GUS Displays Call-up Tuning

Last Revision Date: 12/2/97
Version: 1.0

1. Definitions, Terms and Acronyms

Before diving into any Performance Tuning, it is necessary to go over some definitions and terms.

GPB GUS Picture Builder/Runtime (a.k.a. GUS Display process), an application for creating and executing process control displays that accesses both LCN and HCI data.
HOPC Honeywell OLE Process Control, a subsystem of the GUS System providing LCN data access based on OLE framework.
LCN Honeywell Local Control Network

GUS Display Call-up Time The elapsed time that is measured from GUS Display invocation action until the invoked Display fully propagate all LCN Data Access references.

Task

For the purpose of this paper, the word **task** is referred as a series of computer instructions, the execution of which involves work to be performed by one or more GUS components or resources (for example, GPB, HOPC, LCN, or even CPU, memory, hard disk, and network adapters).

The amount of time it takes to complete a task can be divided up among the several GUS components and system resources. Some of task's resources will be involved and responsible for small amounts of the total time, others will be responsible for larger amounts.

Bottleneck

The single resource that consumes the most time during a task's execution is that task's **bottleneck**. Bottlenecks can occur because resources are not being used efficiently, resources are not being used fairly, or a resource is too slow or too small.

One thing to always keep in mind, especially in a multitasking OS like NT, is that ***resolving one bottleneck will always lead to the next one.*** For example, by boosting GUS Display (gpb) process too much, it will slow down the performance of GUS Data Access (HOPC) and therefore, the overall performance of GUS Display call-up maybe slower.

2. GUS Application Tuning

The goal in tuning GUS is to determine what resource is experiencing the greatest demand (bottleneck) during Display call-up, and then adjusting the operation to relieve that demand and maximize total throughput. A GUS system should be structured so that its resources are used efficiently and distributed fairly among the GUS displays. This is not as difficult as it sounds, assuming you use a few good rules/guidelines and have a thorough understanding of the computing environment.

2.1 GPB Boost Duration

Overview

Boost duration registry value indicates the number of seconds for selected threads within GPB process remained at the highest priority (ie. THREAD_PRIORITY_HIGHEST setting) during a GUS Display call-up and shutdown.

The default value of this registry value is 10,000 msec or 10 seconds in decimal.

Recommendation

If users have a pool of GUS Displays and the call-up time is varied from n to m seconds (where m is the longest call-up time), then this value should be set within the (m-x) second range where x is in the range of 0-3 seconds. Here are few suggestions that reflect the dependency of value x on the nature of GUS displays:

- If GUS displays contain a lot of Data Access references, such as more than 400 LCN point parameters which are located within multiple group Ids and collection rates. In this case, x should be in the range of 2-3 seconds.
- If GUS displays do not contain many Data Access references, but rather have multiple complex graphic drawing objects (eg. animation objects, trend OCX, etc), then x should be in the range of 0-1 seconds.
- Leave at the default setting if GUS Display call-up is satisfactory.

Refer to Appendix 1 for more details of how to modify this registry setting.

2.2 Data Change Wait Limit

Overview

During GUS Display call-up, the “Data Change Wait Limit” registry value specifies the number of seconds in delaying activation of Data Change thread (within GPB process) till all LCN Data Access have been read. In this case, it delays the drawing until all LCN Data Access references have been read. Hence, this reduces multiple interim drawing tasks occurred at start-up. Users may be expected to see “white screen” window shows up longer than the default setting but the overall Display call-up time will be more likely reduced.

If the GUS Display call-up is satisfactory, users can leave it at default setting which is 0 seconds.

Recommendation

If users have a pool of GUS Displays and the call-up time is varied from n to m seconds (where m is the longest call-up time), then this value should be within the (m – z) second range where z should be in the range of 0-2 seconds.

Refer to Appendix 1 for more details of how to modify this registry setting.

2.3 LCN Data Collection Settings

There is some performance trade-off for a display call-up when having too much variety of collection rates and Group Ids. This means the higher variety of collection rate and group ids in a GUS Display, the slower the display call-up is.

In general, transactions between devices is usually much more expensive than the size of each transaction. Therefore, the higher the number transactions between HOPC and LCN, the slower the GUS Displays call-up.

Hence, normally all static parameters of the same Data owner should be set at the Collection Rate equal to zero (ie. only collecting once when Display start-up) and should be all in one group ID. However, if there are already LCN parameters configured to be collected from same set of LCN points then their few static parameters (eg. name, ptdesc) should be regrouped together in the same group ID and collection rate (even not zero). This will allow to improve performance on GUS display call-up. While doing this, GUS Displays still do not compromise any performance hit for subsequence Data Change Collections.

Please refer to section **Fine Tuning Performance** in **GUS Authoring Tutorial** document for more information about this topic.

2.4 OnDisplayStartup scripts

Reducing the amount of executed time consuming work in OnDisplayStartup scripts will improve the performance GUS Display call-up.

Please refer to section **Scripting Events Properly** in **GUS Authoring Tutorial** document for more information about this topic.

3. NT system Tuning

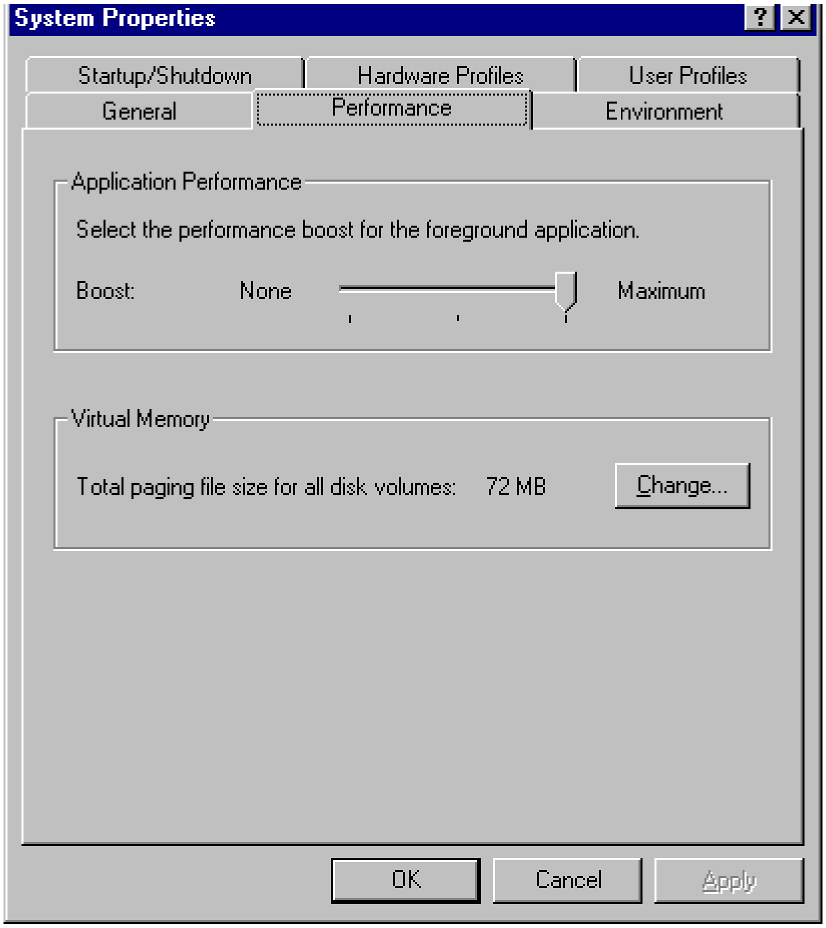
3.1 Select the performance boost for the foreground application

On an NT system, a foreground application is defined as the application having the active window appeared at the top of the screen (relative to a background application). Each application (as well as each thread) in the system has a set priority.

Recommendation

Setting the performance boost for the foreground application to maximum. This allows the foreground application receives the most processor time, and so responds better, than applications running in the background. As a trade-off, the background application may react slower than before.

You can control the priority system-wide by changing the following in Control Panel | System | Performance.



Use this dialog box to change the relative responsiveness of applications that are running at the same time. When more than one application is running in Windows NT, by default the foreground application receives more processor time, and so responds better, than applications running in the background.

3.2 NT page file setting

Assuming that all pct files are located in one physical drive.
Relocated system virtual page file to separate physical (not logical disk partition) drive that is the least busy. This will improve the disk access for GPB Display call-up.

Refer to Appendix 2 for more details of how to set up this NT virtual page file.

Appendix 1

Here is the procedure the change these registry values:

<u>WARNING:</u> Using Registry Editor incorrectly can cause serious, system-wide problems that may require you to reinstall Windows NT to correct them.

1. Logon the machine with administrator privilege.

2. Start Registry Editor (REGEDT32.EXE) and locate the following Registry sub-key in the HKEY_LOCAL_MACHINE sub-tree:

   \SOFTWARE\Honeywell\GPB

3. Select the following value name: Boost Duration

4. From the Edit menu, select DWORD.

5. Click on the Decimal Radio button and enter your decimal value in msec, eg. 5000 (ie. 5 secs).

6. Click on OK button.

7. From the Edit menu, select Add Value.

8. In the Value Name edit box, type in: Data Change Wait Limit
Remember: you must type in exactly the registry Value Name as stated.

9. For Data Type, choose REG_DWORD

10. Click on OK button.

11. Select the following value name: Data Change Wait Limit

12. From the Edit menu, select DWORD.

13. Click on the Decimal Radio button and enter your decimal value in msec, eg. 5000 (ie. 5 secs).

14. Close the Registry Editor application.

15. Log off and log on the machine.

Remember: you must log off and log on the machine before the new settings will take effect.

APPENDIX 2

NT Page file Configuration

1. From Start/Programs/Administrative Tools launch the Disk Administrator application.

2. Select the second physical drive.

3. Using Right Click and select Create partition (Max size), follow by commit changes.

4. Format the newly created partition using NTFS.

5. Close the Disk Administrator application.

6. From Start/Settings/Control Panels/System, select Performance tab, follow by Change button.

7. Create the new page file on the new partition by entering a value equal to 3 times the size of the RAM size on machine for both Initial and Maximum Size. Commit the changes by using Set button.

8. Select the old page file on C: drive and delete it. Ignore the warning message (yes to continue).

9. Reboot machine for new Changes to take affect.

APPENDIX 3

Version 1.4

Presented by: Scott B. Suhy

Senior Consultant with Microsoft Consulting Services, responsible for enterprise architecture, design, and optimization for Fortune 500 companies.

E-mail Scottsu@microsoft.com

Baseline

Introduction

Would it not be nice if there were no traffic ***bottlenecks*** during your everyday ***task*** of going to work? No traffic lights, fender benders, car problems, detours, people pulling out in front of you, people in the left hand lane going less than the speed limit, four lane highways narrowing down to two lanes.... This is rather unrealistic, just like with a computer system it is unrealistic to expect at some point in time there will not be a limit to the amount of memory, CPU, or I/O being consumed by internal or external processes.

You might also say that it might be nice to know how long it was going to take you to get to work in the morning (*with some expected normal variation*). Users of a computer system have the same expectation. They expect their jobs to finish in an acceptable amount of time without bottlenecks in the system slowing them down.

If there were bottlenecks on your way to work each day, I suppose you could ***optimize or tune*** the trip (reduce bottlenecks) by possibly finding an alternate route, car pooling, taking advantage of a car pool lane, taking a bus, or even changing your working hours (possibly to the evening when there is no traffic and the only thing keeping you from getting to work any faster is the speed limit and possibly the size of your engine). Computer systems have the same optimizations (run jobs during off peak hours, etc.). As with transportation systems, there is also the same lack of environmental control with a computer system. For example, it is not realistic to think that there will always be the same amount of traffic (on the road or in your computer system), it's also not realistic to think that you have control over the traffic (on the road or in your computer system). Problems always occur (a rain storm causing increased slowdowns on the road or one user consuming a great deal of the bandwidth of the server's memory, CPU, or I/O). Managing, as well as expecting, the problems, and knowing what to do when they occur is the key.

Once you feel you have the trip optimized, you might also think about taking some statistics, daily, weekly, or monthly, such as the amount of time it takes you to arrive at the office, number of *red* lights you got rather than green, and so on. This type of information will allow you to make future decisions on such things as "If I stop to get gas in the morning, how much earlier will I have to leave the house?" Of course you would also have to know how much time it would take you at the gas station (another set of statistics). The same thing goes for your computer system. It's called Capacity Planning.

The following information provides you with tips on areas of the Microsoft® Windows NT™ operating system in which you should pay attention (What to Watch). It also gives you a few rules/guidelines to use to optimize the system (What You Can Do). Once you take each of these areas into consideration, your system should be optimized. Once you feel your system is optimized it is then time to gather data on current capacity. The data will allow you to do the following:

- Project how much the workload at the memory, CPU, I/O, and bandwidth levels will increase in response to business growth and new Microsoft BackOffice applications.
- Diagnose problems by comparing subsequent measurements.

This information is rather technical in nature and assumes that you already know a great deal about Microsoft Windows NT™ Workstation and Microsoft Windows NT Server operating systems. However, it only touches the surface of optimization. Many books could be written on the subject. Consequently, this paper neglects to explain many details and assumes you know where to get information about the hardware and software concepts mentioned. If you stumble upon a concept that is not explained in detail, you may want to refer to the Microsoft Windows NT Resource Kit, Server Message Block specification (which can be obtained from Microsoft), Microsoft TechNet, or any book that details network architecture (such as the book *Local Area Networks* by James Martin or *LAN Times Encyclopedia of Networking* by Tom Sheldon).

Definitions

Before diving into any Performance Tuning, it is necessary to go over some definitions and terms.

Task

For the purpose of this paper, I refer to the word **task** as a series of computer instructions, the execution of which involves work to be performed by one or more computer components or resources (for example, CPU, memory, hard disk, and network adapters).

The amount of time it takes to complete a task can be divided up among the several resources that are involved in the task's execution-some resources will be responsible for small amounts of the total time, others will be responsible for larger amounts.

Bottleneck

The single resource that consumes the most time during a task's execution is that task's **bottleneck**. Bottlenecks can occur because resources are not being used efficiently, resources are not being used fairly, or a resource is too slow or too small. Let me try to elaborate on this point with the following example.

