A File and Web Server Comparison:

Microsoft Windows NT Server 4.0 and Red Hat Linux 5.2 Upgraded to the Linux 2.2.2 Kernel

Mindcraft, Inc.
170 Knowles Drive, Suite 206
Los Gatos, CA 95032-1833
(408) 364-2860   Fax: (408) 364-2862
www.mindcraft.com
sales@mindcraft.com

April 13, 1999

Contents

NOTICE:

The information in this publication is subject to change without notice.

MINDCRAFT, INC. SHALL NOT BE LIABLE FOR ERRORS OR OMISSIONS CONTAINED HEREIN, NOR FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES RESULTING FROM THE FURNISHING, PERFORMANCE, OR USE OF THIS MATERIAL.

This publication does not constitute an endorsement of the product or products that were tested. This test is not a determination of product quality or correctness, nor does it ensure compliance with any federal, state or local requirements.

Mindcraft is a registered trademark of Mindcraft, Inc.

Product and corporate names mentioned herein are trademarks and/or registered trademarks of their respective companies.

The testing reported herein was performed without independent verification by Ziff-Davis and Ziff-Davis makes no representations or warranties as to the results of the test.

Section
1

Executive Summary

Microsoft Windows NT Server 4.0 is 2.5 times faster than Linux as a File Server and 3.7 times faster as a Web Server

Mindcraft tested the file-server and Web-server performance of Microsoft Windows NT Server 4.0 and Red Hat Linux 5.2 upgraded to the Linux 2.2.2 kernel (in this report referred to simply as Linux) on a Dell PowerEdge 6300/400 server. For Linux, we used Samba 2.0.3 as the SMB file server and Apache 1.3.4 as the Web server. For Windows NT Server we used its embedded SMB file server and Internet Information Server 4.0 Web server.

Figure 1 summarizes the file server peak throughput measured for each system in megabits per second (Mbits/Sec). It also shows how many test systems were needed to reach peak performance. The results show that, as a file-server, Windows NT Server 4.0 is 2.5 times faster than Linux with Samba. In addition, Windows NT Server reaches its peak performance at 2.3 times the number of test systems that Linux with Samba does.

Figure 1: File Server Peak Performance
(larger numbers are better for all metrics)

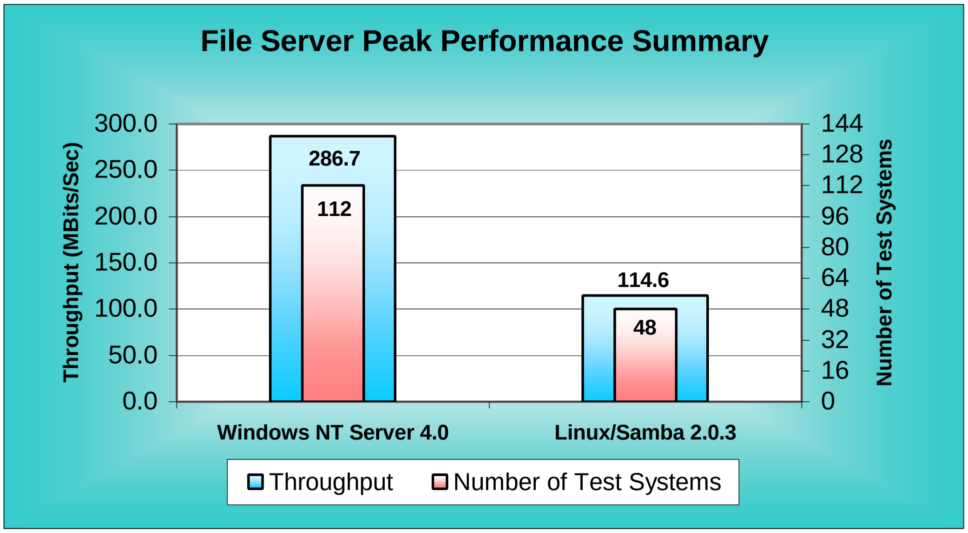
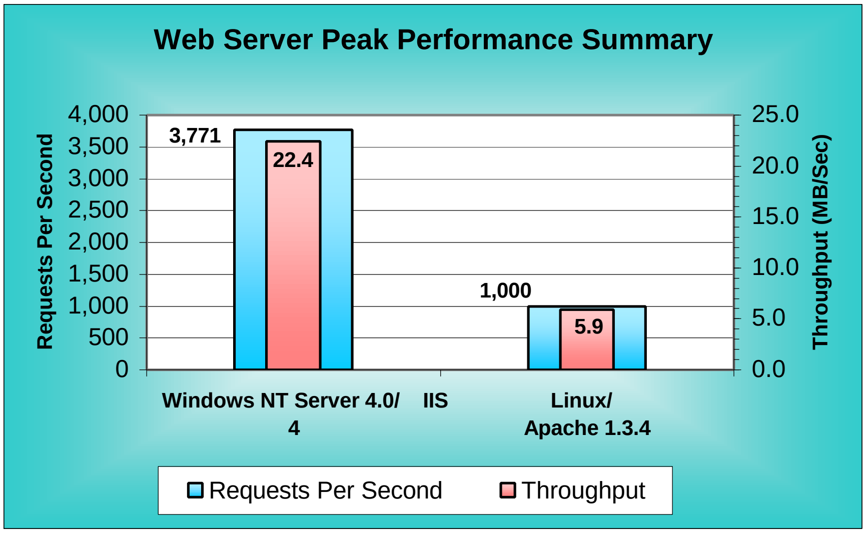


Figure 2 shows the Web server peak performance measured in HTTP GET requests per second and throughput measured in megabytes per second (MB/Sec). The Web server results show that Windows NT Server 4.0 is over 3.7 times faster than Linux with Apache. As discussed in the *Web-Server Performance* section below, the performance of Linux with Apache drops to 7% of the peak level when we increased the number of test threads above 160. Thus, Linux/Apache performance becomes unreliable under heavy load. Windows NT Server, on the other hand, continues to increase its performance up through 288 test threads. We believe that we did not reach the true peak performance of the system under Windows NT Server 4.0 because we did not have more test systems available.

