SEPTEMBER 1995

NOVELL® RESEARCH

An Introduction to Novell's NetWare Client32 for Windows 95

KELLI FRAME
Technical Documentation
Operating Systems Division

KURT KRIEGER-JAMES
Technical Documentation
Operating Systems Division

Novell's new 32-bit client for Windows 95 sports a redesigned architecture and many new or enhanced features to make it faster, more reliable, and more usable than previous NetWare client software. This Application Note provides a brief overview of the NetWare Client32 software and outlines a few of the most significant new features and improvements. The purpose of this document is to give network designers, system administrators, and users an advance look at the new client's capabilities.

ACKNOWLEDGEMENTS

Thanks to Brad Young at Novell for his help with this AppNote.

TRADEMARKS

NetWare, the N-Design, and Novell are registered trademarks and the NetWare Logotype (teeth logo), NetWare Directory Services, NDS, NetWare Loadable Module, and NLM are trademarks of Novell, Inc in the United States and other countries. UNIX is a registered trademark in the United States and other countries, licensed exclusively through X/Open Company, Ltd. UnixWare is a registered trademark of Novell, Inc. in the United States and other countries.

Macintosh is a registered trademark of Apple Computer, Inc. IBM and OS/2 a re registered trademarks of International Business Machines Corporation. Microsoft, MS-DOS, and Windows are registered trademarks of Microsoft Corporation. All other product names mentioned are trademarks of their respective companies or distributors.

DISCLAIMER

Novell, Inc. makes no representations or warranties with respect to the contents or use of these Application Notes (AppNotes) or of any of the third-party products discussed in the

Novell, Inc.
122 East 1700 South
Provo, Utah 84606 USA

Introduction

Novell's new 32-bit client for Windows 95 sports a redesigned architecture and many new or enhanced features to make it faster, more reliable, and more usable than previous NetWare client software. This Application Note provides a brief overview of the NetWare Client32 software and outlines a few of the most significant new features and improvements. The purpose of this document is to give network designers, system administrators, and users an advance look at the new client's capabilities. Future AppNotes will provide more detailed information about NetWare Client32, including a theory of operations and installation/ configuration tips.

Overview of NetWare Client32

Client32 is a 32-bit protected mode NetWare client for use on computers running MS Windows 95. It provides connectivity to NetWare 2.2, NetWare 3.1x, and NetWare 4.x servers.

Architecture

Several fundamental changes were made in the design of the Client32 requester. The new requester comprises two fundamental parts: the NetWare I/O Subsystem (NIOS) and client NetWare Loadable Modules (NLMs).

The NIOS insulates the core client modules from the host operating system by providing an OS abstraction layer that core modules can use to access system services. Instead of making OS calls to access system services, Client32 core modules make NIOS calls.

In addition, NIOS provides services to manage client NLMs, using dynamic, self-configurable parameters where possible. For example, take the number of open IPX sockets. If more sockets are opened, IPX.NLM dynamically allocates more memory to handle the socket.

There are certain things that all the Client32 NLMs have in common:

- They are dynamically loadable and unloadable.
- They use NLM executable format.
- They run in a 32-bit flat memory model.
- They allocate memory that is guaranteed not to move or be discarded.

- They are fully language enabled.
- They require no static configuration information from users.
- They port across platforms, transport protocols, and name services.

Hardware Requirements

The NetWare Client32 requires at least an Intel 80386 processor. It requires only 4KB of conventional or UMB (Upper Memory Block) memory. The remaining portion of the client is loaded into XMS (Extended Memory Specification) memory. Client32 requires HIMEM.SYS or an equivalent memory manager. Currently the XMS memory requirement for the client is 800KB, excluding cache memory.

Installing the Client32 Software

Installing Client32 is easier than installing a NetWare client has ever been. There are four ways to install Client32:

- At the same time as Windows 95
- Auto Client Update
- Single-user, typical install
- Single-user, customized install

Each of these methods uses the Windows 95 Network Device Installer (NDI) and .inf script files to ensure full integration with the Windows 95 environment. In addition, Novell's Client32 installation program incorporates Windows 95 *property pages* that have been created specifically to set Client32 configuration parameters contained in the Windows 95 Registry. (Property pages are the graphical user interface that replace editing the network configuration settings by hand.)

Installing at the Same Time as Windows 95

To install the NetWare Client32 software at the same time as Windows 95, the administrator simply puts the Windows 95 install image on a server and then updates it to default to Novell's *NetWare Client32* instead of Microsoft's *Client for NetWare Networks*. The NDI will then install Client32 from the .INF files and property sheets.

Auto Client Update

This is a very important feature for network administrators who are faced with the task of updating numerous client workstations when a new version of a client is released. To configure the client on any number of workstations, the administrator need only specify the user containers or groups to be affected by the install, make any necessary changes to .INF files and property sheets, and write any necessary scripts.

When users in the specified groups or containers next log in, a script executes which checks to see if the client is older than the latest client being installed. If it is, the install program executes automatically without any input from the user.

Single-user, Typical Install

This is an administrator-driven, automated install. Users do not need to be knowledgeable about networking concepts to execute this install. In this method, an executable file removes the Microsoft NetWare client or any other NetWare network component. It then sets up certain files that the Microsoft NDI will use in

installing the Novell NetWare Client32.

The typical install (with administrator-set defaults) installs the Novell NetWare Client32, a Novell ODI Driver, and all existing network components that the administrator has specified in a NetWare administator information file.

Single-user, Customized Install

A knowledgeable network user has the option of configuring network parameters manually, including the client, protocol, and network adapter. This option removes the Microsoft NetWare client and any other NetWare network component. Then it sets up files that the Microsoft NDI will use in installing the Novell NetWare Client32.

The custom install uses the Windows 95 Network Control Panel applet and property page features. The screen shots below show some of the parameters being set.

Figure 1 shows the screen in which you select the client being installed.

Figure 1: In the "Select Network Client" screen, you choose to install the Novell NetWare Client32 software.