Example. *If a task takes 2.2 seconds to complete, with .2 seconds spent executing instructions in the CPU and 2 seconds retrieving data from the disk (assuming both are not overlapping in time), the disk is the bottleneck in the task. If the CPU were replaced with one twice as fast, task execution time would drop from 2.2 to 2.1 seconds. This would be approximately a 4.5% increase in productivity. However, if the disk controller were replaced with one twice as fast, it would drop the disk access time from 2 seconds to 1 second, dropping the total execution time from 2.2 to 1.2 seconds. This would be approximately a 45% increase in productivity.*

It would be easy if the previous example were on a workstation running the Microsoft MS-DOS® operating system, but we are dealing with a multitasking OS. One thing to always keep in mind, especially in a multitasking OS, is that ***resolving one bottleneck will always lead to the next one.***

Windows NT System Tuning

The goal in tuning Windows NT is to determine what hardware resource is experiencing the greatest demand (bottleneck), and then adjusting the operation to relieve that demand and maximize total throughput. A system should be structured so that its resources are used efficiently and distributed fairly among the users. This is not as difficult as it sounds, assuming you use a few good rules/guidelines and have a thorough understanding of the computing environment. For example, in a file and print server environment, most of the activity at the server is in support of file and print services. This tends to cause high disk utilization because of the large number of files being opened and closed. It also causes the network interface card(s) to endure a heavy load because of the large amount of data that is being transferred. Memory typically does not get a heavy load in this environment (memory usage however can be heavy due to the large amount of system memory that may be allocated to file system cache). Processor utilization is also typically low in this environment. In contrast, a server application environment (for example, other Microsoft BackOffice products such as Microsoft SQL Server™ database server for PC networks, Microsoft Exchange electronic mail system, Microsoft Systems Management Server centralized management for distributed systems, and Microsoft SNA Server) is much more processor and memory bound than a typical file and print server environment because much more actual processing is taking place at the server. The disk and network tend to be less utilized, due to a smaller amount of data being sent over the wire and to the disk. Understanding these generalizations is not enough; the only way to get an idea of the utilization of the resources is to monitor them, and one of the most powerful tools that you can use is the Windows NT Performance Monitor.

Performance Monitor is a graphical tool for measuring the performance of your own Windows NT-based computer or other Windows NT-based computers on a network. It is located in the Administrative Tools group of both the Windows NT Workstation and Windows NT Server products. On each computer, you can view the behavior of objects such as processors, memory, cache, threads, and processes. Each of these objects has an associated set of **counters** that provide information on such things as device usage, queue lengths, and delays, as well as information used for throughput and internal congestion measurements. It provides **charting, alerting,** and **reporting** capabilities that reflect current activity along with ongoing **logging.** You can also open log files at a later time for browsing and charting as if they were reflecting current activity.

Before spending money to add more hardware or replace existing hardware with faster, it's best to use Performance Monitor to first tune the system to make the most efficient use of existing resources. Here are a couple of examples of where the tool may be useful:

Example. *If we find that the CPU is 100% utilized, before replacing it with a faster CPU or adding another one, we should identify and analyze the process that is utilizing the bulk of the CPU time. We may find that the processor cycles are being consumed by a disk controller requiring PIO. In this case a DMA disk controller will then reduce processor utilization.*

Example. *If we determine the hard disk is full, before adding additional disk drives, identify how much of the page file is being utilized. You may find that the system page file size is initialized at 100 MB, but there is never more than 40 MB of it being used. Instead of purchasing another disk, we could adjust the size of the page file.*

Typical Questions

If you talk to our product support engineers or our consultants in the field and ask them about the tuning questions they most frequently hear, you may find the following:

1. How do I determine how well an application is performing?
2. How can I support my environment in a proactive manner?
3. How do I know what component of my system is the most limiting (the bottleneck)?
4. How can I ensure my system is performing the best it possibly can perform?
5. How do I determine what size system I need based on the following criteria?
6. How do I know when to upgrade?

All of these questions play some part in performance tuning. We are going to focus mostly on answering questions 2, 3, and 4, primarily by focusing our attention on exploring each of the primary components of a computer system-the memory, processor, and the I/O subsystem (e.g., disks and networks). From this standpoint, performance tuning means ensuring that every user gets a fair share of available resources of the entire system. Once you feel you have 2, 3, and 4 under control, you can start focusing on 5 and 6, which are more capacity planning issues. Once you have 5 and 6 under control, you will be able to answer number 1, and more important, do “What If” analysis.

Tuning for "Memory" Performance

Lack of memory is by far the most common cause of serious performance problems in computer systems. If you read no further in this document you could just answer by saying “Memory!”, if anyone ever asks you how to improve the performance of a system.

Memory contention arises when the memory requirements of the active processes exceed the physical memory available on the system; at this point, the system is out of memory. To handle this lack of memory the system starts *paging* (moving portions of active processes to disk in order to reclaim physical memory). At this point, performance decreases dramatically. Consider the following example. If the average instruction in a computer takes approximately 100 nanoseconds to execute and disk access takes somewhere on the order of 10s of milliseconds, how many times slower would the machine run, if there were 1 paging operation per instruction? If you answered 100,000 you would be correct! Let’s hope things don’t get that bad....

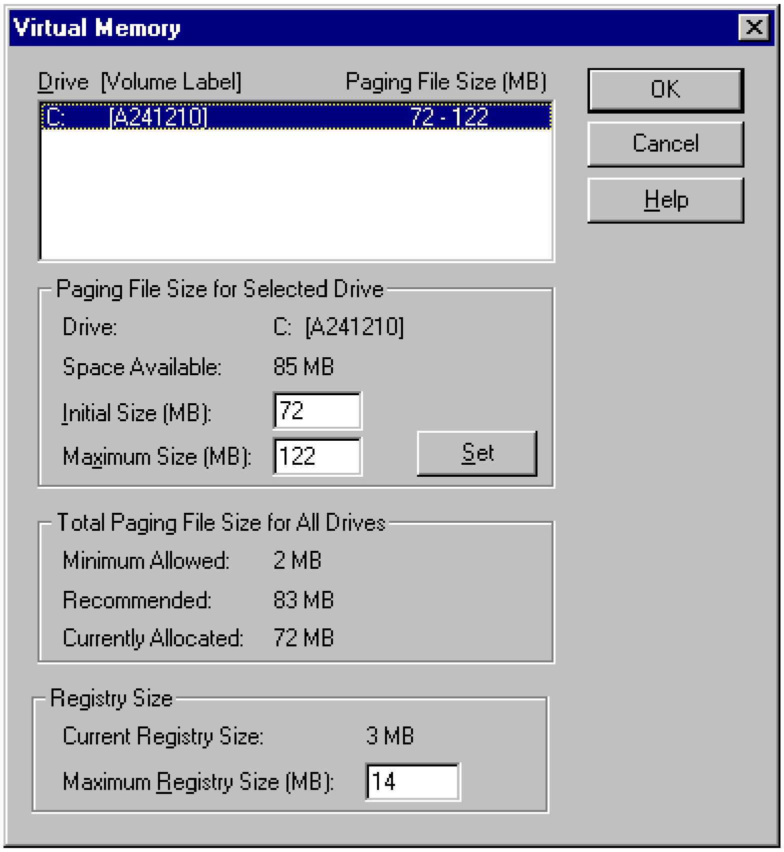
To optimize overall performance, steps must be taken to ensure that main memory is used as efficiently as possible and thus paging is held to a minimum. As you will see in the next section, you can tell how loaded system memory is by watching how the system pages.

What to Watch

- The counter “Memory **Available Bytes”** displays the amount of free virtual memory. If this counter stays consistently below 4 MB on Workstations or Windows NT servers tuned as ‘Application’ Servers , or 1 MB on Servers tuned as File/Print Servers (see the ‘Tuning for Network Performance‘ section for more information), paging is occurring and performance is less than optimal.
- The Performance Monitor counter “Memory **Pages/sec”** is the number of pages read from the disk or written to the disk to resolve memory references to pages that were not in memory at the time of the reference. As a rule, you can assume that if the <u>average</u> of this counter is consistently greater than 10, then memory is probably becoming a bottleneck in the system. Once this counter starts to average consistently at 20 or above, performance is significantly degraded and disk thrashing is probably occurring.

  If “Memory **Pages/sec**” is increasing, yet “Memory **Available bytes”** is not decreasing then you may actually not have a memory bottleneck. In this case you would want to check for an application that is doing a great deal of disk IO (reads or writes) and the data is not in cache. The “Memory **Pages/sec**” counter also increases each time a non-cached read from disk occurs-hence on database systems with many reads this counter becomes less useful.
- If the actual size of the page file is greater than its initial size (typically physical RAM + 12), time is being spent growing the page file and dealing with page file fragmentation. It is best that the page file not be required to grow during the operation of the system because it adds time to the paging processes (additional disk access to allocate the needed sectors, update any allocation, and free sector tables used by the various file systems). Another result of this behavior is fragmentation, causing the file to exist on many areas of the disk (the initial page file is created using contiguous disk space).

  To set the page file size use the Control Panel System Icon and Press the Virtual Memory ‘Change’ button. The following dialog box will be displayed:



- In the previous figure there is a setting called '**Registry Size Limit**'(RSL). This is not a setting to use to tune a system, however it is very important that it be understood because if it is set wrong a system can be unstable and act as if there is a memory problem such as:

  "The RPC server is too busy to complete this operation."

  "The RPC server is unavailable."

  "Not enough resources are available to complete this operation."

  "Not enough server storage is available to process this command."

  The Registry Size is the data stored in the Registry Hives *(Note: On a server all of the User account data is also stored in the registry).* The Registry Size Limit dictates how large this data can become.

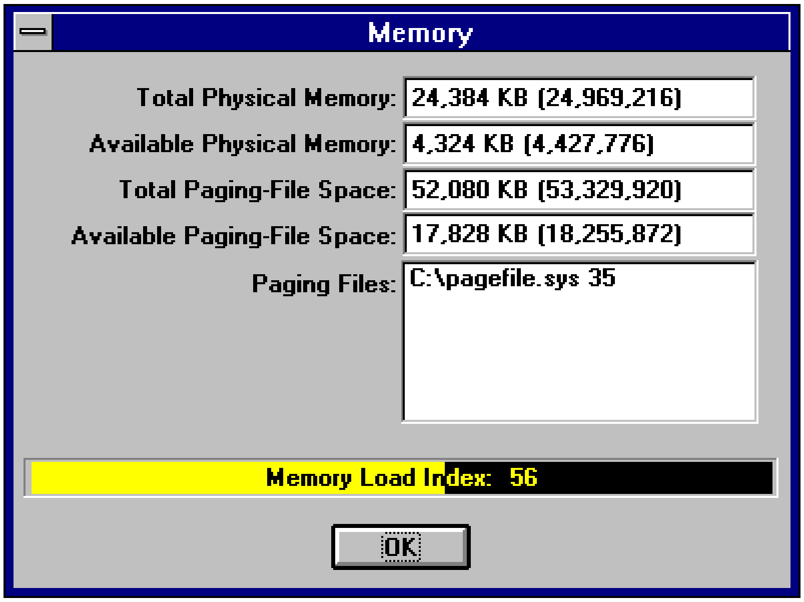
  By default, Registry Size Limit is 25 percent of the size of paged pool. Setting up the size of paged pool (see PagedPoolSize value of the Registry key (HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\Session Manager\Memory Management) also affects the size of RSL.

  RSL sets a maximum, not an allocation. Setting a large RSL will NOT cause the system to use that much space unless it is actually needed by the Registry. It also does NOT guarantee that that much space will be available for use in the Registry. In Windows NT version 3.1, paged pool defaults to 32 MB, so the default RSL is 8 MB (enough to support approximately 5000 user accounts). In Windows NT 3.5, paged pool can be set to a maximum of 128 MB, so RSL can be no larger than about 102 MB (enough to support approximately 80,000 users; however, other system limitations might keep this number of users considerably lower).

  RSL includes space in the hives themselves, as well as some of the Registry's runtime structures. Other runtime structures are either billed against standard quota, or are protected by size limits and serialization. To ensure that you can always at least boot and edit the Registry if you set

RSL incorrectly, quota checking is not turned on until after the first successful loading of a hive (that is, loading a user profile).

- A quick way to tell if your system is struggling for memory is to call up WINMSD.EXE (located in %System Root%\system32) and look at the Memory dialog.



It details the total memory in your system, the current available memory ready for allocation to applications you may start, available space within your page file, and the Memory Load Index. The Memory Load Index specifies a number between 0 and 100 that gives a general idea of current memory utilization, in which 0 indicates no memory use and 100 indicates full memory use. This index is based on the amount of available memory pages are free in the system. If AvailablePages < 100 (~500K assuming a 4K page) then the index would be at 100. If the AvailablePages > 1100 (4MB assuming a 4K page) the index would be at 0. Any value in between would be calculated with the following formula:

(100-((Available Pages-100)/10)

This dialog is built with a call to the Microsoft Win32® application programming interface GlobalMemoryStatus() in the SDK.

(Note: The MemoryLoadIndex is no longer available in Windows NT 4.0's WINMSD.)

- "Memory **Committed Bytes"** displays the size of virtual memory (in bytes) that has been committed (as opposed to simply reserved). If this counter is greater than the amount of main memory, it indicates that main memory MAY not be large enough to accommodate all functions of all currently active processes-some paging MAY be inevitable. However, before making such an assumption, you should check "Memory **Pages/sec**" and "Memory **Page Faults/sec."** If the "Memory **Pages/sec**" is greater than 10 (10 is a reasonable guideline, but varies with disk hardware) and "Memory **Page Faults/sec"** is greater than "Memory **Cache Faults/sec**" then you are paging too much.

- If you are trying to determine if adding more memory to your system will benefit your Microsoft SQL Server system, then you may want to monitor the "SQLServer **Cache Hit Ratio**" while the system is under a typical load. If the hit ratio is relatively high (over 90%), adding more memory will usually not be beneficial. This is because additional memory can mainly be used for additional Microsoft SQL Server data cache, thereby increasing the hit ratio. In this case, the hit ratio is already high, and the maximum available improvement quite small. If the hit ratio is consistently lower than this, adding more memory may improve the hit ratio and thereby performance, if the locality of reference is such that it can be "bracketed" by economically or technically feasible amounts of memory.

- When "Memory **Committed bytes"** approaches the "Memory **Commit Limit"-**and the page file has already reached maximum page file size, there are simply no more pages available, in main memory or in the page file. The "Memory **Commit Limit"** is the amount of virtual memory that can be committed without extending the page file. If this occurs on a server running Windows NT Server, you may experience 3 errors in the Event Log. (EVENTVWR.EXE is located in the Administrative Tools group). They are from the source: SRV.
  - 2020: The server was unable to allocate from the system paged pool because the pool was empty.
  - 2001: The server was unable to perform an operation due to a shortage of available resources.
  - 2016: The server was unable to allocate virtual memory.

  If this occurs, it is generally related to a memory leak in another process. To determine the process at fault you can monitor each process's Page File bytes or Working Set.

  *(Note: Windows NT 4.0 has a counter called "Memory **% Committed Bytes In Use**" This counter is the ratio of the Commited Bytes to the Commit Limit. This represents the amount of available virtual memory in use. Note that the Commit Limit may change if the paging file is extended. This is an instantaneous value, not an average.)*

- Another condition you may want to be aware of is the following nonpaged pool error in the server's Event Log:
  - 2019: The server was unable to allocate from the system nonpaged pool because the pool was empty.

  Nonpaged pool pages cannot be paged out to the paging file, but instead remain in main memory as long as they are allocated. NonPagedPoolSize is calculated using complex algorithms based on physical memory size. However, you can use the following formulas to 'approximate' these values for an X86-based computer.

