in Table 8.43.

TABLE 8.27 *IIS 5 Application Process Operations Run Time Options*

Application Protection Level	Availability	Performance Level	Description
Low	Highest Risk	Highest	Web application processes are run in the same process as web services (Inetinfo).
Medium	Medium Risk	Medium	Web application processes are run in an isolated pooled process in which other web applications are also run.
High	Lowest Risk	Lowest	Web application processes are each run in isolated process separate from other processes.

By default IIS 5.0 runs web applications in an out-of-process state (Medium Application Protection level) so they cannot directly interfere with IIS operations and possibly crash IIS. This approach improves the likelihood that a poorly written web application does not adversely affect the Inetinfo.exe pro-

cess and subsequently IIS 5. However, this approach provides a lower level of performance, since context switch overhead is much higher for each web application since it involves a complete thread change from different processes. The default approach also consumes more memory. For maximum performance, run your well-written and tested web applications in the IIS process (Low Application Protection Level). Of course, there is always a trade-off: if your well-written web application panics while running in Application Protection Level Low, it will most likely cause IIS 5 to stop serving web pages!

You can tune which application protection level your web site runs on your web server either through the MMC IIS snap-in under the Home Directory Tab (shown in Figure 8-10) or via the command line using the IIS metabase which is accessed using the adsutil.vbs administrative utility located in the c:\Inetpub\AdminScripts directory structure or Metaedit GUI-based metabase editor.

It is important to note that, when making any changes to the IIS 5 metabase, those changes can be reflected across the entire web server or for a specific web site instance (you may host multiple web sites on one physical IIS 5 Web Server) that you are tuning! This is when the ability to manipulate the IIS 5 metabase really begins to shine. If you wanted to change the performance configuration of 200 web sites that your web server is hosting, do you really want to manually click through 200 screens? By using the command option with tools such as adsutil.vbs, you can change all 200 web sites, correctly, with one point and click! Once you have written and tested your script, of course.

TURN OFF DEBUGGING OVERHEAD FROM PRODUCTION SERVERS

Just in case your developers used some of the debugging features in IIS, you can ensure they are turned off and not adding CPU overhead (i.e., constricting IIS to a single threaded mode of operation) by setting the IIS 5 metabase variable disablesocketpooling to true. To set this variable, run the adsutil.vbs (Active Directory Services Interfaces ADSI) utility in the following manner:

C:\inetpub\adminscripts\cscript.exe adsutil.vbs set w3svc/ X/AppAllowDebugging False

The value of X corresponds to the web site on your web server on which you are turning debugging off. You will need to run this command for each web site—0, 1, 2, etc.—where you wish to turn debugging off.

Another debugging feature you can ensure is not running on your production web server is to set AspAllowSessionState to FALSE. When this is changed, commands must explicitly override this setting in pages that need to make use of the session object.

C:\inetpub\adminscripts\cscript.exe adsutil.vbs set w3svc/ AspAllowSessionState false

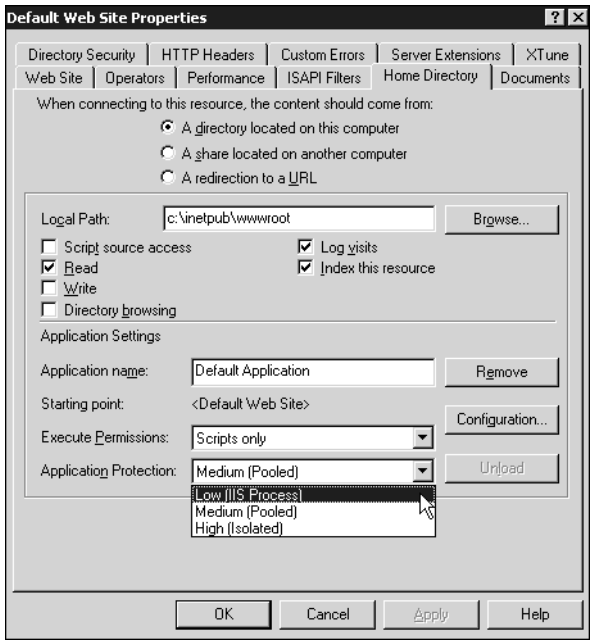

FIGURE 8–5 *Tuning the Application Protection (Performance Level) of an IIS Web Site (Note: These settings are implemented on a per web site basis, not web server basis. Here we edit the default web site)*