[Figure Not Available]

To select the network adapter that matches your hardware, you use the screen shown in Figure 2.

Figure 2: The Select Device screen allows you to specify the manufacturer and model of your network adapter.

[Figure Not Available]

Configuring Network Parameters. It is best to configure the client before the actual file copying takes place so that the necessary files can be copied. After choosing which network components will be installed, you can configure any of them by highlighting that component and choosing *Properties* to bring up the property pages associated with that component.

A key property page for connecting to the network is the *Novell NetWare Client32 Requester Property Page*, shown in Figure 3. In this screen, the user specifies the preferred server, preferred tree, and a name context. The first network drive = F is automatically entered by Client32 Install.

Figure 3: On the NetWare Client32 "Requester" property page, you specify your preferred server, tree, and name context.

[Figure Not Available]

Once you have selected and configured all the components to install, the Network Control Panel applet comes back. The list of network components is now filled out, as shown in Figure 4.

Figure 4: The Network "Configuration" sheet lists the network components you have selected to install.

[Figure Not Available]

Pressing OK at this point will start the actual file copying stage of the installation process.

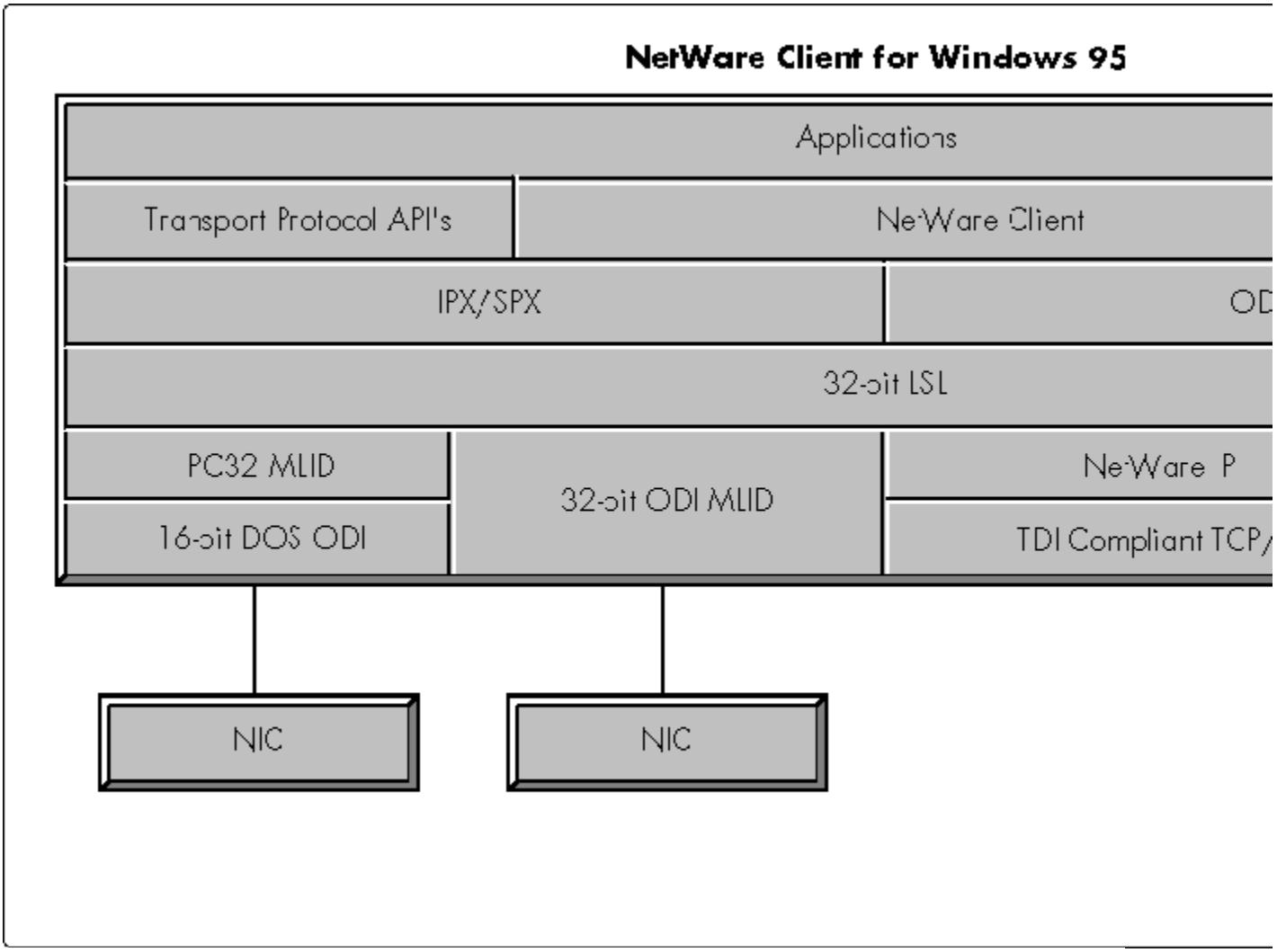
Protocol and LAN Driver Support

Like other NetWare clients, the NetWare Client32 for Windows 95 allows users to use a number of different protocols, clients, and LAN drivers (MLIDs). It supports ODI drivers for 16-bit network interface cards (NICs) and 32-bit NICs, as well as NDIS (Network Driver Interface Specification) controlled drivers.

Client32 supports a number of different methods for connecting from the workstation to the server. Figure 5 shows three different paths to a network interface card that are possible with Client32:

- Standard 32-bit ODI MLID
- PC32MLID over a 16-bit DOS ODI MLID
- VMLID over NDIS

Figure 5: NetWare Client32 supports several different combinations of protocols, clients, and LAN drivers.