  MinimumNonPagedPoolSize = 256K
  MinAdditionNonPagedPoolPerMb = 32K
  DefaultMaximumNonPagedPool = 1 MB
  MaxAdditionNonPagedPoolPerMb = 400K
  PAGE_SIZE=4096

  NonPagedPoolSize = MinimumNonPagedPoolSize +
  ((Physical MB - 4) * MinAdditionNonPagedPoolPerMB)

  Example. On a 32 MB x86-based computer:

  MinimumNonPagedPoolSize = 256K
  NonPagedPoolSize = 256K + ((32 - 4) * 32K) = 1.2 MB

  MaximumNonPagedPoolSize = DefaultMaximumNonPagedPool +
  ((Physical MB - 4) * MaxAdditionNonPagedPoolPerMB)

  If MaximumNonPagedPoolSize < (NonPagedPoolSize + PAGE_SIZE * 16),
  then MaximumNonPagedPoolSize = (NonPagedPoolSize + PAGE_SIZE *16)

  Example On a 32 MB x86-based computer:

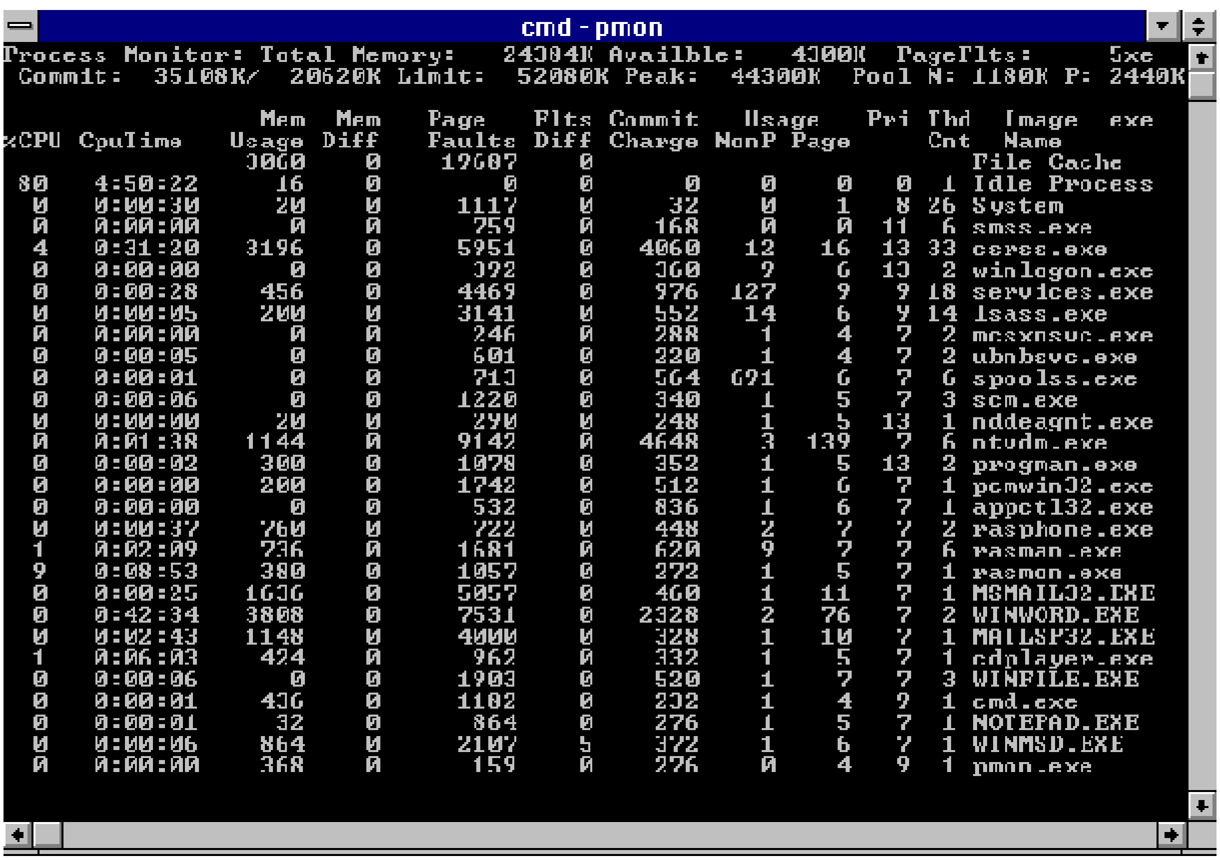
  MaximumNonPagedPoolSize = 1 MB + ((32 - 4) * 400K) = 12.5 MB

  You can monitor the system's nonpaged pool allocation with the "Memory Pool Non Paged Bytes" counter. If there is a shortage of nonpaged pool, you may also see the following error on a remote system or even the local system:

- Not enough storage available to process this command.

  If this occurs, start looking at each process's nonpaged pool allocation. This is generally caused by an application incorrectly making system calls and using up all allocated nonpaged pool.

- If you are concerned that one application is consuming a great deal of memory (paged or nonpaged) then you may want to use a utility such as the Win32 Software Development Kit utility PMON.EXE (this is also included in the Windows NT Resource Kit volume 3 utilities) to monitor its load on the system. At the top of the PMON display you see some system global statistics: memory size and available bytes, the virtual memory commitment, and pool sizes. Then, for each process, PMON shows processor usage during the last update interval. The next column is total processor time. The third column 'Mem Usage' is how many pages each process is using (this is the same as the Performance Monitor counter "Process **Working Set**"), and then the change since the last update. PMON also shows how many Page Faults have occurred in the process and the change since the last update. Next is the virtual memory commitment charge**--** Commit Charge (this is the same as the Performance Monitor counter "Process **Private Bytes**"), and then the pool usage estimates for the process. Finally you see process priority and the number of threads. There's nothing here that is not in Performance Monitor (you could get the same information by looking at such counters as "Process **Page Faults/sec"**), but it is a very handy overview and is quicker to start up, as well as being "pre-configured" to show you the system at a glance. Here is how it looks:



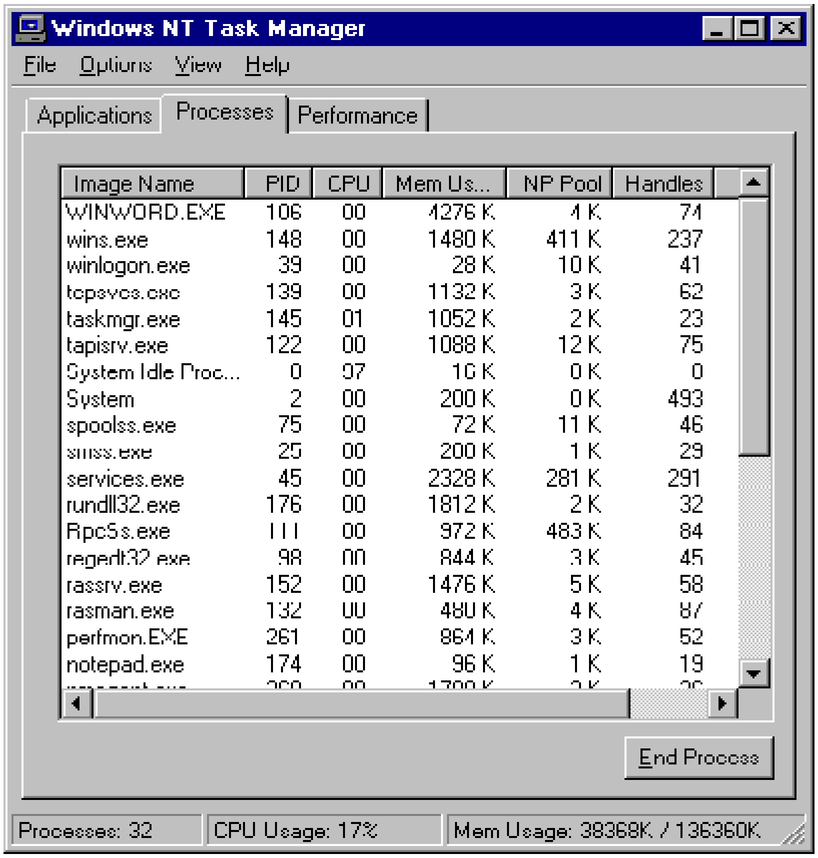
cmd - pmon

Process Monitor: Total Memory: 24304K Availble: 4300K PageFlts: 5xe
Commit: 35108K/ 20620K Limit: 52080K Peak: 44300K Pool N: 1180K P: 2440K

%CPU	CpuTime	Mem Usage	Mem Diff	Page Faults	Flts Diff	Commit Charge	Usage NonP	Usage Page	Pri	Thd Cnt	Image Name	exe
		3060	0	19607	0						File Cache	
80	4:50:22	16	0	0	0	0	0	0	0	1	Idle Process	
0	0:00:30	20	0	1117	0	32	0	1	8	26	System	
0	0:00:00	0	0	759	0	168	0	0	11	6	smss.exe	
4	0:31:20	3196	0	5951	0	4060	12	16	13	33	csrss.exe	
0	0:00:00	0	0	392	0	360	9	6	13	2	winlogon.exe	
0	0:00:28	456	0	4469	0	976	127	9	9	18	services.exe	
0	0:00:05	200	0	3141	0	552	14	6	9	14	lsass.exe	
0	0:00:00	0	0	246	0	288	1	4	7	2	mcsxnsvc.exe	
0	0:00:05	0	0	601	0	220	1	4	7	2	ubnbsvc.exe	
0	0:00:01	0	0	713	0	564	691	6	7	6	spoolss.exe	
0	0:00:06	0	0	1220	0	340	1	5	7	3	scm.exe	
0	0:00:00	20	0	290	0	248	1	5	13	1	nddeagnt.exe	
0	0:01:38	1144	0	9142	0	4648	3	139	7	6	ntvdm.exe	
0	0:00:02	300	0	1078	0	352	1	5	13	2	progman.exe	
0	0:00:00	200	0	1742	0	512	1	6	7	1	pcmwin32.exe	
0	0:00:00	0	0	532	0	836	1	6	7	1	appctl32.exe	
0	0:00:37	760	0	722	0	448	2	7	7	2	rasphone.exe	
1	0:02:09	736	0	1681	0	620	9	7	7	6	rasman.exe	
9	0:08:53	380	0	1057	0	272	1	5	7	1	rasmon.exe	
0	0:00:25	1636	0	5057	0	460	1	11	7	1	MSMAIL32.EXE	
0	0:42:34	3808	0	7531	0	2328	2	76	7	2	WINWORD.EXE	
0	0:02:43	1148	0	4000	0	328	1	10	7	1	MAILSP32.EXE	
1	0:06:03	424	0	962	0	332	1	5	7	1	cdplayer.exe	
0	0:00:06	0	0	1903	0	520	1	7	7	3	WINFILE.EXE	
0	0:00:01	436	0	1102	0	232	1	4	9	1	cmd.exe	
0	0:00:01	32	0	864	0	276	1	5	7	1	NOTEPAD.EXE	
0	0:00:06	864	0	2107	5	372	1	6	7	1	WINMSD.EXE	
0	0:00:00	368	0	159	0	276	0	4	9	1	pmon.exe	

(Note: PMON.exe and other console applications can be run on a remote system by using another Resource Kit utility service called RCMD. See the Resource Kit for more information.)

With Windows NT 4.0 you can also use the Task Manager to get the same information. Simply right click on the task bar and choose 'Task Manager'. (*Note: To configure the columns choose 'View | Select Columns'*.) The following dialog will be displayed:



What You Can Do

- Schedule memory-intensive applications during off-peak hours. You can use the AT scheduler that ships with Windows NT. For example, it would not make much sense to do a Tape backup while a system is being heavily utilized by users.
- Distribute memory-intensive applications/processes across multiple machines. For example, if you are running Systems Management Server on the same machine that you are running SQL server you may want to consider moving the SQL server to another system.
- Add more memory. To determine ABOUT how much memory to add, use the following formula:

  "Paging File **% Usage MAX"** * Page file size = number of bytes used

  Add together the bytes used for all page files. This is the amount of memory that would need to be added to allow all of the applications to perform their operations with minimum paging. For example, if your page file is 100 MB and the % Usage MAX is 20%, then you would need 20 MB additional RAM to have a system that does minimal paging. The reason this formula only gives you an idea ABOUT how much memory to add is that a) not all page file "in use" code is accessed all of the time; and b) the formula ignores the requirements for code and mapped files not backed by the paging file. Therefore this estimate is neither an upper bound, nor a lower bound-it is only an “indication.” The truth is that there is no good way to know how much memory to add at this time. A more accurate way to measure the amount of memory an application would require is to run the application on a very large machine and measure the needs under some slight memory pressure. (There is a tool in the Windows NT Resource Kit volume 3 utilities called Response Probe that can aid in this area.)

Gotcha. Adding memory without upgrading the secondary cache size sometimes degrades processor performance. This is because the secondary cache now has to map the larger memory space, usually resulting in lowered hit rates in the cache. This slows down processor-bound programs because they are scattered more widely in memory after memory has been added. (Secondary cache refers to the physical cache memory chip(s) usually located on the motherboard, as opposed to within the processor itself. In the future, processors will be built with secondary cache on the same substrate as the processor chip, or even within the processor chip itself.)

- If you determine that a great deal of memory is being consumed by an application for which you have the source code, you may want to investigate tuning the application to be less memory intensive. Good tools to use to profile your applications’ memory allocation are the Working Set Tuner and the VADUMP tools in the Win32 SDK.
- Spreading paging files across multiple disk drives and controllers generally improves performance as multiple disks can process I/O requests concurrently. After all, you can have up to 16 separate page files. Also, since Windows NT has several system files that are frequently accessed, you may want to experiment with locating the paging file on one disk and the Windows NT system files on another. You should also locate the page file(s) on separate disk(s) from application files to allow for page file I/O and application file I/O to occur concurrently. This will only work if the disk driver(s) and controller(s) used can accommodate asynchronous I/O requests. Keep in mind that most IBM-compatible “non-super servers” have an ATDISK as the default and the ATDISK driver can have only one I/O request pending at a time. If your system mixes high-speed disks and low-speed disks, use the fastest disks for all your paging.
- Use the Control Panel | System | Virtual Memory and set the page file size such that extension of it will rarely occur.
- Use the Control Panel | Services to turn off unnecessary Windows NT services, and Control Panel | Network to un-install any unnecessary Windows NT device drivers. This can free up both CPU and memory. One example might be the **Spooler** service. If you don’t have a printer connected to the workstation or server there may not be any reason to have it running. On NT 4.0 it can save you 600K of committed memory and about 10K of non paged pool--Check it out with the Task Manager or the PMON.exe tool in the Windows NT Resource Kit. Look at the ‘Commit Charge’ column (this is the same as the “Process: **Private Bytes”** counter in Performance Monitor).