Figure 2: Web Server Peak Performance
(larger numbers are better for all metrics)



Mindcraft tested file server performance using the Ziff-Davis Benchmark Operation NetBench 5.01 benchmark. We used the Ziff-Davis Benchmark Operation WebBench 2.0 benchmark to test Web server performance. We tuned each operating system, file server, and Web server according to available documentation and tuning parameters available in published benchmarks. The *Products Tested* section on page 19 gives the detailed operating system tuning we used.

Although much has been written about the performance and stability of Linux, Samba, and Apache, our tests show that Windows NT Server 4.0 performs significantly faster and handles a much larger load on enterprise class servers.

Section
2

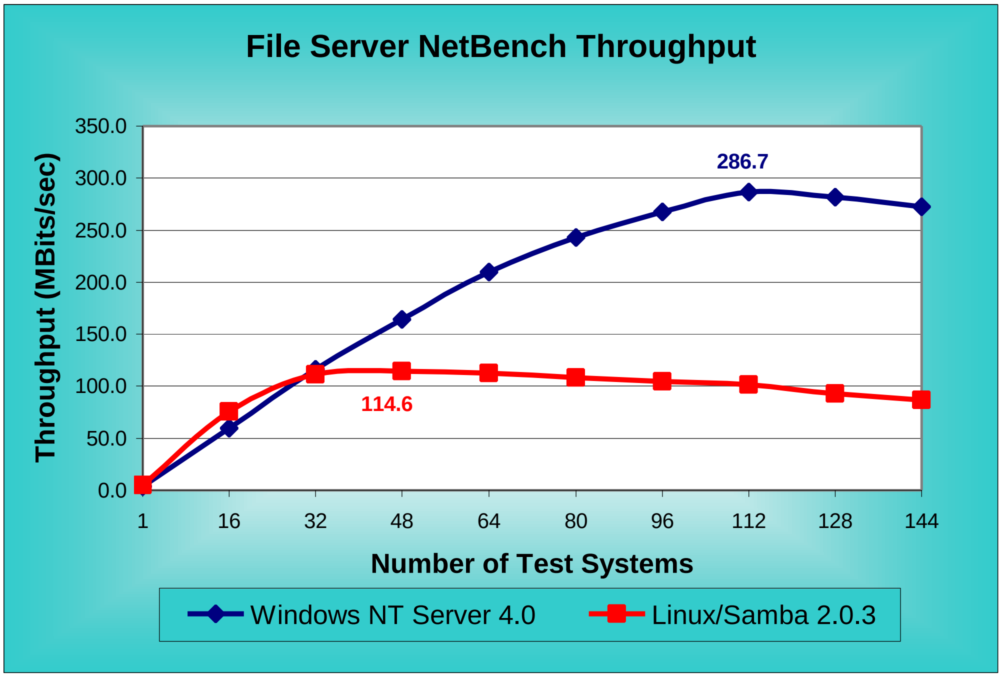
Performance Analysis

Looking at NetBench Results

The NetBench 5.01 benchmark measures file server performance. Its primary performance metric is throughput in bytes per second. The NetBench documentation defines throughput as "*The number of bytes a client transferred to and from the server each second. NetBench measures throughput by dividing the number of bytes moved by the amount of time it took to move them. NetBench reports throughput as bytes per second.*" We report throughput in megabits per second to make the charts easier to compare to other published NetBench results.

We tested file-sharing performance on Windows NT Server 4.0 and Linux on the same system. We used Samba 2.0.3 to provide SMB file sharing for Linux. Figure 3 shows the throughput we measured plotted against the number of test systems that participated in each data point.

Figure 3: NetBench Throughput Performance (larger numbers are better)



Understanding how NetBench 5.01 works will help explain the meaning of the NetBench throughput measurement. NetBench stresses a file server by using a number of test systems to read and write files on a server. A NetBench test suite is made up of a number of mixes. A mix is a particular configuration of NetBench parameters, including the number of test systems used to load the server. Typically, each mix increases the load on a server by increasing the number of test systems involved while keeping the rest of the parameters the same. We modified the standard NetBench `NBDM_60.TST` test suite to increase the number of test systems to 144 and the increment in test systems for each mix to 16 in order to test each product to its maximum performance level. The *NetBench Test Suite Configuration Parameters* section on page 17 shows the parameters we used.

NetBench does a good job of testing a file server under heavy load. To do this, each NetBench test system (called a client in the NetBench documentation) executes a script that specifies a file access pattern. As the number of test systems is increased, the load on a server is increased. You need to be careful, however, not to correlate the number of NetBench test systems participating in a test mix with the number of simultaneous users that a file server can support. This is because each NetBench test system represents more of a load than a single user would generate. NetBench was designed to behave this way in order to do benchmarking with as few test systems as possible while still generating large enough loads on a server to saturate it.

