SEPTEMBER 1998

NOVELL® RESEARCH

What's New in the NetWare 5 Operating System?

EDWARD A. LIEBING
Senior Research Engineer
Novell Developer Information

Focusing just on the NetWare 5 OS itself, this AppNote provides an overview of its new features and capabilities, along with some implementation tips.

Contents

Introduction

NetWare 5 features some important changes and additions to its operating system kernel that make it the premier networking platform on the market today. This AppNote outlines the key new features in the operating system, which include:

- Modifications to the NetWare 5 OS loader
- Changes that allow for NCP independence over multiple protocols
- New server memory model, including Virtual Memory support
- Multi-processor kernel
- Use of the NetWare Configuration file for keeping track of server configuration and SET parameters
- Full support for the NetWare Peripheral Architecture (NWPA)
- Support for PCI Hot Plug and Intelligent I/O (I2O) technologies
- Support for the WinSock 2 programming interface
- Java-based GUI console
- New server console commands

This AppNote also tells where to find a list of the hardware and software products that are certified to run under NetWare 5. This online list is continually being updated as new products are tested and certified.

NetWare 5 OS Loader Changes

One new feature in NetWare 5 concerns the OS loader itself. Among other changes, you no longer have to type LOAD before the name of the NetWare Loadable Module (NLM) or utility you want to invoke. For example, to bring up the new MONITOR utility, you can simply go to the server prompt and type:

```
MONITOR <Enter>
```

The NetWare loader will proceed to start the MONITOR utility.

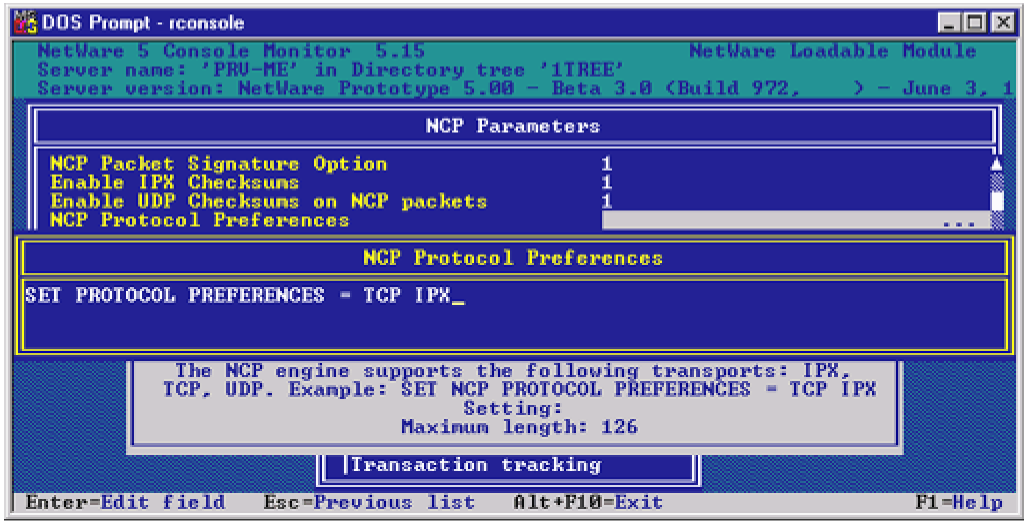
NCP Protocol Independence

The NetWare 5 operating system is NCP protocol independent. This means that internally to the NetWare OS, NetWare Core Protocols (NCPs) can make and receive specific requests for services which are handled either through IP, IPX, UDP, or a combination of these protocols. To see which protocols are loaded and in which order, NetWare 5 has a number of console commands and SET parameters to assist you.

The server console commands specific to NCPs include NCP STATS, NCP ADDRESSES, NCP TRACE, and NCP DUMP. To see which IPX, TCP and UDP addresses are loading and in which order, type **`NCP ADDRESSES <Enter>`** at the server prompt. This list shows the order in which the protocols were loaded through the AUTOEXEC.NCF or the NetWare Configuration file.

If you thought you were loading the IP address before IPX and you see that IPX is indeed loading before IP, you can use the SET parameter to change the order in which they are designated, regardless of which protocol you bind first through the AUTOEXEC.NCF file or through the NetWare Configuration file. In the MONITOR utility, select the Server Parameters option from the "Available Options" window. Select NCP option, then highlight the NCP Protocol Preferences and press **`<Enter>`**. You can then type in the order that you wish the protocols to be used by incoming service requests. Once saved, this information is written in the NetWare Configuration file and used each time the server is brought up.

Figure 1: *NetWare 5 Console Monitor screen where you set the NCP protocol preference.*



```
NetWare 5 Console Monitor  5.15                      NetWare Loadable Module
Server name: 'PRV-ME' in Directory tree '1TREE'
Server version: NetWare Prototype 5.00 - Beta 3.0 (Build 972,      ) - June 3, 1

                            NCP Parameters
  NCP Packet Signature Option                   1
  Enable IPX Checksums                          1
  Enable UDP Checksums on NCP packets           1
  NCP Protocol Preferences                                   ...

                        NCP Protocol Preferences
SET PROTOCOL PREFERENCES = TCP IPX_

        The NCP engine supports the following transports: IPX,
       TCP, UDP. Example: SET NCP PROTOCOL PREFERENCES = TCP IPX
                             Setting:
                        Maximum length: 126

                    Transaction tracking
Enter=Edit field   Esc=Previous list   Alt+F10=Exit                    F1=Help
```

The NCP STATS command initially shows you the number of NCP requests that have been processed through the OS engine up to that point. For instance, the numbers shown to the left of the ProcessNCPPacket Requests and the ProcessNCPPacketWithLength Requests entries are NCP requests that were processed through the IPX or IP protocols, as well as through CLIB. While not in use at this time, the ExecuteNCPPacket allows developers to designate the packet receive buffer length of differing sizes. Finally, the NCPPacketReceiveHandler entry shows only the NCP requests that were processed through the IPX protocol.

The NCP TRACE command allows you to view incoming NCP requests either on the server console screen or save that information to a file. The NCP DUMP command allows you to view how all NCPs are currently being implemented as well as often they are executed. This information is mostly useful to developers and hardcore system integrators.

For more information on implementing SLP and IP or running IP/IPX mixed environments, refer to the articles entitled "Migrating to Pure IP with NetWare 5" and "Compatibility Mode Installation and Configuration" in this issue.