TUNING THE IIS QUEUE SIZE

CPU resources are not so inexpensive that you want them to sit around idle, so let's ensure they are put to good use. Under heavy web workloads, keeping IIS 5.0 properly fed with data is important. First determine if you have additional CPU capacity available by monitoring Sysmon's Processor object's %Processor Time counter and the System object's Processor Queue Length for the CPUs in your web server. Not all workloads are distributed well across all processors, so watch each processor's usage closely. If your processor usage looks relatively well distributed, and only 30% of processor resources are being used, consider increasing the workload that IIS will accept. To increase the workload IIS will accept, add the following registry value: `HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\Inet-Info\Parameters\ListenBacklog=0x1E`. This increases the maximum active connections that are held in the IIS queue from the default of 25, to 300 (150 per CPU to start). Whenever this queue length limit is reached, IIS will reject any new connections until the queue length shortens. Tune this queue length lower if a system bottleneck begins to occur and subsequently increase this value if web requests are being rejected and you have the CPU resources to

support more workload. Setting the ListenBackLog value high when you do not have the CPU resources available to support the increased workload will cause longer waiting periods for your web server's end users. This is sometimes worse than no response at all!

Based on this same concept, you can increase the number of IIS threads that are used to respond to web requests by adding the following registry entry: `HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\InetInfo\Parameters\MaxPoolThreads=20`. This registry entry specifies maximum network request threads per processor. The default is 10. Here we increased the value to 20, or 10 per CPU. Use extreme caution when tuning IIS; with all of these CPU specific options more is not always better. You must always ensure that you have sufficient processor resources, as it is possible to increase these values to high and actually lower your server's overall performance. If Inetinfo threads—which can be observed under Perfmon's processor object processor usage counter—are overworking your CPUs, lower the MaxPoolThreads and ListenBackLog values. Related to the MaxPoolThreads is the PoolThreadLimit, which relates to the number of threads that can be created in IIS. This variable is a hard limit and should always be greater than or equal to MaxPoolThreads.

Tuning IIS 5 for Common Gateway Interface (CGI) Operations

What can you optimize under IIS 5 if you are running CGI scripts? By default, IIS5 sets the maximum number of threads for CGI operations to four. Thus, only four CGI applications can run simultaneously. If your web server has the CPU resources to support additional workload (e.g., processor usage is low and there are no processor queues) increase the MaxPoolThreads value by 50%. MaxPoolThreads parameter specifies the number of pool threads to create per processor. Each pool thread watches for a network request and processes it. The MaxPoolThreads count does not include threads that are consumed by ISAPI applications. The MaxPoolThreads is located at `HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\InetInfo\Parameters\MaxPoolThreads`.

LOWERING OVERHEAD ASSOCIATED WITH SSL OPERATIONS

The initial creation of the SSL connection for a customer is very expensive in terms of CPU cycles. To help mitigate this you could turn SSL (this is not realistic for security reasons) off or try to lower how often the initial SSL connection must be created. This can be accomplished by lengthening SSL session time-out period. In this manner, if someone starts a SSL session, but works on something else for a short time, then continues to work with the SSL encrypted session, it still will be active and no new connection will be required. This is more art than science, as you must estimate how long you expect your customer to need the SSL session. You can control the SSL ses-

sion time-out period by editing the `HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\InetInfo\Parameters\ServerCacheTime` registry entry.

OPTIMIZING ACTIVE SERVER PAGE OPERATIONS

If you operate your web site using Active Server Pages (ASP), which is quite common, there are several ASP-related metabase settings that can have a significant impact on the performance and queuing of your web server, based on your environment and workload. This is almost like tuning inside of tuning. We tuned Windows 2000 to run IIS 5 well, then we optimized IIS 5 web server operations, then we looked into tuning how ASPs interact with IIS 5 and the rest of the system.

The following outlines the different parameters for ASP tuning grouped by functions and when they should be used. All of the parameters below are IIS 5 metabase parameters and are tuned using the adsutil.vbs utility or the metaedit metabase tuning tool. The standard syntax for implementing these parameters is:

C:\inetpub\adminscripts\cscript.exe adsutil.vbs set w3svc/ X/PARAMETER value, where X is the integer representing the web site you are tuning.