  Here are some of the Windows NT 4.0 services and the amount of memory each process consumes:

  (Note: These numbers may be different on other systems depending on how the system is configured)

Service	Pool Nonpaged Bytes	Private Bytes	Working Set Bytes
AtSvc	10,308	253,952	765,952
clipsrv	1,908	237,568	1,114,112
netdde	11,142	368,640	1,286,144
nmagent	2,532	1,810,432	2,727,936
alerter (thread in services.exe)	52	4096	24576

messenger (thread in services.exe)	156	16384	49152
spoolss	11,452	618,496	1,105,920
tapisrv	12,624	499,712	1,433,600

- User accounts are stored in a registry hive, which means each account consumes paged pool on a Primary Domain Controller or Backup Domain Controller. Therefore the limit on the number of user accounts depends on the amount of memory and swap file space in your PDC and BDCs. User accounts take about 1K each, so 10,000 is about 10 MB. You may want to consider a second domain (possibly a different domain model) if you have more than 15,000 user accounts. However, the only answer may be to add more memory.
- If you are running SQL server, when you first installed it you allocated memory to be dedicated to the SQL server application. The system can be configured to allocate memory as it needs it (working set) and never de-allocate the memory. There is an advanced configuration option called "set working set size" that will configure SQL Server to lock down the working set so it doesn't get paged out. The key is to not over allocated this memory. To check on the allocation on a production server (you would only want to do this if you were happy with the SQL performance, it's use was not going to increase, and the server was in production for some time) pull up the Performance Monitors counter "Process: **Working Set**" for the SQL server EXE instance. If this counter is a great deal less than the amount you first allocted for SQL (you can check this with the SQLAdmin program) you may want to consider re-evaluating the amount of memory pre allocated to SQL.
- Some machines provide the ROM BIOS shadowing option. While this feature provides an advantage with MS-DOS, it is NOT an advantage with Microsoft Windows NT. ROM BIOS shadowing is the process of copying the BIOS from ROM into RAM and using either hardware or 386 enhanced mode to remap the RAM into the normal address space of the BIOS. Because reading RAM is much faster than reading ROM, BIOS-intensive operations are substantially faster. For example, MS-DOS uses the BIOS to write to the screen; therefore, with ROM BIOS shadowing, directory listings run more quickly. Windows NT does not use the BIOS (except during startup); therefore, no performance is gained by shadowing. If ROM BIOS shadowing is not used, more RAM is available. With Windows NT, there is an advantage to disabling the ROM BIOS shadowing option. This applies to other BIOS shadowing schemes as well. Typically the CMOS settings allow the system to shadow any BIOS. This includes the following: System BIOS, Video BIOS, Other adapters ROM BIOS (in a given select range).

Tuning for "Processor" Performance

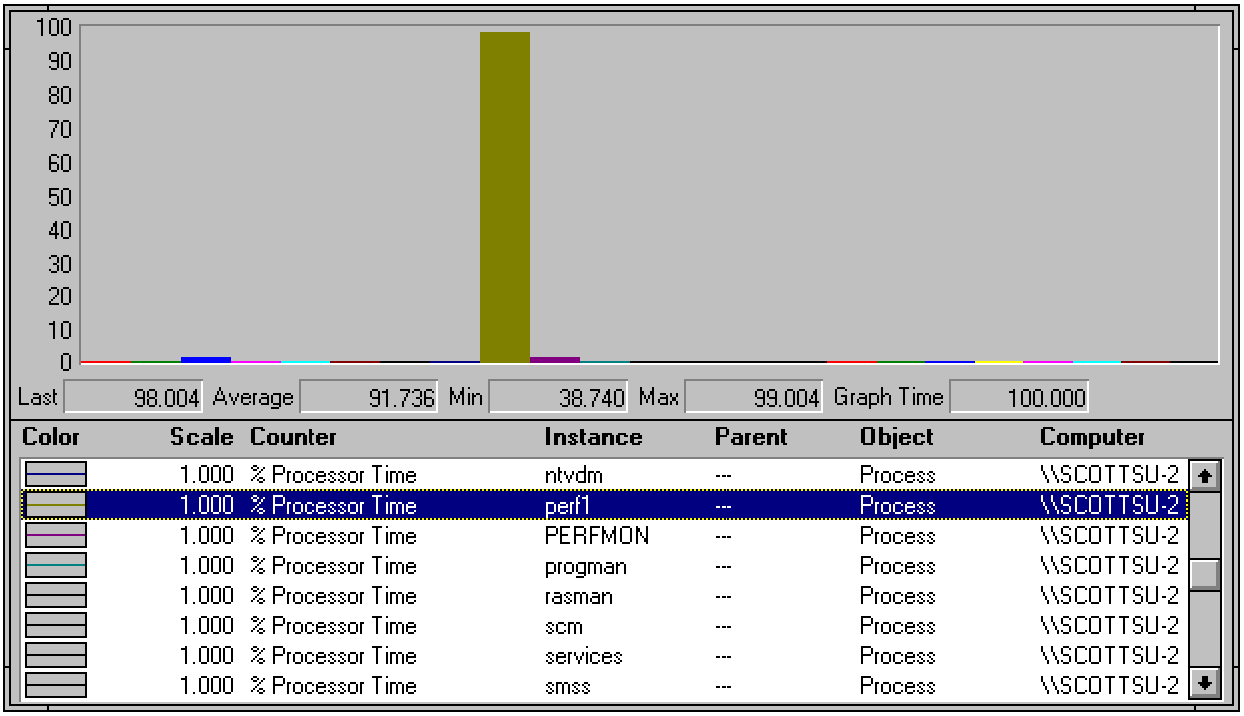
A processor (running at a given clock speed) can execute a set number of instructions per second. Therefore, if a processor is switched among multiple threads that all have work to do, a given thread will take *x* (*x* being the number of simultaneously executing threads) times longer to complete a given task.

There are times when a thread has no work to do, such as when waiting for user input, or when waiting for another thread to finish a related operation. As long as the thread is in this waiting state, it will not be scheduled for execution and, thus, does not take up any CPU time. Since most Microsoft Windows®-type applications spend a considerable amount of time with their threads in this waiting state, there may be little performance degradation when running multiple Windows-based applications.

Some applications are considered CPU intensive. A CPU-intensive application almost always has work to do and spends very little, if any, time in the waiting state. For example, the following C program consumes 100% of the CPU. When additional applications are started, their performance, and that of the CPU-intensive application, will be less than optimal since all must share the processor's time. This is an example of how NOT to write an application; a better approach would be to create an event or wait on a semaphore.

```
main(){
     while(1){}
}
```

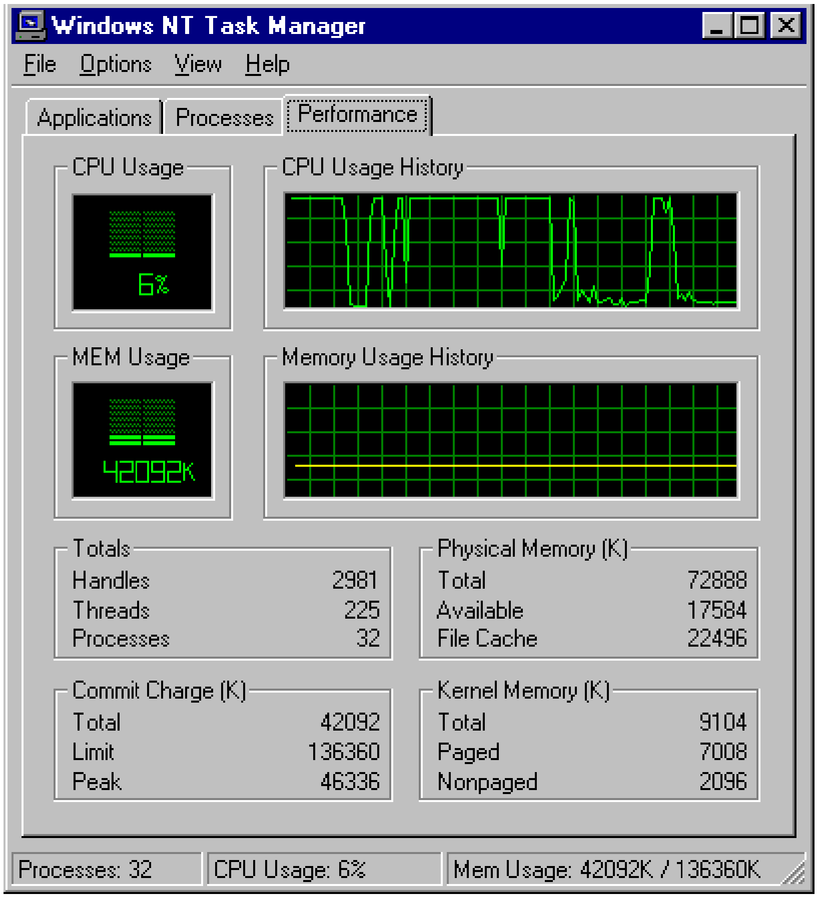
The figure below shows the application's utilization of the CPU.



What to Watch

- If the "Processor **% Processor Time**" counter consistently registers at or near 100%, the processor may be the bottleneck. ('System **% Total processor time**" can be viewed for multiprocessor systems.) If this occurs you need to determine WHO or WHAT is consuming the CPU. To determine which process is using up most of the CPU's time, monitor the "Process objects **% Processor Time"** for all of the process instances (as in the previous figure).

  Windows NT 4.0 has a quick look feature that you can use from the Task Manager to view the CPU usage as well as memory statistics for a system:



- You can tell if the CPU activity is due to applications or to servicing hardware interrupts by monitoring "Processor **Interrupts/sec.**" This is the number of device interrupts the processor is experiencing. A value over 1000 should cause you to look at the efficiency of hardware I/O devices such as the disk controllers and network cards.
- You can also monitor "System **System Calls/sec.**" Systems Calls/sec is the frequency of calls to Windows NT system service routines. These routines perform all of the basic scheduling and synchronization of activities on the computer and provide access to non graphical devices, memory management, and name space management. If there are many more interrupts per second than system calls, it could indicate that a hardware device is generating an excessive number of interrupts.
- Monitor the "System **Context Switches/sec**" as well. Too frequent context switching can be caused if semaphores or critical sections (see the Windows NT SDK for more information) are placed at too low a level in order to attain high concurrency. The only way to solve this problem is to re-evaluate the priority place on the source code.

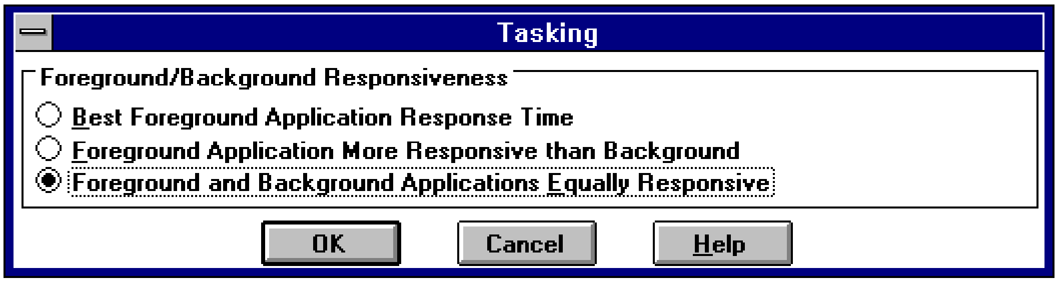
What You Can Do

- Schedule CPU-intensive applications during off-peak hours. You can use the AT scheduler that ships with Windows NT.
- If you have control over the application source, you may want to investigate tuning the application to be less CPU intensive. There are a number of tools available with the Windows NT SDK that allow you to do this, such as WAP (Windows API Profiler), CAP (Call Attributed Profiler), FIOSAP (File I/O and Synchronization Win32 API Profiler), and Win32 API Logger.

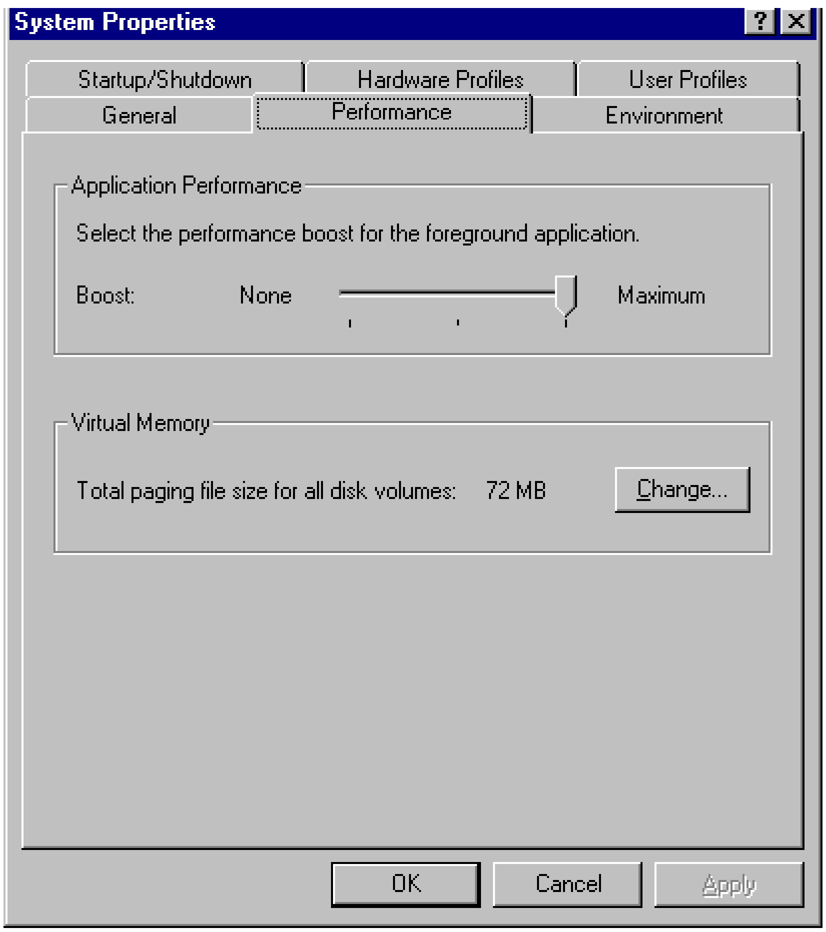
- Distribute applications and processes across multiple machines.
- Upgrade the processor if possible. Keep in mind that Windows NT runs on MIPS and Digital Alpha AXP machines as well as the Intel (386, 486, and Pentium). Most servers are either file servers or application servers. Even though they use the same operating system each uses the machine's resources in a different way. A file server generally maximizes system bus utilization and under-utilizes the processor. A 486 clock doubler chip in this machine would not provide a big performance enhancement over a typical 486 chip. An application server (such as a database server running Microsoft SQL Server and Systems Management Server), however, utilizes the processor subsystem significantly more than the file servers. You will find that this is the environment where a more powerful CPU chip will pay off.

  If you are in a situation where you are trying to determine if moving to a RISC processor will increase performance, you should look at the counter "System **Context Switches/sec."** This is the rate of switches from one thread to another. Moving to a RISC machine will only be a good idea if the **Context Switch** rate is NOT dominating processor activity.
- Add more processors assuming there is more than 1 thread capable of asynchronous execution. If you have a multiple processor computer, Windows NT will assign separate threads to different processors (interrupts are also distributed). The thread execution load is then distributed across the multiple processors. For example, if a CPU-intensive thread is executing on processor A, processor B will be free to process other threads.
- Upgrade the secondary cache. In this same regard, you may consider upgrading the CPU to a chip with a 16K First Level cache such as a 486 DX4/100 (Unified Instruction and data cache) or a Pentium (8K data cache and 8K instruction cache).
- Assuming you have at least a 486, if you are in a server environment, part of your problem may be the network or disk adapter cards you have chosen. 8-bit cards use more processor time than 16-bit or 32-bit cards. The number of bits here refers to the amount of data moved to memory from the adapter on each transfer. The most efficient cards use 32-bit transfers to adapter memory or direct memory access (DMA) to move their data. Adapters that don't use memory-mapped buffers or DMA must use processor instructions to move data, and that makes the processor busy. DMA uses the memory bus, and that can slow the processor down but it is still more efficient than individual instructions. There is more information on this topic in the "Tuning for Disk Performance" section of this document. Keep in mind while reading this section and the "Disk Performance" section that replacing PIO devices will almost always reduce processor bottlenecks.
- In a resource-sharing environment, a greater improvement can be found by upgrading to a faster processor rather than increasing the number of CPUs. In a client-server environment, the addition of another CPU will typically give a better performance increase than upgrading to a faster or more advanced processor because of the multithreaded design of all Microsoft BackOffice products.
- Each application (as well as each thread) in the system has a set priority. You can control the priority system-wide by changing the following in Control Panel | System | Tasking.