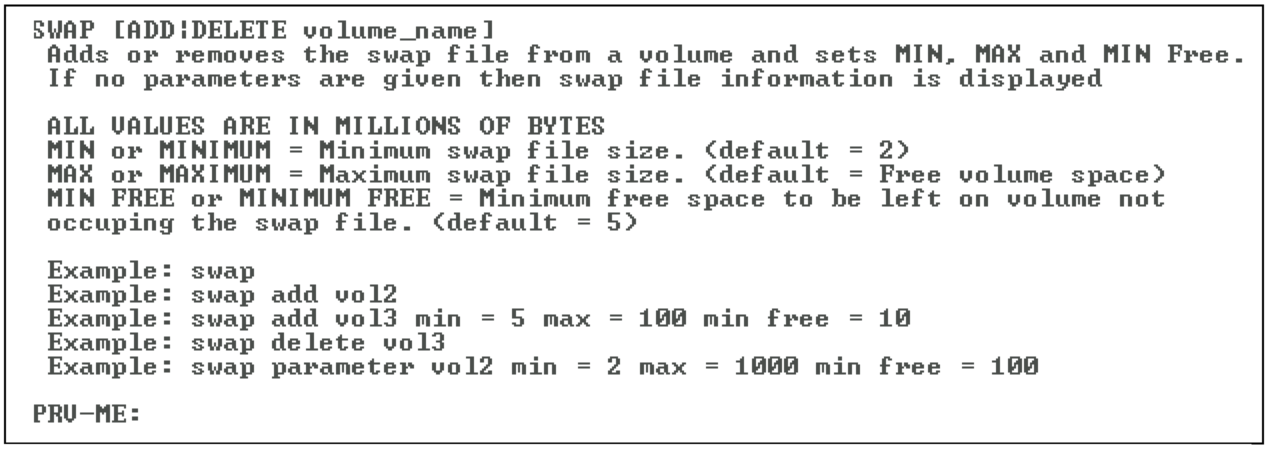
NetWare 5's Memory Model

Virtual Memory (VM) support has been added to the NetWare 5 operating system kernel and allows you to use more than your primary memory. In earlier versions of NetWare, if you have a server with 64MB of RAM, that was all the memory that the server could use to load applications and NLMs, as well as offer memory for printing, user data creation and manipulation, and application access. With Virtual Memory, server applications and NLM programs can be swapped into and out of memory and stored on the hard disk.

By default, VM allocates a 2MB swap file that is placed at the root of the SYSTEM volume. From there, the swap file will grow and shrink, depending on the number of NLMs and server applications you have running, and how much overall memory is needed by the system to maintain other server apps and users. You can also select a different volume upon which to place the swap file (for performance reasons, it's best to move the swap file to the least-used volume so the disk drive heads can better accommodate swap file page-ins and page-outs). Or you can have multiple swap files, and NetWare 5 will stripe them automatically for better performance (note that swap files are not cached).

Use the SWAP console command to either change the swap file to a different volume or to add another swap file. To see all of the options of the SWAP console command, type **`HELP SWAP <Enter>`** at the server console prompt. You will see a help screen similar to the one shown in Figure 2 with instructions on how to use this command.

Figure 2: *The SWAP command options of the HELP SWAP screen.*



```
SWAP [ADD|DELETE volume_name]
 Adds or removes the swap file from a volume and sets MIN, MAX and MIN Free.
 If no parameters are given then swap file information is displayed

 ALL VALUES ARE IN MILLIONS OF BYTES
 MIN or MINIMUM = Minimum swap file size. (default = 2)
 MAX or MAXIMUM = Maximum swap file size. (default = Free volume space)
 MIN FREE or MINIMUM FREE = Minimum free space to be left on volume not
 occuping the swap file. (default = 5)

 Example: swap
 Example: swap add vol2
 Example: swap add vol3 min = 5 max = 100 min free = 10
 Example: swap delete vol3
 Example: swap parameter vol2 min = 2 max = 1000 min free = 100

PRV-ME:
```

The Virtual Memory model contains three basic elements: primary storage, secondary storage, and the swap file. Primary storage is the physical memory that the server has; secondary storage is how much of running applications can be placed on the server's hard disk in the swap file. Just how much of the server applications actually remain in memory or is placed on the server's disk is up to VM to decide.

The goal of VM is to allocate memory to those processes of an application that is needed and necessary to keep it running smoothly, while swapping out to disk those parts that don't affect performance. For example, when loading, applications may allocate large memory chunks for future processing needs; these memory chunks can be allocated to the swap file and then primary storage can be allocated when the application actually needs physical memory. This effort helps balance system response to demanding processes that have present memory needs.

However, if your system does not have enough physical memory to handle all of the essential elements of the programs you are running, then the essential elements are written out to the swap file, thereby impacting performance. For performance reasons, applications should not "page out" to the swap file more than twice the physical memory that you have in the server. While you can load more NLMs than twice the physical server memory, such actions will start impacting performance.

In short, VM gives NetWare an infrastructure to support shared memory, to over-commit memory by allowing you to load more server applications. VM also allows servers to have larger physical memory configurations (up to 4 GB) and implements the use of expandible and compressible stacks. VM supports

Java and Java-based applications, thereby expanding the server's role as an application development environment, for which VM is the infrastructure.

Features of Virtual Memory

Some of the features of VM include memory states, shared memory, kernel pageable memory, expandible/collapsible stacks. VM has different memory states, which are tied to performance issues. These states include reserved memory, committed memory, auto-committed memory, and locked memory.

Reserved memory allows applications to perform their own memory management (Java uses this a lot). *Committed memory* is a means to reserve the necessary resources to have memory available without actually allocating memory. So when an application asks for 10MB of memory, it is allocated instantaneously, which guarantees that the resources will be available)either as actual memory or as a swap file. As the application actually needs the physical memory it asked for, VM will make those memory pages available. Similarly, *auto-committed* memory is used to perform stack allocation and file locking allocation.

To learn more about how VM is set up on the server through the SET parameters, type **MONITOR !H <Enter>** at the server console prompt (the !H option displays all of the hidden SET parameters). Choose the Server Parameters option at the bottom of the Available Options screen, then select the Memory option from the Select A Parameter Category window. You can then read through the help screens on the different VM options that are available. (There will also be follow-up AppNotes and NetNotes describing the different SET parameters that are available to network administrators.)

OS Address Space and User Protected Space.
In NetWare 5, VM also added a user protected address space model that is now separate from the OS address space within physical memory. Since Java loads like an NLM, the VM model provides a method to load the Java Virtual Machine and its applets into its own protected address space.