ASP Caching

Each process that hosts an ASP will have its own ASP template and script engine caches. By default, that is just one process because ASP applications run at Application Protection Level medium, but could be more depending on how your system is tuned. When an ASP gets a request for a page, it checks the ASP template cache first. If there's an instance of that page cached there, the request is forwarded to the script engine cache. If the requested page is not in the template cache, it is compiled into a template and forwarded to the ASP script engine cache. This avoids the cost of reparsing the template into byte code. If there is no script engine associated with the page, ASP takes the precompiled template from the ASP template cache, creates a new script engine and has it compile the template into byte code, and then executes it. The following three metabase parameters influence the ASP cache operations.

- The AspScriptEngineCacheMax parameter determines the maximum number of script engines to cache in memory. A hit in the script engine cache means that you can avoid recompiling the template into byte code. If you have a significant number of unique pages being generated, increasing this value should improve overall performance and lower CPU operations. The default value is 50. Increase at 100% increments and test for your specific environment.
- The AspScriptFileCacheSize parameter specifies the number of precompiled script files to store in the ASP template cache. If 0, no script

files will be cached. If –1, all script files requested will be cached. The default value is 256. Increase at 100% increments and test for your specific environment.

- The AspBufferingOn parameter default behavior allows all output from an application to be collected in the buffer before the buffer is flushed to the client browser. If this property is set to FALSE, output from ASP scripts will be written to the client browser as it becomes available. Ensure this property is set to true on all production web servers.

ASP CPU Optimization

Increasing Workload That IIS 5 Will Accept

IIS 5.0 dynamically changes the number of threads it operates in response to changing workloads if the AspThreadGateEnabled parameter is set to true. The maximum number of threads that IIS will allow per ASP process is AspProcessorThreadMax multiplied by the number of processors on your server. When dynamic tuning is activated, IIS performs thread gating which dynamically controls the number of concurrently executing threads in response to varying load conditions. When processor utilization drops below 50%—which could indicate that threads are blocked and waiting for an external database server to return the results of a query—IIS 5.0 increases the number of active threads so that other requests can be serviced in a timely manner. When processor utilization exceeds 80%, indicating a heavier load, IIS 5.0 deactivates threads to reduce the amount of context switching. Both lower and upper limits can be set: AspThreadGateLoadLow defaults to 50%, while AspThreadGateLoadHigh defaults to 80%. Try it. Set AspThreadGateEnabled parameter to true. For many environments it works well.

If your web server is under a heavy workload and accepts too much work, then many people get slow responses. If your web server accepts less work, some people get great responses while others get none. This simple example of providing web services illustrates the need to know your environment. If you have significant resources available, you should consider increasing the AspRequestQueueMax variable another 20% to enable more concurrent ASP requests to be serviced. AspRequestQueueMax specifies the maximum number of concurrent ASP requests that are permitted into the queue.

Dropping Distracted Customers: Managing ASP Connections

Would you want to service a request if a customer was distracted and discounted from your web server? Probably not. There is no reason to waste limited server resources. If a request has been in the queue longer than the

queue connection test time, the web server checks to see that the client is still connected before beginning execution. You can adjust the queue connection time by tuning the AspQueueConnectionTestTime parameter. Again, you must know your environment. If your web server typically generates web pages that take a long time to download, this parameter should be raised. If the opposite is true, consider lowing the value. Based on this same concept of connection time, consider disconnecting inactive customers if they are inactive. This is accomplished by tuning the ConnectionTimeout parameter that is 15 minutes by default.

Optimizing IIS for Web Publishing

If you support customers who run the Microsoft Front Page web-publishing application, you can optimize operations for those important customers, too. Don't forget that as you tune the web server for web publishing customers, you will be taking memory resources that IIS uses for delivering those published pages! Whenever possible, do not allow multiple direct publication of web content to highly loaded web servers. Instead, use a staging web server for development and then use other techniques—such as one-way replication software—to move data from the staging web server to the production web server. An added value of this staging server approach is that you can optimize the staging server for its mission and optimize the production web server for its associated mission.

IIS 5 provides a friendly interface to access the key tunables to optimize web publishing that are installed when you install server extensions. Access this GUI by selecting Start / Programs / Administrative Tools / Computer Management / Services and Applications. Right-click and select Properties for Internet Information Services (not a specific web site, the top-level IIS bar), then select the Server Extensions Table. Once you follow these steps, Figure 8-11 is presented.