  Windows NT 3.X:



Windows NT 4.X:



Use this dialog box to change the relative responsiveness of applications that are running at the same time. When more than one application is running in Windows NT, by default the foreground application receives more processor time, and so responds better, than applications running in the background. (You can also use the Windows NT SDK utility PVIEW to set individual application priorities.)

You may also use the START command to alter the priority of a program as it is started. This command can take /low, /normal, /high, and /realtime switches to start programs with varying levels of priority.

Gotcha. Never start processor-bound applications at real-time priority.

Considerations for 16-Bit Applications

- You can monitor the performance of 16-bit MS-DOS-based applications, however they are difficult to identify as instances because the program name does not appear. This is because each MS-DOS-based application shows up in its own Virtual DOS Machine (NTVDM). You would have to look at the individual threads (that is, "Thread **Processor Time**") for the NTVDM.EXE application. An easy way to identify the thread associated with the application you want to monitor is to stop all other 16-bit MS-DOS-based applications and choose the remaining thread. Another way to identify the application is to copy the NTVDM.EXE process to another name and editing the following path in the Registry:

  SYSTEM\CurrentControlSet

  Control\WOW

  cmdline

  16-bit Windows-based applications execute in one NTVDM by default, but can be started in separate NTVDMs.
- If you are not satisfied with the performance of your MS-DOS-based applications running on Windows NT Workstation, try full-screen mode. In full-screen mode, most applications can run with native performance directly on the installed video adapter. Windows maps VGA memory to the appropriate place in the VDM and maps the relevant registers from the application to the video adapter. To get in and out of full-screen mode, press ALT+ENTER.
- When running MS-DOS or Windows version 3.1, serial communications applications that directly access serial port hardware, you may enhance performance of these applications by using software handshaking (xon/xoff) instead of hardware handshaking (cts/rts). Because hardware must be virtualized under Windows NT, checking the cts/rts signals directly will incur an unavoidable performance degradation. Using xon/xoff handshaking avoids this problem since xon/xoff handshaking does not require accessing the serial port hardware directly.
- Windows NT 3.51 supports disk compression (via the COMPACT.EXE utility). You should not compress your \temp directory \%system root%\system32 directory or any other directory that gets used a great deal. You should also not compress any files that are already compressed, such as Microsoft Email MMF files.

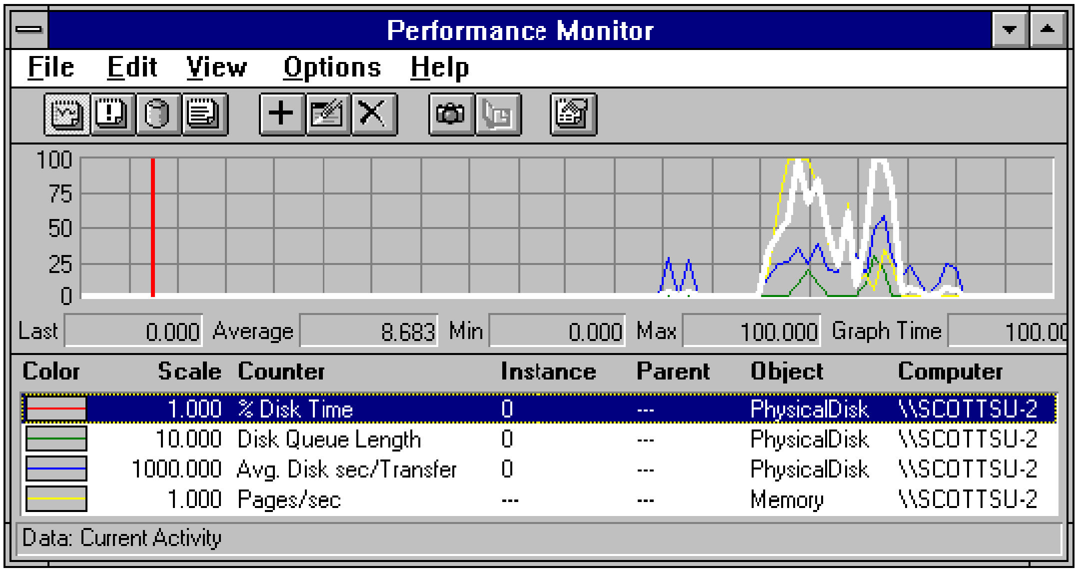
Tuning for "Disk" Performance

As you might have guessed, disk performance is the single most important aspect of I/O performance. It affects many other aspects of system performance. Good disk performance enhances virtual memory performance and reduces the elapsed time required to load programs that perform a great deal of I/O, and so on.

If you discover a disk bottleneck, the first thing you need to determine is whether it's really more memory that you need. If you are short on memory, you will see the lost performance reflected as a disk bottleneck.

Gotcha. Because disk counters can increase disk access time by approximately 1.5% on a 386/20, Windows NT does not automatically activate these counters at system startup. To activate disk counters, type **diskperf -y** at the command prompt and restart the computer. On a 486 or better system, the hit is not apparent.

What to Watch



- If the "Physical Disk object's **% Disk Time**" counter consistently registers at or near 67%, the physical disk is the bottleneck. This counter is the percentage of elapsed time that the selected disk drive is busy servicing read or write requests, including time waiting in the disk driver queue.
- If "Physical Disk **Disk Queue Length**" (pending disk I/O requests) is greater than 2, it generally indicates significant disk congestion. *(Note: This same rule applies to most all I/O devices.)*
- Determine the portion of the disk I/O used for paging with the following function "% disk time used for paging = 100 * ('Memory **Pages/sec**' * 'PhysicalDisk **Avg,DiskSec/Transfer**')". If this is more than 10% of the total disk activity then paging is excessive. **Avg. Disk sec/Transfer** is the time in seconds of the average disk transfer. This formula does not include the case where you may be paging over the network.

What You Can Do

- Install a faster disk and/or controller. Determine if the controller card does 8-bit, 16-bit, or 32-bit transfers. The more bits in the transfer operation, the faster the controller moves data. You may also want to choose a different drive technology. IDE (integrated drive electronic) has a 2.5 MB/s throughput, ESDI has a 3 MB/sec, SCSI-2 has a 5 MB/s throughput, and a Fast SCSI-2 has a 10 MB/sec throughput.
- Create mirrored data sets. The I/O system can issue concurrent reads to 2 partitions. The first portion of the read will be to partition A, while the next portion of the read will be to partition B. (Assuming the disk driver and controller can handle asynchronous I/O).
- Create striped data sets. Multiple disks (between 3 and 32)can process I/O requests concurrently (assuming the disk driver and controller can handle asynchronous I/O).
- Add memory (RAM) to increase file cache size.
- Change to a different I/O bus architecture. EISA, MCA, and local bus (VESA or PCI) buses transfer data at a much higher rate than ISA buses. PCI is fast because it transfers data at 33 MHz, a double word at a time (33 MHz * 4 = 132 Mb/sec) whereas ISA maxes out at about 5 Mb/sec and EISA about 32 Mb/sec (EISA transfers at 8 MHz * 4 bytes). There has been talk about raising the PCI clock rate to 66 MHz (to get a 264 Mb/sec transfer rate) but most

manufacturers are resisting the idea (at about 50 MHz or so, getting past FCC class B certification is a nightmare).

- When choosing a I/O device such as a disk adapter, consider the architecture of the card. For example here are some of the points to consider about each architecture:
  - **PIO:** PIO (programmed I/O) requires intervention by the CPU. For example, the Adaptec 1522 is a PIO device and can do either 16-bit PIO or 32-bit PIO. However, CPU-usage is quite intensive (30-40%) and it will slow down your system during a large transfer or a CD-ROM access. As such, most high-performance systems don't use a PIO device because they adversely impact system throughput. *BYTE* magazine did a comparison of Adaptec 2940 (PCI) against a Future Domain adapter (PIO). While the Future Domain and Adaptec 2940 provide almost identical benchmark results, the Future Domain consumes a hefty 40% of CPU time whereas the 2940 does not. However, all PIO devices are much cheaper to manufacture- the FD is about half the price of the 2940. Another thing to keep in mind is that the standard ATDISK disk (most IDE drives) does PIO.
  - **DMA:** ISA DMA has only 24-address lines so it can physically address 16 MB. However, if you happen to have 32 MB of RAM, the OS can see all of the memory. Therefore, if the OS wants to transfer a block of memory (which happens to be located at memory location above 16 MB, which the ISA DMA card, such as the Adaptec 1542C, cannot physically see), it will have to copy that block down to an area in the 0-15 MB range (where the Adaptec 1542c can see) so the 1542C can initiate the DMA transfer (double buffering). This copying down to 0-15 MB range and also copying up (16 MB and up) ***takes quite a bit of time*** (using Intel repsb, repsw, repsd) so that explains the slow down. However, you don't have that problem with either VL, PCI, or EISA as they all have 32-bit DMA address lines and can physically see up to 4 GB. PIO devices can see all of the memory, including those above 16 MB. The only problem is that it takes the processor to do any kind of data transfer. The last thing to keep in mind is that some devices do both PIO and DMA. If your system is not an ISA computer WITH more than 16 MB of RAM, you should always run with the controller in DMA mode.

    Gotcha. The Adaptec 154x ISA busmaster, has a hard coded limit of a DMA speed of 5 MB/sec transfer rate. This is hard coded in the Windows NT driver.
  - **Bus Master:** Bus master devices have their own intelligence and offload this work from the CPU. The CPU can resume doing its own thing while the bus-master device is doing all the I/O. When it's done, it hands the result to the CPU. These cards are by far the best solution.

    Gotcha. Make sure that you check the Windows NT Hardware Compatibility List before you purchase a controller. This will tell you if the controller is supported by Microsoft and has a certified driver.

- On a 2 SCSI disk daisy-chained system, the SCSI controller has more of an impact on your total performance than your disk drive. You would be better off buying a slower, cheaper disk and investing in a better SCSI controller.
- Adding more physical drives in a RAID 5 configuration can result in significant performance improvements when the disk subsystem is the bottleneck. However, adding more controllers usually does not significantly improve performance. When using high-performance disk controllers, the physical drive access times are usually the performance limiting factor for the disk subsystem.
- Choose a disk with a low seek time (the time required to move the disk drive's heads from one track of data to another). The ratio of time spent seeking to time spent transferring data is usually 10 to 1, and often much higher.
- Distribute the workload as evenly as possible among different disk drives. This will allow you to take full advantage of the system's I/O bandwidth. For example, if you have one user population

that does a great deal of reads and writes to directory \\server\ExcelData and another user population that does a great deal of reads and writes to a directory \\server\WordData then you may want to consider putting the ExcelData directory on a different disk and/or controller than the WordData directory. You can take advantage of the auditing facility of Windows NT and the NTFS file system to track how certain network files are being used. User Manager lets you enable file access auditing, and File Manager lets you specify the users and files whose access you want to record.

- If you choose a FAT file system, with time it tends to become fragmented. As the file system becomes full, pieces of files tend to be scattered over the disk; the system cannot find enough contiguous blocks to store a new file in one place, so it must fit the file in empty spaces between other files. As files are added, deleted, truncated, and expanded, the file system becomes increasingly disorderly. Performance suffers because the disk drive cannot read a file with a sequential group of operations. Instead, it must constantly seek for different pieces of the file. To avoid fragmentation, use a Defrag utility, such as Executive Software's DiskKeeper, to adjust files in a sequence (for more information refer to their web site at http: \\www.earthlink.net\execsoft on the internet).
- NTFS is best for use on volumes of about 400 MB or more. This is because performance does not degrade with larger volume sizes under NTFS as it does under FAT. As the size of the volume increases, performance with FAT will quickly decrease. When using the FAT file system, the disk space taken by files is more than the space taken when using NTFS. FAT file system uses clusters to allocate disk space for files. Clusters are the smallest allocation units that the file system uses to allocate space for the files. For example, for a 1-byte file, 1 cluster will be allocated, thus wasting all of the unused space. When a large number of small files are stored on a FAT partition, the cluster size may tend to waste a large amount of disk space. The cluster size is dependent on the size of the logical drive. FAT can only track a maximum of 64K clusters since there are 64K entries in the File Allocation Table. That would indicate that the cluster size will increase for large drives, in order to access the whole drive. The maximum cluster size is 64K, thus making the largest logical drive size to be 4 gigabytes. With NTFS there is a limit, however it's 2^{64}.
- Disabling short name generation on an NTFS partition will greatly increase directory enumeration performance especially in the case where individual directories contain a large number of files/directories with non-8.3 filenames. To disable short name generation, use REGEDT32.EXE to set a registry DWORD value of 1 in the following Registry location:

  SYSTEM\CurrentControlSet

  Control\Filesystem

  NtfsDisable8dot3NameCreation

  Gotcha. This may cause compatibility problems with 16-bit MS-DOS- and Windows-based applications.

Tuning for "Network" Performance

Network performance problems can have basically three forms, each of which cause the network protocol to have to transmit each block of data many times (or error out) causing performance problems.

- A server can be overloaded

  The server is being asked to do more than it can based on an inadequate resource, possibly from a lack of another resource such as memory.
- A network can be overloaded

  The amount of data that needs transferred is greater than the capacity of the physical medium.

- A network can lose data integrity

  The network is faulty and intermittently transfers data incorrectly.

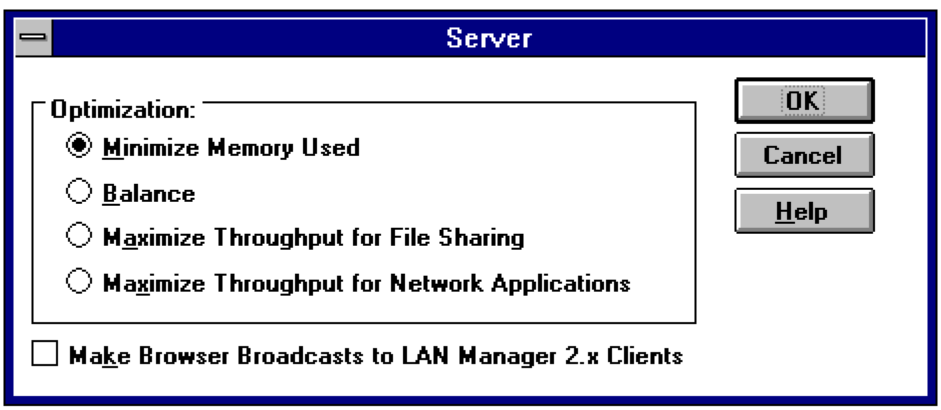
We can examine each of these problems from the perspective of the OSI (Open Systems Interconnect) networking model. From the application layer's point of view, there are the **Server** service and **Workstation** (redirector) service components as well as other application layer support entities such as **Netlogon**, **Replicator,** and other services. From the Transport layer's point of view, there are the transport components such as **TCPIP, NetBEUI, NWLINK,** and so on. From the Datalink/Physical layer's point of view there are the Adapter cards and NDIS drivers.