Those NLMs that use Novell's approved and documented SDK interface don't need to be changed in order to be able to load into a protected address space. VM is backward compatible with any application that is using CLIB, and every program that runs in a protected address space and is using CLIB will automatically use VM. For example, GroupWise is written to CLIB and can therefore run in VM without modification.

User address space is shielded from the NetWare kernel OS address space, so whatever is running in the user address space cannot crash the kernel. You can load multiple NLMs to run in a single address space, which can share code and symbol tables. NetWare 5 can also shared code and symbol tables across one or more user address spaces as needed. For example, one code image of CLIB can be used by a number of programs in different user address spaces; however, they will have their own data files for each user address space.

Memory Protection Console Options

NetWare 5 comes with a number of server console commands that you can use to create user address spaces. The options are described below:

PROTECTION [[[NO] RESTART]address space name]

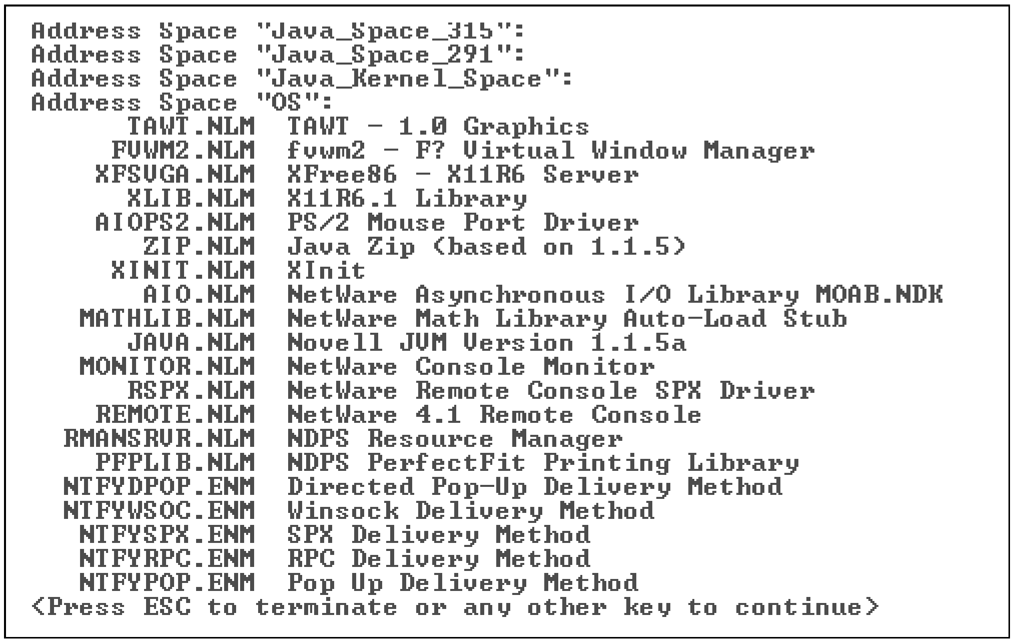
RESTARTFlag address space as restartable

ADDRESS SPACE=AS_NAME Load module in a protected address space specified by AS_NAME

If you just type **PROTECTION <Enter>**, you will see a list of the address spaces that are presently designated on the server, as well as the NLMs that are loaded in each address space, as illustrated in

Figure 3. (When you type **MODULES** at the server console prompt, you can also see a more detailed list of the modules that are loaded and in which address space they reside.)

Figure 3: *By typing "PROTECTION" you see a list of address spaces and corresponding loaded NLMs.*



```
Address Space "Java_Space_315":
Address Space "Java_Space_291":
Address Space "Java_Kernel_Space":
Address Space "OS":
      TAWT.NLM  TAWT - 1.0 Graphics
     FVWM2.NLM  fvwm2 - F? Virtual Window Manager
    XFSVGA.NLM  XFree86 - X11R6 Server
      XLIB.NLM  X11R6.1 Library
    AIOPS2.NLM  PS/2 Mouse Port Driver
       ZIP.NLM  Java Zip (based on 1.1.5)
     XINIT.NLM  XInit
       AIO.NLM  NetWare Asynchronous I/O Library MOAB.NDK
   MATHLIB.NLM  NetWare Math Library Auto-Load Stub
      JAVA.NLM  Novell JVM Version 1.1.5a
   MONITOR.NLM  NetWare Console Monitor
      RSPX.NLM  NetWare Remote Console SPX Driver
    REMOTE.NLM  NetWare 4.1 Remote Console
  RMANSRVR.NLM  NDPS Resource Manager
    PFPLIB.NLM  NDPS PerfectFit Printing Library
  NTFYDPOP.ENM  Directed Pop-Up Delivery Method
  NTFYWSOC.ENM  Winsock Delivery Method
   NTFYSPX.ENM  SPX Delivery Method
   NTFYRPC.ENM  RPC Delivery Method
   NTFYPOP.ENM  Pop Up Delivery Method
<Press ESC to terminate or any other key to continue>
```

You can also use the **PROTECT** console command to load modules that are called up by an .NCF file into a protected name address space (using the name of the .NCF file to designate the address space name):

PROTECT **[**`NCF filename`**]**

Loads NLMS from .NCF file in a protected address space.

The NetWare 5 MONITOR utility now has a Virtual Memory option within the "Available Options" window that has two subcategories in which you can view information about Virtual Memory. The initial screen shows VM information, the second shows address space information, and the last is the swap file information. All of this information will be covered in a future AppNote or NetNote

Multi-Processor Kernel (MPK)

The NetWare 5 operating system contains a multi-processor kernel (called the MPK) that fully supports symmetrical multi-processing (SMP) hardware. Because of this, NetWare 5 does not need to load a second SMP driver as the NetWare 4.11 version of SMP NetWare does. NetWare 5's MPK is fully multithreaded to perform thread scheduling interrupt and exception handling, multiprocessor synchronization, and it also adds support for pre-emption.

During the installation process, NetWare 5 checks the server hardware to see if it has more than one processor. Because MPK was written to the Intel 82489DX interrupt controller specification, it can support up to 32 processors (this number can increase in future releases as different controller specifications are implemented).

Server-based NLMs that have been written to work on SMP NetWare 4.11 will also work with NetWare 5's MPK without modification. Developers writing to NetWare 5's CLIB can use its multi-threaded APIs and therefore implement MPK support to their NLMs.