From this interface, you can select the number of web pages that your web site supports. In general, select a range of pages that matches your environment and you are off like a herd of turtles! This typically works, but you will want to understand how the standard settings really map to the IIS 5 metabase if you want to truly optimize your web server publishing services.

If you would like to customize your tuning even more than the standard settings provide, or learn which IIS 5 metabase values are changed, select the upper settings box shown in Figure 8-11. This will provide a detailed performance screen shown in Figure 8-12.

The following is a description of what each IIS 5 metabase tunable does to support web publishing and how the standard settings preset options map to the custom IIS metadata variables. These definitions and variable settings

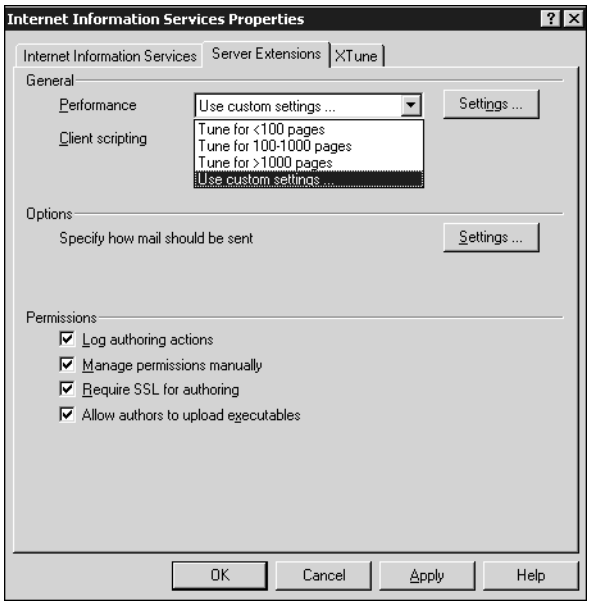

FIGURE 8–6 *Tuning IIS to Support Web Publishing Customers*

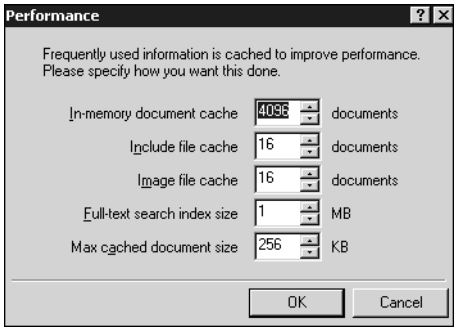

FIGURE 8–7 *Custom Tuning IIS to Support Web Publishing Customers*

are based on information obtained from the Microsoft Q article found at: http://support.microsoft.com/support/kb/articles/q201/8/03.asp.

In-Memory Document Cache (IIS 5 Metabase Value: CacheMaxDocMeta)

CacheMaxDocMeta represents the maximum number of documents (property information, such as link maps and Web parameters) that can be held in the cache. For example, if you specify 4,096 documents, when the 4,097th

document is read, then the cache clears itself and only includes the 4,097th document. The cache will then increase in size as documents are read and clear itself again after the 4,096th limit is reached again. If you have the available memory, it is better to have these settings slightly larger than is projected in case your estimates are incorrect. A complete cache flush will slow down other web publishers working with your web server.

When you use the preset options, you set the values shown in Table 8.44.

TABLE 8.28 *Preset Values for CacheMaxDocMeta*

Preset	CacheMaxDocMeta
Tune for less than 100 pages	Predefined for 4,096 (documents)
Tune for 1 to 1,000 pages	Predefined for 4,096 (documents)
Tune for greater than 1,000 pages	Predefined for 16,384 (documents)

Include File Cache (IIS 5 Metabase Value: CacheMaxInclude)

CacheMaxInclude represents the number of files that you want to keep available in memory for inclusion in other files. For example, there might be header, footer, and copyright files that you want to include in some or all of a web site's web pages. The predefined setting is set for 16. Consider increasing if you have more than 16 files that are used in the development of other web pages.

TABLE 8.29 *Preset Values for CacheMaxInclude*

Preset	CacheMaxInclude
Tune for less than 100 pages	Predefined for 16 (documents)
Tune for 1 to 1,000 pages	Predefined for 16 (documents)
Tune for greater than 1,000 pages	Predefined for 16 (documents)

Image File Cache (IIS 5 Metabase Value: CacheMaxImage)

CacheMaxImage is the number of image files in memory that the FrontPage Server Extensions can use to create layered pictures in web pages. For example, one file may consist of a background, another of a Navigation button. The server extensions can compose a picture by adding the background to a web page and then overlaying the Navigation button image. The picture can be composed faster if the component files are in a cache. Adjust to match your development environment.