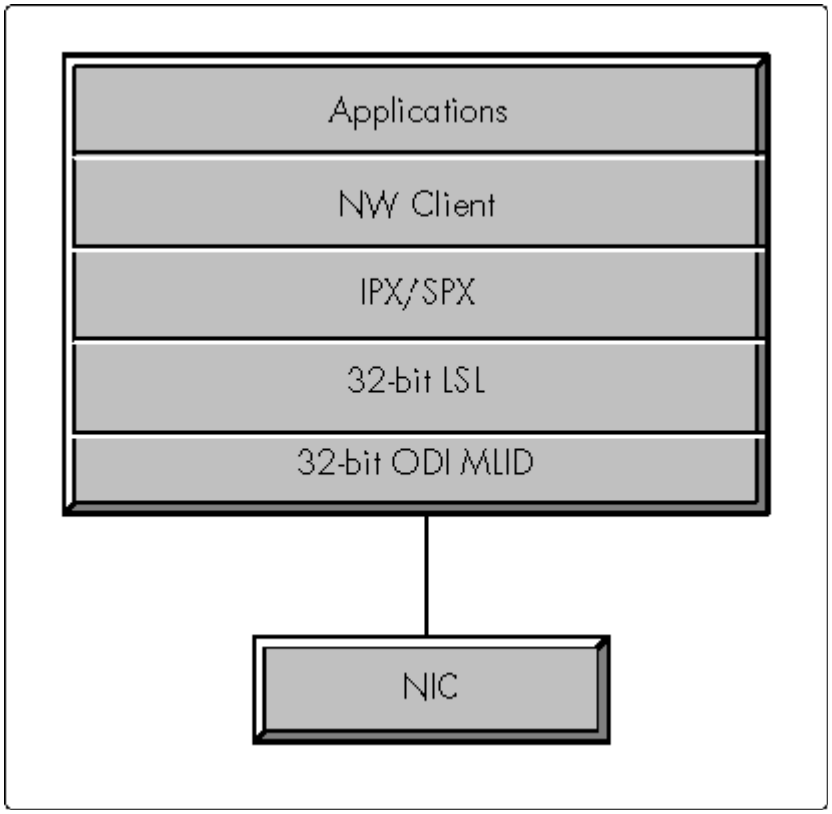


Note: The darker shaded portions in Figure 4 indicate components supplied by Microsoft.

Standard 32-bit ODI LAN Driver

If a user chooses to go with the standard Novell configuration using a 32-bit ODI LAN driver, the underlying architecture is simple and easy to configure (see Figure 6).

Figure 6: With a 32-bit ODI LAN driver, a client uses this standard Novell configuration to connect to a server.



PC32MLID over 16-bit DOS ODI Driver

PC32MLID (Pseudo C 32-bit Multiple Link Interface Driver) is a program that enables a workstation running NetWare Client32 to use a 16-Bit DOS ODI driver when a 32-Bit MLID is not available. This program allows drivers written to the 16-Bit DOS ODI specification to function as if they were written to the new 32-Bit C-ODI specification. Thus Client32 can use unmodified NetWare OS-compatible (3.x and 4.x) LAN drivers, which provides a huge pool of proven, certified drivers for the Client32 environment.

PC32MLID functions like a 32-Bit LAN driver, but it transmits and receives 16-bit data packets to and from the network. On transmissions, the PC32MLID converts data packets from the 32-bit protected mode memory format into the conventional real mode memory format before sending. On reception, the PC32MLID translates the 16-bit real mode memory addresses of the data packets into 32-bit protected mode addresses for the protocol stack.

PC32MLID Installation. The PC32MLID program is installed during the installation of the NetWare Client for Windows 95. After you have selected the Novell NetWare Client 32 Workstation under *Network Clients*, choose the ODI Driver sheet and select "16 Bit ODI Driver", as shown in Figure 7.

Figure 7: To install PC32MLID, select the "16 Bit ODI Driver" option when installing Windows 95.

[Figure Not Available]

The PC32MLID program will automatically load and the installation of the NetWare Client for Windows 95 will continue.

NDIS and VMLID

Users may also choose to use the NetWare Client with a LAN driver that adheres to Microsoft's NDIS specification. VMLID (Virtual MLID) is a NetWare program that enables NetWare's 32-bit LSL to communicate with Microsoft's NDIS stack.

ODINSUP is a NetWare program that enables the SMB Client (or the Microsoft Client) to communicate with the 32-bit LSL. (Normally, the SMB Client uses NWLINK.)

Multiple NIC Support

The NetWare Client for Windows 95 supports multiple NICs simultaneously. That is, IPX can bind to multiple frame types on multiple LAN adapters at the same time. Once the client is installed, it can send and receive packets to different frame types without the user having to reinstall the client or change the configuration. This allows the user to communicate with different kinds of network topologies simultaneously and transparently. For example, the user could make a call to a server on an Ethernet network and then make another call to a different server on a Token Ring network, without making any adjustments.

Client32 will not only support bindings to multiple NICs, it will also support auto-binding to dynamically loaded LAN drivers.

Transport Protocols

IPX/SPX and TCP/IP are supported equally with this client. In addition to running with NetWare's native IPX/SPX protocol, Client32 can run over any TDI-compliant TCP/IP stack, thus offering users freedom of choice.

Note: Most protocols will be ported to an NLM format for use with Client32.

IPX/SPX. The standard transport protocol provided with the Client32 for Windows 95 is Novell's IPX/SPX protocol. This protocol is implemented in the IPX.NLM file.

Novell's Internetwork Packet eXchange (IPX) is a connectionless datagram protocol. IPX is supported by most router venders and can be used to build all sizes of networks, from a single LAN segment to large geographically dispersed networks. By contrast, Novell's Sequenced Packet eXchange (SPX) is a connection-oriented, message-based protocol. SPX uses IPX for full access to the network, but uses additional control information to guarantee that data is transferred reliably and in the correct order.