When comparing NetBench results, be sure to look at the configurations of the test systems because they have a significant effect on the measurements that NetBench makes. For example, the test system operating system may cache some or all of the workspace in its own RAM causing the NetBench test program not to go over the network to the file server as frequently as expected. This can significantly increase the reported throughput. In some cases, we've seen reported results that are 75% above the available network bandwidth. If the same test systems and network components are used to test multiple servers with the same test suite configuration, you can make a fair comparison of the servers.

File-Server Performance Analysis

With this background, let us analyze what the results in Figure 3 mean (the supporting detail data for this chart are in Appendix 1 on page 17). The three major areas to look at are:

- **Peak Performance**

  This tells you the maximum throughput you can expect from a file server. NetBench throughput is primarily a function of how quickly a file server responds to file operations from a given number of test systems. So a more responsive file server will be able to handle more operations per second, which will yield higher throughput.

- **Shape of the Performance Curve**

  How a product performs as a function of load is perhaps the most meaningful information NetBench produces. If performance drops off rapidly after the peak, users may experience significant unpredictable and slow response times as the load on the server increases. On the other hand, a product whose performance is flat or degrades slowly after the peak can deliver more predictable performance under load.

- **Where Peak Performance Occurs**

  How quickly these products reach their peak performance depends on the server hardware performance, the operating system performance, and the test system performance. In this case, we tested a fast server platform with significantly slower clients. This test lab setup meant that small numbers of clients could not generate enough requests to utilize the server processors fully. So the part of the throughput performance curve to the left of the peak does not tell us anything of interest. The performance curve after the peak shows how a server behaves as it is overloaded.

File-Server Performance Conclusions

Windows NT Server 4.0 is a high-performance file server that helps users be more productive than a Linux/Samba file server would. We base this conclusion on the following analysis:

- The peak performance for Windows NT Server 4.0 was **286.7 Mbits/second at 112 test systems** while Linux/Samba reached a peak of **114.6 Mbits/second at 48 test systems**. Thus, Windows NT Server reached a peak performance level that was 2.5 times that of Linux/Samba. The test results also show that Windows NT Server 4.0 is 43.5% faster than Linux/Samba at 48 test systems. Only on a lightly loaded server, with 1 or 16 test systems, does Linux/Samba outperform Windows NT Server and then by only 26%.

- The shapes of the performance curves for both Windows NT Server 4.0 and Linux/Samba indicate that we reached peak performance and went beyond it. Performance for both Windows NT Server 4.0 and Linux/Samba degrades slowly as the load is increased past the peak performance load. So both systems should deliver predictable performance even under overload conditions.

- The peak performance for Windows NT Server 4.0 occurs at 112 test systems while that for Linux/Samba occurs at 48 test systems. This means that the Windows NT Server 4.0 can handle over 2.3 times the load of Linux/Samba while delivering significantly better performance.

Looking at WebBench Results

In order to understand what the WebBench measurements mean you need to know how WebBench 2.0 works. It stresses a Web server by using a number of test systems (called clients in the WebBench documentation) to request URLs. Each WebBench test system can be configured to use multiple worker threads (threads for short) to make simultaneous Web server requests. By using multiple threads per test system, it is possible to generate a large enough load on a Web server to stress it to its limit with a reasonable number of test systems. The other factor that will determine how many test systems and how many threads per test system are needed to saturate a server is the performance of each test system.

The number of threads needed to obtain the peak server performance depends on the speed of the test systems and the server. Because of this, it is not meaningful to compare performance curves generated using different test beds. However, it is meaningful to compare the peak server performance measurements from different test beds, as long as the true peak has been reached, because each server sees enough requests from WebBench test systems to make it reach its maximum performance level. In addition, it is meaningful to compare performance curves for different servers based on the number of threads, not systems, at each data point only if the same test bed is used. That is why our graphs below show the number of test threads for each data point.

WebBench can generate a heavy load on a Web server. To do this in a way that makes benchmarking economical, each WebBench thread sends an HTTP request to the Web server being tested and waits for the reply. When it comes, the thread immediately makes a new HTTP request. This way of generating requests means that a few test systems can simulate the load of hundreds of users. You need to be careful, however, not to correlate the number of WebBench test systems or threads with the number of simultaneous users that a Web server can support since WebBench does not behave the way users do.