Gotcha. The following section details many registry entry changes. Let us note that the "out of the box" settings within the registry should allow you to have a well-balanced system. If you alter a setting, it may actually reduce the bottleneck, however it may also create another problem. Set parameters with care. If you do have a problem, use the "Last Known Good" option during system initialization to revert to an unchanged registry.

Server

The Windows NT Server service's responsibility is to establish sessions with remote stations and receive SMB (Server Message Block) request messages from those stations. (SMB requests are typically used to request the Server service perform I/O-such as open, read, or write on a device or file located on the Windows NT Server station).

- You can configure the Windows NT Server service's resource allocation (and associated **nonpaged** memory pool usage) by using the Control Panel Network application. When you use the Control Panel Network application to configure the Windows NT Server service software, you are presented with the following ***Server Optimization Level*** dialog:



  You may want to consider a specific setting, depending on factors such as how many users will be accessing the system and the amount of memory in the system. The amount of memory allocated to the Windows NT Server service (for such resources as InitWorkItems, MaxWorkItems, RawWorkItems, MaxPagedMemory, MaxNonPagedMem, ThreadCountAdd, BlockingThreads, MinFreeConnections, and MaxFreeConnection) differs dramatically based on your choice.

  - The **"Minimize Memory Used"** level is meant to accommodate up to 10 remote users simultaneously using Windows NT Server.
  - The **"Balance"** option is for up to 64 remote users.
  - The **"Maximize Throughput for File Sharing"** is for 64 or more remote users. With this option set, file cache access has priority over user application access to memory (the value of **LargeSystemCache** in the registry changes to 0x1). Use this option if you are using Windows NT Server for file server capabilities. **This is the default setting!**

SYSTEM\CurrentControlSet\Control

Session Manager\Memory Management

LargeSystemCache

- The **"Maximize Throughput for Network Applications"** is for 64 or more remote users. However, with this option set, users' application access has priority over file cache access to memory (the value of **LargeSystemCache** in the registry changes to 0x0).

  This is not valuable to MS-SQL Server because this BackOffice product does it's own memory management. This is only valuable to Client/Server applications that do not do their own memory management.

If the Windows NT Server service runs out of a resource due to one of these settings, you will see the following error in the Windows NT Event Log:

- 2009: Server could not expand a table because the table reached the maximum size.
- Not enough server storage is available to process this command.

Gotcha. If you do over ride any of the LanmanServer\Parameters you are overriding the Server Optimization Level. This may cause problems if you increase the servers memory and forget to remove the modified parameter from the registry.

- If the "Server **Work Item Shortages"** or "Server **Pool Paged/Nonpaged Failures"** are consistently increasing, or if "Server **Context Block queue Time"** (the average time, in milliseconds, a work context block sat in the server's queue waiting for the server service to act on the request) consistently averages greater than about 50 (ms), the server service is acting as a bottleneck for all tasks, on remote stations, that are issuing remote I/O requests to the server. This may be the fault of the Windows NT Server service optimization level, or it may be the fault of other bottlenecked resources (disk, CPU, and memory) on which the Windows NT Server service depends. A WorkItem is the location where the server stores an SMB. The amount of WorkItems that are available fluctuates between a minimum value (InitWorkItems) and a maximum value (MaxWorkItems). The initial value and maximum value are configured based on Server Optimization level and the amount of memory in the machine. If WorkItem shortages are occurring, it may be caused by an overloaded server. You may want to consider identifying and off-loading some of the server's "resource-consuming" tasks.
- Monitor the "Server **Pool paged failures"** and "Server **Pool nonpaged failures."** If they are occurring then the server is running out of the paged/**nonpaged** pool it originally allocated. If this occurs you ***may*** want to consider increasing the resource using the following parameters:

  SYSTEM\CurrentControlSet\Services

  LanmanServer\Parameters

  MaxNonPagedMemoryUsage

  and

  SYSTEM\CurrentControlSet\Services

  LanmanServer\Parameters

  MaxPagedMemoryUsage

  You will also experience one of the following errors in the system Event Log:

  - 2017: The server was unable to allocate from the system nonpaged pool because the server reached the configured limit for nonpaged pool allocations.
  - 2018: The server was unable to allocate from the system paged pool because the server reached the configured limit for paged pool allocations.

- 2019: The server was unable to allocate from the system nonpaged pool because the pool was empty.
- 2020: The server was unable to allocate from the system paged pool because the pool was empty.

This is more than likely being caused by lack of memory in the system. If this occurs you should refer to the “Memory” section of this paper.

There are similar paged/**nonpaged** values for the Macintosh file server service. The “MacFile **PagedMemLimit”** specifies the maximum amount of page memory that the Macintosh file server can use. Performance of the Macintosh file service increases with an increase in this value. However, the value should not be set lower than 1000K. It is especially important that you are well acquainted with memory issues before changing this resource parameter. You cannot change this value from Server Manager.

SYSTEM\CurrentControlSet\Services

MacFile\Parameters

PagedMemLimit (default = 20000 decimal REG_DWORD)

The “MacFile **NonPagedMemLimit”** specifies the maximum amount of RAM that is available to the file server for Macintosh. Increasing this value helps performance of the file server but decreases performance of other system resources.

SYSTEM\CurrentControlSet\Services

MacFile\Parameters

NonPagedMemLimit (default = 4000 decimal REG_DWORD)

- If Other (nonserver service) processes are competing with the server for processor time, you may want to consider increasing the server’s worker threads priority.

SYSTEM\CurrentControlSet\Services

LanmanServer\Parameters

ThreadPriority (default =1 REG_DWORD)

The server threads by default run at “foreground process priority.” Other threads in the system service run at “foreground process priority + 1” such as the XACTSRV threads (the service responsible for supporting remote API requests from Microsoft LAN Manager local area network software version 2.*x* stations). Since the XACTSRV is used to process printing requests, a file server that is also a print server may suffer from server thread starvation because the server threads are at a lower priority than the XACTSRV threads. In this case it makes since to increase the servers ThreadPriority to 2.

Gotcha. Do not increase the priority beyond 2, or the system may not respond normally to other activity.

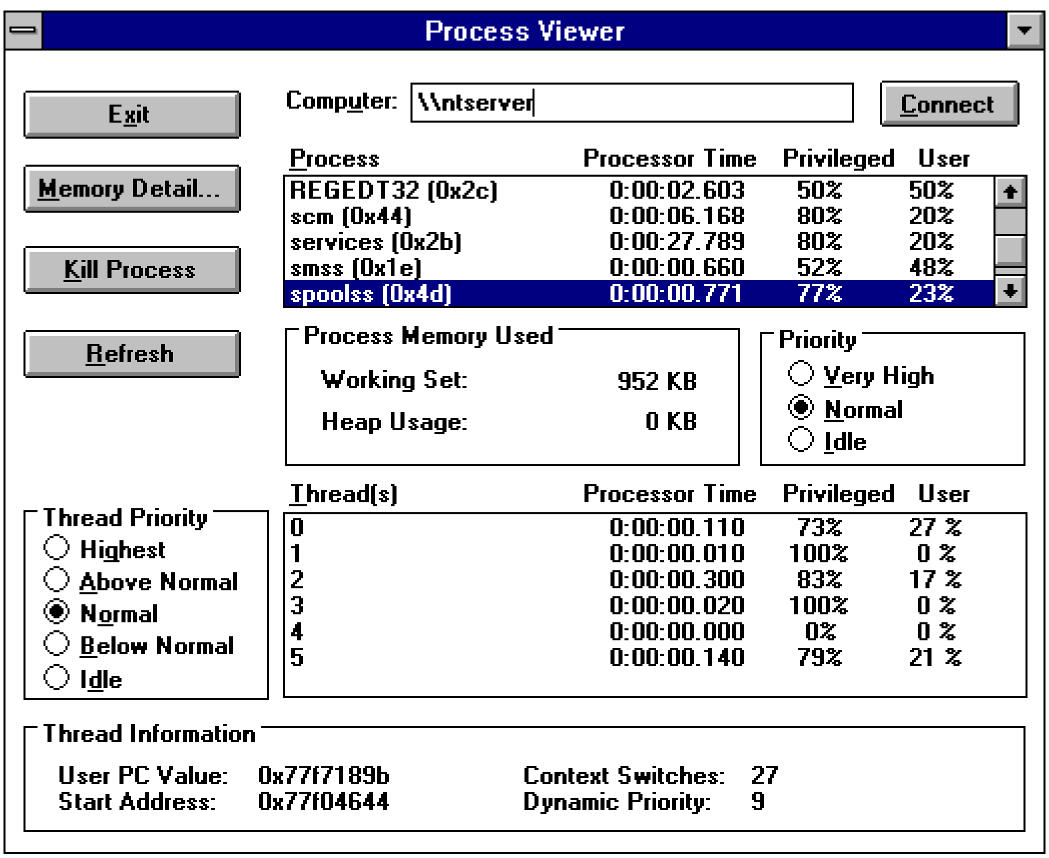
Another alternative is to drop the priority of the Spooler (it runs at 9 by default on NT 3.5 Server). You can do this with the PriorityClass parameter in the registry. It is located in the following location:

SYSTEM\CurrentControlSet

Control\Print

PriorityClass (default=0 REG_DWORD)

You can verify the priority with the PVIEWER.EXE application in the Windows NT Resource Kit. The figure below details the priority with the default setting. If you change the value in the registry and do a ‘net stop spooler’ then a ‘net start spooler’ at the command line the priority will update.


Process Viewer
Exit
Memory Detail...
Kill Process
Refresh
Computer: \\ntserver
Connect
Process Processor Time Privileged User
REGEDT32 (0x2c) 0:00:02.603 50% 50%
scm (0x44) 0:00:06.168 80% 20%
services (0x2b) 0:00:27.789 80% 20%
smss (0x1e) 0:00:00.660 52% 48%
spoolss (0x4d) 0:00:00.771 77% 23%
Process Memory Used
Working Set: 952 KB
Heap Usage: 0 KB
Priority
Very High
Normal
Idle
Thread Priority
Highest
Above Normal
Normal
Below Normal
Idle
Thread(s) Processor Time Privileged User
0 0:00:00.110 73% 27 %
1 0:00:00.010 100% 0 %
2 0:00:00.300 83% 17 %
3 0:00:00.020 100% 0 %
4 0:00:00.000 0% 0 %
5 0:00:00.140 79% 21 %
Thread Information
User PC Value: 0x77f7189b
Start Address: 0x77f04644
Context Switches: 27
Dynamic Priority: 9

- If you see the following event occur in the System log "2001: The server was unable to perform an operation due to a shortage of available resources... with the following included in the hex information 000c0000 005c0001," increase the following:

  SYSTEM\CurrentControlSet\Services

  LanmanServer\Parameters

  MinFreeConnections

  No matter how few connections are actually established, the server will make sure that there are at least "MinFreeConnections" pre-initialized, unused connection blocks ready to be used for a new connection. This value is 2 if you set the server to "Minimize Memory Used" or "Balance" and 4 if you select "Maximize Throughput...."

- If you are limited on hardware resources and want to limit the number of users that can be simultaneously logged on to a server you can manipulate:

  SYSTEM\CurrentControlSet\Services

  LanmanServer\Parameters

  Users

- Since each server connection does take up some amount of memory, you may want to consider tuning the Autodisconnect parameter. This parameter sets the time interval after which inactive connections are terminated if no open files on the connection exist. This will free up a small amount of the server's resources to accommodate active users.

  SYSTEM\CurrentControlSet\Services

  LanmanServer\Parameters

  Autodisconnect (default=15 min.)

- In the Windows NT 3.51 product there have been additions and changes to the server object. A new object has been created called **Server Work Queues**. It's counters are as follows:

  active threads; available threads; available work items; borrowed work items; bytes received/sec;bytes sent/sec; bytes transferred/sec; context block queued/sec; current clients; queue length; read bytes/sec; read operations/sec; total bytes/sec; total operations/sec; work item shortages; write bytes/sec; write operations/sec;

  The **Server** object also has the following new and changed parameters:

  blocking requests rejected; bytes received /sec; bytes total/sec; bytes transmitted/sec; context blocks queued /sec; errors access permissions; errors granted access; errors logon; errors system; file directory searches; files open; files opened total; logon total; logon/sec; pool nonpaged bytes; pool nonpaged failures; pool nonpaged peak; pool paged bytes; pool paged failures; pool paged peak; server session; sessions errored out; sessions forced off; sessions logged off; sessions timed out; work item shortages

Workstation (Redirector)

When applications or users issue Connect, Open, Read, or Write requests on path-names that reference a redirected drive (net use z: \\server\share), the request is forwarded to the local Windows NT redirector. The redirector then packages up the request and forwards it down to the transport (TCP/IP, NBF, or NWLINK) and out onto the wire to be picked up by a server. So, as you can see, a great deal of the redirector's network performance is tied directly to how well the server responds to its requests. However, there are a few issues to be aware of on the redirector side.

- "Redirector **Current Commands"** counts the number of requests to the Redirector that are currently queued for service. If this number is much larger than the number of network adapter cards installed in the computer, then the network(s) and/or the server(s) being accessed are seriously bottlenecked. To try to compensate for the problem locally, you could increase the maximum allowed pending network commands if the redirector application I/O request queue is backed up by increasing:

SYSTEM\CurrentControlSet\Services

LanmanWorkstation\Parameters

MaxCmds (default = 5)

- If you see "Redirector **Network Errors/sec"** then SMB requests are timing out, forcing the redirector to disconnect, reconnect, and recover. If this is occurring, you may need to increase the:

  SYSTEM\CurrentControlSet\Services

  LanmanWorkstation\Parameters

  SessTimeout (default = 45 sec REG_DWORD)

  This specifies the maximum amount of time that the redirector allows an operation that is not long-term to be outstanding.

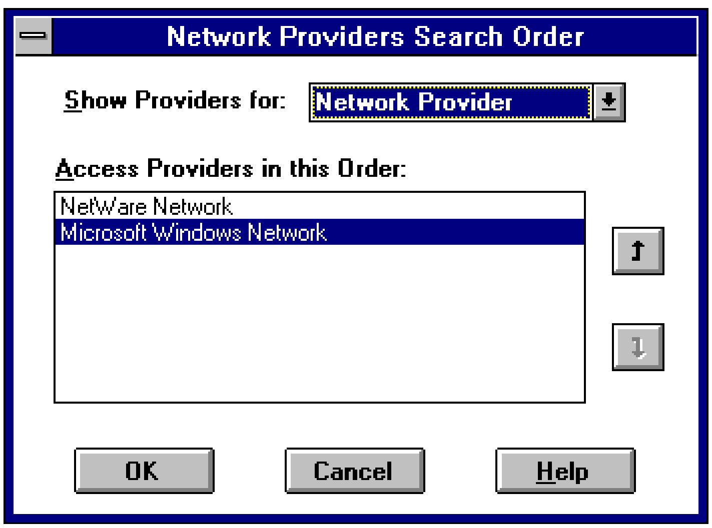
- Increase the redirector's thread count if the redirector can't accommodate overlapped I/O requests. For example, the WriteFileEx() WIN32 function may fail, returning the messages ERROR_INVALID_USER_BUFFER or ERROR_NOT_ENOUGH_MEMORY if there are too many outstanding asynchronous I/O requests.