TABLE 8.30 *Preset Values for CacheMaxImage*

Preset	CacheMaxImage is set to
Tune for less than 100 pages	Predefined for 16 (documents)
Tune for 1 to 1,000 pages	Predefined for 16 (documents)
Tune for greater than 1,000 pages	Predefined for 16 (documents)

Full-Text Search Index Size (IIS 5 Metabase Value: TextMemory)

TextMemory is the maximum amount of disk space that can be allotted for storing a full-text search index. When this amount is reached, no other pages on the web site can be indexed (unless you increase this number). If you have large web sites, increase this value (start with 10MB) as full-text indexes can improve overall performance as long as they do not grow too large.

TABLE 8.31 *Preset Values for TextMemory*

Preset	TextMemory is set to
Tune for less than 100 pages	Predefined for 1 MB
Tune for 1 to 1,000 pages	Predefined for 2 MB
Tune for greater than 1,000 pages	Predefined for 4 MB

Max Cached Document Size (IIS 5 Metabase Value: CacheMaxIncludeSize)

CacheMaxIncludeSize is the maximum size of a document that can be stored in memory. This size limit applies to include files, image files, and other files that may be stored in a cache. Adjust to match your development environment. If you work with 100KB documents, this is a great setting. If you work with 300KB files, tune!

TABLE 8.32 *Preset Values for CacheMaxIncludeSize*

Preset	CacheMaxIncludeSize
Tune for less than 100 pages	Predefined for 256K Windows NT and UNIX, 32K Windows 95/98
Tune for 1 to 1,000 pages	Predefined for 256K Windows NT and UNIX, 32K Windows 95/98
Tune for greater than 1,000 pages	Predefined for 256K Windows NT and UNIX, 32K Windows 95/98

Thinking Outside of the Box: Xtune

There are a lot of potential tunables in this section, both in the registry and in the IIS 5 metabase. Web servers are a dynamic environment and fill many roles. Is there a tool to track all of these tunables and make life simpler? Yes, there is one that covers many of the most important tunables. Xtune from Post Point Software (http://www.PostPointSoft.com) provides a very friendly and free tool for observing many of the registry and IIS metabase tunable parameters reviewed in the IIS 5 tuning section and even provides some recommendations. Screen shots of this tool are shown in Figures 8-13 and 8-14.

Windows 2000 does a good job of setting these tunables for general environments, but tools like Xtune make life a little easier when working in more diverse, performance-challenged environments. Remember, every environment is different. Knowing your environment, workload, server design, web server applications, and how and why Windows 2000 and IIS operate as they do provides you a distinct advantage in improving your web server's performance. Always test your ideas for your specific environment to determine what is best for you.

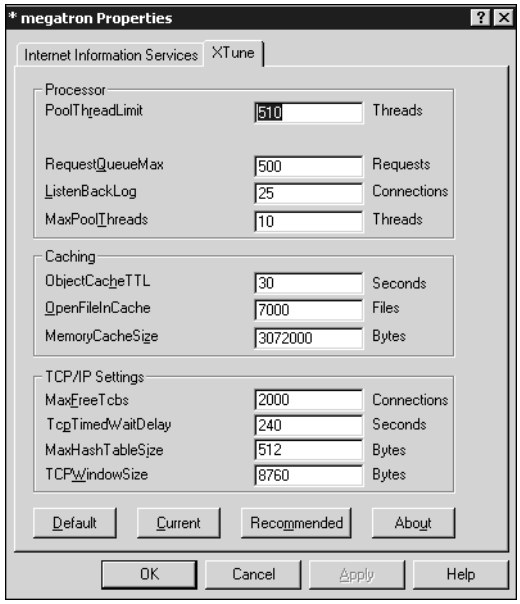

FIGURE 8–8 *Automating Some of the Key Windows 2000 Registry IIS 5.0 Web Server Tuning Options (Note: These are global web server settings that affect the entire web server. These tuning changes are not completed on a web site-by-web site basis.)*