Web-Server Performance Analysis

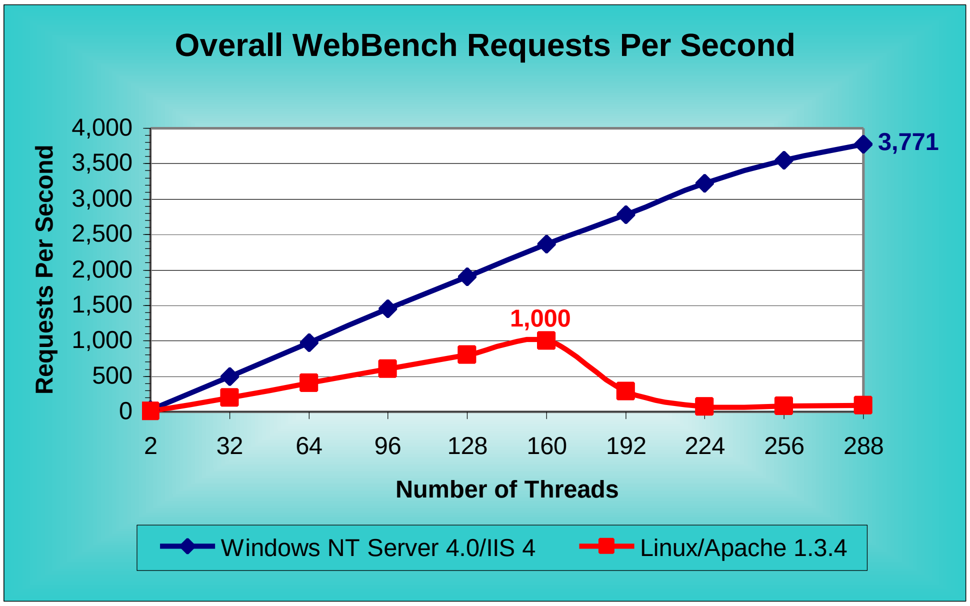
WebBench 2.0 gives two metrics for comparing Web server performance:

- The number of HTTP GET requests per second.
- The number of bytes per second that a Web server sends to all test systems.

We tested both Web servers using the standard WebBench `zd_static_v20.tst` test suite, modified to increase the number of test systems to 144 and the increment in test systems for each mix to 16 in order to test each product to its maximum performance level. This standard WebBench test suite uses the HTTP 1.0 protocol without keepalives.

Figure 4 shows the total number of requests per second for both Windows NT Server 4.0/IIS 4 and Linux/Apache 1.3.4. The x-axis shows the total number of test threads used at each data point; a higher number of threads indicate a larger load on the server. Figure 5 gives the corresponding throughput for each platform.

Figure 4: HTTP Requests/Second Performance (larger numbers are better)



With this background, let us analyze what the results in Figure 4 and Figure 5 mean (the supporting detail data for these charts are in Appendix 2 on page 20). As with NetBench, the three major areas to look at are:

- **Peak Performance**

  This tells you the maximum requests per second that a Web server can handle and the peak throughput it can generate. A more responsive Web server will be able to handle more requests per second, which will yield higher throughput.

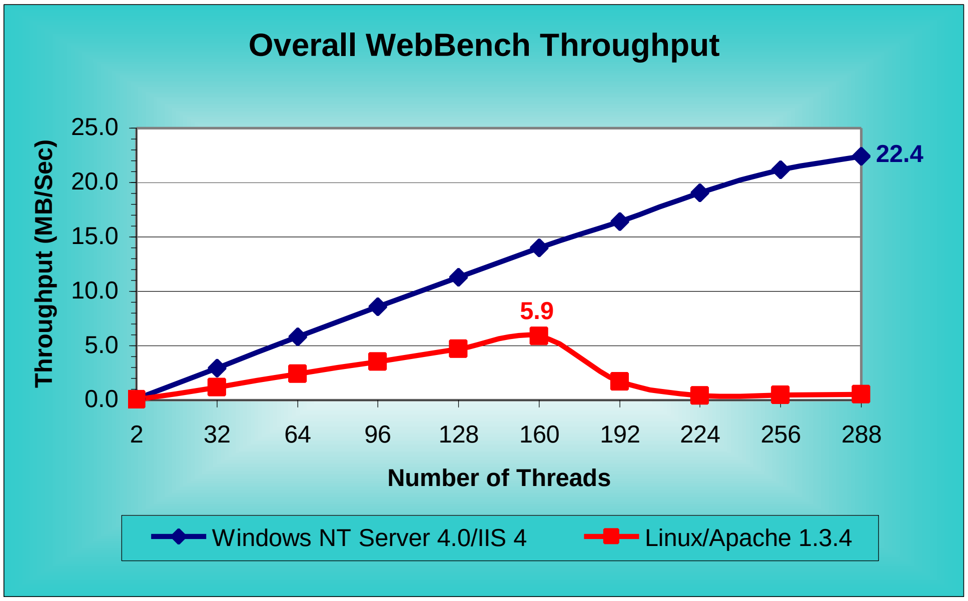
- **Shape of the Performance Curve**

  The shape of the performance curve shows how a Web server performs as a function of load. If performance drops off rapidly after the peak, users may experience significant unpredictable and slow response times as the load on the Web server increases. On the other hand, a Web server that degrades performance slowly after the peak will deliver more predictable performance under load.

- **Where Peak Performance Occurs**

  How quickly a Web server reaches its peak performance depends on the performance of the server hardware, the operating system, the Web server software, and the test systems. For this report, we tested a fast server system with significantly slower clients. This test bed setup meant that small numbers of clients could not generate enough requests to utilize the server processors fully. So the part of the performance curves to the left of the peak does not tell us anything of interest. The performance curves after the peak show how a server behaves as it is overloaded.

Figure 5: Throughput Performance (larger numbers are better)



Web-Server Performance Conclusions

Windows NT Server 4.0/IIS 4 significantly out-performs Linux/Apache 1.3.4 and provides much more predictable and robust performance under heavy load. On a given large workgroup or enterprise-class computer, Windows NT Server/IIS will satisfy a much larger Web server workload than Linux/Apache will. We base these conclusions on the following analysis:

- The peak performance for Windows NT Server 4.0/IIS 4 was **3,771 requests per second at 288 threads** while Linux/Apache 1.3.4 reached a peak of **1,000 requests per second at 160 threads**. Thus, Windows NT Server/IIS reached a peak performance level that was almost 3.8 times that of Linux/Apache. Based on the increasing performance for Windows NT Server/IIS from 256 to 288 threads, we believe that peak performance would have increased if we had more test systems available to us.