A number of core NetWare services are multithreaded and will therefore take advantage of MPK server systems. For example, Java can take advantage of MPK servers by allowing applications to specify which processor to bind to. NetWare 5's memory system and disk subsystem is mostly multithreaded to take better advantage of MPK systems. Other multithreaded support includes the NetWare Debugger, ODI/LSL, Media Manager, Virtual Memory, and Abend recovery.

Currently, the LAN and file subsystems take only partial advantage of MPK multithreading capabilities. But these include important elements, such as memory allocation, DFS read and write paths, as well as data writes through data buffers to cache buffers can be put on other processors. Also the packet receive buffers to cache buffers operation has also been multithreaded.

GroupWise comes with configuration parameters for SMP support. Oracle also has a database setting to allow for parallelism support. (Expect to see NLM updates that enhance the multithreading capabilities of NetWare core operations in the near future from Novell and other NLM vendors.) If you have the Developer Option SET parameter set to ON (default is OFF), those NLMs that don't take advantage of MPK multithreading capabilities will give the following information in the color green as they load on the server:

```
<Modulename> does not have any XDC data.
```

MPK Console Commands

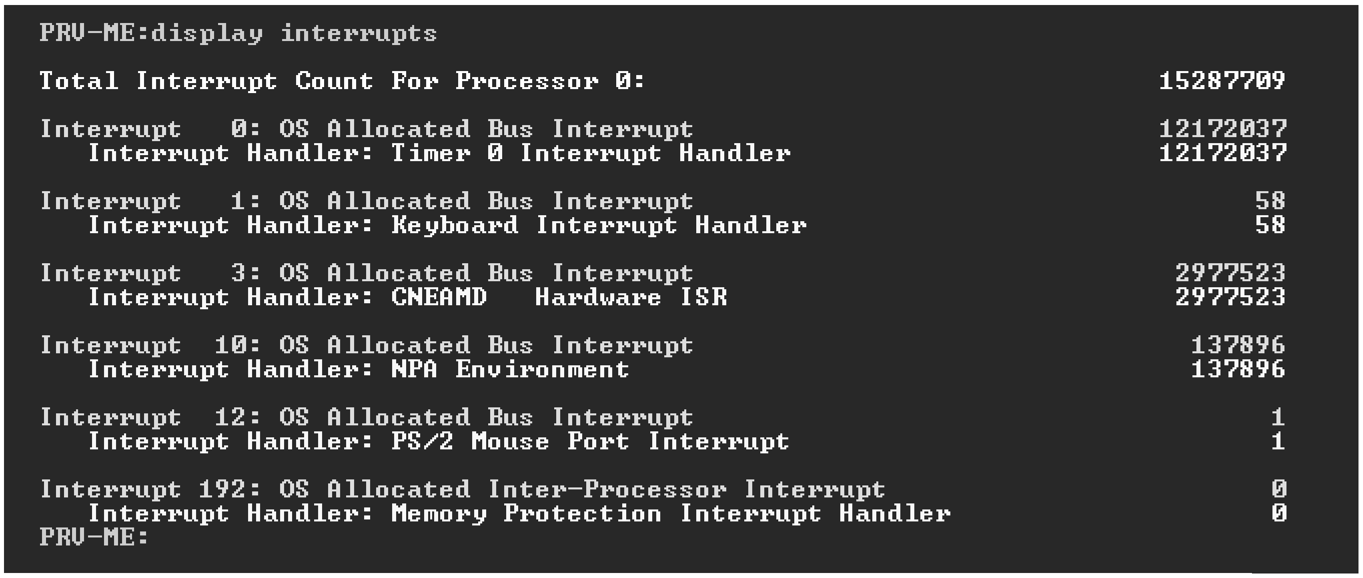
NetWare 5 comes with a number of console commands to help view information about the processors and interrupts running on the server. These commands include:

```
DISPLAY INTERRUPTS [I# I# I# | ALL] [PROC | REAL] [ALLOC]
```

- I# equals the interrupt number you wish to view. If no number is specified, all numbers are displayed.
- ALL will also display all interrupts
- PROC displays per processor interrupt information
- ALLOC displays allocated interrupts
- REAL displays interrupts which occurred while the OS was in real mode and were reflected back to protected mode for servicing.

If you use type **`DISPLAY INTERRUPTS <Enter>`** you will see a screen similar to Figure 4.

Figure 4: *Screen shot displaying the results of the "display interrupts" command.*



```
PRV-ME:display interrupts

Total Interrupt Count For Processor 0:                            15287709

Interrupt   0: OS Allocated Bus Interrupt                         12172037
   Interrupt Handler: Timer 0 Interrupt Handler                   12172037

Interrupt   1: OS Allocated Bus Interrupt                               58
   Interrupt Handler: Keyboard Interrupt Handler                        58

Interrupt   3: OS Allocated Bus Interrupt                          2977523
   Interrupt Handler: CNEAMD   Hardware ISR                        2977523

Interrupt  10: OS Allocated Bus Interrupt                           137896
   Interrupt Handler: NPA Environment                               137896

Interrupt  12: OS Allocated Bus Interrupt                                1
   Interrupt Handler: PS/2 Mouse Port Interrupt                          1

Interrupt 192: OS Allocated Inter-Processor Interrupt                    0
   Interrupt Handler: Memory Protection Interrupt Handler                0
PRV-ME:
```

`DISPLAY PROCESSORS [P# P# P#]`

- P# equals the processor number(s) you wish to view (default is to display all processors).

`START PROCESSORS [P# P# P#]`

- P# equals the secondary processor number(s) you wish to start, while adding no number will start all secondary processors.

`STOP PROCESSORS [P# P# P#]`

- P# equals the secondary processor number(s) you wish to stop, while adding no number will stop all secondary processors.

To ensure that all processors are started at server boot up, place the following command in the STARTUP.NCF file:

`SET AUTO START PROCESSORS`

You can also type **`CPUCHECK <Enter>`** at the server console prompt to see the server's processor speed(s) and the L1 and L2 cache sizes. There is also other information that you can view when using the NetWare 5 MONITOR utility. This information will be covered in a future AppNote or NetNote.

The NetWare Configuration File

NetWare 5 comes with a newer approach to keeping track of information stored in .NCF files, such as the AUTOEXEC.NCF and the STARTUP.NCF. The NetWare Configuration file for NetWare 5 is a repository for commonly shared information about the server configuration, SET parameters, as well as other internal information that is required by Novell's more modular approach to OS processing.