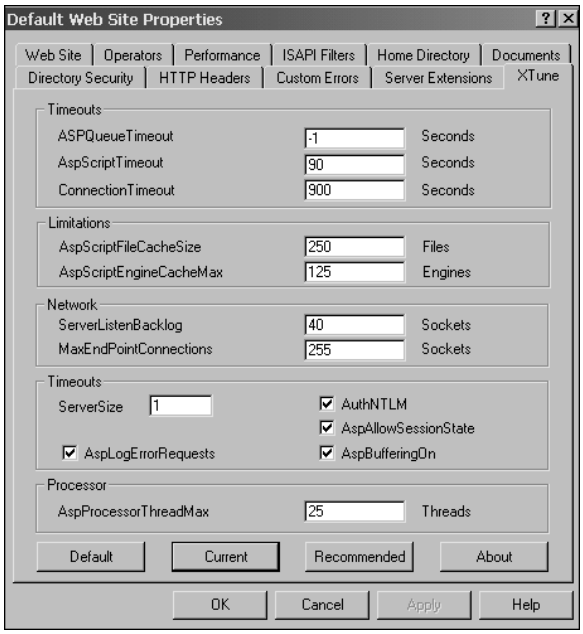

FIGURE 8–9 *Xtune Automating Some of the Key IIS 5.0 Metabase Server Tuning Options (Note: These settings are implemented on a per web site basis, not web server basis)*

Literally Thinking Outside of the Box: Network Load Balancing

What can you do if you have a perfectly sized and tuned Windows 2000 web server and you still need more performance? You could purchase a larger web server or you could implement Network Load Balancing (NLB). NLB is a new technology included in Windows 2000 Advanced Server that enables the implementation of an active-active, shared-nothing web cluster server solution that supports up to 32 nodes. Active-active refers to the fact that all nodes in the cluster provide web services; no node sits idle. Shared-nothing cluster refers to the fact that no servers directly share their disk subsystems to another. NLB is a software-based solution and works by installing on each web server a virtual NIC that load balances the workload across all web servers configured into the virtual web server. One IP address is advertised to the Internet. When the web server request arrives, it is sent to a web server in your cluster based on your configuration settings. Here we do not go into each step needed to install NLB, but Windows 2000 Help provides easy to

follow directions. NLB is installed as a network driver under Start / Settings / Network and Dial-Up Connections / Local Area Connection / Properties. This approach provides a method of scaling your web server solution to meet your performance needs by just adding additional servers that provide a single managed interface to your customers. Note one odd occurrence in NLB environments. If your server is reachable over the network—e.g., you can ping.exe the server—NLB will forward requests to it even if your web application has stopped serving web pages! Although a rare occurrence, it is something you should be aware of. Custom solutions to this challenge are available, but I have yet to see any official third-party products for this special case.

All of the tuning techniques and sizing techniques discussed are directly applicable to NLB environments. The first challenge in NLB environments is to ensure that the web content stays identical on each web server in the cluster. To accomplish this, you can either manually copy files as needed or use software replication over the local area network, ensuring that when one web server is updated, the other web servers in the cluster are automatically updated too. The second challenge is ensuring that no back-end servers that the front-end web server cluster relies on for processing activities such as database services create a bottleneck.

Windows 2000 IIS 5 Web Server Tuning Summary

For the hands-on specifics of why, when, and how these tuning recommendations work in detail and steps to implement them, refer back to the subsequent tuning sections of this section and the previous chapters. Here select information is provided to give you a jump-start into an optimized Windows 2000 IIS 5 web server solution!

IIS WEB SERVER HARDWARE TUNING

- Update the BIOS and Windows 2000 drivers on all server hardware (includes motherboards and the associated adapters.)
- Ensure BIOS is configured for maximum performance.
- Ensure BIOS performance settings match the hardware you have installed.
- Ensure you received the correct hardware.

WEB SERVER APPLICATION TUNING

- Use native application development environment for IIS 5 such as ISAPI and ASP.
- Use static and thin (low KB) web pages whenever possible.

- Limit the use of SSL for when it is absolutely necessary.
- Tune Windows 2000 TCP/IP Stack to match your web workload.
- Tune IIS 5 operational behavior to maximize use of CPU resources and lower overhead.
- Tune IIS 5 operational behavior to maximize use of memory resources to improve caching.
- Ensure the IIS 5.0 Web Server process Inetinfo.exe so it is never paged to disk.
- Configure your web site applications to run in the same process as Inetinfo.exe by setting the Application Protection level to low (ensure these are stable applications).
- Consider Network Load Balancing to scale your web site as needed.
- Follow the tuning and sizing methodologies from chapters 2 and 3.
- Review the Web Server Specific Application Tuning section above for details on these steps.