Together, IPX and SPX can be accessed by applications in one of six ways:

NWSIPX	32-bit Protected mode
NWIPXSPX	16-bit Protected mode
WINSOCK	16-bit Protected mode
WSOCK32	32-bit Protected mode
TLI/XTI	16-bit Protected mode
INT2F/7A	16-bit DOS mode

NWSIPX. NWSIPX is a new 32-bit interface to the IPX/SPX protocol that will be available on Windows 95,

NT, OS/2, and other platforms. NWSIPX is designed to provide the fastest and fullest support of the 32-bit IPX/SPX protocol stacks. The NWSIPX API is implemented in the NWSIPX.DLL file.

NWIPXSPX. In addition to providing a 32-bit interface to IPX, NWSIPX includes several enhancements to the 16-bit NWIPXSPX API. NWSIPX replaces the traditional NetWare IPX and SPX Event Control Blocks with a single NWTCB (transport control block) and unifies the IPX/SPX interfaces into a single set of primitives. Applications can choose to manage events by callback function, polling, blocking, or event notification. Operational parameters can be read and modified by the application for greater control of runtime performance.

Application programs written to the NWIPXSPX API can be converted to utilize the 32-bit NWSIPX API in a straightforward manner. The overall logic flow of an existing application does not have to change, and in many areas can be greatly simplified.

WinSock and WSOCK32. WinSock is an industry-standard application interface to transport protocols. The Client32 WinSock library supports not only the 16-bit WinSock 1.1 specification, but also the IPX/SPX protocol stack. Novell also supports Microsoft's WSOCK32, a 32-bit version of WinSock. Using these two libraries, a 32-bit application written to the WinSock specification can run over either IPX/SPX or TCP/IP.

WinSock, which is currently the number one transport interface for MS Windows, provides functionality similar to BSD UNIX sockets. Because of this functional equivalency, many applications that access the Internet have been ported to MS Windows using the WinSock specifications.

Integrated Support for NetWare/IP

Many system administrators want to run their networks with a single transport protocol. The NetWare Internet Protocol, known as NetWare/IP or NWIP, allows users to utilize the services provided by NetWare and to run NetWare-aware applications in an IP-only environment. To accomplish this, NetWare/IP encapsulates all Internetwork Packet eXchange (IPX) traffic between client and server machines in IP packets using the User Datagram Protocol (UDP). UDP/IP provides the optimal transport for NWIP because it closely mirrors the level of functionality that IPX provides.

The NetWare Client Installation utility makes installing NetWare/IP as easy as selecting *Protocol...Novell...NWIP* on the control panel and filling in the properties on the property page that appears. NetWare/IP is installed between the Link Support Layer (LSL) and Microsoft's TCP/IP stack, and functions as a virtual MLID. Like an MLID, NetWare/IP sends and receives packets, either by encapsulating IPX packets in UDP/IP packets or by translating UDP/IP packets into IPX packets. Unlike an MLID, NetWare/IP does not interact with the network interface card directly. Instead, it interacts with the TCP/IP stack.

Since NetWare/IP is IP v2.1 compliant, it represents the most advanced and efficient means of fully utilizing the benefits of NetWare on an IP-only network.

Support for SMB Client and RAS Client

The Server Message Block (SMB) Client is a component of Microsoft's Windows 95 that provides peer-to-peer connectivity. Remote Access Services (RAS), another component of Windows 95, provide dial-up access to the network over a modem. The SMB Client and RAS run over Microsoft's NWLINK, which is an IPX/SPX stack that interfaces with NDIS.

NWLINK2. To provide the user access to the SMB and RAS when Novell's ODI IPX/SPX stack is loaded, Novell created a shim called NWLINK2 that sits on top of the ODI IPX/SPX stack. NWLINK2 is a VxD that makes Novell's IPX/SPX stack look like NWLINK to any component making calls into it. In this way, Novell provides support for the SMB and RAS clients running over Novell's IPX/SPX protocol. It also allows any module written to utilitize NWLINK to run over Novell's IPX/SPX. These components include Microsoft's 32-bit WinSock and NetBIOS.

New Features

NetWare Client32 for Windows 95 has several new features that significantly improve the user interface over that of the previous client software. This section highlights three of these:

- GUI login
- NetWare Provider
- NetWare Application Launcher (NAL)

GUI Login

If Client32 is loaded on your Windows 95 workstation, you will get the NetWare GUI Login when you log in to your Windows 95 workstation. The GUI login lets you run a login script from a Windows login, which has never been possible before. This feature supports old login scripts (stored in the NetWare file system for NetWare 2.x and 3.x environments, or in Directory Services for NetWare 4.x environments). There is also an option for a simple login, where the user is asked for only a name and password, and never sees other login options.

The Client32 GUI Login allows the user to execute a user or system login script, log in to multiple trees, update search drives, and update environment variables. Both Bindery and NDS connections are supported.

As shown in the following screen shots, there are three categories of choices that can be configured when logging in: login, connection, and script.

Figure 8 shows the initial "Login" sheet where you enter your username and password.

Figure 8: The GUI login's "Login" sheet.

[Figure Not Available]

On the "Connection" sheet, the user can specify a tree, server, and whether this will be a Bindery connection instead of the default DS connection (see Figure 9).

Figure 9: The GUI login's "Connection" sheet.

[Figure Not Available]

If the user has an alternate login or profile script, it can be specified in the Script sheet (see Figure 10).

Figure 10: The GUI login's "Script" sheet.

[Figure Not Available]

After the login script executes, the user can request to see the results of the login, including server authentications and drive mappings as shown in Figure 11.

Figure 11: The GUI login allows the user to see the results of the login.

[Figure Not Available]

The NetWare Provider

The NetWare Provider for Windows 95 takes full advantage of the Windows 95 user interface to offer NetWare and Netware Directory Services functionality, including access and browsing of network resources, and working with resources on multiple trees. Novell's NetWare Provider is fully integrated with the *My Computer*, *Explorer*, and *Network Neighborhood* features of Windows 95. It contains many of the same features as the NETWARE.DRV driver provides for Windows 3.1 clients.