The NetWare Configuration file stores names and values within a hierarchical database tree structure that consists of branches, nodes, and leaves. Because configuration and parameter information is stored within the Configuration file, changes are retained and implemented whenever the server is brought up or when a change in configuration is performed. Consequently, you don't have to add SET parameters to the AUTOEXEC.NCF or the STARTUP.NCF files, as these changes become the new parameters for the system even if the server is brought down.

The NetWare Configuration file is saved in the SERVCFG.000 file that is stored in the C:\NWSERVER directory as well as in the SYS:SYSTEM directory. System configuration information and parameter settings are first read out of the Configuration file, then out of the .NCF files if they are present for compatibility reasons. Changes made to the system through NetWare 5's MONITOR utility are saved to the NetWare Configuration file (NetWare 5 combines the SET parameter functionality of the SERVMAN utility into its MONITOR utility).

Helpful Console Commands

The NetWare 5 OS comes with a number of server console commands that you can use to see how SET parameters and system configuration information are presently set. While this is not an exhaustive list, below are the console commands that can help you better understand how your NetWare 5 server is presently configured.

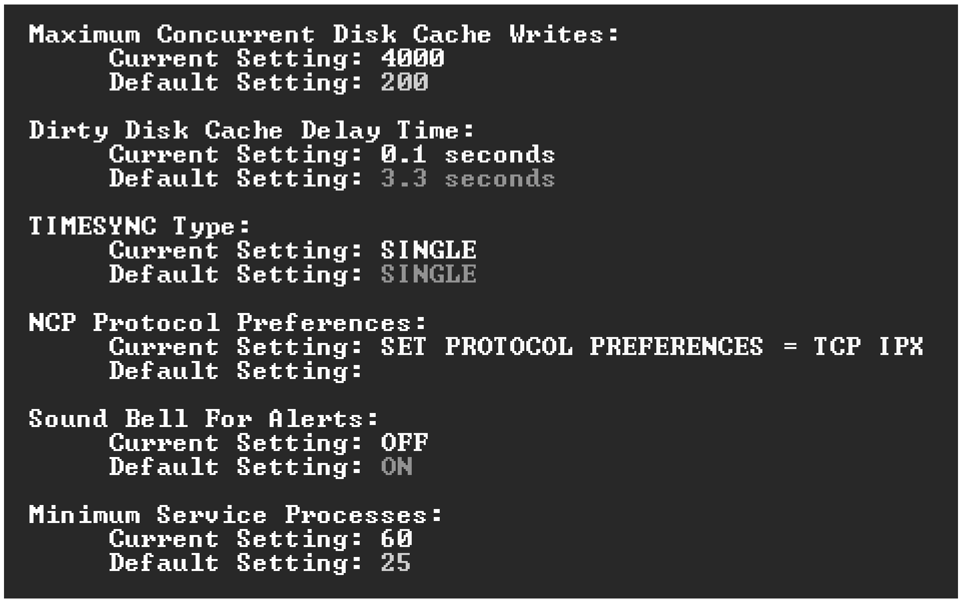
DISPLAY ENVIRONMENT

This command displays the current search paths and the current values of the set parameters for the server. The name of the SET parameter(s) displayed is highlighted in the color White with the current value in the color Yellow. The Display Environment command only displays SET parameters that are not marked as Hidden. You will also see the minimum and maximum values, as well as the default setting of the set parameter (if present) for each SET parameter displayed.

DISPLAY MODIFIED ENVIRONMENT

Use this command to display all of the "persistent" SET parameters that have been changed from their default value either through the NetWare Configuration file or through an .NCF file. You will see both the current setting and the default setting (see Figure 5).

Figure 5: *Display of the current and default settings of the SET parameters.*



```
Maximum Concurrent Disk Cache Writes:
     Current Setting: 4000
     Default Setting: 200

Dirty Disk Cache Delay Time:
     Current Setting: 0.1 seconds
     Default Setting: 3.3 seconds

TIMESYNC Type:
     Current Setting: SINGLE
     Default Setting: SINGLE

NCP Protocol Preferences:
     Current Setting: SET PROTOCOL PREFERENCES = TCP IPX
     Default Setting:

Sound Bell For Alerts:
     Current Setting: OFF
     Default Setting: ON

Minimum Service Processes:
     Current Setting: 60
     Default Setting: 25
```

RESET ENVIRONMENT

Use this command to display every set parameter that has been changed from its default value. You will be asked whether you want to change the value of each SET parameter back to its default value (Yes), to skip to the next modified SET parameter (No), change all of the SET parameters back to the default without any intervention (All), or quit the console command (Quit).

SAVE ENVIRONMENT (filename.ext)

Use this command to save the current environment (see DISPLAY ENVIRONMENT) to a specified filename that will be stored at the root of the SYS volume. You can then use the information in the file to help troubleshoot and document your server environment along with the NLMs that are loading.

SAVE MODIFIED ENVIRONMENT (filename.ext)

Use this command to save all the set parameters that have been modified to a specified filename that will be stored at the root of the SYS volume. You can use this command to help troubleshoot and document your present server environment along with the NLMs that you are loading

NetWare Peripheral Architecture

While implemented in earlier versions of NetWare/intraNetWare, the NetWare Peripheral Architecture (NWPA) was written to provide a broader and more reliable driver support for third-party host adapters and storage devices. The NWPA architecture comes as the storage configuration for NetWare 5 and replaces existing NetWare DDFS driver specifications for developers.

NWPA takes a very modular approach to driver support by breaking them into two components types: the Host Adapter Module (HAM) and the Custom Device Module (CDM). HAM aligns itself to adapter hardware, while CDM associates with storage devices or autochangers attached toa host adapter bus.

Also supported in NWPA for NetWare 5 is driver hot replacement capabilities. This ability allows users to dynamically swap out a driver that is in the server's memory with a newer version of the driver without downing the server in the process. This ability can save systems administrators a lot of down time when upgrading drivers supporting large disk volumes.

How NWPA Works

HAMs and CDMs are loaded as NLMs and they must provide resource-need information as they load, as well as how to deallocate their resources when they are unloaded. Components making up the NWPA architecture are as follows:

Media Manager.
The Media Manager is the storage management layer and the "brain" that runs the NWPA .This component provides the storage management interface between NLM applications and storage device drivers. The Media Manager take NLM application I/O requests and converts them to messages that are compatible with the NWPA architecture. The Media Manager is the layer between NWPA and the storage application.