- The shapes of the requests per second and throughput performance curves for Windows NT Server 4.0/IIS 4 indicate that we probably did not reach the maximum performance levels possible with the Dell PowerEdge 6300 system. On the other hand, the performance curves for Linux/Apache indicate that we did reach peak performance and went beyond it. These results show very serious performance degradation from 1,000 requests per second at 160 threads to 68 requests per second at 224 threads. Please see our comments in the next section, *Observations*, for more information about this.

- The peak performance we measured for Windows NT Server/IIS occurred at 288 threads while that for Linux/Apache occurred at 160 threads. This means that the Windows NT Server/IIS can handle over 1.8 times the load of Linux/Apache. In addition, the test results show that Windows NT Server/IIS is 140% faster than Linux/Apache at 160 threads, the peak for Linux/Apache.

Section
3

Observations

The comments in this section are based on observations we made during the testing.

Linux Observations

- The Linux 2.2.x kernel is not well supported and is still changing rapidly. The following observations led us to this conclusion:

  - We started the tests using Red Hat Linux 5.2 but had to upgrade it to the Linux 2.2.2 kernel because its Linux 2.0.36 kernel does not support hardware RAID controllers and SMP at the same time. In addition, there are comments in the Red Hat Linux 5.2 source code noting that the SMP code is effectively Beta-level code and should not be used at the same time as the RAID driver. For this reason, we upgraded to the Linux 2.2.2 kernel, which has full support for both hardware RAID controllers and SMP to be used simultaneously. As of the date this report was written, Red Hat did not ship or support a product based on the Linux 2.2.x kernel.
  - The instructions on how to update Red Hat Linux 5.2 to the Linux 2.2.x kernel at the Red Hat Web site were complete but require care from the user. It is quite possible to put the system in a state where you must reload all software from scratch since you need to recompile and reinstall the kernel.
  - We contacted Red Hat for technical support after we saw that Linux was getting such poor performance. They told us that they only provided installation support and that they did not provide any support for the Linux 2.2.2 kernel.
  - We posted notices on various Linux and Apache newsgroups and received no relevant responses. Also, we searched the various Linux and Apache knowledge bases on the Web and found nothing that we could use to improve the performance we were observing.
  - Linux kernels are available over the Internet from www.kernel.org and its mirror sites. The issue is that there are many updates to the kernel. For example, as of the time of writing this report, we found the following kernel update history:

Linux Kernel Version	Release Date
Linux 2.2.0	January 25, 1999
Linux 2.2.1	January 28, 1999
Linux 2.2.2	February 22, 1999
Linux 2.2.3	March 8, 1999
Linux 2.2.4	March 23, 1999
Linux 2.2.5	March 28, 1999

- Linux performance tuning tips and tricks must be learned from documentation on the Net, newsgroups, and trial-and-error. Some tunes require you to recompile the kernel. We came to this conclusion from the following observations:

- The documentation on how to configure the latest Linux kernel for the best performance is very difficult to find.
- We were unable to obtain help from various Linux community newsgroups and from Red Hat.
- We were unable to find any books or web sites that addressed performance tuning in a clear and concise manner. At best we found bits and pieces of information from dozens of sites.
- The kernel source code contains comments regarding tuning and configuration.

Samba Observations

- Samba was easy to set up for file sharing once you spent a day or two learning how it fits with Linux. For people not familiar with UNIX/Linux systems, it may take longer to do the installation.
- The documentation available with Samba and in books is clear and easy to follow.

Apache Observations

- Apache's performance on Red Hat Linux 5.2 upgraded to the Linux 2.2.2 kernel is unstable under heavy load. We came to this conclusion from the following observation:
  - Performance collapses with a WebBench load above 160 threads. We verified that the problem was with Apache, not Linux, by restarting Apache at the 256 threads data point during a WebBench test run. After the restart, Apache performance climbed back to within 30% of its peak from a low of about 6% of the peak performance.
  - We tried many configurations suggested in Apache books and in comments in the Apache high performance configuration file.
  - There were no error messages in the Web server error log or operating system logs to indicate why Apache performance collapsed.

Section
4

Products Tested

Configuration and Tuning

We used the same Dell PowerEdge 6300/400 to test both Windows NT Server 4.0 and Red Hat Linux 5.2 upgraded to the Linux 2.2.2 kernel. Table 1 shows the system configuration we used.