WINDOWS 2000 CPU SUBSYSTEM RESOURCE TUNING

Key disk subsystem–related tuning you should consider:

- Set the application response optimization to background services (Control Panel / System / Advanced / Performance Options).
- Off-load CPU operations to I2O-enabled I/O adapters.
- If Secure Socket Layer (SSL) encryption is in use, consider obtaining an I/O card that off-loads all SSL calculations from the main CPUs or plan for the additional CPU overhead by monitoring your server closely.
- Since multiple CPUs are in use with Multiple NIC and SCSI adapters, consider using the Microsoft Interrupt-Affinity Filter Control Tool (IntFiltr; see chapter 4) to ensure interrupts are properly distributed between NICs and CPUs.
- Do not use a 3GL or other fancy screen saver.
- Check all services running on your server. Operate only those you need and stop all others.
- Avoid running another CPU-intensive application, such as a database, on your web server.

WINDOWS 2000 MEMORY SUBSYSTEM RESOURCE TUNING

Key memory-related tuning you should consider:

- Check the Windows 2000 memory management strategy. IIS Web Server relies heavily on the File System Cache. If you have over 1GB of RAM in your IIS web server, use the system cache setting Maximize Data Throughput for file sharing. With 1GB of memory in the server, Windows 2000 will commit 432MB for the system cache. The remaining memory can then be used by IIS. However, if you have less than

1GB of memory in your IIS Web server, configure Windows 2000 system cache algorithm to Maximize Data Throughput for network applications. These options are selected under: Start / Programs / Settings / Network and Dial-Up Connections / Properties of your LAN adapter / File and Print Sharing for Microsoft Networks.

- Check all services running on your server. Operate only those you need and stop all others.
- Ensure the pagefile is set to twice the size of physical RAM in your server to ensure physical memory can be committed to action.
- Set the pagefile size to twice the amount of RAM in your server and to a fixed size.
- Never place the pagefile on a RAID 5 array.
- If your system must page, consider multiple pagefiles on physically separate disk drives which are all configured to the exact same size.
- Force Windows 2000 to keep the Windows 2000 executive (kernel) from paging any of its executive system drivers to disk and ensure that they are immediately available if needed.
- Do not use a 3GL or other fancy screen saver.
- Do not use exotic wallpaper on the server console; it wastes memory resources.

WINDOWS 2000 DISK SUBSYSTEM RESOURCE TUNING

Key disk subsystem–related tuning you should consider:

- When formatting the disk drive with NTFS, select the ALU size that best matches your disk workload characteristics. For this scenario, use an 8KB ALU size for all disk subsystems.
- Match the stripe size of the low-level RAID format with the ALU size used (see chapter 6 for details).
- Since we are using large NTFS volumes, increase the size of the NTFS log file size to 64MB, by using the *chkdsk* command: chkdsk drive-letter: /l:65536.
- Turn on write-back caching and read-ahead caching on hardware RAID adapters.
- Use only I2O-enabled RAID adapters and those with their own CPUs.
- Keep storage capacity on the disk subsystem under 60–80% so that the fastest portion of the disk drive is used and so that there is room on the disk drives to perform defragmentation.
- Defragment your disk subsystem on a regular basis; this will result in significant performance improvements!
- Evenly distribute disk workload across the physical disk drives in the server.
- If DOS file names are not required (i.e., legacy DOS-based desktops are not in use), disable short name (8.3) file generation. Note: To disable short name generation, use regedt32.exe to set the reg-

istry DWORD value of 1 in the following Registry location: HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Filesystem\NtfsDisable8dot3nameCreation. More information on this technique in chapter 6.

- Disable last access updates. `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\FileSystemNtfsDisableLastAccessUpdate`. Changing the default REG_DWORD value of this key from 0 to 1 will stop Windows 2000 from updating the last access time/date stamp on directories as directory trees are traversed.
- Understand the characteristics of all RAID levels and leverage different RAID levels as needed in the same web server solution.
- Use more than one RAID level in your IIS solution!
- Group similar disk activities on the same physical disk subsystem and do not separate them via logical partitions on the same physical disk drive/array.
- Use one Windows 2000 partition per physical disk device/array.
- Group similar disk activities on the same physical disk subsystem and do not separate them via logical partitions on the same physical disk drive/array.
- Add disk drives as needed to keep the sec/transfers low and average queue size less than twice the number of disk drives in the disk array.
- Check all services running on your server. Operate only those you need and stop all others.