The Media Manager adds value to the devices presented by NWPA and provides the APIs set used to configure and manage the IO System. One of the added value features Media Manager provides is software fault tolerance via the HotFix and Mirror support. HotFix reserves space on the media as spares for other parts of the media that are bad. IO requests to the bad areas are transparently redirects by HotFix to the spares. The Mirror support duplicates data by writing the same data to multiple media. Should one of the media in the Mirror set go bad data is still preserved and obtained from the remaining media in the set. Media Manager also supports all aspects of removable media.

Host Adapter Module (HAM).
HAMs are the driver components that are associated with a specific host adapter hardware. Third-party developers who are writing to the NWPA supply these program modules with their host adapters. HAMs are loaded as NetWare Loadable Modules (NLMs) and are used to route requests to the bus where a specified device is attached.

Host Adapter Interface (HAI).
The HAI is a set of APIs within the NWPA that provides an interface for HAMs to communicate with the NWPA.

Custom Device Module (CDM).
CDMs are the driver components that are associated with storage devices and are supplied by third-party developers. CDMs build device-specific commands from the I/O messages received from the Media Manager, and are also loaded as NLMs.

Custom Device Interface (CDI).
The CDI is another layer within the NWPA that provides an interface for CDMs to communicate with the NWPA.

CDM Message.
The CDM message is a data structure for an I/O message that is received by a CDM. The NWPA receives an I/O request from the Media Manager and converts the request to a CDM message to be passed to a CDM. It is from the contents of this structure that a CDM builds a request structure (SuperHACB) specific to a particular hardware-bus protocol.

Super Host Adapter Control Block (SuperHACB).
The SuperHACB is a data structure built by a CDM and contains device-specific commands. Each SuperHACB contains a HACB as one of its data members along with some additional data space.

Host Adapter Control Block (HACB).
The HACB is the protocol-specific request structure containing the data that is essential to communicate with the HAM layer. It is an I/O data structure contains a protocol-specific command block (such as SCSI or

IDE-ATA) . All I/O requests to the HAM are in the form of HACBs, and the HAM passes the commands comtained in the command block on to the devices attached to the hardware bus.

NetWare Bus Interface (NBI).
The NBI is a hardware abstraction layer that allows hardware developers to write platform independent modules. Some platforms may support more than one bus at a time and each bus can be quite different from another bus. The NBI makes platform-related issues transparent to the software modules.

Novell Event Bus (NEB).
The NEB allows multiple event producers to communicate with multiple event consumers in a synchronous or asynchronous manner. Consumers and producers must register with the event bus in order to interact with producers and consumers.

Now that the modules have been explained, here is how the flow of events occurs:

1. An NLM application, or the OS, issues an I/O request to the Media Manager, which then converts the raw request into a Media Manager Message.
2. The NWPA converts the Media Manager's message to a CDM message and passes a pointer to the CDM Message to the CDM's I/O entry point that it specifies during CDM initiation and registration.
3. The CDM builds a SuperHACB from the data in the CDM Message. The CDM then passes a pointer to this SuperHACB to the NWPA through the CDI interface.
4. The NWPA routes the HACB portion of the SuperHACB to the HAM supporting the target device associated with the I/O request.
5. The HAM sends the device command in the HACB to the appropriate adapter registers to where it is attached.
6. After the device finishes processing the command, the HAM is notified (usually by an interrupt).
7. The HAM layer does whatever is necessary to complete the HACB I/O request, places the completed information in the HACB, and then passes a pointer to the HACB to the NWPA through the HAI interface.
8. The NWPA then performs a callback to the CDM by passing a pointer to the original SuperHACB. At this point, the CDM checks to see if the requested action completed and to determine the device's error status. The CDM then returns to the NWPA the completion status.
9. The NWPA then returns the Media Manager message back to the Media Manager.
10. The Media Manager then calls the application back with the completed message.

PCI Hot Plug

NWPA supports hot plugable adapters. The Hot Plugging feature reports when an adapter error occurs, (CDMs return an adapter error code within an I/O message) and then unloads that adapter. In order to do this, the server must have loaded the Hot Plug Monitor and Hot Plug Controller Driver modules. The Hot Plug Monitor NLM monitors the adapter's status, gives commands to unload adapter support, and turns off power to the adapter that is being replaced. The Hot Plug Controller Driver turns the power off to the slots and controls the indicator lights. The Controller driver is vendor specific for the Hot Plug hardware. Currently there are only two supported platforms:

- Compaq Proliant 6500 and 7000
- IBM Netfinity 5500, and 7000 M10

Expect to see more in the future as other hardware manufacturers begin implementing these Hot Plug capabilities. As a troubleshooting note, the events to control this capability are sent and received through the Novell Event Bus (NEB). To enable these hot plug capabilities, be sure the NEB, ODINEB, NCM, NCMCON, HWDETECT, and the SBD modules are all loaded. You can type **`MODULES <Enter>`** at the server console prompt to see a list of all presently loaded NLMs.

There are some console commands to view the current status of components of the NWPA architecture. These include SCAN ALL, SCAN FOR NEW DEVICES, LIST STORAGE ADAPTERS, and LIST STORAGE DEVICE BINDINGS.

I2O Support

NetWare 5 also supports the Intelligent I/O (I2O) Architecture Specification, which is an open architecture for developing device drivers and can run independent of operating systems, processor platforms, or the system I/O bus. This allows hardware vendors to develop host/OS-independent I/O controllers that can offload much of the I/O processing burdon from the main CPU(s).

Such implementations include RAID controllers for network data storage and retrieval, ATM controllers, and NICs. Thus by exporting interrupt calls to another device, the server's CPU can spend its time performing other functions and services that don't involve interrupt processing, thereby increasing a server's I/O scalability.

The main objectives of the I2O Architecture Specification are as follows:

- To specify an architecture that is operating system or vendor-independent, and adapts to existing operating systems.
- To define an environment that coexists with existing device drivers; legacy device drivers can be ported to the new environment at the vendor's discretion.
- To provide an architecture that isolates the intelligent I/O subsystem from the host operating system. The execution environment created by the architecture enhances both system performance and functionality.
- To create an architecture that allows device drivers to scale across system platforms, from high-end workstations to high-end servers.
- To enable device drivers to port across target processors; portability refers to the device driver source code written in ANSI C.