The Provider establishes connections only to file and print objects (either through UNC paths or map and capture), so these are the only objects displayed in any of the Windows 95 utilities. Since Windows 95 supplies doesn't supply any APIs for displaying user and other objects, no provider that follows the Windows 95 provider architecture will have this capability. This means *Network Neighborhood* and *Explorer* cannot expose all NetWare functions.

Novell's NetWare Provider supplies functionality beyond *Network Neighborhood* and *Explorer* specific to NetWare and NDS. As before, Novell supplies other utilities, such as NWAdmin and various command line utilities, to perform operations on these other objects and to take full advantage of the power of NetWare.

The screen shots that follow show a subset of the NetWare Provider's functionality: browsing for NetWare resources, viewing NDS objects, and changing a context.

Browsing NetWare Resources. To browse a Novell network, choose *Network Neighborhood*. In the resulting display, NDS trees are shown as tree icons. Figure 12 shows an example screen listing the NDS trees visible to this user.

Figure 12: When using Network Neighborhood to browse, you can see all the NDS trees that are available on your network.

[Figure Not Available]

Viewing NDS Objects. To see, for example, the file and print objects contained in a given context, you need only *Explore* on a given part of the tree. The results of this action are shown in Figure 13.

Figure 13: When exploring a particular part of the tree, you can see the NDS objects contained in a particular context.

[Figure Not Available]

Changing Context. Changing your NDS context is as easy as right-clicking on an NDS tree and selecting *Change Context.* The resulting window shows your current context and allows you to change to a different default context (see Figure 14).

Figure 14: The NetWare Provider allows you to change your NDS context.

[Figure Not Available]

User Profile and System Policy Support. The Provider supports Windows 95 User Profiles and System Policies.

User Profiles allow users to have the same desktop no matter where they log in on a network. To make this

possible, Windows 95 copies each user's profile information (configuration preferences and options) both to a local drive and to the network in a specified directory. Microsoft suggests that this directory be a subdirectory of the system mail directory (*server**sys**mail**user_id*), but such a directory is not always present in an NDS environment.

To support this option, the Provider need only return to Windows 95 the path from which to access the user profile. In a Bindery connection, the Provider will return the mail path specified above. In an NDS connection, the Provider returns the path specified as the *Home Directory* in NWAdmin (*Environment*/ *Home Directory*). (If the user or system administrator has not specified a *HomeDirectory*, then the Provider informs Windows 95 that no path has been specified, and User Profiles cannot be supported.)

System Policies allow the system administrator to customize and control the users' desktops, both in terms of what they can see and what they can access. These decisions are stored in a file called CONFIG.POL, which overrides the settings in the Registry under USER.DAT and SYSTEM.DAT. When a user logs in, Windows 95 checks the user's configuration information to find where the policy file is stored, and then copies the policy settings into the Registry.

The Provider supports System Policies by copying the CONFIG.POL file to the public area on the SYS volume of the user's preferred server (*server**sys**public*\), and making sure the preferred server has this area.

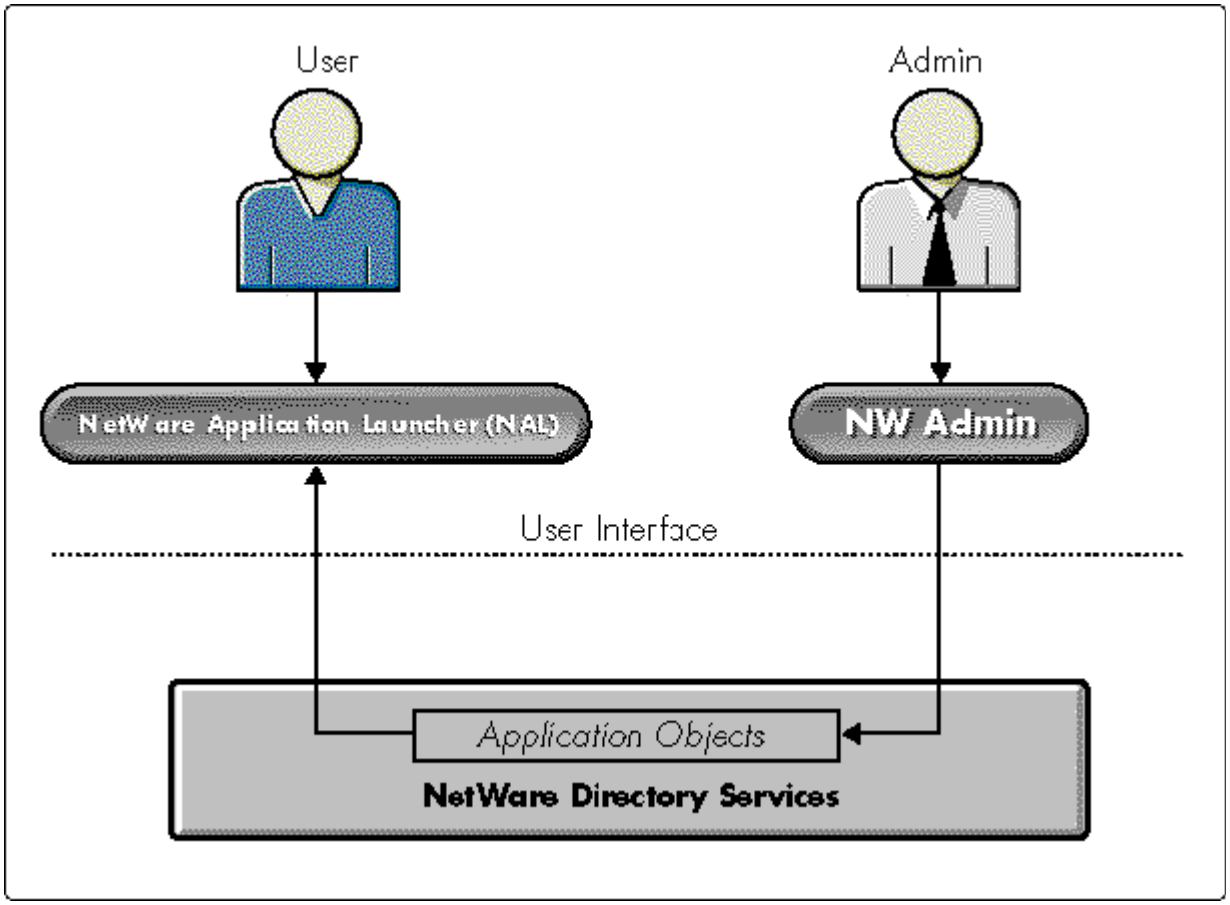
NetWare Application Launcher (NAL)