  SYSTEM\CurrentControlSet\Services

  LanmanWorkstation\Parameters

  MaxThreads (default=17)

- If you have more than 1 redirector loaded on your Windows NT Workstation (for example, Client Services for NetWare, and so on), consider the order of providers. When a WNet API is called, it routes the call to the first provider DLL (dynamic-link library) in the "ProviderOrder" and then waits for this provider to return before submitting it to the next provider. You can see the provider order by looking in the Network Control Panel and pressing the Network button



  or, if you are interested in the value on a remote machine, you can use the registry editor (REGEDT32.EXE) and view the following registry entry:

  SYSTEM\CurrentControlSet\Control

  NetworkProvider

  Order:ProviderOrder

- There is a new SMB that is now supported under Windows NT called **NtTransact_NotifyDirectoryChange.** This allows an application to know when a directory structure has been updated on the server. If an application causes one of these SMBs to be submitted, RAW SMB I/O cannot be accomplished. (Note: RAW I/O is much faster than CORE I/O. However, it must have the session's full attention. Since there is an outstanding request on the session, RAW cannot be accomplished.) Windows NT File Manager causes one of these SMBs to be submitted if you are focused on a redirected drive. This can cause a slowdown on large reads and writes from other applications. You can shut this feature off for File Manager in the registry by adding the following value:

HKEY_CURRENT_USER\SOFTWARE\Microsoft\File Manager

Settings

ChangeNotifyTime (default=0 REG_SZ)

Netlogon

One of the primary jobs of Netlogon is to keep the user account database in sync on all of the backup domain controllers with the primary domain controller.

- Increase Netlogon service update notice periods on your Primary Domain Controllers, as well as the server announcement period if you are concerned with the amount of maintenance traffic the Windows NT Server is creating and the load on the primary domain controller.

  Use the registry editor and modify the following path:

  HKEY_LOCAL_MACHINE\SYSTEM

  \CurrentControlSet\Services\Netlogon\Parameters

Value Name	Default Value	Minimum Value	Maximum Value
PulseConcurrency	20	1	500
Pulse	300 (5 minutes)	60 (1 minute)	3600 (1 hour)
Randomize	1 (1 second)	0 (0 seconds)	120 (2 minutes)

  - **Pulse** defines the typical pulse frequency (in seconds). All User/Security account database changes made within this time are collected together. After this time, a pulse is sent to each BDC needing the changes. No pulse is sent to a BDC that is up-to-date.
  - **Randomize** specifies the BDC back off period (in seconds). When the BDC receives a pulse, it will back off between zero and Randomize seconds before calling the PDC.
  - **PulseConcurrency** defines the maximum number of simultaneous pulses the PDC will send to BDCs.

- Netlogon sends pulses to individual BDCs. The BDCs respond by asking for any database changes. To control the maximum load these responses place on the PDC, the PDC will only have PulseConcurrency pulses "pending" at once. The PDC should be sufficiently powerful to support this many concurrent replication RPC calls (related directly to server service tuning as well as the amount of memory in the machine). Increasing PulseConcurrency increases the load on the PDC. Decreasing PulseConcurrency increases the time it takes for a domain with a large number of BDCs to get a user account database change to all of the BDCs. Consider that the time to replicate a database change to all the BDCs in a domain will be greater than:

  ((Randomize/2) * NumberOfBdcsInDomain) / PulseConcurrency

You have to keep in mind the amount of traffic that 1 change to the user account database causes between Primary Domain Controllers and Backup Domain Controller. Each password change or addition of a user takes approximately 1K of data (it can be greater depending on the amount of comments etc.). Each group takes approximately 4K of data. When doing the math once you know how many changes your users will be making (you can control this with User Manager Account Policies) you also have to consider how many servers you have in the domain because each server changes its internal password randomly each 7 days).

Netlogon also has controls in place to ensure that the bandwidth is not compensated. Netlogon has a REPLICATIONGOVERNER that lets it use only 25% of the available bandwidth.

In Windows NT 3.51, there are two new counters in Performance Monitor to assist you in Domain Performance analysis. They are "Server **Logon Total"** and "Server **Logon/Sec.**".

Print Browser

When you share a printer in Windows NT, the spooler creates a thread that broadcasts a message on all of the print server's installed protocols to all other Windows NT print servers, informing the other print servers of the new print share. Each of these print servers adds the new print share name to its local printer browse list. Also, each print server rebroadcasts the list of its local printers to all other print servers every 10 minutes. This ensures that all Windows NT print servers have current browse lists, but it also causes network traffic. To disable the printer browse thread (Disabling the thread will reduce network traffic, but it also keeps other print servers from knowing about new print shares) update the following registry path to a 1 rather than a 0.

HKEY_LOCAL_MACHINE\SYSTEM

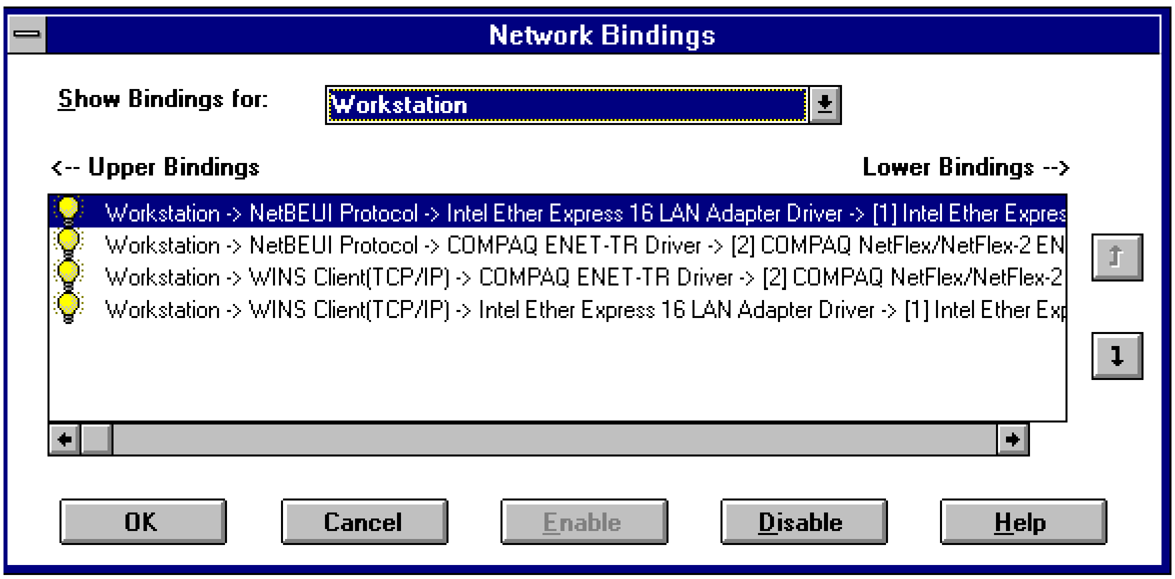
CurrentControlSet\Control\Print\Providers

ServerThreadRunning (default = 0 REG_SZ)

Transport (NBF, TCP/IP, NWLink, and so on)

The transport drivers function is to transport network data submitted by applications (such as the redirector, e-mail, Microsoft SQL Server, and so on) to other network stations. Windows NT ships with a variety of transport drivers such as TCP/IP, NBF (NetBEUI), and NWLink. All of these transports export a TDI interface on top and an NDIS (Network Driver Interface Specification) on the bottom. (Windows NT also ships with AppleTalk and DLC, however, these do not have a TDI interface.)

- If the protocol used on most stations that you will connect to is first in the bindings list, average connection time decreases. This is because when you request a connection to shared resources on a remote station, the local workstation redirector submits a TDI connect request to all transports simultaneously, and when any one of the transport drivers completes the request successfully, the redirector waits until all higher priority transports return. In the following figure you will see that the NetBEUI / Intel Ether Express binding has the highest priority.



- Having more than 1 transport protocol also causes more network traffic for session startup. The following trace was taken from a client with TCPIP installed (noted in **bold**), as well as, Netbeui (noted in *italic*). TCPIP is bound higher than Netbeui. The client does a 'Net Use' to a server called 'SERVER'. The server also has both TCPIP and Netbeui installed. Note that the session gets setup on both transport stacks, however the session via Netbeui is disconnected (frame 19) once the session is also successfully setup via TCPIP. This occurs because TCPIP is bound higher than Netbeui. The point to understand here is that because the workstation also had Netbeui installed it sent out 16 additional frames, one of which was a multicast (frame 2).

	Source	Dest.	Frame Type	
1	**WORKSTATION**	***BROADCAST**	**NBT**	**NS: Query req. for SERVER**
2	*WORKSTATION*	**NETBIOS Multi*	*NETBIOS*	*Name Query*
3	*SERVER*	*WORKSTATION*	*NETBIOS*	*Name Recognize*
4	*WORKSTATION*	*SERVER*	*NETBIOS*	*Name Query*
5	*SERVER*	*WORKSTATION*	*NETBIOS*	*Name Recognize*

```
6     SERVER      WORKSTATION NBT   NS: Query
7     WORKSTATION SERVER      LLC   SABME
8     WORKSTATION SERVER      TCP   ....S.
9     SERVER      WORKSTATION LLC   UA
10    SERVER      WORKSTATION TCP   .A..S.
11    WORKSTATION SERVER      LLC   RR
12    WORKSTATION SERVER      TCP   .A....
13    WORKSTATION SERVER      NBT   SS: Session Request
14    SERVER      WORKSTATION LLC   RR
15    SERVER      WORKSTATION NBT   SS: Positive Session Response
16    WORKSTATION SERVER      NETBIOS     Session Initialize
17    SERVER      WORKSTATION LLC   RR
18    SERVER      WORKSTATION NETBIOS     Session Confirm
19    WORKSTATION SERVER      NETBIOS     Session End
20    WORKSTATION SERVER      LLC   RR
21    SERVER      WORKSTATION LLC   RR
22    WORKSTATION SERVER      LLC   DISC
23    WORKSTATION SERVER      SMB   C negotiate
24    SERVER      WORKSTATION LLC   UA
25    SERVER      WORKSTATION SMB   R negotiate
26    WORKSTATION SERVER      SMB   C session setup & X
27    SERVER      WORKSTATION SMB   R session setup & X
```

Note that changing the binding order of the Server component does not impact server performance. he Server listens on all protocols and responds when it gets a connect request regardless of the binding order.

- Name resolution via NetBIOS (how someone finds another computer on the network) on a system has to be understood and configured correctly or it can cause unnecessary traffic and longer session setup wait times. On systems that are running the NetBEUI transport there is not much that must be configured. However, on a system running NetBIOS over TCPIP there are a number of NetBIOS name resolution options.

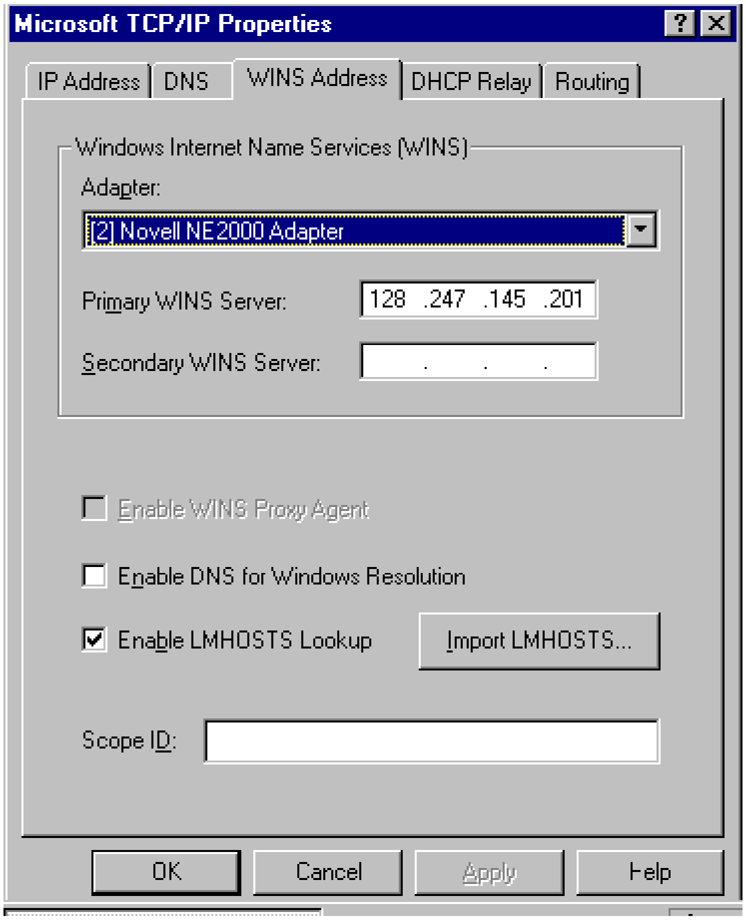
  With Windows NT 3.1 there was only 1 option and that was:

  - **B-node -**This implementation uses broadcasts to resolve names.

  With Windows NT 3.5 and above the WINS (Windows Internet Naming Service) three other options were added:

  - **P-node -**This implementation uses point-to-point communications with a name server to resolve names.
  - **M-node -** This implementation uses b-node first (broadcasts), then p-node (name queries) if the broadcast fails to resolve a name.
  - **H-node -** This implementation uses p-node first for name queries, then b-node if the name service is unavailable.

  NetBIOS name resolution is also available via the DNS server, however because of the non -dynamic nature of a DNS server this is not a very good choice. You can configure this choice via the TCPIP properties dialog box:



You can tell what node your system is using by typing '**ipconfig /all**'.

The %systemroot%\system32\drivers\etc\LMHOST file also can play a part in name resolution. If a system has a #PRE next to it in the LMHOSTS file then it is loaded into the cache when the system is started (or Nbtstat -R is run).

The reason all of this is important from a tuning perspective is that if a system is configured as a H-node client it first checks is #PRE cache and if that fails it checks the WINS server, then it broadcasts, then it checks the local LMHOST file and then finally the DNS server (via multiple queries if multiple 'Domain Suffix Search Orders' are specified). This can all take quite some time (and create unnecessary traffic) if the NetBIOS name is not located in one of the first places the client checks. Both TCPIP and the Redirector have a 45 second time out so you might not make it through all of then name resolution attempts anyway.

- Each transport has its own way of doing windowing (typically the amount of packets sent before an acknowledgment is required). By increasing the window size, you can send more packets to the other side before you have to wait for an acknowledgment. This can have a slight increase in performance (less packets = less I/O), however, it can also increase the risk of retransmission. This is NOT a recommended practice.
  - For NBF you can modify:

    SYSTEM\CurrentControlSet\Services

    NBF\Parameters

    LLCMaxWindowSize (default = 10)

This is how many LLC I-frames NBF can send before it must stop and wait for an acknowledgment.