Novell's Support for I2O

With NetWare 5, Novell ships a number of NetWare Loadable Modules which you can use, depending on the type of hardware system your server uses. These NLMs include:

I2OPCI.NLM	Free BuilD
IOPX.NLM	Free Build, used to testing performance issues

BKSTROSM.HAM	Free Build, used to testing performance issues
SCSIOSM.HAM	Free Build, used to testing performance issues

The next list of modules depends on the type of protocol you are running. These modules include the following:

ETHEROSM.LAN	Ethernet LAN driver support
FDDIOSM.LAN	FDDI LAN driver support
TOKENOSM.NLM	Token-Ring LAN driver support
NBI.NLM	NetWare Bus Interface-the hardware abstraction layer support
MSM.NLM	Media Support Manager layer support
ETHERTSM.NLM	Ethernet support module
TOKENTSM.NLM	Token-Ring support module
FDDITSM.NLM	FDDI support module

NetWare 5 will automatically detect if the proper hardware in place, such as I2O-aware mother board designs and add-on network boards. With them, NetWare servers should be able to achieve faster throughput for both the I/O channels and the NetWare OS services. By splitting the workload to embedded processors on I/O controllers, user/server-requested data reads and writes can improve dramatically under heavy workloads.

WinSock 2 Implementation

A big plus for developers, NetWare 5 fully supports the WinSock 2 programming interface as an industry standard. WinSock 2 is adapted from the Microsoft Windows WinSock 1.1 specification and uses sockets as its means to transport data.

NetWare 5 supports WinSock 2 on the server by supporting all of the operational modes that apply to the NetWare OS. This includes traditional blocking and non-blocking modes, as well as the new asynchronous message mode (NetWare 5 does not support Windows-specific Asynchronous Windows Message Mode). Novell considers WinSock 2 the preferred interface for applications that need to access a number of different protocols simultaneously for transport independence. This means you can change the plumbing underneath without affecting the application that is using the plumbing.

Novell's implementation includes backwards compatibility for WinSock 1.1 at the source code level. Source code compatibility means that the WinSock 1.1 APIs are preserved, allowing WinSock 1.1 application source code to run on a WinSock 2 system by including the ws2nlm.h header file and relining the code to the WinSock 2 libraries. (All header files are found on Novell's Developer SDK and for Novell's implementation, developers will need to recompile their code to implement those changes).

The WinSock 2 architecture also allows for simultaneous access to multiple transport protocols, which Novell fully supports. While WinSock 1.1 was implemented on TCP/IP only, WinSock 2 contains the Windows Open System Architecture (WOSA) compliant architecture, allowing applications access to protocols other than TCP/IP. WinSock 2 also provides for a name resolution mechanism when more than one transport protocol is in use.

Novell's Java-based GUI and Utilities

NetWare 5 comes with a server-based Java graphical user interface (GUI) that you use when installing the operating system itself. You will also see a Java-based GUI when you restart NetWare 5 after installation. The NetWare GUI allows you to run Java programs and applets on the NetWare server. Its menu is preconfigured with java utilities.

Accessible from the Start Menu of the Java GUI is ConsoleOne. It is a graphical utility written in Java that runs on both the NetWare Server and Windows workstations, that allows you to perform selected server and NDS Administration tasks. For instance, ConsoleOne provides the ability to browse the NDS tree, create and modify four NDS object types: user, group, organization and organizational unit; and full rights management within the directory structure.

ConsoleOne allows you to perform certain file system functions from the server, something you have never been able to perform up to this time. Some of the file system functions include creating, deleting, and renaming files, moving and copying files, and viewing DOS volumes. However, with this Proof of Concept version, the user's attributes and trustee assignments won't be moved or copied, nor will Read-Only files be moved as Read-Only. There is also presently no support input of Asian characters.

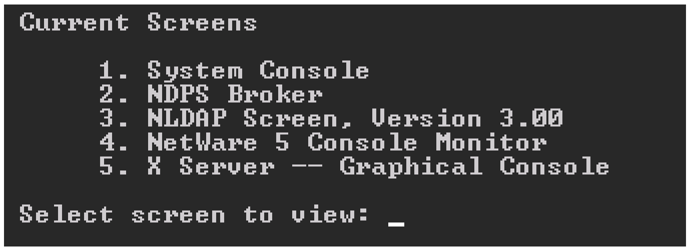
Note: It is important to remember that the minimum requirements for running ConsoleOne management utility on the server is 128MB of server memory with at least 200MHz of CPU power. Otherwise, you may be disappointed in what you see.

Because this is a "proof of concept" utility, feel free to tell Novell what you would like to see in this and future Java-based server utilities. To do this, send e-mail to consoleone@novell.com.

Another thing to remember when working at the server console is that after 15 minutes, the ConsoleOne and the common GUI shell (known as the Graphical Console screen) is written to VM disk to conserve physical memory and other server resources. Because of this, if you are using the Alt+ESC keys to thumb through the server console screens and everything comes to a standstill when you pull up the Graphical Console screen, know that it's because the screen must first be reloaded into physical memory from the VM swap file.

To avoid this unnecessary wait, instead of using the Alt+Esc keys for perusing the console screens, use the Ctrl+Esc keys. This key combination will bring up the Current Screens window, listing the different console screens that are presently open (see Figure 6). Simply type in the number of the screen you want to go to and that screen will be displayed.

Figure 6: *By using the Ctrl+Esc keys the server console brings up a Current Screens menu to allow you to choose a screen without having to toggle.*



```
Current Screens

     1. System Console
     2. NDPS Broker
     3. NLDAP Screen, Version 3.00
     4. NetWare 5 Console Monitor
     5. X Server -- Graphical Console

Select screen to view: _
```

Another suggestion found in the README file is to exit ConsoleOne when you are not using it, then relaunch it when you want to use it. You can unload it by pulling up the Graphical Console server screen, clicking on the Novell icon in the bottom left-hand corner, then selecting the Exit GUI/Yes option. (You can also exit ConsoleOne by clicking on the Graphical Console server screen and then selecting the Exit GUI/Yes option.)

To disable the Graphical Console screen from loading when the server comes up, load the NWCONFIG utility (this is the NetWare INSTALL utility renamed), select the NCF Files Options, then the Edit AUTOEXEC.NCF File, and look for the following lines at the bottom of the screen:

```
#ConsoleOne Startup NCF

STARTX.NCF

#C1START.NCF
```

Simply put a pound sign in front of the STARTX.NCF line to prevent the GUI Console splash screen from automatically loading. (The C1START.NCF line comes REMarked out, preventing ConsoleOne from loading, as well as the GUI Console splash screen. Each Java-based application that needs the common console screen will load it as the application initializes.) You can then simply type C1START at the server console prompt whenever you want to load ConsoleOne.