One of the persistent challenges of administering or using a network has been managing access to network applications. The NetWare Application Laucher (NAL) is a new NetWare utility that drastically simplifies the network administrator's work in setting up, configuring, and upgrading network applications, and in saying who has access to the application.

With Client32, the network administrator can use NWAdmin to configure applications on all client workstations, and the information is stored in NDS. NAL then accesses NDS to get the object information for the user, and the user will see a window come up with all available applications. This design takes good advantage of the power of NetWare Directory Services.

As shown in Figure 15, the administrator uses NWAdmin to store application object information in NDS, and NAL retrieves that information for each user. There are two steps in the process of launching applications. First, the administrator creates and configures the application object. Second, the user launches the application.

Figure 15: Design of the NetWare Application Launcher (NAL).



Setting Up an Application Object. To set up an application, the network administrator must first select the container for an object (can be a user, group, or container), and then create the object. As Figure 16 shows, there are four new classes of application objects available in NWAdmin when the network administrator chooses to create an object: DOS applications, Windows 3.x applications, Windows 95 applications, and Windows NT applications

Figure 16: Four new objects are available in NWAdmin for different types of applications.

[Figure Not Available]

Note: These application objects are available because of a snap-in module that extends the NDS schema for Client32 to include a new class of object the application object.

After creating the object, the administrator can configure all of the application information, including mapping drives, setting up execution parameters, capturing printer ports, specifying command line parameters, and assigning associations. Figure 17 shows a sample input screen for configuring an application object for WordPerfect for Windows.

Figure 17: Once an application object is created, you can configure its parameters in this screen.

[Figure Not Available]

Launching the Application. After all the parameters are configured for the application, the application is visible to all users specified by the network administrator. The user will log in and have access to any application object associated with that user's user object, group, or NDS container. As shown in Figure 18, the available applications are displayed in a window.

Figure 18: Once application objects are created and configured, they are available to authorized users in an Applications window.

[Figure Not Available]

Key Enhancements

In NetWare Client32, significant improvements have been made over previous NetWare clients in several key areas, including speed, performance, reliability, portability, and usability. This section focuses on the following specific enhancements:

- Enhanced NDS support
- Simplified printing
- Reliability (autoreconnect)
- Performance (client-side caching, packet burst, LIP, and multiplexed NCP connections)
- Security (packet signing)
- Network management

Enchanced NDS Support

Client32 allows greater access to the power of NDS than ever before, offering several significant new features.

Multiple Tree Support. A user may now authenticate to more than one NDS tree at a time. This capability is supported in the Provider, in GUI Login, and in NWCALLS.

Login and Authentication Done at the Requester. All code for login and authentication has been moved to NDS. This is significant mainly to DOS applications, which will realize a savings in conventional memory when the large libraries are no longer statically linked into the login executable. (Statically linked DOS applications will need to be relinked with new SDKs to realize this conventional memory savings.)

NDS Name Resolution Done at the Client. Name resolution will be done at the client when it is advantageous to do so. This will take some of the burden off the server, as well as allow the best NDS partition to be used when an object name is resolved to multiple addresses.

Simplified Connection Licensing. Licensing must be handled differently on NDS networks than on Bindery networks. In a Bindery network, when the user authenticates to a NetWare 2 or 3 server, that user is automatically authenticated to file and print services as well. In NDS (NetWare 4 servers), the user can authenticate to Directory Services, but not necessarily to file and print services. Since Directory Services are a free service, no license is used until an actual license NCP is sent by an application.

In addition to separating licensing from authentication, the Client32 requester is smarter about recognizing services that need to be licensed. It will license a connection automatically if certain NCPs are used, even if

the issuing application does not request a license. This relieves applications of some of the burden of deciding exactly which NCPs require a license. (Of course, doing explicit licensing and unlicensing may be advantageous for an application, as it will help the application track resources and errors more efficiently.)

Simplified Printing

Client32 supports "deviceless" printing. That is, it is no longer necessary to use print captures, and to associate a printer port with a specific printer. Instead, when you set up a printer, Windows 95 writes all of the printer configuration information to the Registry. To use the printer, a user simply clicks on that printer's icon.

The process of setting up the printer is very straightforward. The user selects *Network Neighborhood*, browses for the printer being configured, and then right-clicks on that printer and selects *Install*.

Note: When installing network printers, it is important to use the *Network Neighborhood*, which can see DS printers. The *Add Printer* option on *My Computer* can only see Bindery printers.

Figure 19 shows the first screen a user sees when installing a printer. It asks if the user will be printing from MS-DOS-based programs. DOS applications still require that printer ports be captured, so if the user answers yes, the next step is to capture printer ports.

Figure 19: If you still need to print from DOS applications, you must answer Yes and then set up a printer port capture.

[Figure Not Available]

If no DOS applications are being used, the user is next asked to select the make and model of the printer being configured so that the appropriate driver can be loaded (see Figure 20).

Figure 20: In this screen, you select the manufacturer and model of your printer.

[Figure Not Available]

Note: If the system administrator has previously configured the Windows 95 *Point and Print* option, the above step is skipped. *Point and Print* automatically associates a driver on the server with a printer, making it unnecessary for users to load drivers from a disk.