Table 1: Dell PowerEdge 6300/400 Configuration

Feature	Configuration
CPU	4 x 400 MHz Pentium II Xeon Cache: L1: 16 KBI + 16 KB D; L2:1 MB
RAM	4 GB 100 MHz SDRAM ECC
Disk	PowerEdge RAID II Adapter, 32 MB cache, RAID 0, BIOS v1.47, stripe size = 64 KB, wirte policy = writeback, read policy = adaptive, cache policy = directIO, raid across two channels, with two logical drives: Drive C/OS: 1 x 9 GB Seagate Cheetah, Model ST39102LC, 10,000 RPM; two partitions – one for each OS Drive D/Data: 8 x 4 GB Seagate Barracuda, Model ST34573WC, 7,200 RPM; two partitions – one data partition for each OS
Networks	4 x Intel EtherExpress Pro 100B Network Interface Cards

Windows NT Server 4.0 Configuration

- Windows NT Server 4.0 Enterprise Edition with Service Pack 4 installed
- Used 1024 MB of RAM (set maxmem=1024 in `boot.ini`)
- Server set to maximize throughput for file sharing
- Foreground application boost set to NONE
- Set registry entries:
  HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services:
  - \NDIS\Parameters\ProcessorAffinityMask=0
  - Tcpip\Parameters\Tcpwindowsize = 65535
- Used the NIC control panel to set the following for all four NICs:
  - Receive Buffers = 200 (default is 32; this setting is under “Advanced Settings”)
  - NIC speed = 100 Mbit (default is “auto”)
- Spooler service was disabled
- Page file size set to 1012 MB on the same drive as the OS
- The RAID file systems were formatted with 16 KB allocation unit size (the /a option of the `format` command) and an NTFS file system
- Increased the file system log on the RAID file system to 65536 K using the `chkdsk f: /l:65536` command
- Used the affinity tool to bind one NIC to each CPU (ftp://ftp.microsoft.com/bussys/winnt/winnt-public/tools/affinity/)

- Rebuilt the NetBench file system between each run

Internet Information Server 4 (IIS 4) Configuration

- Used the NIC control panel to set the following for all four NICs:
  - Coalesce Buffers = 32 (default is 8)
  - Receive Buffers = 1023
  - Transmit Control Blocks = 80 (default is 16)
  - Adaptive Transmit Threshold = on (default is on)
  - Adaptive Technology = on (default is on)
  - Adaptive Inter-Frame Spacing = 1 (default is 1)
  - Map Registers = 64 (default is 64)
- SMTP, FTP, MSDTC, and Browser services were disabled
- Set registry entries: HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services:
  - \InetInfo\Parameters\ListenBackLog=200
  - \InetInfo\Parameters\ObjectCacheTTL=0xFFFFFFFF
  - \InetInfo\Parameters\OpenFileInCache=0x5000
- Using the IIS Manager
  - Set Logging – "Next Log Time Period" = "When file size reaches 100 MB"
  - Set performance to "More than 100,000"
  - Removed all ISAPI filters
  - Removed all Home directory application mappings except .asp
  - Removed permissions for "Application Settings"
- Logs on the F: drive (RAID) along with the WebBench data files
- Server set to maximize throughput for applications when doing WebBench tests

Linux Configuration

- Followed the Red Hat instructions for upgrading Red Hat Linux 5.2 to the Linux 2.2.x kernel (http://www.redhat.com/support/docs/rhl/kernel-2.2/kernel2.2-upgrade.html)
- Used the AMI 0.92 version of the MegaRAID driver for Linux (this was the latest driver available from the AMI Web site)
- Compiled the Linux 2.2.2 kernel using gcc version 2.7.2.3
- Kernel automounter support = no (was yes)
- NFS file system support = yes
- Enabled SMP support
- The following processes were running immediately before the NetBench and WebBench tests: init, (kflushd), (kpiod), (kswapd), /sbin/kerneld, syslogd, klogd, crond, inetd, bash, /sbin/mingetty [on tty2, tty3, tty4, tty5, and tty6], update (bdflush), and portmap
- The Linux kernel limited itself to use only 960 MB of RAM

Samba 2.0.1 Configuration

- Set HAVE_SHARED_MMAP = 1, HAVE_MMAP = 1, and CFLAGS = -O before compiling
- Compiled Samba using gcc version 2.7.2.3 and glibc 2.0.7
- Changes in /usr/local/samba/lib/smb.conf:
  - wide links = no
  - getwd cache = yes
  - read prediction = yes
  - status = no
  - raw read = yes
  - raw write = yes
- Rebuilt file system on the RAID between NetBench runs using the command `mke2fs –b 4096 /dev/sdb1.` Note that `mke2fs` does not support file systems with block sizes above 4096 bytes.

Apache 1.3.4 Configuration

- Set OPTIM = "-04 –m486" before compiling
- Set `EXTRA_CFLAGS=-DHARD_SERVER_LIMIT=500`
- Compiled Apache using gcc version 2.7.2.3 and glibc 2.0.7
- Disabled the following modules:
  - mod_env
  - mod_setenvif
  - mod_negotiation
  - mod_alias
  - mod_userdir
  - mod_autoindex
  - mod_access
  - mod_auth
  - mod_include
  - mod_cgi
  - mod_actions
  - mod_status
  - mod_imap
- The following parameters were set in the Apache config.h file:
  - `MinSpareServers 1`
  - `MaxSpareServers 290`
  - `StartServers 10`
  - `MaxClients 290`
  - `MaxRequestsPerChild 10000`
  - `.htaccess` file access was disabled
  - `LogFormat "%h %l %u %t \"%r\" %>s %b" common`
  - `KeepAlive Off`

Section

5

Test Lab

Test Systems and Network Configuration

Mindcraft ran these tests using a total of 144 test systems made up of two types. Table 2 and Table 3 show the system configurations. We used 72 Type A systems and 72 Type B systems.