WINDOWS 2000 NETWORK SUBSYSTEM RESOURCE TUNING

Key network subsystem–related tuning you should consider:

- If SSL sessions are in use, expect more bandwidth to be needed due to the encryption overhead. More of the data stream is taken up with encryption data; thus either a larger network connection is needed or more time to deliver the encrypted data. Expect at least a 10% increase.
- Team/trunk multiple NICs (Ethernet channels) together as needed to add network bandwidth and availability to your IIS Web Server (seamlessly to your customers).
- Remove all unnecessary protocols and Redirectors.
- Standardize on one network protocol: TCP/IP.
- If more than one network protocol is in use, set binding order such that the most commonly used protocol is first on the list.
- Tune the NIC driver to off-load IPSEC security operations and TCP operations on the NIC itself instead of the main system CPUs.
- Tune the NIC driver to increase its send and receive buffers.
- Use only I2O-Enabled NICs.

- Use the fastest I/O bus-enabled NIC possible (64-bit, 66MHz PCI slot).
- Tune the Windows TCP/IP stack to your environment (TCP Hash Table, MTU Window Size) as per the guidance above for IIS network tuning.
- Do not use autodetect for your NIC. Set it to the best performance level the internetwork devices will support. For example: 100BaseTX full duplex.

THIRD-PARTY TOOLS FOR TUNING WINDOWS 2000 IIS WEB SERVERS

Key third-party tools to consider for enhancing the performance of your SQL environment:

- Obtain a third-party disk defragmentation tool that provides enhanced defragmentation features such as scheduling and support of multiple NTFS ALU sizes. Recommended vendor to consider: Norton Speed Disk http://www.Norton.com.
- If, after analyzing your system, you find that specific web server files are being used heavily or another area of the disk subsystem is used heavily such as c:\temp, consider implementing a RAM disk. RAM disks create a file system in memory, so from the application's perspective, it just sees a logical disk drive, such as "r:" to interact with, so no application changes are needed. When applications interact with the RAM disk, they run a lot faster than they do on a physical disk drive. The trade-off for this performance boost is the risk of data loss, so do not use this technology for critical data. This technology was investigated in chapter 6 and the resulting tests looked fantastic—performance improvements greater than 200%. RAM disks are not native to Windows 2000. The RAM disk tested was a product from Super Speed Systems (http://www.eecsys.com) obtained through SunBelt Software (http://www.sunbeltsoftware.com).
- To assist in tuning the many registry settings associated with IIS, use the free Xtune utility software tool from http://www.postpointsoft.com. This tool provides a GUI front end to the most common (not all) and beneficial registry tuning options for IIS and also provides recommended settings based on your environment. A very helpful and easy tool to use.
- To improve cache-hit ratios of ASPs, consider using the Xcache product. This product works to improve the caching of ASP-based web requests. When effective, this product lowers the number of back-end database queries and local ASP processing load. Xcache is available for download at http://wwww.postpointsoft.com. From my testing, this product looked promising.

Web Server Solution Scenario Summary

In this scenario, we focused on sizing and tuning of your Windows 2000–based web server from a system perspective to improve overall performance. It is important both to understand the type of web content you are providing and to closely monitor your web server resources. Armed with this information and knowing what tuning options are available, you are empowered to make informed decisions for maximizing the performance of your web server. Windows 2000 Server provides a very capable web server platform; now you can truly take advantage of its capabilities!

Summary

In this chapter, the tuning and sizing methodologies and strategies discussed in chapters 2 and 3 are applied to develop real-world solutions with specific step-by-step, hands-on recommendations and guidelines. Using sound and repeatable methodologies when developing your solution will give you more confidence that the solution will meet your requirements. These case studies used the various tuning and sizing concepts presented in chapters 4–7 in conjunction with new, application-specific optimizations to develop the overall solutions. It becomes much easier to intelligently size and tune Windows 2000 and back office solutions when you have an understanding of what, how, and why operations are occurring in the system from both a software application and hardware perspective. These solution scenarios provide you with a template to size and tune your own solutions but, as always, remember that they are guides! Your environment and experiences are unique, so customize as needed. These solution scenarios get you into the game. From there it is up to you!