We should also mention that ConsoleOne can run on Windows 95/98 or Windows NT client workstation. To do this, you should have the NetWare 5 Client installed, and the workstation should have a minimum of a 200 MHZ processor and 64 MB of actual memory, with a 64 MB swap file minimum defined and 150 files allocated in the CONFIG.SYS file. Also, set the monitor to 800x600 resolution for easier screen manipulation.

ConsoleOne Architecture

ConsoleOne consists of a common shell and a set of snap-in programming interfaces. This allows developers to use the shell as the common framework where they write management pieces that "snap-in" and therefore provide management utilities.

The design of the snap-in architecture provides a console snap-in, content snap-ins and namespaces definitions. The console snap-in provides the common shell framework which developers can use to build on and provides menu-item specifications for tool bar and menu bar placements. The content snap-ins are the management utilities that Novell and third-party companies can design to snap-in to the console shell framework in order to extend network management at the server.

The snap-in architecture can also display network objects or resources within a defined namespace. These can be in logical namespaces, such as NDS, LDAP, NetWare file systems, NT Domains, or they can be physical name spaces, such as network or segment maps or inventory databases.

(For more information on the snap-in architecture, see "Novell ConsoleOne: Common Console for Management and Administration Utilities" in the March 1998 issue of *Novell Developer Notes*.)

The beginnings of the snap-in architecture can be seen through the ConsoleOne utility, which will eventually exhibit all of the functionality that you now see in the NWADMIN utility. And, because it is written in Java it will run on both the server and the workstation.

Some New NetWare 5 Console Commands

While listing all of the console commands that have changed will entail its own AppNote or two, here are some of the more pertinent commands. For more information about any of the server commands, type **`HELP <console command> <Enter>`**, at the server console prompt. To view the available console commands, type **`HELP <Enter>`** at the server console prompt.

ALIAS

While not a new command, the ALIAS command does contain some extended functionality in its NetWare 5 release. ALIAS allows you to create more easily remembered names for your most used console commands. By typing ALIAS <Enter> at the server console prompt, you will see a list of those console commands that have already been given an alias name or an abbreviated name, as shown below:

PRV-ME: ALIAS

The following is a list of defined aliases:

1. PROTOCOLS alias for <PROTOCOL>
2. SERVERID alias for <IPX INTERNAL NET>
3. VOLUMES alias for <VOLUME>
4. M alias for <MODULES>

These alias names are "persistent," so if you create an alias name for a command, it is written to the NetWare Configuration file, and will appear when you type **`ALIAS <Enter>`** again, even after you reboot the server. You can use much shorter names for console commands if you like. For example, when you type **`HELP ALIAS <Enter>`** at the console prompt, the example it uses is the letter V as an alias for VOLUME. You can also overwrite the aliases that are already in place if you choose the same letter as one that is already in use.

Note: Do not create an alias to call another alias. If you do you may receive an UNKNOWN COMMAND ??? designation if you try the command after you down the server and bring it back up again.

You can also use the ALIAS command to set up shortcuts for calling NLMs. For example, you can type **`ALIAS MON MONITOR <Enter>`** to set up an alias that calls up the MONITOR utility whenever you type **`MON <Enter>`** at the server console prompt. Again, this alias for the NLM will remain a part of the NetWare Configuration file and will be available the next time you bring the server up.

However, if you are calling up a series of modules together, it's best not to use ALIAS, but to create an .NCF file and place all of the commands needed in that file.

MODULES or M

To see the different NetWare Loadable Modules that are presently loaded on your NetWare 5 server, you can type **`M <Enter>`** at the server console prompt to use the MODULES's alias name. The different colors you see when NetWare 5 initially loads or when you bring up a modules listing is rather interesting. The color equation goes like this:

BlueThese NLMs are loaded from a hard-coded internal list that is called when SERVER.EXE executes.

RedThese NLMs are also bound in to SERVER.EXE, but are loaded from the startup partition at

C:\NWSERVER. For instance DSLOADER.NLM is always colored Red because it needs to load from the C: drive.

PurpleThese NLMs are autoloaded by another NLM as the server initially comes up. For example, LONG.NAM, CLIB, STREAMS, are colored Purple when they initially load, but they are then colored Red when you see them listed through the MODULES command.

WhiteThese NLMs are loaded from the Novell Configuration file, from any .NCF files (i.e., the AUTOEXEC.NCF or STARTUP.NCF files), or from the server console prompt.

YellowInformational messages referring to symbol information about the modules that are loading.

GreenInformational messages on the modules that are loading.

For instance, you will often see in the color green the message:

```
NLMmodulename does not have any XDC data.
```

This message appears as the multiprocessor feature of the NetWare OS loader checks to see if these NLMs are MPK-enabled. If they are not, you see this informational message, which basically means that these modules will be run in single processor mode, regardless of how many processors the server has.

Ctrl+Alt+Esc

You can use this key sequence to gracefully down the server if the server console is hung and you can't seem to get back to the server prompt. When you press Ctrl+Alt+Esc, you will see a screen that gives you two options:

1. Down the File Server and Exit to DOS.

2. Cancel the Volume Mount.

You can also press the ESCape key to exit out of this screen if you need to. If you choose 1, you will be asked again if you want to down the server and exit to DOS. Answering Y will do just that.

The NetWare Debugger

You need to use the new VDB5.EXE file in order to look at the NetWare 5 core dumps, as the old VDB for NetWare 4.x does not support NetWare 5. To get into the debugger, press Shift+Shift+Alt+Escape at the same time at the server console prompt. (Note that all server activity is suspended when the server is in debugger mode.) Here's a small list of commands for getting around in the debugger:

h or .hPulls up Help

?Instruction pointer

ddsDumps the stack

swPerforms a stack walk

lWrites to the Abend log

vViews the different console screens states before the server crash

.mDisplays the modules and their versions

.cPerforms a core dump, with a new mini core dump feature added when don't have enough disk space for a full core dump.

Compatible Software and Hardware for NetWare 5

As NetWare 5 ships, there are over 200 appplications that have been tested and certified to run on this platform. A list of these applications and products can be found on the World Wide Web at the following URL:

http://developer.novell.com/netware5

This list will be constantly added to as additional software and hardware are tested for compatibility.

Summary

This AppNote covered some of the major features in the NetWare 5 operating system, such as NCP independence over multiple protocols, a new server memory model, multi-processor kernel with MPK server hardware support, and the use of the NetWare Configuration file for keeping track of server configuration and SET parameters.

Future AppNote articles will delve into more detail on the administrative elements assigned to NetWare 5's management utilities and SET parameters.