Table 2: Type A Test Systems Configuration

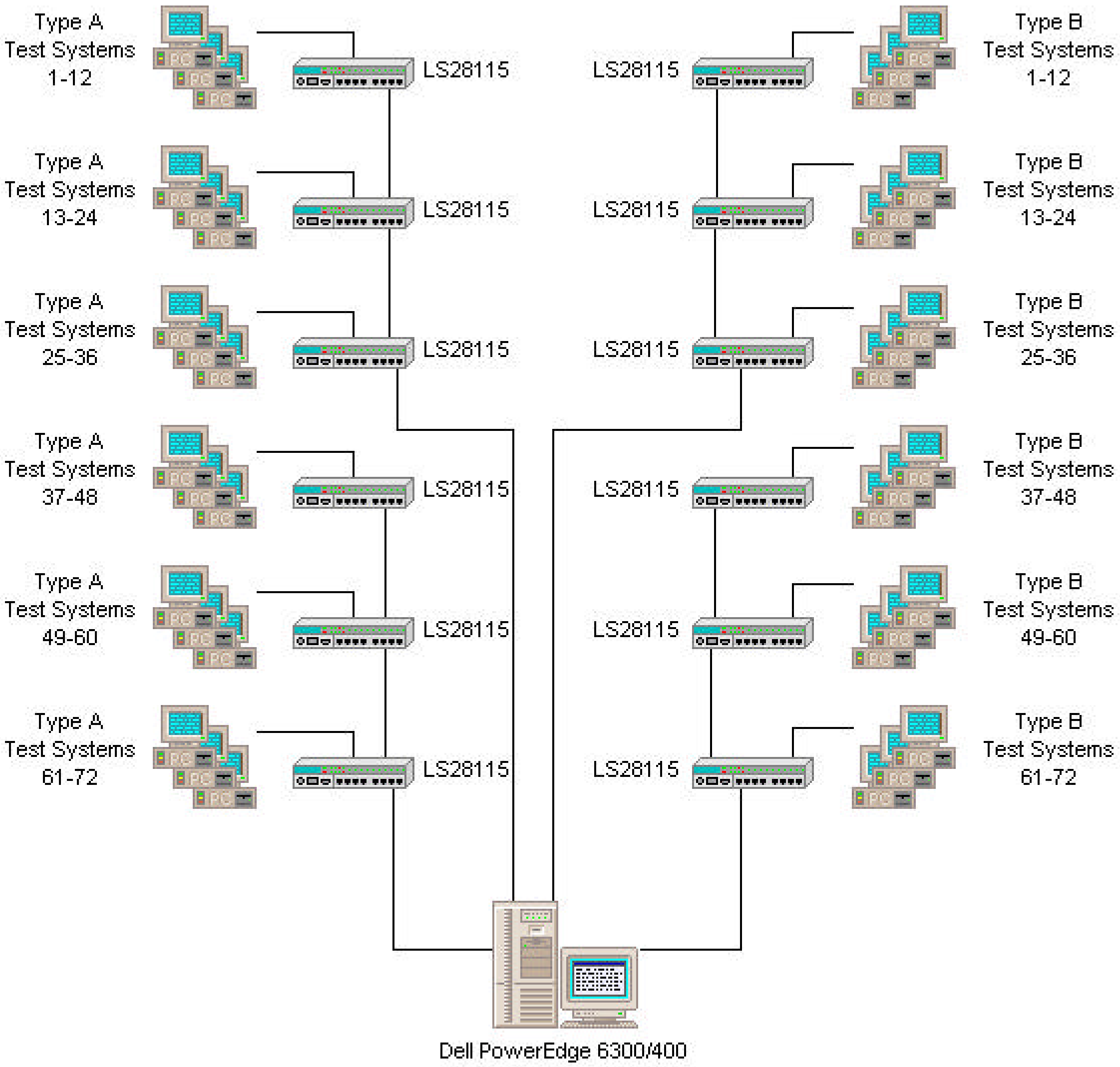
Feature	Configuration
CPU	133 MHz Pentium. All are identical Mitac systems.
RAM	64 MB
Disk	1 GB IDE; standard Windows 95 driver
Network	All systems used Intel E100B LAN Adapter (100Base-TX) using e100b.sys driver version 2.02 Network software: Windows 95 TCP/IP driver.
Operating System	Windows 95, version 4.00.950

Table 3: Type B Test Systems Configuration

Feature	Configuration
CPU	133 MHz Pentium. All are identical Mitac systems.
RAM	64 MB
Disk	1 GB IDE; standard Windows 98 driver
Network	All systems used Intel Pro/100B LAN Adapter (100Base-TX) using e100b.sys driver version 2.02 Network software: Windows 98 TCP/IP driver.
Operating System	Windows 98

Two switched networks made up of 12 Bay Networks LS28115 switches connected the test systems to the Dell PowerEdge 6300. Figure 6 shows the test lab configuration.

Figure 6: Test Lab Configuration



Section

6

Mindcraft Certification

Mindcraft, Inc. conducted the performance tests described in this report between March 10 and March 13, 1999. Microsoft Corporation sponsored the testing reported herein.

Mindcraft certifies that the results reported accurately represent the file-server performance of Microsoft Windows NT Server 4.0 and Red Hat Linux 5.2 upgraded to the Linux 2.2.2 kernel with Samba 2.0.1 running on a Dell PowerEdge 6300/400 as measured by NetBench 5.01. Also, we certify that the Web-server performance reported for Windows NT Server 4.0 with IIS 4 and for Red Hat Linux 5.2 upgraded to the Linux 2.2.2 kernel with Apache 1.3.4 accurately represent the WebBench 2.0 measurements we made on a Dell PowerEdge 6300/400.