The user then has the option of naming the printer and deciding whether this printer will be the default printer for Windows-based applications (see Figure 21).

Figure 21: You can then enter a name for your printer.

[Figure Not Available]

At this point the printer is installed as network printer. It will be visible from the *Network Neighborhood* and can be printed to with a single click.

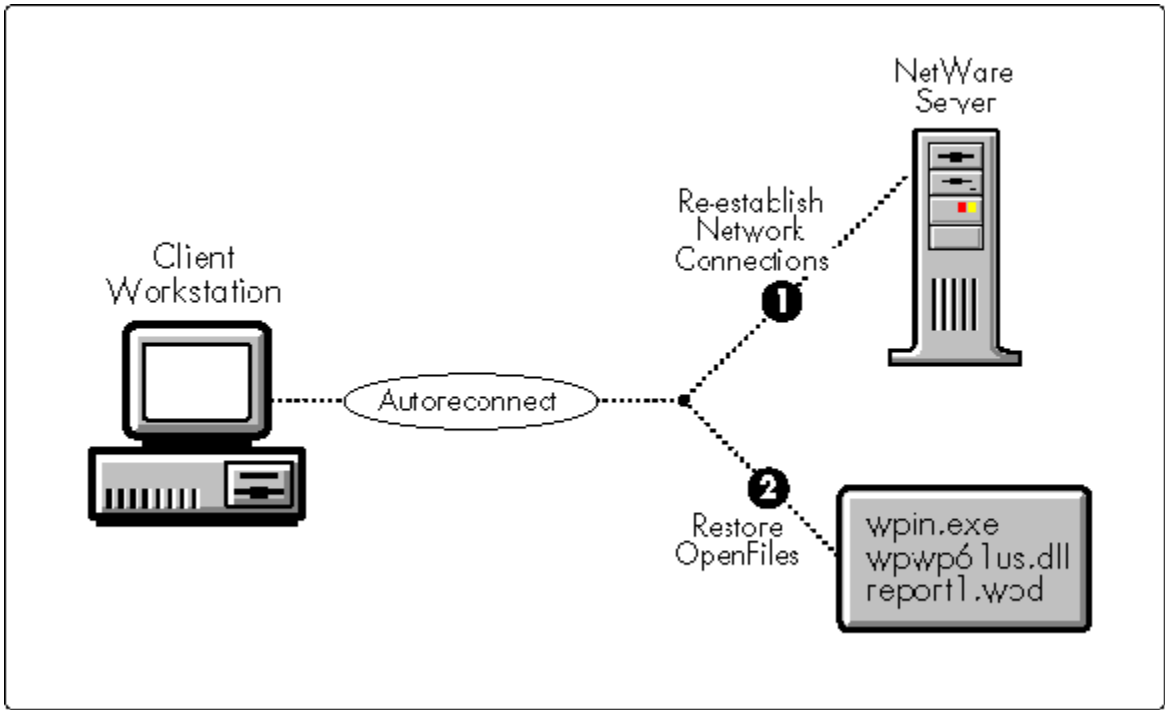
Autoreconnect

A server connection may become invalid by events such as the server going down, the server's watchdog timing out a connection, or as a result of a mobile operation. When a connection becomes invalid, it must be revalidated through the process of auto-reconnection. To make autoreconnection possible, Client32

stores information regarding each open resource at the client, so that the resource can be re-established after a connection loss.

In the event of a network failure, NetWare Client32 reestablishes the entire network environment including connections, drive mappings, printer captures, open files, and file locks. Not only does Client32 automatically detect errors and rebuild lost connections and drive mappings, it also restores open files, file locks, and other user state information (see Figure 22). This improves the overall reliability of the client and minimizes the effects of a network failure on the end user.

Figure 22: With autoreconnect, the workstation client reconnects the network connections, and then restores open files as well.



Client32 provides five levels of autoreconnect support. Each level adds to the support provided by the previous level.

Level	Description	Notes
4	Read/write files, file locks, and record locks are tracked and restored	Manual intervention will be required only when accessing files that cannot be fully cached or when network connectivity cannot be rebuilt. This option uses True Commit* to ensure integrity of files during autoreconnect when communicating with servers running NetWare 4.02 and earlier versions.

3	Read/write files are restored (without True Commit*)	This is the default level for autoreconnect support. Clients will have read-write files restored, but these files will be rebuilt without True Commit. If your server is stable but network connectivity is sporadic, this level of support is a viable option. Manual intervention will be required only when accessing files that cannot be fully cached or when network connectivity cannot be rebuilt.
2	Read-only files are tracked and restored	Clients will have their read-only files reopened at the server if the connection was lost. Manual intervention will be required only when network connectivity cannot be rebuilt.
1	Connection table, drive mappings, directory handle table, and printer redirection information are tracked and rebuilt	The user sees messages with abort, retry, and ignore options that require manual intervention. Users will be asked to manually intervene only if they attempt to access a resource when network connectivity cannot be rebuilt. This level and higher also provides mobile support for the specific resources supported.
0	No autoreconnect support	With no autoreconnect support, the user will see messages with abort, retry, or fail options that require manual intervention. This level of autoreconnect support does not track authentication information or store reconnection information. It is not available if the workstation is configured for mobile support.

*True Commit means that the server acknowledges write requests only after actually writing the data to disk. When using True Commit, after a file's data is written to the server, it is held for a time before being released. If autoreconnect is initiated within this time, this data is rewritten. True Commit slows down the server and write operations, but helps to ensure file data integrity.

Client-side Caching

To improve client performance, the Client32 requester caches files, directories, and file locks.

Caching Files. Because user reads and writes are the most frequently requested network operations, network performance improves by buffering these reads and writes, thereby reducing network access. This is called *caching*.

The Client32 requester has an integrated cache that buffers network reads and writes so that network access is made less frequently and in bigger blocks. Instead of having every user read and write go directly to the server, the requests are instead sent to a cache buffer.