- For TCP/IP you can modify:

  SYSTEM\CurrentControlSet\Services

  Tcpip\Parameters

  TcpWindowSize (default = 8192 for NT 3.1. In NT 3.5x it depends on the Maximum Transmission Unit and it rounds to the nearest multiple, so for Tokenring it ends up at 16K and for Ethernet it ends up about 8700.)

This is the amount of data that can be accepted in a single transaction.

- For NWLINK you can modify 3 entries:

  SYSTEM\CurrentControlSet\Services

  NWNBLink\Parameters

  AckWindow (default = 2)

This specifies the number of frames to receive before sending an acknowledgment.

SYSTEM\CurrentControlSet\Services

NWNBLink\Parameters

RcvWindowMax (default = 4)

This specifies the maximum number of frames the receiver can receive at one time.

SYSTEM\CurrentControlSet\Services

NWLink\Parameters

WindowSize (default = 4)

This specifies the window to use in the SPX packets.

- If you are on an NBF network and have a server on a very slow link, you may want to consider increasing the following:

  SYSTEM\CurrentControlSet\Services

  NBF\Parameters

  DefaultT1Timeout (default = 600 ms; grows dynamically to 10 sec)

  The T1 value controls the time that NBF waits for a response after sending a logical link control (LLC) poll packet before resending it. The default value you specify here is only used upon link establishment. It is then dynamically changed every 30 seconds.

- If you are on an NBF network and have a server on a very busy link you may want to consider increasing the following:

  SYSTEM\CurrentControlSet\Services

  NBF\Parameters

  LLCRetries (default = 8)

  This value specifies the number of times that NBF will retry polling a remote workstation after receiving a T1 time out. After this many retries, NBF closes the link.

Physical (Network Adapter)

- If the sum of the “Server **Bytes Total/sec”** (the number of bytes the server has sent to and received from the network) is roughly equivalent to the maximum transfer rate of your network, you may need to segment your network. On an Ethernet segment this value is ~1.2 megabyts per sec *[(10,000,000 bits /sec) / 8 = 1,250,000 bytes / sec]*, once you include the overhead of the network.

- An Ethernet segment is shared by every user of every system on the network. Therefore, it is a relatively limited resource with many users. This situation can be alleviated somewhat by adding sub-networks, but no matter how complex the network's topology, a network basically consists of many systems communicating through a single piece of wire. If one user is accessing a very large file across the network, that user may be slowing down the network for ***all*** users.
- Match adapter to the system bus. If you have a 16-bit bus, use a 16-bit network adapter; if you have a 32-bit bus, use a 32-bit network adapter.
- Avoid sending from fast adapters to slow adapters.
- If you need to transfer huge amounts of data between different computer systems, Ethernet may not be the appropriate medium to use; the basic Ethernet cable is limited to 10 megabits per second (considerably less when you include network overhead). Other media are now available that offer significantly higher sustained transfer rates (FDDI, and so on).
- It's not difficult to figure out how much data can be send down a segment once you know the line speed. For example, if you are dealing with a 9600 BPS line, that means that you should be able to get a true throughput of 4.32MB per hour.

  9600 BPS/8 bits/byte * 60 sec/min * 60 min/hour ==> 4.32MB/hour

  The same goes for a higher speed line such as a 38.4K line. The math says you should be able to get approximately (4*4.32) 17.28MB down the pipe. This does not account for retransmissions and line noise, it's a theoretical number. When doing calculations such as this you should probably use some type of factor such as (.80 * theoretical value).
- If you install the SNMP service you will see an additional object added to the Performance Monitor-**Network Interface**. With the Network Interface counters you can determine the number of bytes sent and received by your system. Comparing these numbers to the total throughput of the network segment can give you some indication to the amount of bandwidth your system is consuming over some period of time.
- The Network Monitor (provided with Systems Management Server) is a very good tool to use to monitor and trouble shoot general network performance problems. The Network Monitor tool utilizes the **Network Monitor Agent** that ships with Windows NT to gather statistics about the network segment traffic and can be used as a general network analyzer.   If you install the Network Monitor Agent (use the Control Panel Network Icon to install the service) you will see an additional object added to the Performance Monitor-**Network Segment**.   The Network Segment counters are:
  - **% Network Utilization** represents what percentage of the network bandwidth is being used.
  - **Frames per Second** is the number of frames being transmitted on the network per second.
  - **Bytes Per Second** is the number of bytes being transmitted on the network per second.
  - **Broadcasts per Second** represents the number of broadcast frames on the network per second.
  - **Multicasts per Second** represents the number of multicast frames on the network per second.
  - **Network Card (MAC) Statistics** represents the cumulative total number of frames, bytes, broadcasts, and multicasts seen on the network by the network card since the capture has begun.
  - **Network Card (MAC) Error Statistics** indicates the cumulative errors seen from the network card. These include CRC Errors and frames dropped because of no buffer space as well as frames dropped because of hardware constraints.

- By sorting the Network Monitor Broadcasts Multicasts column in the Station Statistics pane (bottom pane), you can find the source(s) of a broadcast storm to see which machine(s) is/are sending the most Broadcast frames.
- An increase in the amount of Broadcasts/Multicasts per second can relate directly to machine performance. Each broadcast/multicast causes every card on the net to generate an interrupt to allow the packet to be passed up to the transport. This can cause serious CPU utilization problems. As a general rule, a broadcast/multicast rate of over 100/sec should cause you to investigate a cause as well as a cure. The cure may be as easy as identifying a jabbering network card or configuring a router to not enable TCP ports 137 and 138. Note: NBF is not a routable transport.
- "Network Segment **% Network Utilization**" should be considered when things start slowing down to the point they are no longer acceptable. Some say that this point is around 40-50%. Then the network is the bottleneck.
- Ethernet ***collisions*** occur when your system starts sending data at the same time as another system on the network. When your system detects a collision, it waits a random amount of time and retransmits the packet. Collisions are normal events and don't indicate hardware problems. However, the probability of two hosts transmitting at the same time, increases as the network is more heavily utilized, so collisions are an extremely good indicator of network load. The number of collisions should be, at most, 15% of the total number of output packets. The only solution for this problem is to rearrange the network in a way that reduces traffic. Ethernet networks start to have significant collisions at about 66.67% utilization, or 833375 bytes per second. You can measure collisions with a tool such as a Network General Sniffer. Note that Version 1.0 of Network Monitor does not report collisions.

Capacity Planning

Now that you have your system optimized to where you are very comfortable with its performance (today), it's time to start collecting data that will help you in the future. The following counters are a good starting point for capacity planning:

Object	**Counter(s)**
Processor	% Processor Time
Memory	Pages/sec, Available Bytes, Commit Limit, Committed Bytes, Pool Non Paged Bytes
Paging File	% Usage Peak
Physical Disk	% Disk Time, Avg. Disk Seconds/Transfer
Logical Disk	% Free Space
Network Seg.	% Network Utilization
Network Int.	Bytes sent, Bytes received, Bytes total

The data must be sampled periodically all day long, every day and stored for analysis. Along with this data must be stored system changes such as operating system upgrades, software installs, hardware installs, system reconfigurations, network segment configuration changes etc. Without this type of history it will be difficult to determine what may have cause a change in the history of the data.

The data can be used for both supportability purposes and planning purposes. Each day the data can be reviewed to see if thresholds are being hit *(Note: It would be ideal if you could be notified as soon as a threshold was hit*). For example, if %Processor time is > 80%, or if Available bytes < 1 MB on a File and Print server, or if the Physical Disk %Disk Time > 67%, or if % Free Space < 5%, or if Network Segment % Network Utilization > 40 % on an Ethernet segment for an extended period of time the Problem Managment group within a company may want to know about the system that is having the problem. From a planning standpoint, if you have data history you can make better judgments about adding users or applications to a system.

Some Capacity Planners choose to use the Performance Monitor Logging functionality to capture the data. You can setup some very elaborate Performance Monitor Log setting files to capture performance statistics on a number of remote servers. Then once a day dump the data into a database via some automated script.

Other Capacity Planners choose to use other tools. For example, there is a service included in the Windows NT Resource kit called **DATALOG.EXE.** It will allow you to capture the data and forward it to a data store where it can be gathered up later to be used for trend analysis. The Performance Monitor is an application and needs to have a user logged onto the system for it to run locally *(Note that Performance Monitor can run remotely)*. Datalog is a service and can be run while the system is logged off. With the Datalog service you can also setup Alerts. For example, you may want to set an alert on the "Physical Disk **Free Megabytes"** on your file server's logical drives if it hits a certain threshold, "Paging File **% Usage"** if it hits 80 or 90%, and "Redirector **NetworkErrors/sec."**

There is also a utility in the Resource Kit called **Perf2Mib.EXE**. This utility allows you to create an SNMP MIB that you can put on a Windows NT server. With the MIB you can use an SNMP manager such as HP Openview's Network Node Manager to query the Performance Monitor Counters.

The following steps describe how to get the Memory, Processor, Network Segment and PhysicalDisk performance monitor counters via SNMP:

1. Run the 3.51 Resource Kit utility Perf2Mib.exe as follows:

   perf2mib perfmib.mib perfmib.ini memory 1 mem processor 2 cpu "Network Segment" 3 net PhysicalDisk 4 disk

2. Run the 3.51 Resource Kit utility MIBCC.EXE to compile the mib as follows:

   mibcc -oc:\reskit\mib.bin -n -t -w2 c:\reskit\smi.mib c:\reskit\LMMIB2.MIB c:\reskit\mib_II.mib perfmib.mib

3. Stop the SNMP agent (e.g. "net stop snmp")
4. Rename the %systemroot%\system32\mib.bin to mib.old
5. Copy the following files to the system32 directory:

   perfmib.dll (from the resource kit)

   perfmib.ini (created when you ran Perf2Mib.exe)

   mib.bin (created when you ran MIBCC.EXE)

6. Register perfmib.reg in the registry (e.g. "regini perfmib.reg")

   (Note: Regini is a utility in the resource kit.)

   The perfmib.reg file should contain the following:

```
\Registry\Machine
    Software
      Microsoft
        PerformanceAgent
          CurrentVersion
            Pathname = REG_EXPAND_SZ
%SystemRoot%\System32\perfmib.dll
    System
      CurrentControlSet
```

Services

SNMP

Parameters

ExtensionAgents

MicrosoftPerformanceAgent=

SOFTWARE\Microsoft\PerformanceAgent\CurrentVersion

7. Restart the SNMP agent (e.g. "net start SNMP")
8. Add "perfmib.mib" to the management station (such as HP Openview) using the appropriate method.
9. Now you can query the mib with NT utilities (as well as something like Openview Network Node Manager) such as SNMPUTIL.exe from the resource kit.

   For example:

   For %Network Utilization:

   snmputil get localhost public .iso.org.dod.internet.private.enterprises.microsoft.software.systems.OS.WinNT.Performance.2.1.2.0

   For %Processor Time:

   snmputil walk localhost public .iso.org.dod.internet.private.enterprises.microsoft.software.systems.OS.WinNT.Performance.3.1.6

   For %Disk Time

   snmputil get localhost public .iso.org.dod.internet.private.enterprises.microsoft.software.systems.OS.WinNT.Performance.4.1.3.0

   For Memory Available Bytes:

   snmputil get localhost public .iso.org.dod.internet.private.enterprises.microsoft.software.systems.OS.WinNT.Performance.1.1

   Remember that there are other devices within a network that can effect the capacity of a server such as the routers. It is also important to monitor the capacity of these devices if it is possible. For example on most cisco routers you can get performance information via SNMP. Here are a couple of examples:

- The amount of free memory in bytes.

  snmputil get ipaddress public .iso.org.dod.internet.private.enterprises.9.2.1.8.0
- The CPU busy percentage in the last 5 second period.

  snmputil get ipaddress public .iso.org.dod.internet.private.enterprises.9.2.1.56.0
- The 1 minute exponentially-decayed moving average of the CPU busy percentage.

  snmputil get ipaddress public .iso.org.dod.internet.private.enterprises.9.2.1.57.0
- The 5 minute exponentially-decayed moving average of the CPU busy percentage.

  snmputil get ipaddress public .iso.org.dod.internet.private.enterprises.9.2.1.58.0
- The five minute average of input bits per second.

snmputil walk ipaddress public .iso.org.dod.internet.private.enterprises.9.2.2.1.1.6

- The five minute average of output bits per second.

  snmputil walk ipaddress public .iso.org.dod.internet.private.enterprises.9.2.2.1.1.8

- The estimated interface's current bandwidth

  snmputil walk ipaddress public .1.3.6.1.2.1.2.2.1.5

  There is a great deal more information about Capacity Planning, and the issues surrounding it, in the Windows NT Resource Kit volume 3 by Russ Blake. It details issues relating to Log concentration and Archiving as well as other important details. There are also good books on the market that cover Capacity Planning in depth. One that comes to mind is "Capacity Planning for Computer Systems" by Tim Browning (ISBN 0-12-136490-9). This book includes: Metrics and foundations for capacity planning, Cost controls and Analysis, Planning vs Forecasting, Contingency, Systems Optimization, etc. The book covers operating systems such as NT, OS/2, UNIX, VMS, MVS, HP-UX, among others.

Summary

The motivation behind system tuning is to get the most you can out of the hardware you already own. If you decide that an upgrade is your only solution, you will find that your investment in performance tuning pays off. Your work will show you how the system should be upgraded. If you have done your homework, you will know whether you need more memory, faster disks, or a completely new processor. However, if you recorded your system's performance history, you not only did your homework, but you've studied enough to pass the test because now you can tell about latent demand and system growth.

References

Software Companies You May Want to Investigate

- BSG Systems, Inc.
  (617) 891-0000
  E-mail best1@bgs.com
- Datametrics
  (800) 869-3282
  E-mail experts@datametrics.com
- Metron
- Candle Software
- Legent
- Ziff (www.ziff.com) Serverbench and Netbench are very good benchmarking tools.
- SPEC
- BAPCO (www.bapco.com)
- Computer Capacity Management (ICCM) in Phoenix, Arizona.
- StonyBrook Services, Inc., in Bohemia, New York
- Intrak, Inc., San Diego TrendTrak
- Network General Corp., Menlo Park, California, "Reporter"

- The Information Systems Manager, Inc. (ISM)
  http://www.infosysman.com/perfman/
  The Makers of PerfMan: Performance Management for Windows NT

Literature

- *Optimizing Windows NT*-Windows NT Resource Kit Volume 3 by Russ Blake (ISBN 1-55615-619-7)
- *Windows NT Advanced Server Concepts and Planning Guide*
- *Capacity Management Review* (602-997-7374, $195.00)
- Computer Measurement Group (newsletter)
  414 Plaza Drive, Suite 209
  Westmont, IL 60559
- *High Performance Computing*-O'Reilly & Associates
- *Capacity Planning and Performance Modeling* (ISBN 0-12-035494-5)
- *Configuration and Capacity Planning for Solaris Servers* (ISBN 0-13-349952-9)
- Microsoft TV: Initial Air Date: September 5, 1995

  Microsoft Studios Catalog#: 95-MSTV-0905

  Windows NT Advanced Technical Workshop #1: Capacity Planning for Windows NT

  To order the tape call 1-800-369-5718
- The Microsoft Solution Providers that teach our ATEC Courses will be offering a Seminar called "**Capacity Planning in a Windows NT Server Environment**"