Our test results should be reproducible by others using the same test lab configuration, the same Dell computer, and the software configurations and modifications documented in this report.

Appendix
1

NetBench Configuration and Results

NetBench Test Suite Configuration Parameters

Items in blue were modified from the standard WebBench 2.0 NBDM_60.TST test.

Parameter	Value	Comment
Ramp Up	30 seconds	This is the amount of time at the beginning of a test mix during which NetBench ignores any file operations that occur.
Ramp Down	30 seconds	This is the amount of time at the end of a test mix during which NetBench ignores any file operations that occur.
Length	660 seconds	The total time for which NetBench will run a test. It includes both the Ramp Up and Ramp Down times.
Delay	5 seconds	How long a test system is to wait before starting a test after it is told by the controller to start. Each test system will pick a random number less than or equal to this value to stagger the start times of all test systems.
Think Time	2 seconds	How long each test system will wait before performing the next piece of work.
Workspace	20 MB	The size of the data files used by a test system, each of which has its own workspace.
Save Workspace	Yes	The last mix has this parameter set to No to clean up after the test is over.
Number of Mixes	10	Each mix tests the server with a different number of test systems. Mix 1 uses 1 system, Mix 2 uses 16 systems, and subsequent mixes increment the number of test systems by 16.
Number of Clients	144	The maximum number of test systems available to be used by any test mix. The actual number of test systems that participate in a mix depends on the number specified in the mix definition and whether an error occurred to take a test system out of a particular mix.

Windows NT Server 4.0 on a Four-Processor Dell PowerEdge 6300/400

Mix Name	Clients Participating	Total Throughput (bytes/sec)	Total Throughput (Mbits/sec)
dm_1_client	1	511,955	3.9
dm_16_clients	16	7,839,439	59.8
dm_32_clients	32	15,268,347	116.5
dm_48_clients	48	21,552,015	164.4
dm_64_clinets	64	27,500,569	209.8
dm_80_clients	80	31,828,415	242.8
dm_96_clients	96	35,090,551	267.7
dm_112_clients	112	37,576,067	286.7
dm_128_clients	128	36,936,733	281.8
dm_144_clients	144	35,725,547	272.6

Linux/Samba 2.0.3 on a Four-Processor Dell PowerEdge 6300/400

Mix Name	Clients Participating	Total Throughput (bytes/sec)	Total Throughput (Mbits/sec)
dm_1_client	1	655,218	5.0
dm_16_clients	16	9,893,001	75.5
dm_32_clients	32	14,554,400	111.0
dm_48_clients	48	15,022,527	114.6
dm_64_clinets	64	14,728,998	112.4
dm_80_clients	80	14,156,420	108.0
dm_96_clients	96	13,669,234	104.3
dm_112_clients	112	13,335,449	101.7
dm_128_clients	128	12,145,976	92.7
dm_144_clients	144	11,408,018	87.0

Appendix
2

WebBench Configuration and Results

WebBench Test Suite Configuration Parameters

Items in blue were modified from the standard WebBench 2.0 `zd_static_v20.tst` test. This is a 100% static workload that uses HTTP 1.0 without keepalives.

Parameter	Value	Comment
Ramp Up	30 seconds	This is the amount of time at the beginning of a test mix during which WebBench ignores any file operations that occur.
Ramp Down	30 seconds	This is the amount of time at the end of a test mix during which WebBench ignores any file operations that occur.
Length	300 seconds	The total time for which WebBench will run a test. It includes both the Ramp Up and Ramp Down times.
Delay	0 seconds	How long a test system is to wait before starting a test after it is told by the controller to start. Each test system will pick a random number less than or equal to this value to stagger the start times of all test systems.
Think Time	0 seconds	How long each test system will wait before performing the next piece of work.
Number of Threads	2	The number of worker threads used on each test system to make requests to a Web server. The total number of threads in a mix is the number of threads times the number of clients in that mix.
Receive Buffer	4096 bytes	The size of the buffer WebBench uses to receive data sent from a Web server.
% HTTP 1.0 Requests	100 %	The percentage of HTTP requests that are made according to the HTTP 1.0 protocol. WebBench does not support keepalives for HTTP 1.0.
Number of Mixes	10	Each mix tests the server with a different number of test systems. Mix 1 uses 1 system, Mix 2 uses 16 systems, and subsequent mixes increment the number of test systems by 16.
Number of Clients	144	The maximum number of test systems available to be used by any test mix. The actual number of test systems that participate in a mix depends on the number specified in the mix definition and whether an error occurred to take a test system out of a particular mix.

Windows NT Server 4.0/IIS 4 on a Four-Processor Dell PowerEdge 6300/400

Number of Threads	Requests per Second	Throughput (MB/sec)
2	25	0.1
32	493	3.0
64	976	5.8
96	1,456	8.6
128	1,907	11.3
160	2,363	14.0
192	2,778	16.4
224	3,222	19.1
256	3,544	21.1
288	3,771	22.4

Linux/Apache 1.3.4 on a Four-Processor Dell PowerEdge 6300/400

Number of Threads	Requests per Second	Throughput (MB/sec)
2	12	0.1
32	199	1.2
64	403	2.4
96	601	3.5
128	802	4.7
160	1,000	5.9
192	287	1.7
224	68	0.4
256	78	0.5
288	87	0.5