The goals of the integrated cache are threefold:

- To buffer small reads and writes (small meaning smaller than the size of the cache blocks)
- To buffer frequently accessed data so that network traffic and delays are reduced.
- To maintain cached data across file open/close operations for frequently accessed files.

The integrated cache monitors file activity (opens, creates, closes, remote-copies, and flush functions), altering cache status accordingly. Files that are being shared may not be cached, thus ensuring that no one gets bad data for example, no one reads data that is being written over by someone else.

A client determines the access mode (read or write locks) of a file in one of two ways: by noting the mode in which the file was opened, or by using a bidirectional NCP.

Bidirectional NCPs allow a client to cache data even though others are granted access to that data. As long as actual access has not occurred (that is, no one has read the file even though they could), the client may continue caching. As soon as the server notifies the client that actual access has been made to the file (someone opened it to read or write), the client flushes any dirty buffers (in the case of a write) or invalidates read buffers (in the case of a read) before continuing.

Caching Directory Information. The functions that return directory information use caching to improve performance. When a search context is opened, a check is made to see if the contents of the cache for that context are still synchronized with the server. If they are, requests are serviced from the cache; otherwise, the context is treated as the first time used and directory entry information is placed in the cache as the entries are enumerated.

Caching File Locks. If a file is opened exclusively, the physical-record locks on that file will be cached since the file is already protected by the exclusive open. If that file's access status changes (by a bidirectional NCP perhaps), those physical locks will be issued without application intervention.

Enhanced Packet Burst and LIP Support

Client32 has built-in support for Packet Burst, a protocol built on top of IPX that speeds the transfer of multiple-packet file reads and writes. Working as an entity within the NetWare Core Protocol (NCP), Packet Burst speeds the transfer of NCP data between a work-station and a NetWare server by eliminating the need to sequence and acknowledge each packet.

With the added efficiency of the Packet Burst protocol, Client32 can send a whole set (burst) of packets before it requires an acknowledgment. By taking into account the total network performance, Packet Burst regulates the number of packets sent in each burst and the time delay between each burst. In this manner, it optimizes the available resources and avoids network congestion.

The Large Internet Packet (LIP) feature makes it possible to send large internet packets across networks, in addition to sending them within a single network. The Client32 LIP algorithm has been improved over the VLM version. It can discover packet size more quickly, and in fewer packets.

Client32's Packet Burst and LIP support have both been optimized for overall network performance, including wide area network performance.

Multiplexing NCP Connections

In the past, if a user had a pending request on an NCP connection, the user had to wait for a reply from that connection before using other NCP connections. Client32 can service multiple NCP connections simultaneously, without having to wait for the reply to be completed from any one request.

Packet Signing

To increase the security of the network, Client32 supports packet signing. Packet signing means that when the client and server communicate, each one "signs" packets with a unique signature before trasmitting the packet. The receiver then compares the received packet to a private key, and rejects it if the signature doesn't match the key. This feature thereby precludes the possibility of an intruder faking a packet from a client or server.

Client32 signs the first 64 bytes of each packet. Due to the computational resources required to generate the signature, packet signing necessarily incurs a performance hit. To help mediate this, the user is able to select one of four levels of packet signing:

Level	Description
0	Packet signing is disabled; the client will never sign packets.
1	Packet signing is enabled; the client is capable of signing packets with the server, but prefers not to. Signing will only occur if the server requires it.
2	Packet signing is preferred; the client is capable of signing and would prefer to sign packets unless the server cannot sign packets.
3	Packet signing is required; the client will not communicate with a server without signing packets.

The client and server negotiate the level of packet signing that will be mutually supported when the connection is first created.

To speed up the process of signing packets, Client32 has implemented a patented process whereby it "predicts" the response from the server while waiting for a reply from the server. For example, if the client has requested access to a resource, it might predict that the answer from the server will be affirmative. To speed up the interaction, the client would use the time it spends waiting for a reply from the server to build the 64-byte, signed reply it expects. When the actual response packet arrives, the client need only do a comparison rather than a build and a comparison. If the response from the server does not match, no time was lost, since the time it spent building the guess was time spent waiting for the server anyway.

Network Management Services

NetWare Client32 includes several components that provide information about network configuration and performance. The key components are described below.

SNMP Agent. The SNMP NLM features a full implementation of the SNMP v1 protocol. It provides a

generic API that can be used by various transport providers and is currently used to implement SNMP over IPX and UDP. The SNMP NLM implements MIB-2's system group, interfaces group, and SNMP group. (Other MIB-2 groups are implemented by the TCPIP NLM.) The SNMP NLM also provides an extensible agent API for other NLMs to instrument various MIB objects to the agent. For example, the HostMIB NLM uses this API to implement the Host Resources MIB.

Host Resources MIB. The HostMIB NLM is an implementation of the Host Resources MIB as defined by RFC 1514 and Novell's extensions to it. The HostMIB NLM uses the services provided by the SNMP NLM. Management consoles, such as Novell's NMS console, use the SNMP protocol to communicate with the HostMIB NLM.

NetWare Diagnostics. NetWare Diagnostics are implemented as part of the IPX NLM. NetWare Diagnostics provide applications with a method for identifying internetwork nodes, building internetwork maps, identifying software components on each node, and retrieving information about those components.

Network Management Responder (NMR). The NMR is a client NLM that provides general configuration information beyond what is available through the standard Diagnostic Services. The NMR actually includes two responders:

- The Machine Configuration Responder, which returns information about the workstation hardware
- The ODI Responder, which returns information about the workstation's networking software, such as MLIDs and the LSL

Generic Network Management Agent (GNMA). GNMA is a software component implemented to extend IPX diagnostics functionality. GNMA provides a way for developers to extend the information that can be retrieved using the existing IPX diagnostic infrastructure. This component provides a method of communi-cation for various types of modules to make available to the network their unique services or information. GNMA includes code to maintain backward compatibility with previously implemented 16-bit GNMA APIs.